Quality-Aware Sampling and Its Applications in Incremental Data Mining

Kun-Ta Chuang, *Member*, *IEEE*, Keng-Pei Lin, and Ming-Syan Chen, *Fellow*, *IEEE*

Abstract—We explore in this paper a novel sampling algorithm, referred to as algorithm *PAS* (standing for **P**roportion **A**pproximation **S**ampling), to generate a high-quality online sample with the desired sample rate. The sampling quality refers to the consistency between the population proportion and the sample proportion of each categorical value in the database. Note that the state-of-the-art sampling algorithm to preserve the sampling quality has to examine the population proportion of each categorical value in a pilot sample a priori and is thus not applicable to incremental mining applications. To remedy this, algorithm *PAS* adaptively determines the inclusion probability of each incoming tuple in such a way that the sampling quality can be sequentially preserved while also guaranteeing the sample rate close to the user specified one. Importantly, *PAS* not only guarantees the *proportion consistency* of each categorical value but also excellently preserves the *proportion consistency* of multivariate statistics, which will be significantly beneficial to various data mining applications. For better execution efficiency, we further devise an algorithm, called algorithm *EQAS* (standing for **E**fficient **Q**uality-**A**ware **S**ampling), which integrates *PAS* and random sampling to provide the flexibility of striking a compromise between the sampling quality and the sampling efficiency. As validated in experimental results on real and synthetic data, algorithm *PAS* can stably provide high-quality samples with corresponding computational overhead, whereas algorithm *EQAS* can flexibly generate samples with the desired balance between sampling quality and sampling efficiency. In addition, while applying the sample generated by algorithms *PAS* and *EQAS* to incremental mining applications, a significant efficiency improvement can be obtained without compromising the resulting precision, showing the prominent advantage of both proposed algorithms to be the quality-aware sampling means for incremental mining applications.

Index Terms—Sequential sampling, incremental data mining.

1 Introduction

1.1 Motivations

Recently, important applications have called for the need for incremental mining to discover up-to-date patterns hidden in the continuous input data [1], [2], [3], [4], [5]. It is believed that the demand of online sampling techniques is increasing since they can prominently reduce the computational cost of the incremental mining applications [6]. However, using sampling prior to the targeted applications inevitably leads to the result being inconsistent with that obtained without sampling. If using sampling leads to a very inconsistent mining result, its usefulness for scaling up is in question. In practice, the level of consistency between results obtained in the whole population and those in a sample solely depends on the quality of the sample. Thus, how to guarantee the quality of samples is deemed the key to the success of sampling techniques [7]. In the literature, a common and successful measure of the sampling quality is to measure the consistency between the population proportion and the sample proportion of every measured pattern [8], [9], [10], [11].

Traditionally, random sampling is the most widely utilized sampling strategy for data mining applications. According to the *Chernoff bounds*, the consistency between the population proportion and the sample proportion of a measured pattern can be probabilistically guaranteed when the sample size is large [9], [12]. However, the overhead of the posterior mining applications will be increased when the sample size is large, thus inevitably degrading the benefit of sampling. Sampling mechanisms to guarantee a high sampling quality without increasing the sample size are still strongly demanded. To achieve this, the state-of-the-art sampling approach, named algorithm *EASE*, was proposed in [8] to guarantee the quality of generated samples with a desired sample size. Specifically, the goal of *EASE* is to precisely preserve the population proportion of each categorical value in the sample. According to the proposed epsilon-approximation method, algorithm *EASE* will obtain the final sample with the desired sample size by a process of repeatedly halving the intermediate samples. As such, the difference between the sample proportion and the population proportion of each categorical value in the final sample can be limited below ε, where the magnitude of ε depends on the desired sample size (a large sample size leads to a small ε and, in contrast, a small sample size leads to a large ε). Algorithm *EASE* is shown to be an effective sampling means to provide the prominent proportion consistency in the sample with a specified size. As compared to random sampling, the result in [8] demon-

- *K.-T. Chuang is with the Graduate Institute of Communication Engineering, National Taiwan University, No. 1, Sec. 4, Roosevelt Road, Taipei, Taiwan, ROC. E-mail: doug@arbor.ee.ntu.edu.tw.*
- *K.-P. Lin is with the Department of Electrical Engineering, National Taiwan University, No. 1, Sec. 4, Roosevelt Road, Taipei, Taiwan, ROC. E-mail: kplin@arbor.ee.ntu.edu.tw.*
- *M.-S. Chen is with the Department of Electrical Engineering and the Graduate Institute of Communication Engineering, National Taiwan University, Taipei, Taiwan, ROC. E-mail: mschen@cc.ee.ntu.edu.tw.*

Manuscript received 27 Jan. 2006; revised 5 Oct. 2006; accepted 9 Oct. 2006; published online 19 Jan. 2007.
For information on obtaining reprints of this article, please send e-mail to: tkde@computer.org, and reference IEEECS Log Number TKDE-0036-0106.
Digital Object Identifier no. 10.1109/TKDE.2007.1005.

strates that preserving the population proportion of each categorical value in the sample can significantly improve the resulting model accuracy of various posterior applications such as association-rule mining and the χ^2 test for independence, to name a few.

Note, however, that algorithm *EASE* in essence cannot sequentially generate the sample, thus it is not applicable to incremental applications. Formally, *EASE*, which can be categorized as a two-phase sampling mechanism [10], requires a pilot sample of the whole population to be generated beforehand, indicating that the population data set will be treated as a static one as opposed to a dynamic one. Such a constraint is infeasible in incremental mining applications, where they usually deal with time-variant data and each tuple is unknown before we receive it. Moreover, in *EASE*, each tuple will be repeatedly examined until it has been decided whether it is to be selected or discarded, implying that the sampling quality is acquired at the cost of execution efficiency. The required computational overhead of generating samples compromises the spirit of sampling to speed up the execution. Consequently, it is essential to develop a new sampling algorithm to incrementally generate high-quality samples while not compromising the sampling efficiency and not increasing the sample size.

As a result, we present in this paper a novel sampling approach, called algorithm *PAS* (standing for **P**roportion **A**pproximation **S**ampling) to achieve the goal of generating a high-quality online sample with the user-specified sample rate. As with *EASE*, the sampling quality in *PAS* is measured as the level of the *proportion consistency*, i.e., the consistency between the sample proportion and the population proportion of each measured pattern. Note that, in addition to applications of the frequent-pattern mining and the χ^2 test studied in [8], it is also reported that guaranteeing the *proportion consistency* provides a great benefit to many different mining applications such as supervised learning, clustering [9], [11]. We thus believe that providing the high *proportion consistency* of each measured pattern can lead to general-purpose and high-quality samples for different application needs. Furthermore, the *proportion consistency* can be guaranteed in two ways, namely, *absolute proportion consistency* and *relative proportion consistency*. Specifically, assuming d_i and s_i denote the population proportion and the sample proportion of a measured pattern, respectively, the value of $\left(1-\frac{|d_i-s_i|}{d_i}\right)$ can be viewed as the *relative proportion consistency* of the pattern. Conversely, its *absolute proportion consistency* is equal to the value of $(1-|d_i-s_i|)$. As opposed to guaranteeing the *absolute proportion consistency* (the goal in algorithm *EASE*), algorithm *PAS* aims to guarantee the *relative proportion consistency* because recent probabilistic thresholding methods for wavelet synopses point out that minimizing the relative error is the more desirable measure for data reduction techniques [13], [14]. In addition, due to the time-variant nature of real data, *PAS* sequentially reads

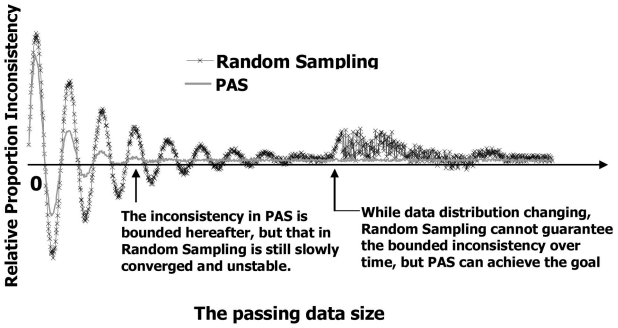


Fig. 1. Illustration of the relative proportion inconsistency over time.

the incoming population and generates the sample with the given sample rate on the fly, where the *sliding window* model is imposed. Therefore, the up-to-date population characteristics can be precisely maintained in the generated sample, allowing *PAS* to be directly applicable to incremental mining applications.

Briefly, the basic idea behind *PAS* is to adaptively determine the inclusion probability of each incoming tuple as time advances, where the inclusion probability will be determined according to two criteria: 1) The *relative proportion "inconsistency"* of every attribute value can be guaranteed toward a user-specified error bound ε and 2) the sample rate is close to the user-desired sample rate p. The concept is illustrated in Fig. 1. Specifically, *PAS* strives to minimize the *relative proportion inconsistency* of each attribute value progressively until the difference is smaller than ε. While ε is specified close to zero, *PAS* will quickly and stably keep the *relative proportion inconsistency* close to zero, as shown in Fig. 1, even though the data distribution is not stationary in a time-variant data source. In contrast, simple random sampling cannot guarantee that the relative error always approaches to zero in a time-variant data source, especially when the data distribution changes suddenly.

In addition, for multidimensional data, it is required to have an atomic unit of measured patterns, which is a multivariate statistic. However, maintaining the *relative proportion consistency* of every multivariate pattern incurs a large computational overhead, which is prohibitive in many applications. For efficiency purposes, *PAS* maintains *proportion consistency* of every attribute value, i.e., every single variate statistic, rather than every multivariate statistic, the same as in algorithm *EASE*. Importantly, even though *PAS* only maintains the *relative proportion consistency* of each categorical value, as shown in our analytical and algorithmic results, the *relative proportion consistency* of multivariate statistics can also be excellently preserved. In contrast, *EASE* may preserve the *proportion consistency* of each categorical value, but lose the *proportion consistency* of multivariate statistics.

Formally, as in algorithm *EASE*, algorithm *PAS* also unavoidably incurs the computational overhead to guarantee the sampling quality by continuously tracking the *relative proportion consistency* of each categorical value. To provide the flexibility of striking a compromise between the

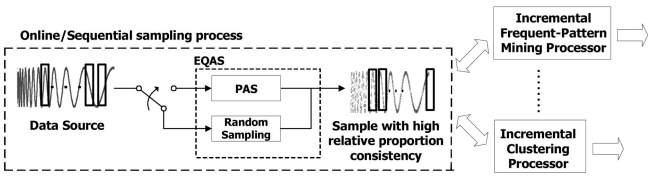


Fig. 2. Framework of sequential sampling PAS and EQAS over sliding windows.

sampling efficiency and the sampling quality, we further devise in this paper another sampling algorithm, named algorithm *EQAS* (standing for **E**fficient **Q**uality-**A**ware **S**ampling). The framework of algorithm *EQAS* is exhibited in Fig. 2. Specifically, from our empirical studies, algorithm *PAS* can quickly and stably guarantee the *relative proportion consistency* with the corresponding computational overhead. On the other hand, random sampling is very efficient, but the *relative proportion consistency* can only be slowly reduced. Random sampling is also highly sensitive to the burst sampling error, which is common when the data distribution is time-variant [15]. Due to their complementary properties, algorithm *EQAS* is devised by integrating random sampling and algorithm *PAS*. By appropriately switching between random sampling and *PAS*, algorithm *EQAS* is able to preserve the advantages of these two schemes while diminishing their side effects. In addition, while applying the sample generated by algorithms *PAS* and *EQAS* to incremental mining applications, a significant efficiency improvement can be obtained without compromising the resulting precision, showing the prominent advantage of both proposed algorithms to be quality-aware sampling mechanisms for incremental mining applications.

1.2 Related Works

The scalability problem in database applications has been fully explored with the help of data reduction techniques such as sampling, histogram [14], and wavelet decomposition [13]. The discussion here is limited to sampling techniques. For other data reduction techniques, which are out of scope for this paper, the reader is asked to follow the pointers in [16]. In practice, sampling techniques have been successfully used in various social and scientific applications. The general introduction of sampling can be found in many well-known works, such as [7], [17]. Here, we focus on discussing sampling techniques related to obtaining high-quality samples for data mining applications.

In essence, sampling has a rich history in statistics with many variants, including *simple random sampling* with/without replacement [7], adaptive sampling [18], and so on. Among them, *simple random sampling* is the most widely employed strategy due to its generality and its simplicity. Recent advances in streaming analysis and database query optimization specifically pay attention to utilizing the variants of *simple random sampling*, called sequential random sampling [19] and reservoir sampling [20] due to their high sampling efficiency. Explicitly, reservoir sampling maintains a random sample with a fixed size M as time advances and Method D in [19] will progressively generate the sample with the specified sample rate p. Those two sampling methods can skip some data elements without processing them while guaranteeing the generated sample is a uniform random sample. However, it is also reported that random samples suffer from insufficiency of the sampling quality, thus resulting in generating a model with low accuracy [8], [21], [22].

To obtain a model with high accuracy, new sampling approaches to generate high-quality samples are required. Algorithm *EASE* [8] is devised to guarantee high *absolute proportion consistency* of each categorical value. As we have discussed in Section 1.1, the goal of *EASE* is to limit the *absolute* difference between the sample proportion and the population proportion of each categorical value in the final sample below ε, where the magnitude of ε depends on the desired sample size and other parameters. Note that, for a measured pattern with the population proportion close to one, its *absolute proportion consistency* and *relative proportion consistency* are roughly the same. However, if the population proportion is small, they are different. For example, representing $d_i = 0.01$ by $s_i = 0.02$ and representing $d_i = 0.1$ by $s_i = 0.11$ have the same *absolute proportion consistency*. However, the former has a 100 percent error rate and the latter has a 10 percent error rate, which can be estimated as the *relative proportion consistency*. In this case, $s_i = 0.02$ deviates quite far from $d_i = 0.01$. Equally minimizing the *absolute* proportion difference of every attribute value may result in poor performance for applications which are sensitive to case, such as $d_i = 0.01$ in the previous example. Clearly, most data mining applications will almost prefer the *relative proportion consistency* since mining algorithms is usually devoted to the discovery of uncommon/surprising patterns (usually with a small occurrences) as opposed to the discovery of common sense knowledge (usually with a large occurrence) [23].

In addition to algorithm *EASE*, density biased sampling (abbreviated as *DBS*) is another sampling strategy which recently received a great deal of attention in the data mining research community [21]. Specifically, *DBS* is devised based on the observation that the distribution of clusters' sizes in real data is usually highly skewed. In such cases, random samples may miss points from small but dense regions, thus resulting in the loss of small clusters after sampling. *DBS* oversamples the regions with high spatial density and downsamples the regions with low spatial density. Note that the goal of *DBS* intrinsically differs from ours in this paper. First, we fairly reduce the difference between the population proportion and the sample proportion of each attribute value, whereas *DBS* emphasizes the density differences of each spatial region. Second, the targeted applications are quite different. *DBS* focuses on identifying small clusters rather than preserving the consistency between clustering results in the whole population and the sample. Using *DBS* is thus not appropriate for other clustering applications such as subspace clustering [24]. In contrast, we aim to make the resulting model obtained in the sample be consistent with that obtained in the population, which is applicable to various mining applications.

Progressive sampling/dynamic sampling [25], [22] is another way to improve the resulting model accuracy in the literature since the sample size estimated by *Chernoff bounds* is conservative and is usually too large for specified applications [12]. Progressive sampling algorithms are devised by iteratively executing the targeted application on random samples whose sizes are progressively increased and the process will be terminated when the mining

accuracy is no longer significantly improved. Finally, a satisfactory model accuracy can be obtained without a prohibitively large sample size. However, the targeted application may be executed on many samples with varying sample sizes, which is also time-consuming.

Many recent applications, including credit card fraud protection and network intrusion detection, call for the need of incremental mining algorithms. Since their data are usually time-variant and the data characteristics may drift as time advances, traditional algorithms which are devised for mining on static data will fail in such cases. Various incremental mining algorithms, e.g., incremental mining of frequent itemsets [4], [6], incremental conceptual clustering [3], and concept-drifting classification [26], are thus specifically devised. We omit the details of these algorithms and concentrate on the discussion of sampling strategies in the incremental mining scenario. Due to the time-variant nature, incremental mining algorithms are designed to analyze the most recent data in order to retrieve up-to-date patterns [15]. Two common approaches are usually utilized to deal with old data in such cases. The first one is *aging* [27], where each data is assigned a weight and more recent data have higher weights. The other approach is to use a *sliding window* [4], where only the most recent data covered by a window are considered. Formally, sampling approaches such as algorithm *EASE* will fail either in the *aging* or in the *sliding window* model since *EASE* assumes the population proportion in the whole population is static and can be known in advance. For incremental mining, only online and sequential sampling approaches can be utilized, such as Method D [19], reservoir sampling [20], and priority sampling [15], where, however, random samples are generated, rather than high-quality samples such as the one generated by *EASE*.

1.3 Our Contributions

Our contributions in this paper are many:

1. We propose algorithm *PAS* to sequentially generate a sample in which the *relative proportion inconsistency* of each categorical value can be minimized toward a user-specified bound ε while also guaranteeing the sample rate close to the user specific one. Importantly, although *PAS* targets on guaranteeing the *relative proportion consistency* of each categorical value, as shown in our analytical and algorithmic results, the *relative proportion consistency* of multivariate statistics can also be excellently preserved, which will be significantly beneficial to data mining applications.
2. For better execution efficiency, we further devise another sampling algorithm, *EQAS*, to provide the flexibility of striking a compromise between the sampling efficiency and the sampling quality.
3. We complement our analytical and algorithmic results by a thorough empirical study on real data and synthetic data and show that algorithm *PAS* can provide high-quality samples with slight computational overhead and algorithm *EQAS* can flexibly generate samples with the desired balance between sampling quality and sampling efficiency. We also explore their benefits for incremental mining applications. The result demonstrates their prominent advantages to be the effective quality-aware sampling means for incremental mining applications.

TABLE 1
Major Notations in This Paper

Notation	*Notation Description*
W_k	The k^{th} population sliding window
S_k	The k^{th} sample sliding window
$N^k(a_j)$	# of tuples in W_k which contain a_j
$s\_N^k(a_j)$	# of tuples in S_k which contain a_j
$\sup(a_j, W_k)$	The population proportion of a_j in W_k
$\sup(a_j, S_k)$	The sample proportion of a_j in S_k

This rest of the paper is organized as follows: Section 2 introduces algorithm *PAS*. In Section 3, we give the details of algorithm *EQAS*. The experimental results are shown in Section 4. Finally, this paper concludes with Section 5.

2 Online Sampling for Guaranteeing Relative Proportion Consistency

2.1 Fundamental Mathematical Model

In this section, we derive our model to generate online samples of guaranteed relative proportion consistency. For simplicity and effectiveness, we intend to follow the idea of random sampling without replacement. The variant lies in the strategy of determining the inclusion probability. As opposed to the fixed inclusion probability in random sampling without replacement, our model will dynamically determine the inclusion probability of each incoming tuple so that we can guarantee the relative proportion consistency on the fly while also ensuring that the size of generated sample is under the user's control. We then discuss the analytical details step by step. For ease of reference, Table 1 shows a summary of major symbols used in this paper.

2.1.1 Problem Description

Suppose that D is a relational table with schema $(A_1, A_2, \ldots, A_h)$, where $A_1, \ldots, A_h$ are attributes and h is the number of attributes in D. Let $t_i = (x_{i1}, x_{i2} \ldots, x_{ih})$ be the ith tuple in D, where $x_{ij} \in A_j$ for $1 \leq j \leq h$. Moreover, assuming that a_j denotes an attribute value in the domain of A_j, $1 \leq j \leq h$, a_j is said to be *contained* in t_i, i.e., $a_j \in t_i$, iff $a_j = x_{ij}$. Without loss of generality, we assume that 1) the order i is able to indicate the receiving order of t_i, 2) D contains infinite tuples, and 3) A_j contains the finite domain, for $1 \leq j \leq h$ (continuous attributes can be discretized using methods such as that described in [28]). Note that those assumptions will be equally applicable to infinite streams and finite data sets.

To formalize the window-based sampling model,[1] we assume that D is segmented into disjoint windows, $\{W_1, W_2, \ldots, W_n, \ldots\}$, in light of a predefined time granularity such as "day," "business-week," "month," "quarter," and "year" to name a few. As such, W_k will consist of a set of tuples, $\{t_{k1}, t_{k2}, \ldots, t_{ki}\}$, where each one is received

1. We adopt the sliding window-based model in our work. Note that the sliding window-based sample is also applicable to the *aging*-based incremental mining applications.

within the corresponding time period of W_k. Let $|W_k|$ and $N^k(a_j)$ be the number of tuples in W_k and the number of tuples *containing* the value a_j in W_k, respectively. Then, we have the population proportion of a_j in W_k, denoted by $\sup(a_j, W_k)$, where $\sup(a_j, W_k) = N^k(a_j)/|W_k|$. In addition, let S_k denote the sample window corresponding to W_k, where the set of tuples in S_k is a subset of tuples in W_k. Also, let $|S_k|$ and $s\_N^k(a_j)$ denote the number of tuples in S_k and the number of tuples *containing* the value a_j in S_k, respectively. We have the sample proportion of a_j in S_k, denoted by $\sup(a_j, S_k)$, where $\sup(a_j, S_k) = s\_N^k(a_j)/|S_k|$.

Our goal in this paper is to efficiently and sequentially generate a high-quality sample. Following the consideration in [8], the sampling quality considered in this study also refers to the consistency between the sample proportion $\sup(a_j, S_k)$ and the population proportion $\sup(a_j, W_k)$ of each attribute value a_j in a window W_k. However, the solution proposed in [8] merely attempts to reduce the *absolute* proportion difference as possible, i.e., minimizing $|\sup(a_j, W_k) - \sup(a_j, S_k)|$ for every value a_j. As pointed out earlier, minimizing the *relative* error is the more desirable measure for applications to discover uncommon/surprising patterns. Therefore, to further ensure the generated sample can better characterize the time-variant data source, we attempt to generate a sample in which *relative* proportion difference can be bounded below the specified error threshold ε.

In general, one way to achieve the bounded *relative* proportion difference is to increase the sample size. However, a large sample size is usually prohibitive. The sample rate/sample size should be under the user's control to prevent generating a large sample. We then formally present our goal as follows:

Proposition 1. *Given a desired sample rate p and the relative error bound ε, we attempt to generate a sample $\{S_1, \ldots, S_n\}$ from the population $\{W_1, \ldots, W_n\}$, in which 1) $\frac{|S_k|}{|W_k|} \approx p$, where $1 \le k \le n$, and 2)*

$$|\sup(a_j, W_k) - \sup(a_j, S_k)| \le \varepsilon \times \sup(a_j, W_k),$$

for every attribute value a_j.

2.1.2 Online Sampling Model with Equivalent Problem Transformation

Formally, it is difficult to devise an approach to simultaneously consider those two heterogeneous criteria in Proposition 1 since various variables need to be taken into consideration at the same time. To solve the problem, we derive Theorem 1 below:

Theorem 1. *Suppose that, in a sample $S = \{S_1, \ldots, S_n\}$, we have $\left|\frac{s\_N^k(a_j)}{N^k(a_j)} - p\right| \le \frac{\varepsilon}{2+\varepsilon} p$, for every value a_j in each window. Then, the sample also satisfies: 1) $(1-\varepsilon)p \le \frac{|S_k|}{|W_k|} \le (1+\varepsilon)p$, where $1 \le k \le n$, and 2)*

$$|\sup(a_j, W_k) - \sup(a_j, S_k)| \le \varepsilon \times \sup(a_j, W_k),$$

for every value a_j.

In the interests of space, proofs are given in the Appendix for interested readers.

Theorem 1 points out that a sample in which $\left|\frac{s\_N^k(a_j)}{N^k(a_j)} - p\right| \le \frac{\varepsilon}{2+\varepsilon} p$ for every value a_j will also satisfy the goal in Proposition 1. As such, our goal shown in Proposition 1 can be equivalently transformed to generate a sample in which $\left|\frac{s\_N^k(a_j)}{N^k(a_j)} - p\right| \le \frac{\varepsilon}{2+\varepsilon} p$ for every value a_j.

However, the most important problem in the considered model is that, in the scenario of incremental mining, the frequency of the attribute value a_j in a window W_k, i.e., $N^k(a_j)$, will be dynamic and up-to-date. Moreover, each tuple should be processed on the fly, meaning that, once a tuple is selected or discarded, we cannot revoke this decision of this tuple. To meet such a constraint of sampling algorithms for incremental mining, we reasonably assume that the latest arriving tuple is the last tuple in the window and, thus, we shall determine the inclusion probability of this tuple so as to achieve our goal. In light of Proposition 1 and Theorem 1, we then formally present Proposition 2 as our new goal to generate online high-quality samples.

Proposition 2. *At the arrival of the tuple t_i, where $t_i \in W_k$, we aim to determine the inclusion probability of t_i in such a way that we can have $\left|\frac{s\_N_i^k(a_j)}{N_i^k(a_j)} - p\right| \le \frac{\varepsilon}{2+\varepsilon} p$ for every categorical value a_j appearing in the window, where $s\_N_i^k(a_j)$ and $N_i^k(a_j)$ denote the frequency of a_j in the sample window S_k and the population window W_k after the selection/discard of t_i, respectively.*

For simplicity, let $s\_N_i^k(a_j)/N_i^k(a_j)$ be denoted by $F_k(a_j, i)$. Importantly, $|F_k(a_j, i) - p|$ is equal to $|F_k(a_j, i-1) - p|$ for every attribute value a_j if $a_j \notin t_i$, implying that selecting t_i or not will not affect the proportion consistency of a_j when $a_j \notin t_i$ (for simplicity, hereafter we assume t_i and t_{i-1} belong to the same window W_k). Therefore, when the tuples sequentially arrive, we only need to ensure $|F_k(a_j, i) - p| \le \frac{\varepsilon}{2+\varepsilon} p$ for the value a_j belonging to the arriving tuple t_i. This observation implies that the inclusion probability of t_i can be determined by considering at most h homogeneous variables.

Nevertheless, due to the inherent limit of sampling, it is difficult to ensure $|F_k(a_j, i) - p| \le \frac{\varepsilon}{2+\varepsilon} p$ for every attribute value a_j in the presence of a small ε and a small p. The problem will be apparent when a window is just initialized or $N_i^k(a_j)$ is very small. Note that the small ε and the small p are indeed two conflict goals (in algorithm *EASE*, a small sample size will inevitably incur a large ε). Importantly, we can still pursue the goal in Proposition 2 by minimizing the difference between $F_k(a_j, i)$ and p until the difference is smaller than $\frac{\varepsilon}{2+\varepsilon} p$. The feasibility of such a concept is shown in Theorem 2 below:

Theorem 2. *Suppose that $\frac{s\_N^k(a_j)}{N^k(a_j)} = (1+\xi_j) \times p$ for every value a_j in the window W_k, which indicates that $\left|\frac{s\_N^k(a_j)}{N^k(a_j)} - p\right| = |\xi_j| \times p$. Let*

$$\Gamma = \frac{\sum_{j=1}^{|A|} \xi_j \times N^k(a_j)}{\sum_{j=1}^{|A|} N^k(a_j)},$$

where $|A|$ is the number of distinct attribute values in W_k. We will have the sample rate $\frac{|S_k|}{|W_k|}$ equal to $p \times (1+\Gamma)$. Furthermore, the relative proportion difference of the attribute value a_j will be equal to

$$\frac{\left|\sup(a_j, W_k) - \sup(a_j, S_k)\right|}{\sup(a_j, W_k)} = \left|1 - \frac{(1+\xi_j)}{1+\Gamma}\right|.$$

Formally, Theorem 2 provides the basis that, if we can minimize $|\xi_j|$ or ξ_j^2, i.e., minimize the difference between $\frac{s\_N^k(a_j)}{N^k(a_j)}$ and p, the resulting sample rate $\frac{|S_k|}{|W_k|}$ will be close to p and the *relative* proportion difference of each attribute value a_j will be close to zero. Based on the foregoing, we therefore aim to determine the inclusion probability of t_i when t_i arrives according to the criterion to minimize $[F_k(a_j, i) - p]^2$, where $a_j \in t_i$.

2.1.3 Inclusion Probability with Minimized Relative Proportion Difference

Before presenting the details of the approach to determine the inclusion probability by minimizing $[F_k(a_j, i) - p]^2$ for $a_j \in t_i$, we first introduce the *proportion-preserved values*, defined as follows:

Definition 1 (Proportion-Preserved Values). *A value $a_j \in t_i$ is called a proportion-preserved value of t_i if we have $|\sup_i(a_j, W_k) - \sup_i(a_j, S_k)| \le \varepsilon \times \sup_i(a_j, W_k)$, whether t_i will be sampled or not, where $\sup_i(a_j, W_k)$ and $\sup_i(a_j, S_k)$ denote the population proportion and the sample proportion of a_j after the selection/discard of t_i, respectively.*

Note that our essential goal is to have the *relative* proportion difference of every attribute value a_j being bounded below ε and *proportion-preserved values* indeed satisfy this requirement. Therefore, we will only need to concentrate on minimizing the proportion differences of the remaining attribute values of t_i (for ease of presentation, we defer the advantage of excluding *proportion-preserved values* to Section 3.4). In view of this, *proportion-preserved values* of t_i will be excluded when we determine the inclusion probability of t_i. Note that, before the determination of the inclusion probability of t_i, we can determine whether a value is a *proportion-preserved value of t_i* or not in light of Lemma 1 below.

Lemma 1. *For the value $a_j \in t_i$, a_j is a proportion-preserved value of t_i if two criteria are satisfied:*

1. $\left|\frac{s\_N_{i-1}^k(a_j)+1}{|S_{k,i-1}|+1} - \frac{N_{i-1}^k(a_j)+1}{|W_{k,i-1}|+1}\right| \le \varepsilon \times \frac{N_{i-1}^k(a_j)+1}{|W_{k,i-1}|+1}$ *and*
2. $\left|\frac{s\_N_{i-1}^k(a_j)}{|S_{k,i-1}|} - \frac{N_{i-1}^k(a_j)+1}{|W_{k,i-1}|+1}\right| \le \varepsilon \times \frac{N_{i-1}^k(a_j)+1}{|W_{k,i-1}|+1}$,

where $|S_{k,i-1}|$ and $|W_{k,i-1}|$ denote the number of tuples in the sample window S_k and the population window W_k after the selection/discard of t_{i-1}, respectively.

Suppose that V_i denotes the set of attribute values of t_i, excluding *proportion-preserved values* of t_i. We then discuss the way to determine the inclusion probability of t_i to minimize $[F_k(a_j, i) - p]^2$, where $a_j \in V_i$. Note that, for the value $a_j \in V_i$, $N_i^k(a_j)$ will be equal to $N_{i-1}^k(a_j) + 1$. However, due to randomness, $s\_N_i^k(a_j)$ will be uncertain before t_i has been sampled or discarded. We will have $s\_N_i^k(a_j) = s\_N_{i-1}^k(a_j) + 1$ if t_i is sampled or have $s\_N_i^k(a_j) = s\_N_{i-1}^k(a_j)$ if t_i is discarded. Since $s\_N_i^k(a_j)$ cannot be exactly identified before selecting/discarding t_i, $E[F_k(a_j, i)]$ will be the best estimator of $F_k(a_j, i)$ in such situations, where $E[F_k(a_j, i)]$ is the expectation of $F_k(a_j, i)$. Specifically, to select or to discard t_i is a *Bernoulli trial* [7] for every value $a_j \in t_i$, indicating that $E[F_k(a_j, i)]$ is equal to $\frac{s\_N_{i-1}^k(a_j)+p_r(t_i)}{N_{i-1}^k(a_j)+1}$, where $p_r(t_i)$ is the inclusion probability of t_i. Let $|V_i|$ denote the number of attribute values in V_i. Following the conclusion of Theorem 2, we thus aim to minimize $\sum_{j=1}^{|V_i|}(E[F_k(a_j, i))] - p)^2$. Suppose that $\widehat{p_r}(t_i)$ denotes the inclusion probability of t_i corresponding to the minimization of $\sum_{j=1}^{|V_i|}(E[F_k(a_j, i))] - p)^2$. We can derive the closed form of $\widehat{p_r}(t_i)$, as shown in Theorem 3 below:

Theorem 3. *Note that $\widehat{p_r}(t_i)$ can be formalized as*

$$\widehat{p_r}(t_i) = \underset{0 \le p_r(t_i) \le 1}{\arg\min} \left[\sum_{j=1}^{|Vi|}\left(\frac{s\_N_{i-1}^k(a_j) + p_r(t_i)}{N_{i-1}^k(a_j)+1} - p\right)^2\right].$$

Suppose that $a = \sum_{j=1}^{|Vi|}\left(\frac{1}{N_{i-1}^k(a_j)+1}\right)^2$ and

$$b = \sum_{j=1}^{|Vi|}\left[\left(\frac{1}{N_{i-1}^k(a_j)+1}\right)\left(\frac{s\_N_{i-1}^k(a_j)}{N_{i-1}^k(a_j)+1} - p\right)\right].$$

The closed form of $\widehat{p_r}(t_i)$ will be

$$\widehat{p_r}(t_i) = \begin{cases} -\frac{b}{a}, & \text{if } 1 \ge -\frac{b}{a} \ge 0 \\ 1, & \text{if } -\frac{b}{a} > 1 \\ 0, & \text{if } -\frac{b}{a} < 0. \end{cases}$$

Consequently, we can devise algorithm *PAS* as a sequential sampling mechanism which determines the inclusion probability of t_i as $\widehat{p_r}(t_i)$ when t_i arrives. As such, the goal in Proposition 2 will be achieved, meaning that our essential goal in Proposition 1 is also achieved.

Furthermore, the previous discussion only shows how to guarantee the precision of the marginal distribution, i.e., the *relative proportion consistency* of each categorical value, rather than how to guarantee the precision of the joint distribution. Importantly, Theorem 4 below shows that our model will also minimize the *relative proportion inconsistency* of multivariate statistics, thus ensuring the preservation of the joint distribution.

Theorem 4. *Let M_i denote a multivariate statistic in the database. PAS will minimize the relative inconsistency between its sample proportion and the population proportion.*

Since mining applications usually concentrate on finding interesting multidimensional knowledge, it is clear that *PAS* can generate more desirable sample for different application needs.

2.2 Examples of PAS

We show some examples to illustrate operations of *PAS*. Suppose that the sample rate p and the error bound ε are specified as 33 percent and 0.3, respectively. Our goal is to sequentially generate a sample in which the relative proportion difference of each attribute value can be bounded below 0.3.

Example 2.1. For the first tuple, $t_1 = (A, X, F)$, we have $N_0^1(A) = 0$, $s\_N_0^1(A) = 0$, $N_0^1(X) = 0$, $s\_N_0^1(X) = 0$, $N_0^1(F) = 0$, and $s\_N_0^1(F) = 0$ since the window is initialized. According to Lemma 1, the attribute values A, X, and F are not *proportion-preserved values* of t_1. Therefore, the inclusion probability $\widehat{p_r}(t_1)$ of t_1 is determined by considering all three values:

$$\widehat{p_r}(t_1) = -\frac{\frac{1}{1}(\frac{0}{1} - 0.33) + \frac{1}{1}(\frac{0}{1} - 0.33) + \frac{1}{1}(\frac{1}{1} - 0.33)}{(\frac{1}{1})^2 + (\frac{1}{1})^2 + (\frac{1}{1})^2} = 0.33.$$

As the same as general cases in random sampling, the inclusion probability of the first tuple in *PAS* is equal to p to ensure that $E[\sup(A, S_1)]$, $E[\sup(X, S_1)]$, and $E[\sup(F, S_1)]$ are equal to $\sup(A, W_1)$, $\sup(X, W_1)$, and $\sup(F, W_1)$, respectively. One important property of *PAS* is thus identified: The sample proportion of each attribute value is the unbiased estimator of the corresponding population proportion, which is an essential requirement for probabilistic sampling methods.

Example 2.2. Suppose that two tuples were selected in the sample before the tuple $t_9 = (B, Y, H) \in W_1$ arrives. In addition, we have $N_8^1(B) = 2$, $s\_N_8^1(B) = 0$, $N_8^1(Y) = 2$, $s\_N_8^1(Y) = 0$, $N_8^1(H) = 4$, and $s\_N_8^1(H) = 1$. Note that, for attribute value H,

$$\left|\frac{s\_N_8^1(H) + 1}{2 + 1} - \frac{N_8^1(H) + 1}{8 + 1}\right| \leq 0.3 \times \frac{N_8^1(H) + 1}{8 + 1}$$

and $\left|\frac{s\_N_8^1(H)}{2} - \frac{N_8^1(H)+1}{8+1}\right| \leq 0.3 \times \frac{N_8^1(H)+1}{8+1}$, meaning that H is a *proportion-preserved value* of t_9. Therefore, we will only consider B and Y when we calculate the inclusion probability of t_9, which yields that

$$\widehat{p_r}(t_9) = -\frac{\frac{1}{3}(\frac{0}{3} - 0.33) + \frac{1}{3}(\frac{0}{3} - 0.33)}{(\frac{1}{3})^2 + (\frac{1}{3})^2} = 0.99.$$

In this case, *PAS* prefers to select t_9 with a high probability 99 percent and we can expect $|\sup(B, W_1) - \sup(B, S_1)| \leq \varepsilon \times \sup(B, W_1)$ since $\sup(B, S_1)$ will be equal to $\frac{1}{3}$ with the high probability. Note that, whether t_9 will be selected or not, we always have the $|\sup(H, W_1) - \sup(H, S_1)| \leq \varepsilon \times \sup(H, W_1)$ since H is a *proportion-preserved values* of t_9.

On the other hand, while H is considered in the determination of the inclusion probability of t_9, we will have the inclusion probability be equal to

$$\widehat{p_r}^*(t_9) = -\frac{\frac{1}{3}(\frac{0}{3} - 0.33) + \frac{1}{3}(\frac{0}{3} - 0.33) + \frac{1}{5}(\frac{1}{5} - 0.33)}{(\frac{1}{3})^2 + (\frac{1}{3})^2 + (\frac{1}{5})^2} = 0.93.$$

Comparing $\widehat{p_r}(t_9)$ with $\widehat{p_r}^*(t_9)$, it can be seen that considering *proportion-preserved values* when we determine the inclusion probability of the incoming tuple will lead to the lower probability to have the bounded *relative* proportion differences of other attribute values. In this case, we demonstrate the feasibility to exclude *proportion-preserved values* when we determine the inclusion probability of the incoming tuple.

Example 2.3. Suppose that, before tuple $t_{10000} = (A, Z, F) \in W_{10}$ arrives, we have

$$N_{9999}^{10}(A) = 99, s\_N_{9999}^{10}(A) = 30, N_{9999}^{10}(Z) = 9$$
$$s\_N_{9999}^{10}(Z) = 2, N_{9999}^{10}(F) = 2, s\_N_{9999}^{10}(F) = 0,$$
$$|S_{10,9999}| = 31, \text{ and } |W_{10,9999}| = 100.$$

It can be seen that the data are highly skewed in this window since attribute value A frequently occurs but attribute value F rarely occurs. In this case, attribute value A will be a *proportion-preserved value* of t_{10000} because A satisfies the two criteria stated in Lemma 1. Note that $-\frac{\frac{1}{10}(\frac{2}{10} - 0.33) + \frac{1}{3}(\frac{0}{3} - 0.33)}{(\frac{1}{10})^2 + (\frac{1}{3})^2} = 1.01$. In such cases, *PAS* will determine if $\widehat{p_r}(t_{10000})$ is equal to one so that t_{10000} will be definitely selected in the sample. As a result, the *relative* proportion differences of F and Z are $\left|\frac{3}{101} - \frac{1}{32}\right| / \frac{3}{101} = 0.05 < \varepsilon$ and $\left|\frac{10}{101} - \frac{3}{32}\right| / \frac{10}{101} = 0.05 < \varepsilon$, respectively. In this example, we show the feasibility of *PAS* in skewed data.

3 Efficient Quality-Aware Sampling

In Section 2, we have introduced algorithm *PAS* to generate a high-quality online sample by adaptively determining the inclusion probability of each incoming tuple t_i. We, in this section, present algorithm *EQAS* (standing for Efficient Quality-Aware Sampling) to provide the flexibility of striking a compromise between the sampling quality and the sampling efficiency.

3.1 Framework of EQAS

Before presenting the details of algorithm *EQAS*, we first show the implementation of algorithm *PAS*. Specifically, in *PAS*, the frequency of every attribute value in a window needs to be maintained in main memory. To efficiently achieve this, *PAS* is devised by employing a *hash* structure, called CF (standing for *cumulative filter*). Let $CF(a_j)$ denote the hash function to hash the attribute value a_j, where the hash entry contains the up-to-date frequencies of a_j in the sample window and the population window. While a new window starts, all entries in CF will be released and initialized again, indicating that the memory usage is irrelevant to the size of input data and will be bounded with respect to the count of distinct attribute values in the database. The function of *PAS* to determine whether the

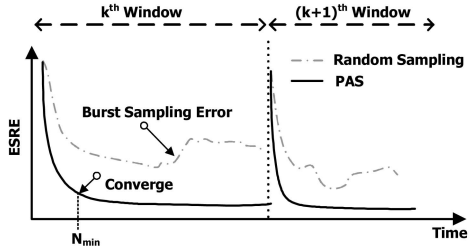


Fig. 3. ESRE in random sampling and PAS.

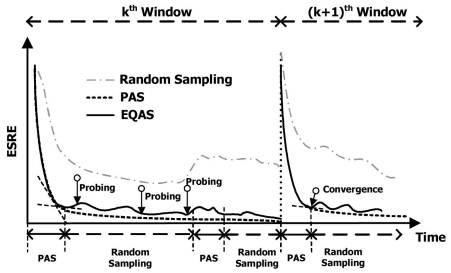


Fig. 4. The illustration of the expected square relative error among three sampling approaches.

tuple t_i is selected in the sample or not is outlined in the procedure *IsSample_PAS*.

Procedure: IsSample_PAS(t_i):
1. for every attribute value $a_j \in t_i$ do begin
2. if $(CF(a_j) = null)$
3. insert $N(a_j) = 0$,$s\_N(a_j) = 0$ into CF with key a_j;
4. $N(a_j) + +$;
5. if a_j is not a proportion-preserved value
6. $V_i = V_i \cup a_j$; //note that V_i is empty initially
7. end
8. calculate $\widehat{p_r}(t_i)$;
9. if $(\widehat{p_r}(t_i) \geq R())$ //$R()$ returns a random number in [0,1]
10. for every attribute value $a_j \in t_i$
11. $s\_N(a_j) + +$;
12. return true; //select t_i
13. else
14. return false; //discard t_i

Indeed, as compared to traditional sequential sampling algorithms such as random sampling, *PAS* will incur the higher computational overhead since the sample proportion and the population proportion of at most h attribute values will be examined when a tuple arrives. As a result, we further devise algorithm *EQAS* to simultaneously achieve high sampling quality and the high sampling efficiency by integrating *PAS* and random sampling. The basic concept behind algorithm *EQAS* is to switch between *PAS* and random sample at the appropriate moment. While deferring the details, we first formally present an important measure of the sampling quality, called the *expected square relative error*.

Definition 2 (Expected Square Relative Error). *The expected square relative error (abbreviated as ESRE) before t_{i+1} arrives is defined as* $E_r(i) = \sum_{j=1}^{|A|} \frac{N_i^k(a_j)}{h \times |W_{k,i}|} \times \left(\frac{\sup_i(a_j,W_k) - \sup_i(a_j,S_k)}{\sup_i(a_j,W_k)}\right)^2$, *where* $\sup_i(a_j, W_k)$ *and* $\sup_i(a_j, S_k)$ *denote the population proportion and the sample proportion of the attribute value a_j in the* kth *window before t_{i+1} arrives, respectively.*

Specifically, *ESRE* is a fair measure for the sampling quality of the sequentially generated sample because *ESRE* represents the expected *relative proportion inconsistency* of an attribute value over time. It is worth mentioning that, as will be validated in our empirical studies, *PAS* can quickly reduce *ESRE* since pursuing the minimization of the *relative* proportion error is the inherent goal of *PAS*. However, as illustrated in Fig. 3, *ESRE* will no longer be significantly reduced by *PAS* while the passing data size in the window exceeds a size, denoted by N_{min} in Fig. 3. Formally, due to the natural limit of sampling, the relative proportion errors of some values may still exceed the desired relative error bound ε after $|W_{k,i}| > N_{min}$. For example, assuming an attribute value occurs only one time during a window (usually deemed as noise), we cannot ensure that its *relative* proportion difference will be bounded below $\varepsilon = 0.1$ when $p = 0.1$. Actually, such a problem is well reported in the literature and a reasonable *sanity bound* ψ is usually used to avoid that the relative error metric is unduly dominated by attribute values with very small occurrences, where the *relative* error is defined as the form of $\frac{\sup_i(a_j,W_k) - \sup_i(a_j,S_k)}{\max\{\psi, \sup_i(a_j,W_k)\}}$ [13]. In our cases, further attempting to reduce their errors by *PAS* will pay for the computational overhead without the prominent improvement of the sampling quality. As such, we can initially execute *PAS* (to pursue the high sampling quality) and switch to random sampling when the window size exceeds N_{min} (to pursue the high sampling efficiency).

Note that random sampling cannot guarantee the sampling quality in the presence of the burst sampling error when the data distribution changes suddenly. Therefore, we periodically perform an offline probing process to examine the *expected square relative error* when random sampling is executed. If the result shows that *ESRE* drastically increases, the sampling approach will be switched back to *PAS* to ensure the sampling quality. Accordingly, the comparison of *ESRE* among random sampling, *PAS* and *EQAS*, is illustrated in Fig. 4, where *ESRE* in algorithm *EQAS* is expected to be close to *ESRE* in algorithm *PAS*. Correspondingly, the sampling time consumed by algorithm *EQAS* will be close to the time consumed by random sampling. As a result, algorithm *EQAS* can achieve the high sampling quality and the high sampling efficiency at the same time.

The overall implementation of *EQAS* is thus outlined below with three algorithm inputs: the data source D, the sample rate p, and the relative error bound ε. In addition, a global variable, called *Status*, indicates the up-to-date variation of the *expected square relative error*. While *Status* is identified as *unstable*, meaning that the variation of *ESRE* is obvious (either drastically increases or drastically decreases), algorithm *EQAS* will execute *PAS* to sample the following tuples for pursuing the high sampling quality. Alternatively, while *Status* is identified as *stable*, meaning that the variation of *ESRE* is insignificant (either slightly increases or slightly decreases), algorithm *EQAS* will execute random sampling to sample the following tuples for pursuing high execution efficiency.

Algorithm: EQAS(D,p,ε):
Global Variables:
Status=unstable; //indicate the up-to-date status of the sample
CF=a hash structure to maintain up-to-date frequencies
1. while has next tuple t_i {
2. if ($t_i \notin W_n$) //t_i belongs to a new window W_{n+1}
3. Status=unstable;
4. CF=$\emptyset$;
5. if (Status=stable) //execute random sampling
6. if (IsSample_RS(t_i)=true) //call random sampling to sample t_i
7. select t_i and output it;
8. else
9. discard t_i;
10. else //execute PAS
11. if (IsSample_PAS(t_i)=true)
12. select t_i and output it;
13. else
14. discard t_i;
15. }

The remaining issue of algorithm *EQAS* is to identify the timing to switch from *PAS* to random sampling and vice versa. We first discuss the timing to switch from *PAS* to random sampling, which can be considered as the process of identifying the convergence of the *ESRE* curve over time.

3.1.1 Convergence Detection

We refer to the technique of the convergence detection utilized in progressive sampling [22]. Formally, the power-law fit [25] and the *linear regression with local sampling* [22] are two common approaches in progressive sampling to detect the convergence point of the learning curve. We follow the idea of the *linear regression with local sampling* since, as demonstrated in [22], it is robust and is the state-of-the-art approach.

Specifically, we periodically examine *ESRE* for a short time duration when *PAS* is executed. Suppose we obtain $E_r(i), E_r(i+1), \ldots, E_r(i+m)$, where m is the length of a duration. Those values are then used to estimate a linear regression line whose slope is compared to zero. If the slope is smaller than a threshold σ, which is a value sufficiently close to zero, the tuple t_i can be deemed as the convergence point. In such cases, the variable *Status* will be modified as *stable* and the sampling strategy will be turned to random sampling.

We then present how to determine the time of switching from random sampling to *PAS*. The procedure is called the probing procedure.

3.1.2 Probing

Note that originally, frequencies of all attribute values in the sample and in the population will not be maintained during the execution of random sampling, thus causing the difficulty of having to examine the up-to-date *ESRE* over time. In practice, we can approximately estimate *ESRE* by only maintaining frequencies of several attribute values. More specifically, at the end of executing *PAS*, we randomly select $|F|$ attribute values from CF (others will be released) and continue to monitor the frequency of those $|F|$ attribute values during the execution of random sampling. Therefore, we can periodically execute the *probing* process to calculate the *estimated ESRE*, which is formularized as $\sum_{j=1}^{|F|} \frac{N_i^k(a_j)}{h \times |W_{k,i}|} \times \left(\frac{\sup_i(a_j,W_k) - \sup_i(a_j,S_k)}{\sup_i(a_j,W_k)}\right)^2$. If the *estimated ESRE* is larger than ρ times the *estimated ESRE* obtained in the end of the former execution of *PAS*, the variable *Status* will be modified as *unstable* and the sampling strategy will be turned to *PAS*. Note that investigating the *estimated ESRE* will increase the complexity by a constant factor since $|F|$ attribute values need to be continuously maintained. However, the overhead is slight as compared to time consumed by *PAS*.

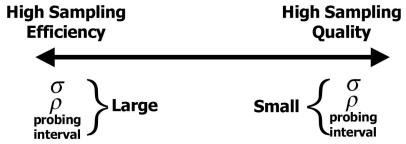


Fig. 5. The trade-off between the sampling quality and the sampling efficiency.

The implementation of the convergence detection and the probing method is outlined in the procedure Status_Detection() below, in which the function linear_reg refers to a function executing the linear regression and returning the slope of the regression line. In addition, the variable pre_e denotes the value of *estimated ESRE* obtained in the end of the former execution of *PAS*.

Procedure: Status_Detection():
1. if (Status=unstable) { //perform the convergence detection
2. for t_c from t_i to t_{i+m} do begin
3. for ever attribute value a_j in CF
4. sum+=$N_c^k(a_j) \times \left[\frac{\sup_c(a_j,W_k) - \sup_c(a_j,S_k)}{\max\{\psi, \sup_c(a_j,W_k)\}}\right]^2$;
5. E[c]=sum/$\left(h \times |W_{k,c}|\right)$;
6. end
7. θ =linear_reg(E[i],...,E[i+m]);
8. if ($\theta < \sigma$)
9. Status=stable; //stage to switch to random sampling
10. random select $|F|$ values and release others from CF;
11. pre_e=$\sum_{j=1}^{|F|} \frac{N_i^k(a_j)}{h \times |W_{k,c}|} \times \left(\frac{\sup_c(a_j,W_k) - \sup_c(a_j,S_k)}{\sup_c(a_j,W_k)}\right)^2$;
12. } else {//perform the probing procedure
13. for i from 1 to $|F|$ do begin
14. sum+=$N_i^k(a_j) \times N_i^k(a_j) \times \left[\frac{\sup_i(a_j,W_k) - \sup_i(a_j,S_k)}{\sup_i(a_j,W_k)}\right]^2$;
15. e=sum/$(h \times |W_{k,i}|)$;
16. if (e>$\rho \times$pre_e)
17. Status=unstable; //stage to switch to PAS
18. initialize CF;
19. }

3.2 Parameters in Algorithm EQAS

Note that, as discussed in related works, it can be seen that sampling quality is usually obtained at the cost of the extra computational overhead, which somewhat compromises the applicability of those sampling algorithms. Indeed, algorithm *EQAS* enables the flexibility between the sampling quality and the sampling efficiency. Without loss of generality, the trade-off between the sampling quality and the sampling efficiency solely depends on the fraction of the execution of algorithm *PAS*. While the sampling quality is the primary concern, the fraction of the execution of algorithm *PAS* can be raised. On the other hand, the fraction of the execution of algorithm *PAS* will be reduced when we pursue the high sampling efficiency. As shown in Fig. 5, three parameters, i.e., σ, ρ, and the probing interval,

in algorithm *EQAS*, will control the execution fraction of *PAS*, indicating the position either close to the high sampling quality or close to the high sampling efficiency. Specifically, setting small σ and ρ implies that we need to strictly check whether random sampling can handle the following tuples or not. In such cases, algorithm *PAS* usually carries over to handle the following tuples. Moreover, the small probing interval will lead to frequently performing the probing process, which will raise the probability of switching from random sampling to *PAS*. In contrast, setting large σ, ρ, and probing interval will tend to frequently switch from *PAS* to random sampling for the following tuples. From our empirical studies, we suggest that $\sigma = 0.01$, $\rho = 1.1$, and the probing interval is equal to 100 (the probing procedure is periodically executed every 100 tuples), which usually leads to the better balance between the sampling efficiency and the sampling quality.

Readers may be able to achieve the analogous flexibility by giving the execution fraction of *PAS* and then periodically switching between random sampling and *PAS* without the need of the probing procedure and the convergence detection. However, such a straightforward solution suffers from the problem that tuples, during the burst sampling error, cannot be precisely handled by *PAS*. It will drastically affect the sampling quality. Note that it may also lead to the meaningless situation of using *PAS* to handle data which are stable and uniformly distributed. Algorithm *EQAS* with the proposed convergence detection and the probing procedure will obviously outperform the naive approach.

The remaining issue in algorithm *EQAS* is to determine the appropriate value of $|F|$, i.e., the number of attribute values whose frequencies will be maintained during the execution of random sampling. Note that maintaining a large $|F|$ will pay for the overhead similar to *PAS*, thus losing efficiency gained from random sampling. Formally, it is meaningful to have $|F|$ larger than 30 in the sense of statistics [7]. We therefore set $|F| = 30$ in default.

3.3 Complexity Analysis

Formally, random sampling is with the linear time complexity and its space complexity is a constant, which implies that the overhead required by *EQAS* is dominated by the complexity of algorithm *PAS*. As such, we show the complexity of *EQAS* by analyzing the one of *PAS* at first.

3.3.1 Time Complexity

The time complexity of *PAS* is $O(h \times |D|)$, where $|D|$ is the data set and h is the number of dimensions in the population. Since either *PAS* or random sampling is linear with respect to the database size, algorithm *EQAS* is also linear with a factor determined by how many tuples are passed by *PAS*. In addition, the execution of the function Status_Detection in algorithm *EQAS* requires constant time to look up all attribute values in *CF*, thus only increasing the complexity of algorithm *EQAS* by a negligible constant factor.

3.3.2 Space Complexity

The space complexity of *PAS* is $O(|A|)$, where $|A|$ is the number of distinct values in a window. While considering the distribution of real data generally follows the Zipf distribution [29], an $O(N^{1/z})$ upper bound of the memory usage is derived, where N is the number of tuples in a window and z is the level of skewness in the distribution of attribute values. Specifically, assuming the distribution follows the Zipf distribution with parameter z, the frequency of the ith rank attribute value is equal to $\frac{N}{\zeta(z)\times i^z}$, where $\zeta(z) = \sum_{i=1}^{|A|} \frac{1}{i^z}$ [29]. Note that $\sum_{i=1}^{\infty} \frac{1}{i^z}$ converges to a small constant[2] when $z > 1$ (this is a common case in real data), implying that $\zeta(z)$ is also a small constant. Since the absolute frequency of an attribute value always exceeds one, we have $\frac{N}{|A|^z} \geq 1$. Therefore, the bound of $|A|$ is $O(N^{1/z})$, indicating that the space complexity of *PAS* is $O(N^{1/z})$. Moreover, since the memory consumed by random sampling is a constant, the space complexities of algorithms *EQAS* and *PAS* are the same.

3.4 Discussions on Quality-Aware Sampling

In this section, we provide more insights into the proposed algorithms. We first describe why *PAS* and *EQAS* utilize probabilistic sampling mechanisms rather than a deterministic sampling mechanism like the one used in algorithm *EASE* [8]. Specifically, readers may argue why we do not simply select or drop the tuple t_i based on which action results in a smaller value of $\sum_{j=1}^{|V_i|}(F_k(a_j, i)) - p)^2$, thus resulting in a deterministic process of selecting tuples. The major reason is that probabilistic models can provide randomness and unbiasedness. Note that, in *EASE*, the data distribution is observed in a pilot sample and it can utilize a deterministic procedure to select the sample based on what it has observed beforehand. In contrast, *PAS* and *EQAS* do not check the population in advance so as to fulfill the need for incremental mining. As can be simply seen, the first tuple is always dropped if $p < 0.5$ when a deterministic procedure is applied in *PAS*, leading to a biased sample (as compared with the discussion in Example 2.1). Since the unbiasedness is an important property for sampling [17], the probabilistic model is more appropriate for our model.

In addition, comparing our quality-aware sampling mechanisms and *EASE* [8], it is clear that the error threshold ε in *PAS* and *EQAS* can be specified by users and, in contrast, the absolute error in *EASE* is guaranteed below a system-determined bound whose magnitude depends on the desired sample size and can be estimated after the first pilot sample is generated. In practice, the small error bound and the small sample size/rate are two conflicting goals due to the inherent limits of sampling (the *relative* error upper bound will be dominated by attribute values with very small occurrences [13]). Coupled with the mechanism of excluding *proportion-preserved values* described in Section 2.1.3, the tunable error threshold will enable the balance between the small error bound and the small sample size. The details will be observed and discussed in our experimental results.

4 Experimental Results

The simulation model of our experimental studies is described in Section 4.1. To assess the performance of *PAS* and *EQAS*, we present empirical studies based on both

2. Refer to http://en.wikipedia.org/wiki/Riemann_zeta_function for details.

TABLE 2
Notations in Data Sets

Notations	Description
z	The level of skew
N	# of attributes
T	Avg. # of distinct values per attribute
D	# of tuples

TABLE 3
Parameters of Data Sets

Name	z	N	T	D
z1.5N30T50	1.5	30	50	500k
z0.5N5T50	0.5	5	50	500k
AlarmLog	-	15	23	160k
Mushroom	-	23	5	8124

synthetic and real data sets. The feasibility and the scalability are examined in Section 4.2. In Section 4.3, the effectiveness of sampling for the various mining applications is demonstrated.

4.1 Simulation Model

In our experiments, synthetic data sets are generated for the sensitivity analysis. Those synthetic data sets are generated as follows: First, a Zipfian data generator was used to produce Zipfian frequencies for various levels of skew. By tuning a parameter z, i.e., the level of skewness in the distribution of attribute values, we generate data sets to simulate highly skewed $(z = 1.5)$ and weakly skewed $(z = 0.5)$ data, where 1.5 and 0.5 are commonly used parameters to investigate the algorithm performance in different levels of skewness [30]. Moreover, the synthetic data generation program takes other parameters, as shown in Table 2, and the values of parameters used to generate the data sets are summarized in Table 3. Data sets of high dimensions with the skewed distribution (named z1.5N30T50) and low dimensions with the approximately uniform distribution (named z0.5N5T50) are both considered.

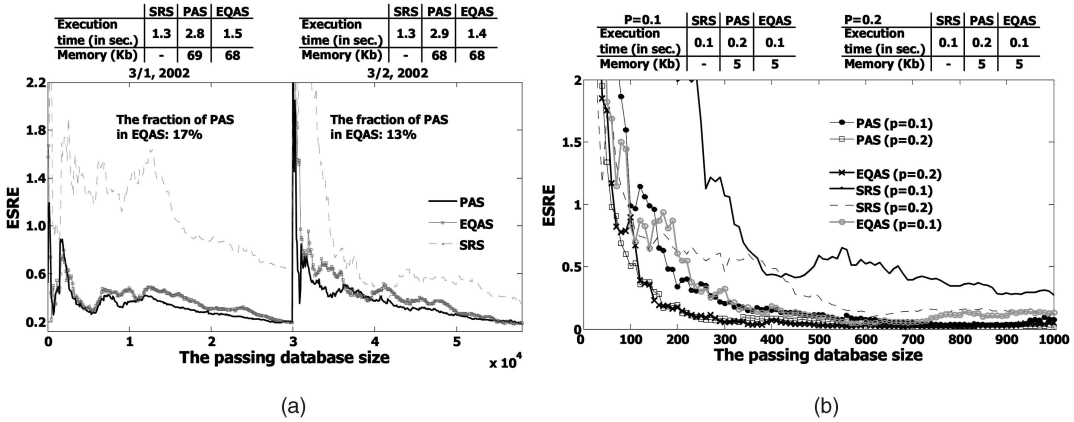
In addition, Table 3 also shows two employed real data sets. The first one is a data set of network alarm logs, named AlarmLog, which is provided by a major telecommunication company in Taiwan. The AlarmLog data set will be utilized to verify the feasibility of various sampling algorithms in the time-variant database. Note that this data records various alarms generated by a huge number of base station controllers and some types of alarms indeed more frequently occur during the weekday while others types of alarms may only occur during the weekend. Another real data set is a well-known public domain data, called the Mushroom data set, which is downloaded from the UCI machine learning repository [31].

The simulation is coded in C++ and performed in an IBM compatible PC with 3.2 GHz CPU and 1.0 GB memory. All implementations employ the common set of functions for performing I/O. There are four sampling algorithms, i.e., *PAS*, *EQAS*, simple random sampling (abbreviated as SRS in the sequel), and *EASE* [8]. Note that *EASE* is the state-of-the-art sampling algorithm used to reduce the *absolute* proportion difference of each attribute value. The code of *EASE* is given from the authors of *EASE*. In addition, the default error threshold ε is set as 0.1 for algorithms *PAS* and *EQAS* and the sample rate $p = 0.1$.

4.2 Sensitivity Analysis of Algorithms PAS and EQAS

4.2.1 On Sampling Quality as Time Advances

We first investigate *ESRE*, i.e., the *expected square relative error* in algorithms *PAS*, *EQAS*, and *SRS* as time advances (algorithm *EASE* cannot be compared in this experiment since it cannot sequentially generate the sample). Fig. 6a shows *ESRE* as time advances, where the time-variant data set AlarmLog is utilized and the time granularity of a window is specified as "day." Note that two windows are shown, where 3/1/2002 is Friday and 3/2/2002 is Saturday. In practice, data distributions in these two windows are quite different, and some types of alarms are more frequently in the weekday (3/1/2002). That is why the curves of *ESRE* are so different in these two windows, which can show the applicability of various sampling algorithms in a time-variant data. It is clear to see that *ESRE* in algorithm *PAS* is on orders of magnitude smaller than that in random sampling. Importantly, we see that the curve of *ESRE* in algorithm *EQAS* is close to that in algorithm



(a) (b)

Fig. 6. The curve of ESRE over time in two real data sets. (a) ESRE in AlarmLog. (b) ESRE in Mushroom.

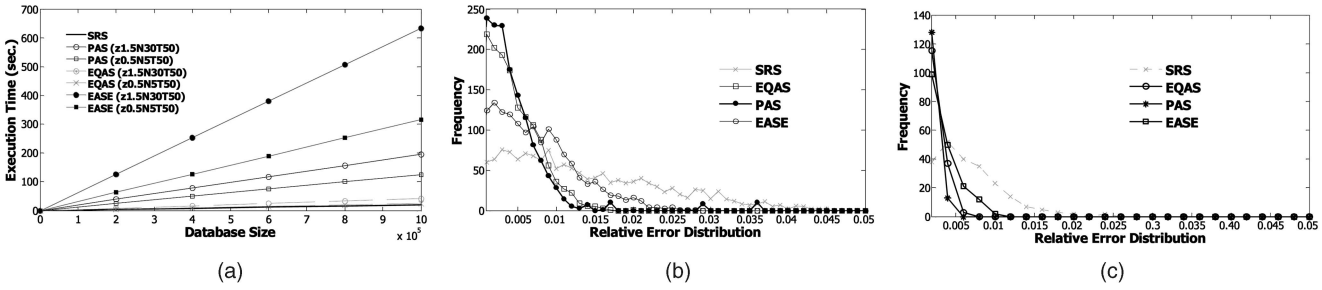


Fig. 7. The relative error distribution and the execution time in various sampling algorithms. (a) The execution time. (b) The relative error distribution (z1.5N30T50). (c) The relative error distribution (z0.5N5T50).

PAS, where only 10 percent $\sim$ 20 percent data in algorithm *EQAS* are handled by algorithm *PAS*. As also shown in Fig. 6a, we find the execution time of algorithm *EQAS* is nearly equal to the time consumed by random sampling. It demonstrates that algorithm *EQAS* can gain the high execution efficiency without much compromising the sampling quality. It is worth mentioning that some types of alarms will emergently and repeatedly occur during the rush time, which incurs the challenge of the burst sampling error. In algorithm *EQAS*, while the burst sampling error is identified by the probing process, algorithm *PAS* can quickly take over and preserve the sampling quality. In this experiment, we demonstrate our claim that algorithms *PAS* and *EQAS* can quickly reduce the *relative* proportion difference and they will be robust to the burst sampling error as compared to random sampling.

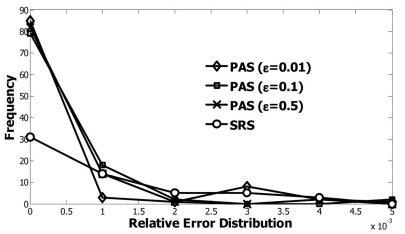
Fig. 6b shows *ESRE* in the Mushroom data set with various sample rates (for ease of presentation, only the observations of the first 1,000 tuples are shown). As can be seen, *ESRE* in algorithm *PAS* is stably and quickly reduced toward a convergent value. In contrast, random sampling suffers from the burst sampling error and *ESRE* cannot be effectively reduced, even though the sample rate is large $(\mathrm{p} = 0.2)$. It is interesting to point out that algorithms *PAS* and *EQAS* have the sampling quality with $p = 0.1$ better than that of random sampling with $p = 0.2$, showing the excellent proportion precision of the proposed algorithms. Note that the memory usages are also shown in Fig. 6. We can find that the memory usage is much smaller as compared to the memory required by the posterior mining applications such as the frequent-itemset mining [4].

4.2.2 On Execution Time and Relative Error Distribution

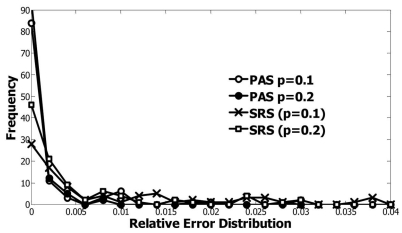
The sampling efficiency is further investigated in the two synthetic data sets, z1.5N30T50 and z0.5N5T50. We also investigate the sampling quality in a different perspective, called the *relative error distribution*. The *relative error distribution* refers to the distribution of the value $\left[\frac{\sup(a_j,W_k)-\sup(a_j,S_k)}{\sup(a_j,W_k)}\right]^2$ of each value a_j at the end of a window. In general, relative proportion errors of most values in a high-quality sample are close to zero so that the *relative error distribution* will be highly left-skewed. In Fig. 7, we show the execution time and the *relative error distribution* obtained in four sampling algorithms, including algorithm *EASE* [8].

We first investigate the scalability of different sampling algorithms on synthetic data sets with various sizes. Note that, for fair comparison of different algorithms, the window size in *PAS*, *EQAS*, and *SRS* will be set equal to the size of the population because *EASE* cannot be directly extended to the window-based scenario. As shown in Fig. 7a, whether data is highly skewed or not, the execution time of each sampling method grows linearly as the data set size increases. Note that the execution time of *SRS* is independent to various parameters of the two synthetic data sets because random sampling did not maintain/analyze the distribution of the population. Thus, we only show one execution time of *SRS* in Fig. 7a. Furthermore, the major reason of *EASE* having the longest execution time results from that *EASE* requires a corresponding time to obtain an initial large sample since *EASE* is a kind of two-phase sampling methods (the same as the size specified in [8], the size of the initial large sample is $0.3 \times |D|$), showing that algorithm *EASE* gains the sampling quality at the cost of sampling efficiency. Formally, the time consumed by *PAS* is also large as compared to the one of *SRS*, particularly in the data set z1.5N30T50 since the number of attributes is large. However, *EQAS* has the execution time very close to that of *SRS*. It is because, in algorithm *EQAS*, the fraction of data passed by *PAS* is relatively small as compared to that passed by *SRS*, thus leading to the insignificant computational overhead.

We then show the *relative error distribution* of generated samples in Fig. 7b and Fig. 7c. For ease of illustration, the *relative* proportion difference larger than 0.05 is truncated in these figures. In the high-dimensional and skewed data (Fig. 7b), each sampling algorithm inevitably leads to the larger *relative proportion inconsistency*. Importantly, the sampling quality of *PAS* is the best one since its *relative error distribution* is highly left-skewed. We also find that the result of *EQAS* is close to *PAS*. Similar results are also obtained in Fig. 7c, where the *relative* proportion difference in the low-dimensional and nonskewed data is relatively small as compared to that shown in Fig. 7b. Note that the result of *EASE* in the high-dimensional and skewed data is not good as compared to those of *PAS* and *EQAS*. Since *EASE* only minimizes the *absolute* proportion difference, the *relative* proportion difference of many attribute values, which rarely occur, will be apparently large in the skewed data set. In practice, the *relative* proportion error is very difficult to be bounded, especially for those attribute values whose population proportions are small. As a result, we



(a)



(b)

ε	Sampling	z1.5N30T50		z0.5N5T50	
		p=0.01	p=0.1	p=0.01	p=0.1
1	SRS	674,439	87,432	139,476	32,743
	PAS	88,232	11,232	15,430	8,324
	EQAS	100,580	12,124	21,758	11,259
0.1	SRS	1,284,438	384,100	324,483	127,435
	PAS	324,549	53,454	94,386	37,886
	EQAS	409,430	87,432	110,847	57,870
0.01	SRS	4,430,400	884,100	672,936	104,320
	PAS	580,498	53,454	127,493	52,301
	EQAS	608,302	87,432	129,103	67,438

(c)

Fig. 8. Studies of parameter sensitivity. (a) The relative error distribution with various error thresholds (Mushroom data set). (b) The relative error distribution with various sample rates (Mushroom data set). (c) The minimum population size to achieve the required relative error bound (synthetic data sets).

show the effectiveness of algorithms *PAS* and *EQAS* to preserve the sampling quality. Clearly, both considering the execution time and the sampling quality, *EQAS* will be the winner.

4.2.3 On Parameter Sensitivity

We investigate the effect of two parameters of *PAS*, i.e., ε and p. In the interests of space and ease of exposition, only *PAS* and *SRS* are compared in this analysis since *EQAS* and *PAS* have the similar sampling quality and only showing the results of *PAS* can demonstrate the pure behavior of our model without the effect from random sampling. Fig. 8a shows the relative error distribution of *PAS* in the Mushroom data set, where the error threshold ε varies from 0.01 to 0.5 ($p = 0.1$). Formally, the results of *PAS* with various ε are not obviously different to each other in this experiment. It is because *PAS* tries to minimize the *relative* proportion error no matter what level of ε is specified. However, on further investigation, we can see that the *relative error distribution* of *PAS* with $\varepsilon = 0.01$ is slightly left-sharper than that of *PAS* with $\varepsilon = 0.5$, but the *relative error distribution* of *PAS* with $\varepsilon = 0.01$ has a few small peaks in high *relative* proportion errors. The reason is that *PAS* simultaneously considers all attribute values of t_i, excluding *proportion-preserved values*, when t_i arrives. Note that it is difficult for each attribute value of t_i to become a *proportion-preserved value* of t_i when ε is small. As such, *PAS* with $\varepsilon = 0.01$ tries to simultaneously minimize the *relative* proportion differences of more attribute values and it leads to a slow convergence of the *relative* proportion errors of a few attribute values. In contrast, although the relative error distribution of *PAS* with $\varepsilon = 0.5$ is not as sharp as that of *PAS* with $\varepsilon = 0.01$, the relative errors of all attribute values are equally reduced, leading to a relative error distribution with less peaks. Clearly, the results of *PAS* all outperform *SRS*, demonstrating the robustness of *PAS*.

The investigation of another parameter of *PAS*, i.e., the sample rate p, is shown in Fig. 8b. As can be seen, the result of *PAS* with $p = 0.1$ is similar to that with $p = 0.2$, showing that *PAS* can guarantee the sampling quality without the need for large sample rates/sizes.

It is worth mentioning that random sampling with a fixed sample rate can also achieve the goal of having the bounded *relative* proportion difference of each attribute value as long as the database size is large enough. Therefore, an interesting question arises: What is the minimal population size to have the *relative* proportion difference of each attribute value being bounded below the specified threshold ε? We show the result in Fig. 8c. As can be seen, the *relative* proportion differences of all attribute values in *SRS* can be bounded while the database size is prohibitively large both in $p = 0.1$ and in $p = 0.01$. In practice, having such a large database within a time window is not prevalent. In contrast, algorithms *PAS* and *EQAS* both require a small database size, which is the reasonable size within a time window, to have the *relative* proportion difference of each attribute value being bounded, thus showing the applicability of algorithms *PAS* and *EQAS*.

4.2.4 On Relative Proportion Consistency of Multivariate Statistics

The relative proportion consistency of multivariate statistics is further investigated in the Mushroom data set. We show the relative error distributions of two-dimensinoal variables and three-dimensional variables in Figs. 9a and 9b, respectively. For ease of illustration, the *relative* proportion error larger than 0.05 is truncated in these figures. Clearly, we can see that *PAS* still excellently preserves the relative proportion consistency of multivariate statistics in orders of magnitude better than random sampling since the relative error distribution of *PAS* is highly left-skewed, thus confirming the statement shown in Theorem 4.

We also observe the *relative proportion consistency* of a randomly selected three-dimensional variable "stalk-shape: enlarging, stalk-root:equal, stalk-surface-above-ring: smooth," whose population proportion is equal to 4.3 percent. Its square relative error over time is shown in Fig. 9c. As compared to *SRS*, *PAS* can quickly and stably ensure a close-to-zero square relative error. Meanwhile, Fig. 9d shows the sampling distribution of the square relative error of this three-dimensional variable, generated from 10,000 runs with a sample rate equal to 0.1. The sampling distribution generated by *PAS* has a sharper curve than that generated by *SRS*, indicating that the variance of the multivariate statistic's sample proportion in *PAS* is much smaller than that in *SRS*. In this experiment, we demonstrate that *PAS* would guarantee the *relative proportion consistency* of multivariate statistics and also show that *PAS* is an excellently unbiased and robust sampling mechanism.

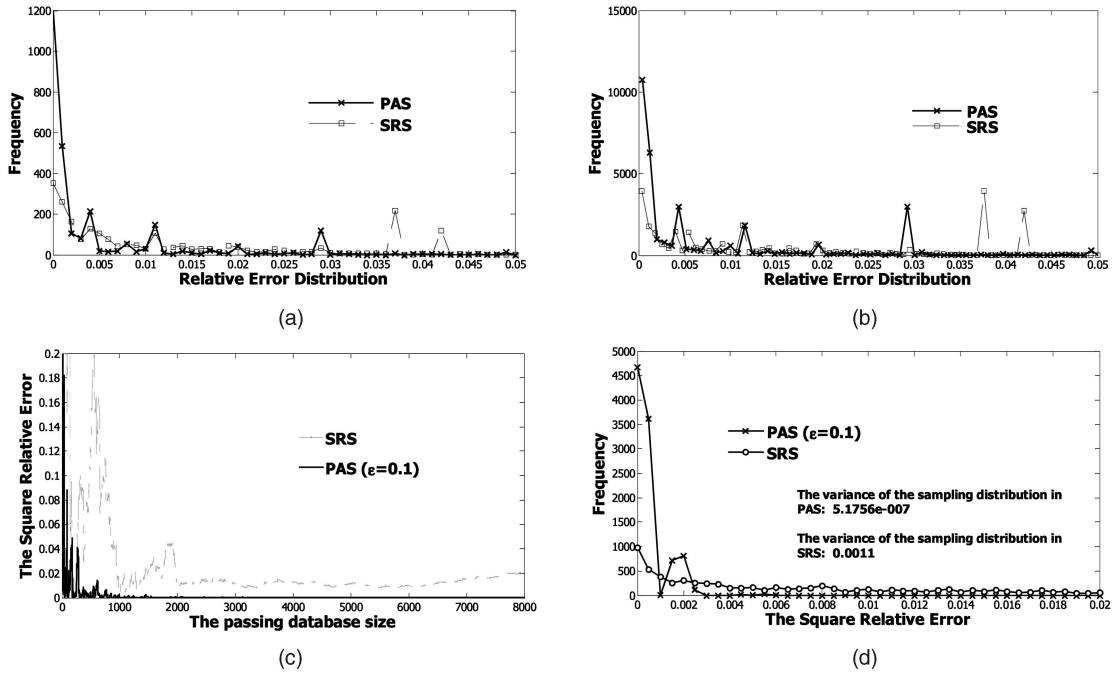


Fig. 9. The relative error distribution of multivariate statistics (Mushroom data set). (a) The relative error distribution of two-dimensional variables. (b) The relative error distribution of three-dimensional variables. (c) The Square Relative Error as time advances (three-dimensional variable: "stalk-shape:enlarging, stalk-root:equal, stalk-surface-above-ring:smooth"). (d) The sampling distribution of the Relative Error (three-dimensional variable: "stalk-shape:enlarging, stalk-root:equal, stalk-surface-above-ring:smooth").

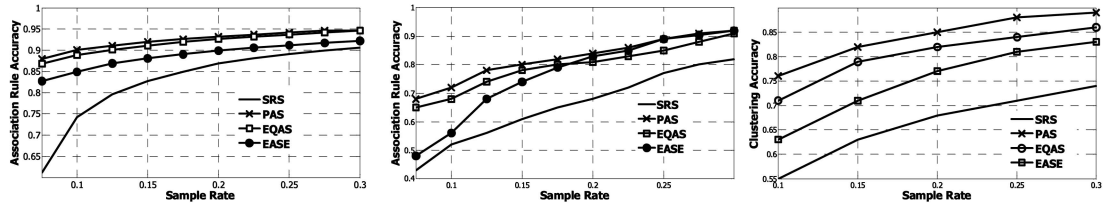


Fig. 10. Sampling effectiveness for frequent-itemset mining and clustering (Mushroom data set). (a) Mining frequent itemsets with $\min\_sup = 0.3$. (b) Mining frequent itemsets with $\min\_sup = 0.15$. (c) EM Clustering.

4.3 Application Studies

To investigate the advantage gained by preserving the *relative proportion consistency*, we first execute algorithm FP-growth, which is downloaded from Christian Borgelt's Web site,[3] on samples generated by *PAS*, *EQAS*, *EASE*, and *SRS*. First, in Figs. 10a and 10b, we show the accuracy of retrieved frequent itemsets in the Mushroom data set, where the minimum supports are specified as 0.3 (2,735 frequent itemsets are discovered in the original Mushroom data set) and 0.15 (98,575 frequent itemsets are identified in the original Mushroom data set). Formally, we use the *F-Score* measurement [8], $F(S)$, to evaluate the accuracy of frequent itemsets which are obtained in the sample S, where $F(S) = \frac{2\times|L(D)\cap L(S)|}{|L(D)-L(S)|+|L(S)-L(D)|}$. $L(D)$ and $L(S)$ denote the sets of frequent itemsets obtained in the original data set D and in the sample S, respectively. We show the accuracy of discovered frequent itemsets of each sample size as the average of 50 runs. As can be seen, algorithms *PAS*, *EQAS*, and *EASE* outperform *SRS* in orders of magnitude, especially when the sample size is small. In addition, note that *PAS* will reduce the *relative* proportion difference as opposed to the *absolute* proportion difference reduced by *EASE*. Reducing the *relative* proportion difference indeed avoids the information loss of some attribute values whose population proportions are close to the specified minimum support. Thus, we can see accuracy of frequent itemsets obtained by *PAS* and *EQAS* both exceed that of *EASE* about 5 percent in average, demonstrating the effectiveness of *PAS* and *EQAS* for mining frequent itemsets. In addition, Fig. 10b shows accuracy of frequent itemsets with the minimum support equal to 0.15. As can be seen, *PAS* outperforms *EASE* in orders of magnitude when the sample rate is small since preserving the *relative proportion consistency* is more important than preserving the *absolute proportion consistency* in the presence of a small minimum support, thus demonstrating the feasibility of *PAS* and *EQAS*.

We also executed EM clustering, which is implemented in *WEKA* [32], on the generated samples. Similarly to the training-and-testing process for evaluating clustering results in [32], the effectiveness of sampling for clustering can

3. The URL is http://fuzzy.cs.uni-magdeburg.de/~borgelt/fpgrowth.html.

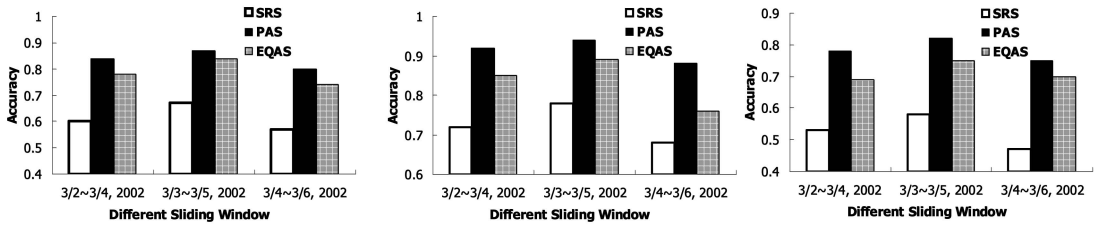


Fig. 11. The sampling effectiveness for incremental frequent-itemset mining (AlarmLog data set). (a) Accuracy with $\mathrm{p} = 0.01$, minimum support = 0.5 percent. (b) Accuracy with $\mathrm{p} = 0.1$, minimum support = 0.5 percent (c) Accuracy with $\mathrm{p} = 0.01$, minimum support = 0.1 percent.

be evaluated by allocating each unsampled tuple to an appropriate cluster (corresponding to the testing process), where clusters are extracted from the sample (corresponding to the training process). Specifically, after blinding the class attribute c_i of every sampled tuple t_i, the *EM* clustering algorithm in *WEKA* is executed to generate two clusters from the sample of the Mushroom data set; one cluster can be regarded as *poisonous* and the other as *edible*. Next, the "classes to clusters" evaluation model in *WEKA* uses a log-likelihood function to assign a class to each unsampled tuple. We then have a 2×2 *contingency table* that shows the relationship between the original class and the estimated class. Finally, the *clustering accuracy*, calculated as the number of correctly clustered tuples divided by the total number of tuples, is deemed as the level of effectiveness of sampling for clustering. We show the clustering accuracy of each sample size as the average over 50 runs.

Fig. 10c shows the results of various sample rates. As can be seen, *PAS* obviously outperforms other sampling algorithms in the clustering task. It is because *PAS* can excellently preserve the *relative proportion consistency* of multivariate statistics. Note that clustering algorithms usually find the group knowledge from correlations between different dimensions. As such, it is clear that *PAS* will have excellent effectiveness for clustering.

Furthermore, the most attractive strength of algorithms *PAS* and *EQAS* lies in the sequential generation of samples, which is particularly important to incremental mining applications. Algorithm *EASE* indeed cannot be applied in such environments since it generates a pilot sample from the whole database in advance. We employ algorithm *SWF*, a sliding window-based mining approach [4] and study the model accuracy obtained by incrementally mining frequent itemsets on samples. The results are shown in Fig. 11 with various sample rates and various minimum supports in the AlarmLog data set. We set the sliding window size equal to three days in this experiment (the time granularity of a window is specified as one day). Clearly, algorithms *PAS* and *EQAS* result in the prominent accuracy of frequent itemsets in each sliding window as compared to that obtained by random sampling, demonstrating the applicability of algorithms *PAS* and *EQAS* to be the prominent means for sequentially generating high-quality samples.

5 Conclusions

This paper has introduced algorithm *PAS*, which is a sampling algorithm to sequentially generate samples in which the *relative* proportion error of each measured pattern can be minimized toward the specified error threshold. Another algorithm, called *EQAS*, was also proposed to integrate *PAS* and random sampling to provide the flexibility of striking a compromise between sampling quality and sampling efficiency. As validated in experimental results on real and synthetic data sets, both proposed algorithms have the prominent advantage of being an effective quality-aware sampling means for incremental mining applications.

Appendix

Proof of Theorem 1. Suppose that $\frac{s\_N^k(a_j)}{N^k(a_j)} = p + \xi_j$, for every attribute value a_j. Assume that the maximum absolute value of ξ_j, i.e., $\xi_{\max}$, can be bounded below $\frac{\varepsilon}{2+\varepsilon}p$, i.e., $\left|\frac{s\_N^k(a_j)}{N^k(a_j)} - p\right| \le \frac{\varepsilon}{2+\varepsilon}p$, $\forall a_j$. We have

$$\sum_{j=1}^{|A|} s\_N^k(a_j) = \sum_{j=1}^{|A|}((p+\xi_j) \times N^k(a_j)),$$

where $|A|$ denotes the number of distinct attribute values in the database. Furthermore, we have h attributes in the tabular database, which yields that

$$\sum_{j=1}^{|A|} s\_N^k(a_j) = h \times |S_k|,$$

and $\sum_{j=1}^{|A|} N^k(a_j) = h \times |W_k|$. Therefore, we have $\sum_{j=1}^{|A|} s\_N^k(a_j) \le \sum_{j=1}^{|A|}(p+\xi_{\max})N^k(a_j)$, yielding that

$$h \times |S_k| \le (p+\xi_{\max}) \times h \times |W_k|; \quad \frac{|S_k|}{|W_k|} \le \left(1 + \frac{\varepsilon}{2+\varepsilon}\right) \times p.$$

Similarly, we have $\frac{|S_k|}{|W_k|} \ge \left(1 - \frac{\varepsilon}{2+\varepsilon}\right) \times p$, yielding that $\left(1 - \frac{\varepsilon}{2+\varepsilon}\right) \times p \le \frac{|S_k|}{|W_k|} \le \left(1 + \frac{\varepsilon}{2+\varepsilon}\right) \times p$. In addition, note that $s\_N^k(a_j) = (p+\xi_j)N^k(a_j)$ for every attribute value a_j and $\left(1 - \frac{\varepsilon}{2+\varepsilon}\right)p|W_k| \le |S_k| \le \left(1 + \frac{\varepsilon}{2+\varepsilon}\right)p|W_k|$. Without loss of generality, we have $\left(1 - \frac{\varepsilon}{2+\varepsilon}\right) > 0$ because ε will be set as a small value (in our experiments, ε is set below 0.1 in general). Therefore, $\frac{s\_N^k(a_j)}{|S_k|} \ge \frac{(p+\xi_j)\times N^k(a_j)}{\left(1+\frac{\varepsilon}{2+\varepsilon}\right)\times p \times |W_k|}$, indicating that $\sup(a_j, S_k) \ge \frac{(1-\frac{\varepsilon}{2+\varepsilon})p}{(1+\frac{\varepsilon}{2+\varepsilon})p}\sup(a_j, W_k)$ ($\because \xi_j \ge -\frac{\varepsilon}{2+\varepsilon}p$). Since $\frac{(1-\frac{\varepsilon}{2+\varepsilon})p}{(1+\frac{\varepsilon}{2+\varepsilon})p} = \frac{1}{1+\varepsilon}$, we have $\sup(a_j, S_k) \ge \frac{1}{1+\varepsilon}\sup(a_j, W_k)$.

Note that $(1-\varepsilon)(1+\varepsilon) < 1$, indicating $\frac{1}{1+\varepsilon} > 1-\varepsilon$, and $\sup(a_j, S_k) \geq (1-\varepsilon)\sup(a_j, W_k)$. Similarly, we can derive $\sup(a_j, S_k) \leq (1+\varepsilon)\sup(a_j, W_k)$. Finally, we have $|\sup(a_j, W_k) - \sup(a_j, S_k)| \leq \varepsilon \times \sup(a_j, W_k)$. □

Proof of Theorem 2. Note that

$$h \times |S_k| = \sum_{j=1}^{|A|} s\_N^k(a_j) = \sum_{j=1}^{|A|}(1+\xi_j) \times p \times N^k(a_j)$$
$$= p \times \left[\sum_{j=1}^{|A|} N^k(a_j)\right] + p \times \left[\sum_{j=1}^{|A|} \xi_j \times N^k(a_j)\right].$$

Moreover, $h \times |W_k| = \sum_{j=1}^{|A|} N^k(a_j)$, which yields that

$$\frac{|S_k|}{|W_k|} = p \times \left(1 + \frac{\sum_{j=1}^{|A|} \xi_j \times N^k(a_j)}{\sum_{j=1}^{|A|} N^k(a_j)}\right) = p \times (1+\Gamma).$$

Furthermore, we will have

$$\sup(a_j, S_k) = \frac{(1+\xi_j) \times p \times N^k(a_j)}{\Gamma \times p \times |W_k|} = \frac{(1+\xi_j)}{1+\Gamma} \times \sup(a_j, W_k),$$

indicating that $\frac{|\sup(a_j,W_k) - \sup(a_j,S_k)|}{\sup(a_j,W_k)} = \left|1 - \frac{(1+\xi_j)}{1+\Gamma}\right|$. □

Proof of Theorem 3.

$$\widehat{p_r}(t_i) = \arg\min_{0 \leq p_r(t_i) \leq 1} \sum_{i=1}^{|V_i|} \left[\frac{1 \times p_r(t_i)}{N_{i-1}^k(a_j)+1} + \left(\frac{s\_N_{i-1}^k(a_j)}{N_{i-1}^k(a_j)+1} - p\right)\right]^2$$
$$= \arg\min_{0 \leq p_r(t_i) \leq 1} \left[a \times \left(p_r(t_i) + \frac{b}{a}\right)^2 + \eta\right],$$

where η denotes these terms without $p_r(t_i)$. Since $a \geq 0$, it implies that $a \times \left(p_r(t_i) + \frac{b}{a}\right)^2 + \eta$ is a convex. The minimum of $\left[a \times \left(p_r(t_i) + \frac{b}{a}\right)^2 + \eta\right]$ will occur when $p_r(t_i) = -\frac{b}{a}$. In addition, $\widehat{p_r}(t)$ must locate in [0, 1] since it must comply with the probability axiom. Hence, if $-\frac{b}{a} > 1$, $\widehat{p_r}(t_i) = 1$. If $-\frac{b}{a} < 0$, $\widehat{p_r}(t_i) = 0$. Therefore, $\widehat{p_r}(t) = -\frac{b}{a}$, subject to $1 \geq \widehat{p_r}(t) \geq 0$. □

Proof of Lemma 1. For the attribute value $a_j \in t_i$, there are two possibilities of its sample proportion after t_i is selected or discarded: 1) $\frac{s\_N_{i-1}^k(a_j)+1}{|S_{k,i-1}|+1}$ or 2) $\frac{s\_N_{i-1}^k(a_j)}{|S_{k,i-1}|}$. Note that $\sup_i(a_j, W_k) = \frac{N_{i-1}^k(a_j)+1}{|W_{k,i-1}|+1}$. Therefore, while

$$\left|\frac{s\_N_{i-1}^k(a_j)+1}{|S_{k,i-1}|+1} - \frac{N_{i-1}^k(a_j)+1}{|W_{k,i-1}|+1}\right| \leq \varepsilon \times \frac{N_{i-1}^k(a_j)+1}{|W_{k,i-1}|+1}$$

and $\left|\frac{s\_N_{i-1}^k(a_j)}{|S_{k,i-1}|} - \frac{N_{i-1}^k(a_j)+1}{|W_{k,i-1}|+1}\right| \leq \varepsilon \times \frac{N_{i-1}^k(a_j)+1}{|W_{k,i-1}|+1}$, we will have the relative proportion difference of a_j after the arrival of t_i being bounded below ε whatever t_i is sampled or not, indicating a_j is a proportion-preserved value of t_i. □

Proof of Theorem 4. Without loss of generality, we first analyze the case of two-dimensional variables. At first, we define $\frac{s\_N^k(a_i)}{N^k(a_i)} = (1+\xi_i) \times p$ for every attribute value a_i in the window W_k, i.e., $s\_N^k(a_i) = (1+\xi_i) \times p \times N^k(a_i)$. Moreover, let $\{b_1, \ldots, b_j, \ldots, b_m\}$ denote the set of distinct values in another attribute B, and let $\frac{s\_N^k(a_i b_\ell)}{N^k(a_i b_\ell)} = (1+\gamma_\ell) \times p$, where $1 \leq \ell \leq m$. Since Theorem 1 says that $|S_k| \approx p|W_k|$, we have the *relative proportion inconsistency* of a two-dimensional variable $a_i b_j$ equal to

$$\frac{\left|\frac{s\_N^k(a_i b_j)}{|S_k|} - \frac{N^k(a_i b_j)}{|W_k|}\right|}{\frac{N^k(a_i b_j)}{|W_k|}} = \frac{\left(1+\gamma_j\right) \times p \times N^k(a_i b_j)}{p|W_k|} - \frac{N^k(a_i b_j)}{|W_k|} = |\gamma_j|.$$

Clearly, $s\_N^k(a_i) = \sum_{\ell=1}^m s\_N^k(a_i b_\ell)$, indicating that

$$s\_N^k(a_i) = p \times \sum_{\ell=1}^m N^k(a_i b_\ell) + p \times \xi_i \times \sum_{\ell=1}^m N^k(a_i b_\ell).$$

As such, we also have

$$p \times \sum_{\ell=1}^m \gamma_\ell \times N^k(a_i b_\ell) = p \times \xi_i \times \sum_{\ell=1}^m N^k(a_i b_\ell).$$

Note that the goal of *PAS* is to minimize ξ_i^2 (recall Theorem 2), implying that *PAS* minimizes

$$\left[\sum_{\ell=1}^m \gamma_\ell \times N^k(a_i b_\ell)\right]^2.$$

While the next tuple containing $a_i b_j$ arrives, minimizing

$$\left[\left(\sum_{\ell=1, \ell \neq 1}^m \gamma_\ell \times N^k(a_i b_\ell)\right) + \gamma_j \times N^k(a_i b_j)\right]^2$$

leads to minimize γ_j^2 since $\left(\sum_{\ell=1, \ell \neq 1}^m \gamma_\ell \times N^k(a_i b_\ell)\right)$ and $N^k(a_i b_j)$ cannot be minimized at the time. Note that the proof can be easily extended to analyze the multivariate statistic with more than two variables. As such, we prove that the *relative proportion inconsistency* of multivariate statistics can be minimized by *PAS*. □

REFERENCES

[1] M. Charikar, C. Chekuri, T. Feder, and R. Motwani, "Incremental Clustering and Dynamic Information Retrieval," *Proc. ACM Symp. Theory of Computing*, 1997.

[2] C.-Y. Chen, S.-C. Hwang, and Y.-J. Oyang, "An Incremental Hierarchical Data Clustering Algorithm Based on Gravity Theory," *Proc. Pacific-Asia Conf. Knowledge Discovery and Data Mining*, 2002.

[3] D. Fisher, "Knowledge Acquisition via Incremental Conceptual Clustering," *Machine Learning*, 1987.

[4] C.-H. Lee, C.-R. Lin, and M.-S. Chen, "Sliding-Window Filtering: An Efficient Algorithm for Incremental Mining," *Proc. Conf. Information and Knowledge Management*, 2001.

[5] T. Zhang, R. Ramakrishnan, and M. Livny, "BIRCH: An Efficient Data Clustering Method for Very Large Databases," *Proc. ACM SIGMOD*, 1996.

[6] S.D. Lee, D.W.-L. Cheung, and B. Kao, "Is Sampling Useful in Data Mining? A Case in the Maintenance of Discovered Association Rules," *Proc. ACM SIGMOD Workshop Research Issues in Data Mining and Knowledge Discovery*, vol. 2, no. 3, pp. 233-262, 1998.

[7] P.S. Levy and S. Lemeshow, *Sampling of Populations: Methods and Applications.* John Wiley and Sons, 1991.

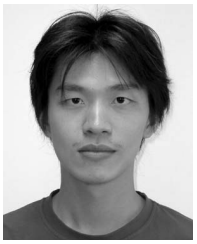
[8] H. Bronnimann, B. Chen, M. Dash, P.J. Haas, and P. Scheuermann, "Efficient Data Reduction with EASE," *Proc. ACM SIGKDD*, 2003.

[9] C. Domingo, R. Gavalda, and O. Watanabe, "Adaptive Sampling Methods for Scaling Up Knowledge Discovery Algorithms," *Data Mining and Knowledge Discovery*, 2002.

[10] B. Gu, F. Hu, and H. Liu, "Sampling and Its Application in Data Mining: A Survey," technical report, School of Computing, Nat'l Univ. of Singapore, 2000.

[11] S. Guha, R. Rastogi, and K. Shim, "CURE: An Efficient Clustering Algorithm for Large Databases," *Proc. ACM SIGMOD,* 1998.
[12] M. Zaki, S. Parthasarathy, W. Li, and M. Ogihara, "Evaluation of Sampling for Data Mining of Association Rules," *Proc. Int'l Workshop Research Issues in Data Eng.,* 1997.
[13] M. Garofalakis and P.B. Gibbons, "Probabilistic Wavelet Synopses," *ACM Trans. Database Systems,* vol. 29, no. 1, 2004.
[14] S. Guha, K. Shim, and J. Woo, "REHIST: Relative Error Histogram Construction Algorithms," *Proc. Int'l Conf. Very Large Data Bases,* 2004.
[15] B. Babcock, M. Datar, and R. Motwani, "Sampling from a Moving Window over Streaming Data," *Proc. 13th Ann. ACM-SIAM Symp. Discrete Algorithms,* 2002.
[16] D. Barbará, W. DuMouchel, C. Faloutsos, P.J. Haas, J.M. Hellerstein, Y.E. Ioannidis, H.V. Jagadish, T. Johnson, R.T. Ng, V. Poosala, K.A. Ross, and K.C. Sevcik, "The New Jersey Data Reduction Report," *IEEE Data Eng. Bull.,* vol. 20, no. 4, pp. 3-45, 1997.
[17] J.A. Rice, *Mathematical Statistics and Data Analysis.* Duxbury Press, 1995.
[18] S.K. Thompson and G.A.F. Seber, *Adaptive Sampling.* Wiley Series in Probability and Statistics, 1996.
[19] J.S. Vitter, "An Efficient Algorithm for Sequential Random Sampling," *ACM Trans. Math. Software,* vol. 13, no. 1, 1987.
[20] J. Vitter, "Random Sampling with a Reservoir," *ACM Trans. Math. Software,* 1985.
[21] C.R. Palmer and C. Faloutsos, "Density Biased Sampling: An Improved Method for Data Mining and Clustering," *Proc. ACM SIGMOD,* 2000.
[22] F. Provost, D. Jensen, and T. Oates, "Efficient Progressive Sampling," *Proc. ACM SIGKDD,* 1999.
[23] J. Han and M. Kamber, *Data Mining: Concepts and Techniques.* Morgan Kaufmann, 2000.
[24] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan, "Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications," *Proc. ACM SIGMOD Int'l Conf. Management of Data,* 1998.
[25] G.H. John and P. Langley, "Static versus Dynamic Sampling for Data Mining," *Proc. ACM SIGKDD,* 1996.
[26] H. Wang, W. Fan, P.S. Yu, and J. Han, "Mining Concept-Drifting Data Streams Using Ensemble Classifiers," *Proc. ACM SIGKDD,* 2003.
[27] J.H. Chang and W.S. Lee, "Finding Recent Frequent Itemsets Adaptively over Online Data Streams," *Proc. ACM SIGKDD,* 2003.
[28] J. Dougherty, R. Kohavi, and M. Sahami, "Supervised and Unsupervised Discretization of Continuous Features," *Proc. Int'l Conf. Machine Learning,* 1995.
[29] A. Metwally, D. Agrawal, and A.E. Abbadi, "Efficient Computation of Frequent and Top-k Elements in Data Streams," *Proc. Int'l Conf. Database Theory,* 2005.
[30] J.X. Yu, Z. Chong, H. Lu, and A. Zhou, "False Positive or False Negative: Mining Frequent Itemsets from High Speed Transactional Data Streams," *Proc. Int'l Conf. Very Large Data Bases,* 2004.
[31] C. Blake and C. Merz, UCI Repository of Machine Learning Databases, http://www.ics.uci.edu/~mlearn/MLRepository.html. 1998.
[32] I.H. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools with Java Implementations.* Morgan Kaufmann, 1999.

Kun-Ta Chuang received the BS degree from National Taiwan Normal University, Taipei, Taiwan, in 2000, and the PhD degree in Communication Engineering from National Taiwan University, Taipei, Taiwan, in 2006. He is currently serving as a software engineer at SYNOPSYS Inc., developing physical verification tools. His research interests include data mining, mobile data management, and electronic design automation. He is a member of the IEEE.



Keng-Pei Lin received the BS degree in computer science and information engineering from National Chiao Tung University, Hsinchu, Taiwan, in 2005. He is currently working toward the PhD degree in the Electrical Engineering Department, National Taiwan University, Taipei, Taiwan. His research interests include data mining and Internet technology.

Ming-Syan Chen received the BS degree in electrical engineering from National Taiwan University, Taipei, Taiwan, and the MS and PhD degrees in computer, information, and control engineering from The University of Michigan, Ann Arbor, in 1985 and 1988, respectively. Dr. Chen is currently a professor in the Electrical Engineering Department, National Taiwan University, Taipei, Taiwan. He was a research staff member at the IBM Thomas J. Watson Research Center, Yorktown Heights, New York, from 1988 to 1996. His research interests include database systems, data mining, mobile computing systems, and multimedia networking and he has published more than 230 papers in his research areas. In addition to serving as a program committee member for many conferences, Dr. Chen served as an associate editor of the *IEEE Transactions on Knowledge and Data Engineering (TKDE)* from 1997 to 2001, is currently on the editorial boards of the *Very Large Data Base (VLDB) Journal*, the *Knowledge and Information Systems (KAIS) Journal*, the *Journal of Information Science and Engineering*, and the *International Journal of Electrical Engineering*, and was a Distinguished Visitor of the IEEE Computer Society for Asia-Pacific from 1998 to 2000, and also from 2005 to 2007. He served as program chairs/vice-chairs and keynote/tutorial speakers for many international conferences. He holds, or has applied for, 18 US patents and seven ROC patents in his research areas. He is a recipient of the NSC (National Science Council) Distinguished Research Award, the Pan Wen Yuan Distinguished Research Award, the Teco Award, the Honorary Medal of Information, the K.-T. Li Research Breakthrough Award for his research work, and also the Outstanding Innovation Award from IBM Corporate for his contribution to a major database product. He also received numerous awards for his research, teaching, inventions, and patent applications. Dr. Chen is a fellow of the IEEE and a fellow of the ACM.