Output Privacy in Data Mining

TING WANG and LING LIU, Georgia Institute of Technology

Privacy has been identified as a vital requirement in designing and implementing data mining systems. In general, privacy preservation demands protecting both *input* and *output* privacy: the former refers to sanitizing the raw data itself before performing mining; while the latter refers to preventing the mining output (models or patterns) from malicious inference attacks. This article presents a systematic study on the problem of protecting output privacy in data mining, and particularly, stream mining: (i) we highlight the importance of this problem by showing that even sufficient protection of input privacy does not guarantee that of output privacy; (ii) we present a general inferencing and disclosure model that exploits the intrawindow and interwindow privacy breaches in stream mining output; (iii) we propose a light-weighted countermeasure that effectively eliminates these breaches without explicitly detecting them, while minimizing the loss of output accuracy; (iv) we further optimize the basic scheme by taking account of two types of semantic constraints, aiming at maximally preserving utility-related semantics while maintaining hard privacy guarantee; (v) finally, we conduct extensive experimental evaluation over both synthetic and real data to validate the efficacy of our approach.

Categories and Subject Descriptors: H.2.8 [**Database Management**]: Database Applications—*Data mining*; H.2.7 [**Database Management**]: Database Administration—*Security, integrity, and protection*

General Terms: Security, Algorithm, Experimentation

Additional Key Words and Phrases: Output privacy, stream mining, data perturbation

ACM Reference Format:
Wang, T. and Liu, L. 2011. Output privacy in data mining. ACM Trans. Datab. Syst. 36, 1, Article 1 (March 2011), 34 pages.
DOI = 10.1145/1929934.1929935 http://doi.acm.org/10.1145/1929934.1929935

1. INTRODUCTION

Privacy of personal information has been arising as a vital requirement in designing and implementing data mining and management systems; individuals were usually unwilling to provide their personal information if they knew that the privacy of their data could be compromised. To this end, a plethora of work has been done on preserving *input privacy* for static data [Agrawal and Srikant 2000; Sweeney 2002; Evfimievski et al. 2002; Chen and Liu 2005; Machanavajjhala et al. 2006], which assumes untrusted data recipients and enforces privacy regulations by sanitizing the raw data before sending it to the recipients. The mining process is then performed over the sanitized data, and produce output (patterns or models) with accuracy comparable to, if not identical to that constructed over the raw data. This scenario is illustrated as the first four steps of the grand framework of privacy-preserving data mining in Figure 1.

This work is partly sponsored by grants from NSF CyberTrust, NSF NetsSE, an IBM SUR grant, and a grant from Intel Research Council.
Author's address: T. Wang; email: twang@cc.gatech.edu.

DOI 10.1145/1929934.1929935 http://doi.acm.org/10.1145/1929934.1929935

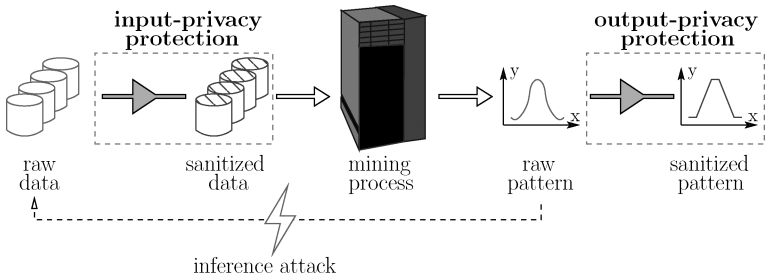

Fig. 1. Grand framework of privacy-preserving data mining.

Nevertheless, in a strict sense, privacy preservation not only requires prevention of unauthorized access to raw data that leads to exposure of sensitive information, but also includes eliminating unwanted disclosure of sensitive patterns through inference attacks over mining output. By sensitive patterns, we refer to those properties possessed uniquely by a small number of individuals participating in the input data. At the first glance, it may seem sufficient to sanitize input data in order to address this threat; however, as will be revealed, even though the patterns (or models) are built over the sanitized data, the published mining output could still be leveraged to infer sensitive patterns. Intuitively, this can be explained by the fact that input-privacy protection techniques are designed to make the constructed models close to, if not identical to, that built over the raw data, in order to guarantee the utility of the result. Such "no-outcome-change" property is considered as a key requirement of privacy-preserving data mining [Bu et al. 2007]. Given that the significant statistical information of the raw data is preserved, there exists the risk of disclosure of sensitive information. Therefore, the preservation of input privacy may not necessarily lead to that of output privacy, while it is necessary to introduce another unique layer of output-privacy protection into the framework, as shown in Figure 1. A concrete example is given as follows.

Example 1.1. Consider a nursing-care records database that collects the observed symptoms of the patients in a hospital. By mining such a database, one can discover valuable information regarding syndromes characterizing particular diseases. However, the released mining output can also be leveraged to uncover some combinations of symptoms that are so special that only rare people match them (we will show how to achieve this in the following sections), which qualifies as a severe threat to individuals' privacy.

Assume that Alice knows that Bob has certain symptoms a, b but not c ($\overline{c}$), and by analyzing the mining output, she finds that only one person in the hospital matching the specific combination of $\{a, b, \overline{c}\}$, and only one having all $\{a, b, \overline{c}, d\}$. She can safely conclude that the victim is Bob, who also suffers symptom d. Further more, by studying other medical databases, she may learn that the combination of $\{a, b, d\}$ is linked to a rare disease with fairly high chance.

The output-privacy issue is more complicated in stream mining, wherein the mining output usually needs to be published in a continuous and in-time manner. Not only a single-time release may contain privacy breaches, but also multiple releases can potentially be exploited in combination, given the overlap of the corresponding input data. Consider the sliding window model [Babcock et al. 2002] as an example, arguably the most popular stream processing model, where queries are not evaluated over the entire history of the stream, but rather over a sliding window of the most recent data

from the stream. The window may be defined over data items or timestamps, that is, item-based or time-based window, respectively. Besides the leakage in the output of a single window (*intra-window* breach), the output of multiple overlapping windows can also be combined to infer sensitive information (*inter-window* breach), even each window itself contains no breach per se. Moreover, the characteristics of data streams typically evolve over time, which precludes the feasibility of global data analysis-based techniques, due to the strict processing time and memory limitations. Hence, one needs to consider addressing output-privacy vulnerabilities in stream mining systems as a unique problem.

Surprisingly, in contrast to the wealth of work on protecting input privacy, output privacy has received fairly limited attention so far in both stream data mining and privacy-preserving data mining in general. This work, to our best knowledge, represents the most systematic study to date of output-privacy vulnerabilities in the context of data stream mining.

1.1. State of the Art

The first naturally arising question might be: is it sufficient to apply input-privacy protection techniques to address output vulnerabilities? Unfortunately, most existing techniques fail to satisfy the requirement of countering inference attacks over mining output. They differ from one to another in terms of concrete mechanisms to provide attack-resilient protection while minimizing utility loss of mining output incurred by sanitization; however, the adversarial attacks over input data (raw records) is significantly different from that over mining output (patterns or models).

As a concrete case, in Example 1.1, one conceivable solution to controlling the inference is to block or perturb those sensitive records, for example, the one corresponding to Bob, in the mining process; however, such record-level perturbation suffers from a number of drawbacks. First, the utility of mining output is not guaranteed. Since the perturbation directly affects the mining output, it is usually difficult to guarantee both that the valuable knowledge (the intended result) is preserved and that the sensitive patterns are disguised. Among these, one significant issue is the possible large amount of false knowledge. For instance, in Example 1.1, if the dataset is prepared for frequent pattern mining, blocking or perturbing sensitive records may make frequent patterns become nonfrequent, or vice versa; if the dataset is prepared for learning classification tree, modifying sensitive records may result in significant deviation of the cut points, crucial for decision making. Second, unlike the scenarios considered in certain existing work [Wang et al. 2007], in real applications, the sensitive patterns may not be predefined or directly observable; rather, sophisticated analysis over the entire dataset is typically necessary to detect potential privacy leakage in the mining output. For example, as we will show in Section 3, in the case of frequent pattern mining, the number of potential breaches needed to be checked is exponential in terms of the number of items. The situation is even more complicated for the case of stream mining wherein multiple windows can be exploited together for inference. Such complexity imposes efficiency issues for record-level perturbation. Third, in a broad range of computation-intensive applications, for instance, neural network-based models, the mining output is typically not directly observable; thus the effect of applying record-level perturbation cannot be evaluated without running the mining process. In all theses cases, record-level perturbation is ineffective to protect sensitive patterns.

Meanwhile, one might draw a comparison between our work and the disclosure control techniques in statistical and census databases. Both concern about providing statistical information without compromising sensitive information regarding individuals; however, they also exhibit significant distinctions. First, the queries of statistical databases typically involve only simple statistics, for instance, MIN, MAX, AVG, while

the output (patterns or models) of data mining applications usually feature much more complex structures, leading to more complicated requirements for output utility. Second, compared with that in statistical databases, the output-privacy protection in data mining faces much stricter constraints over processing time and space, which is especially true for the case of stream mining.

1.2. Overview of Our Solution

A straightforward yet inefficient solution to preserving output privacy is to detect and eliminate all potential breaches, that is, the *detecting-then-removing* paradigm as typically adopted by inference control in statistical databases. However, the detection of breaches usually requires computation-intensive analysis of the entire dataset, which is negative in tone [Chin and Ozsoyoglu 1981] for stream mining systems. Further, even at such high cost, the concrete operations of removing the identified breaches, for example, suppression and addition [Atzori et al. 2008], tend to result in considerable decrease in the utility of mining output.

Instead, we propose a novel proactive model to counter inference attacks over output. Analogous to sanitizing raw data from leaking sensitive information, we introduce the concept of "sanitized pattern," arguing that by intelligently modifying the "raw patterns" produced by mining process, one is able to significantly reduce the threat of malicious inference, while maximally preserving the utility of raw patterns. This scenario is shown as the last step in Figure 1.

In contrary to record-level perturbation, pattern-level perturbation demonstrates advantages in both protecting sensitive patterns and preserving output utility. First, the utility of mining output is guaranteed, for instance, it is feasible to precisely control the amount of false knowledge. For instance, in Example 1.1, all the valuable frequent patterns regarding symptom-disease relationships can be preserved, while no false frequent patterns are introduced. Also, as we will show in Section 5 and 6, in the case of frequent pattern mining, not only the accuracy of each frequent itemset can be controlled, but also their semantic relationships can be preserved to the maximum extent, which is hard to achieve with record-level perturbation. Second, it is possible to devise effective yet efficient pattern-level perturbation schemes that can be performed either online or offline, without affecting the efficiency of (stream) mining process. Finally, since the target of perturbation, the mining output, is directly observable to the perturbation process, it is possible to analytically gauge the perturbation schemes.

Specifically, we present BUTTERFLY*, a light-weighted countermeasure against malicious inference over mining output. It possesses a series of desirable features that make it suitable for (stream) mining applications: 1) it needs no explicit detection of (either intrawindow or interwindow) privacy breaches; 2) it requires no reference to previous output when publishing the current result; 3) it provides flexible control over the balance of multiple utility metrics and privacy guarantee.

Following a two-phase paradigm, BUTTERFLY* achieves attack-resilient protection and output-utility preservation simultaneously. In the first phase, it counters malicious inference by amplifying the uncertainty of sensitive patterns, at the cost of trivial accuracy loss of individual patterns. In the second phase, while guaranteeing the required privacy, it maximally optimizes the output utility by taking account of several model-specific semantic constraints.

Our contributions can be summarized as follows: (i) we articulate the problem and the importance of preserving output privacy in (stream) data mining; (ii) we expose a general inferencing attack model that exploits the privacy breaches existing in current (stream) mining systems; (iii) we propose a two-phase framework that effectively addresses attacks over mining output; (iv) we provide both theoretical analysis and

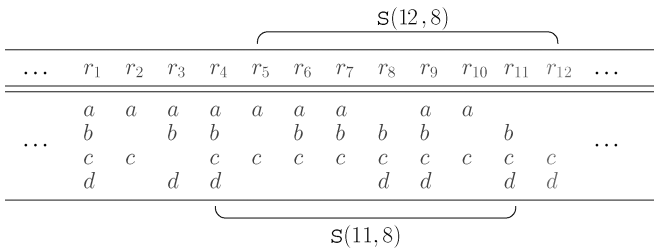

Fig. 2. Data stream and sliding window model.

experimental evaluation to validate our approach in terms of privacy guarantee, output utility, and execution efficiency.

1.3. Roadmap

We begin in Section 2 with introducing the preliminaries of frequent pattern mining over data streams, exemplifying with which, we formalize the problem of addressing output-privacy vulnerabilities in (stream) data mining. In Section 3, after introducing a set of basic inferencing techniques, we present two general attack models that exploit intra-window and inter-window breaches in stream mining output, respectively. Section 4 outlines the motivation and design objectives of BUTTERFLY*, followed by Section 5 and 6 detailing the two phases of BUTTERFLY* and discussing the implicit trade-offs among privacy guarantee and multiple utility metrics. Section 7 examines the impact of perturbation distribution over the quality of privacy protection and utility preservation. An empirical evaluation of the analytical models and the efficacy of BUTTERFLY* is presented in Section 8. Finally, Section 9 surveys relevant literature, and the article is concluded in Section 10.

2. PROBLEM FORMALIZATION

To expose the output-privacy vulnerabilities in existing mining systems, we exemplify with the case of frequent pattern mining over data streams. We first introduce the preliminary concepts of frequent pattern mining and pattern categorization, and then formalize the problem of protecting output privacy in such mining task.

2.1. Frequent Pattern Mining

Consider a finite set of *items* $\mathcal{I} = \{i_1, i_2, \ldots, i_M\}$. An *itemset* I is a subset of $\mathcal{I}$, i.e., $I \subseteq \mathcal{I}$. A *database* $\mathcal{D}$ consists of a set of records, each corresponding to a non-empty itemset. The *support* of an itemset I with respect to $\mathcal{D}$, denoted by $T_{\mathcal{D}}(I)$, is defined as the number of records containing I as a subset in $\mathcal{D}$. Frequent pattern mining aims at finding all itemsets with supports exceeding a predefined threshold C, called *minimum support*.

A *data stream* $\mathcal{S}$ is modeled as a sequence of records, $(r_1, r_2, \ldots, r_N)$, where N is the current size of $\mathcal{S}$, and grows as time goes by. The *sliding window* model is introduced to deal with the potential of N going to infinity. Concretely, at each N, one considers only the window of most recent H records, $(r_{N-H+1}, \ldots, r_N)$, denoted by $\mathcal{S}(N, H)$, where H is the window size. The problem is therefore to find all the frequent itemsets in each window.

Example 2.1. Consider a data stream with current size $N = 12$, window size $H = 8$, as shown in Figure 2, where $a \sim d$ and $r_1 \sim r_{12}$ represent the set of items and records, respectively. Assuming minimum support $C = 4$, then within window $\mathcal{S}(11, 8)$, $\{c, bc, ac, abc\}$ is a subset of frequent itemsets.

One can further generalize the concept of itemset by introducing the negation of an item. Let $\bar{i}$ denote the negation of item i. A record is said to contain $\bar{i}$ if it does not contain i. Following, we will use the term *pattern* to denote a set of items or negation of items, for example, $a\overline{bc}$. We use $\overline{I}$ to denote the negation of itmeset I, that is, $\overline{I} = \{\bar{i}|i \in I\}$.

Analogously, we say that a record r satisfies a pattern P if it contains all the items and negation of items in P and the support of P with respect to database $\mathcal{D}$ is defined as the number of records containing P in $\mathcal{D}$.

Example 2.2. In Fig. 2, r_{10} contains $a\bar{b}$, but not $a\bar{c}$. The pattern $\bar{a}bc$ has support 2 with respect to $\mathcal{S}(12, 8)$, because only records r_8 and r_{11} match it.

2.2. Pattern Categorization

Loosely speaking, output privacy refers to the requirement that the output of the mining process does not disclose any sensitive information regarding individuals participating in the input data.

In the context of frequent pattern mining, such sensitive information can be instantiated as patterns with extremely low supports, which correspond to properties uniquely possessed by few records (individuals), as shown in Example 1.1. We capture this intuition by introducing a threshold K ($K \ll C$), called *vulnerable support*, and consider patterns with (nonzero) supports below K as *vulnerable patterns*. We can then establish the following classification system.

Definition 2.3. (PATTERN CATEGORIZATION) Given a database $\mathcal{D}$, let $\mathcal{P}$ be the set of patterns appearing in $\mathcal{D}$, then all $P \in \mathcal{P}$ can be classified into three disjoint classes, for the given threshold K and C.

$$\begin{cases} \text{Frequent Pattern :} & \mathcal{P}_f = \{P|T_{\mathcal{D}}(P) \geq C\} \\ \text{Hard Vulnerable Pattern :} & \mathcal{P}_{hv} = \{P|0 < T_{\mathcal{D}}(P) \leq K\} \\ \text{Soft Vulnerable Pattern :} & \mathcal{P}_{sv} = \{P|K < T_{\mathcal{D}}(P) < C\}. \end{cases}$$

Intuitively, frequent pattern ($\mathcal{P}_f$) is the set of patterns with supports above minimum support; they expose the significant statistics of the underlying data, and are often the candidate in the mining process. Actually the frequent itemsets found by frequent pattern mining are a subset of $\mathcal{P}_f$. Hard vulnerable pattern ($\mathcal{P}_{hv}$) is the set of patterns with supports below vulnerable support; they represent the properties possessed by only few individuals, so it is unacceptable that they are disclosed or inferred from the mining output. Finally, soft vulnerable pattern ($\mathcal{P}_{sv}$) neither demonstrates the statistical significance nor violates the privacy of individual records; such patterns are not contained in the mining output, but it is usually tolerable that they are learned from the output.

Example 2.4. As shown in Figure 2, given $K = 1$ and $C = 4$, ac and bc are both $\mathcal{P}_f$, and $\overline{ab}c$ is $\mathcal{P}_{hv}$ with respect to $\mathcal{S}(12, 8)$, while bcd is $\mathcal{P}_{sv}$ since its support lies between K and C.

2.3. Problem Definition

We are now ready to formalize the problem of preserving output privacy in the context of frequent pattern mining over streams: For each sliding window $\mathcal{S}(N, H)$, output-privacy preservation prevents the disclosure or inference of any hard vulnerable patterns with respect to $\mathcal{S}(N, H)$ from the mining output.

It may seem at the first glance that no breach exists at all in frequent pattern mining, if it only outputs frequent itemsets (recall $C \gg K$); however, as shown in the next example (with detailed discussion in Section 3), from the released frequent

Table I. Symbols and Notations

Notation	Description
$\mathcal{S}(N, H)$	stream window of $(r_{N-H+1} \sim r_N)$
$T_{\mathcal{D}}(X)$	support of X in database $\mathcal{D}$
K	vulnerable support
C	minimum support
$\mathcal{X}_I^J$	set of itemsets $\{X \mid I \subseteq X \subseteq J\}$
W_p	previous window
W_c	current window
Δ_X^+	number of inserted records containing X
Δ_X^-	number of deleted records containing X

patterns and their associated supports, the adversary may still be able to infer certain hard vulnerable patterns.

Example 2.5. Recall Example 2.4. Given the supports of $\{c, ac, bc, abc\}$, based on the inclusion-exclusion principle [O'Connor 1993], $T(\overline{a}\overline{b}c) = T(c) - T(ac) - T(bc) + T(abc)$, one is able to infer the support of $\overline{a}\overline{b}c$, which is $\mathcal{P}_{hv}$ in $\mathcal{S}(12, 8)$.

3. ATTACK OVER MINING OUTPUT

In this section, we reveal the privacy breaches existing in current (stream) mining systems, and present a general attack model that exploits these breaches.

3.1. Attack Model

For simplicity of presentation, we will use the following notations: given two itemsets I and J, $I \oplus J$ denotes their union, $I \odot J$ their intersection, $J \ominus I$ the set difference of J and I, and $|I|$ the size of I. The notations used in the rest of the paper are listed in Table I.

As a special case of multiattribute aggregation, computing the support of I ($I \subseteq J$) can be considered as generalization of J over all the attributes of $J \ominus I$; therefore, one can apply the standard tool of multiattribute aggregation, a lattice structure, based on which we construct the attack model.

Lattice Structure. Consider two itemsets I, J that satisfy $I \subset J$. All the itemsets $\mathcal{X}_I^J = \{X \mid I \subseteq X \subseteq J\}$ form a lattice structure: each node corresponds to an itemset X, and each edge represents the generalization relationship between two nodes X_s and X_t such that $X_s \subset X_t$ and $|X_t \ominus X_s| = 1$. Namely, X_s is the generalization of X_t over the item $X_t \ominus X_s$.

Example 3.1. A lattice structure is shown in Fig. 3, where $I = c$, $J = abc$, and $J \ominus I = ab$.

For simplicity, in what follows, we use $\mathcal{X}_I^J$ to represent both the set of itemsets and their corresponding lattice structure. Next, we introduce the basis of our inferencing model, namely, *deriving pattern support* and *estimating itemset support*. These two techniques have been introduced in Atzori et al. [2008] and Calders and Goethals [2002], respectively, with usage or purpose different from ours. In [Atzori et al. 2008], deriving pattern support is considered as the sole attack model to uncover sensitive patterns; in Calders and Goethals [2002], estimating itemset support is used to mine nonderivable patterns, and thus saving the storage of patterns. The novelty of our work, however, lies in constructing a general inferencing model that exploits the privacy breaches existing in single or multiple releases of mining output, with these two primitives as building blocks.

Deriving Pattern Support. Consider two itemsets $I \subset J$, if the supports of all the lattice nodes of $\mathcal{X}_I^J$ are accessible, one is able to derive the support of pattern P, $P = I \oplus (\overline{J \ominus I})$, according to the inclusion-exclusion principle [O'Connor 1993]:

$$T(I \oplus (\overline{J \ominus I})) = \sum_{I \subseteq X \subseteq J} (-1)^{|X \ominus I|} T(X).$$

Example 3.2. Recall the example illustrated in Figure 3. Given the supports of the lattice nodes of $\mathcal{X}_c^{abc}$ in $\mathcal{S}(12, 8)$, the support of pattern $P = \overline{a}\overline{b}c$ is derived as: $T_{\mathcal{S}(12,8)}(\overline{a}\overline{b}c) = T_{\mathcal{S}(12,8)}(c) - T_{\mathcal{S}(12,8)}(ac) - T_{\mathcal{S}(12,8)}(bc) + T_{\mathcal{S}(12,8)}(abc) = 1$.

Essentially, the adversary can use this technique to infer vulnerable patterns with respect to one specific window from the mining output.

Estimating Itemset Support. For the support of any itemset is nonnegative, according to the inclusion-exclusion principle, if the supports of the itemsets $\{X | I \subseteq X \subset J\}$ are available, one is able to bound the support of J as follows:

$$\begin{cases} T(J) \leq \sum_{I \subseteq X \subset J} (-1)^{|J \ominus X|+1} T(X) & |J \ominus I| \text{ is odd} \\ T(J) \geq \sum_{I \subseteq X \subset J} (-1)^{|J \ominus X|+1} T(X) & |J \ominus I| \text{ is even.} \end{cases}$$

Example 3.3. Given the supports of c, ac, and bc in $\mathcal{S}(12, 8)$, one is able to establish the lower and upper bounds of $T_{\mathcal{S}(12,8)}(abc)$ as: $\leq T_{\mathcal{S}(12,8)}(ac) = 5$, $\leq T_{\mathcal{S}(12,8)}(bc) = 5$, $\geq T_{\mathcal{S}(12,8)}(ac) + T_{\mathcal{S}(12,8)}(bc) - T_{\mathcal{S}(12,8)}(c) = 2$.

When the bounds are tight, that is, the lower bound meets the upper bound, one can exactly determine the actual support. In our context, the adversary can leverage this technique to uncover the information regarding certain unpublished itemsets.

3.2. Intrawindow Inference

In stream mining systems without output-privacy protection, the released frequent itemsets over one specific window may contain intrawindow breaches, which can be exploited via the technique of deriving or estimating pattern support.

Example 3.4. As shown in Example 3.2, $\overline{a}\overline{b}c$ is $\mathcal{P}_{hv}$ with respect to $\mathcal{S}(12, 8)$ if $K = 1$; however, one can easily derive its support if the supports of c, ac, bc, abc are known.

Formally, if J is a frequent itemset, then according to the Apriori rule [Agrawal and Srikant 1994], all $X \subseteq J$ must be frequent, which are supposed to be reported with their supports. Therefore, the information is complete to compute the support of $P = I \oplus (\overline{J \ominus I})$ for all $I \subset J$. This also implies that the number of breaches needed to be checked is potentially exponential in terms of the number of items.

Even if the support of J is unavailable, that is, the lattice of $\mathcal{X}_I^J$ is incomplete to infer $P = I \oplus (\overline{J \ominus I})$, one can first apply the technique of estimating itemset support to complete certain missing "mosaics," then derive the supports of vulnerable patterns. Possibly the itemsets under estimation themselves may be vulnerable. Following, we assume that estimating itemset support is performed as a preprocessing step of the attack.

3.3. Interwindow Inference

The intrawindow inference attack is only a part of the story. In stream mining, privacy breaches may also exist in the output of overlapping windows. Intuitively, the output of a previous window can be leveraged to infer the vulnerable patterns within the current

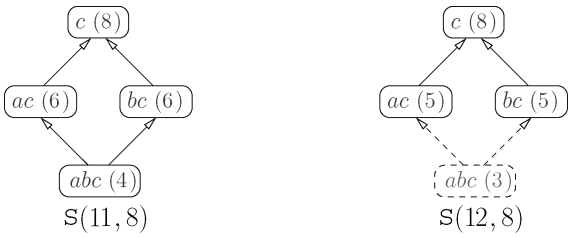

Fig. 3. Privacy breaches in stream mining output.

window, and vice versa, even though no vulnerable patterns can be inferred from the output of each window per se.

Example 3.5. Consider two windows $W_p = \mathcal{S}(11, 8)$ and $W_c = \mathcal{S}(12, 8)$ as shown in Fig. 2, with frequent itemsets summarized in Figure 3. Assume $C = 4$ and $K = 1$. In window W_p, no $\mathcal{P}_{hv}$ exists; while in W_c, abc is unaccessible (shown as dashed box). From the available information of W_c, the best guess about abc is $[2, 5]$, as discussed in Example 3.3. Clearly, this bound is not tight enough to infer that $\overline{a}bc$ is $\mathcal{P}_{hv}$. Both windows are thus immune to intra-window inference.

However, if one is able to derive that the support of abc decreases by 1 between W_p and W_c, then based on the information released in W_p, which is $T_{W_p}(abc) = 4$, the exact value of abc in W_c can be inferred, and $\overline{a}\overline{b}c$ is uncovered.

The main idea of interwindow inference is to exactly estimate the transition of the supports of certain itemsets between the previous and current windows. We below discuss how to achieve accurate estimation of such transition over two consecutive windows.

Without loss of generality, consider two overlapping windows $W_p = \mathcal{S}(N - L, H)$ and $W_c = \mathcal{S}(N, H)$ $(L < H)$, that is, W_c is lagging W_p by L records (in the example just given, $N = 12$, $H = 8$ and $L = 1$). Assume that the adversary attempts to derive the support of pattern $P = I \oplus (\overline{J \ominus I})$ in W_c. Let $\mathcal{X}_p$ and $\mathcal{X}_c$ be the subsets of $\mathcal{X}_I^J$ that are released or estimated from the output of W_p and W_c, respectively. We assume that $\mathcal{X}_p \oplus \mathcal{X}_c = \mathcal{X}_I^J$ $(\mathcal{X}_I^J \ominus \mathcal{X}_c = \mathcal{X}_p \ominus \mathcal{X}_c)$, i.e., the missing part in $\mathcal{X}_c$ can be obtained in $\mathcal{X}_p$. In Fig. 3, $\mathcal{X}_p = \{c, ac, bc, abc\}$, while $\mathcal{X}_c = \{c, ac, bc\}$.

For itemset X, let Δ_X^+ and Δ_X^- be the number of records containing X in the windows $\mathcal{S}(N, L)$ and $\mathcal{S}(N - H, L)$, respectively. Thus, the support change of X over W_p and W_c can be modeled as inserting Δ_X^+ records and deleting Δ_X^- ones, i.e., $T_{W_c}(X) = T_{W_p}(X) + \Delta_X^+ - \Delta_X^-$.

Example 3.6. Recall our running example, with $N = 12$, $H = 8$, and $L = 1$. $\mathcal{S}(N, L)$ corresponds to record r_{11} while $\mathcal{S}(N - H, L)$ refers to record r_4. Clearly, r_4 contains ac, while r_{11} does not; therefore, $T_{\mathcal{S}(12,8)}(ac) = T_{\mathcal{S}(11,8)}(ac) + \Delta_{ac}^+ - \Delta_{ac}^- = 5$.

The adversary is interested in estimating $T_{W_c}(X^*)$ for $X^* \in \mathcal{X}_p \ominus \mathcal{X}_c$. The bound (min, max) of $T_{W_c}(X^*)$ can be obtained by solving the following integer programming problem:

$$\max(\min) \quad T_{W_p}(X^*) + \Delta_{X^*}^+ - \Delta_{X^*}^-.$$

satisfying the constraints:

$$\mathsf{R}_1 : 0 \le \Delta_X^+, \Delta_X^- \le L$$
$$\mathsf{R}_2 : \Delta_X^+ - \Delta_X^- = T_{W_c}(X) - T_{W_p}(X) \qquad X \in \mathcal{X}_p \odot \mathcal{X}_c$$
$$\mathsf{R}_3 : \Delta_X^+(\Delta_X^-) \le \textstyle\sum_{I \subseteq Y \subset X} (-1)^{|X \ominus Y|+1} \Delta_Y^+(\Delta_Y^-) \qquad |X \ominus I| \text{ is odd}$$
$$\mathsf{R}_4 : \Delta_X^+(\Delta_X^-) \ge \textstyle\sum_{I \subseteq Y \subset X} (-1)^{|X \ominus Y|+1} \Delta_Y^+(\Delta_Y^-) \qquad |X \ominus I| \text{ is even.}$$

Here, R_1 stems from that W_p differs from W_c by L records. When transiting from W_p to W_c, the records containing X that are deleted or added cannot exceed L. R_2 amounts to saying that the support change $(\Delta_X^+ - \Delta_X^-)$ for those itemsets $X \in \mathcal{X}_c \odot \mathcal{X}_p$ is known. R_3 and R_4 are the application of estimating itemset support for itemsets in windows $\mathcal{S}(N, L)$ and $\mathcal{S}(N - H, L)$.

Sketchily, the inference process runs as follows: starting from the change of $X \in \mathcal{X}_p \odot \mathcal{X}_c$ (R_2), by using rules R_1, R_3, and R_4, one attempts to estimate $\Delta_X^+(\Delta_X^-)$ for $X \in \mathcal{X}_p \ominus \mathcal{X}_c$. It is noted that when the interval L between W_p and W_c is small enough, the estimation can be fairly tight.

Example 3.7. Consider our running example, $L = 1$, and $\mathcal{X}_p \odot \mathcal{X}_c = \{c, ac, bc\}$. One can first observe the following facts based on R_1 and R_2:

$$\Delta_{ac}^+ - \Delta_{ac}^- = -1, 0 \le \Delta_{ac}^+, \Delta_{ac}^- \le 1 \;\Rightarrow\; \Delta_{ac}^+ = 0, \Delta_{ac}^- = 1$$
$$\Delta_{bc}^+ - \Delta_{bc}^- = -1, 0 \le \Delta_{bc}^+, \Delta_{bc}^- \le 1 \;\Rightarrow\; \Delta_{bc}^+ = 0, \Delta_{bc}^- = 1.$$

Take ac as an instance. Its change over W_p and W_c is $\Delta_{ac}^+ - \Delta_{ac}^- = -1$, and both Δ_{ac}^+ and Δ_{ac}^- are bounded by 0 and 1; therefore, the only possibility is that $\Delta_{ac}^+ = 0$ and $\Delta_{ac}^- = 1$. Further, by applying R_3 and R_4, one has the following facts:

$$\Delta_{abc}^+ \le \Delta_{ac}^+ = 0 \qquad\qquad \Rightarrow\; \Delta_{abc}^+ = 0$$
$$\Delta_{abc}^- \ge \Delta_{ac}^- + \Delta_{bc}^- - \Delta_c^- = 1 \;\Rightarrow\; \Delta_{abc}^- = 1.$$

Take abc as an instance. Following the inclusion-exclusion principle, one knows that Δ_{abc}^+ should be no greater than $\Delta_{ac}^+ = 0$; hence, $\Delta_{abc}^+ = 0$. Meanwhile, Δ_{abc}^- has tight upper and lower bounds as 1. The estimation of abc over W_c is thus given by $T_{W_c}(abc) = T_{W_p}(abc) + \Delta_{abc}^+ - \Delta_{abc}^- = 3$, and the $\mathcal{P}_{hv}$ $\overline{ab}c$ is uncovered.

The computation overhead of interwindow inference is dominated by the cost of solving the constrained integer optimization problems. The available fast off-the-shelf tools make such an attack feasible even with moderate computation power.

4. OVERVIEW OF BUTTERFLY*

Motivated by the inferencing attack model above, we outline BUTTERFLY*, our solution to protecting output privacy for (stream) mining applications.

4.1. Design Objective

Alternative to the reactive, detecting-then-removing scheme, we intend to use a proactive approach to tackle both intrawindow and interwindow inference in a uniform manner. Our approach is motivated by two key observations. First, in certain mining applications, the utility of mining output is measured by metrics other than the exact supports of individual itemsets, for instance, the semantic relationship of their supports (e.g., the order or ratio of supports). It is thus acceptable to trade the accuracy of individual itemsets for boosting the output-privacy guarantee, provided that the desired output utility is maintained. Second, both intrawindow and interwindow inferencing attacks are based on the inclusion-exclusion principle, which involves multiple

frequent itemsets. Trivial randomness injected into each frequent itemset can accumulate into considerable uncertainty in inferred patterns. The more complicated the inference (i.e., harder to be detected), the more considerable such uncertainty.

We therefore propose BUTTERFLY*, a light-weighted output-privacy preservation scheme based on pattern perturbation. By sacrificing trivial accuracy of individual frequent itemsets, it significantly amplifies the uncertainty of vulnerable patterns, and thus blocking both intrawindow and interwindow inference.

4.2. Mining Output Perturbation

Data perturbation refers to the process of modifying confidential data while preserving its utility for intended applications [Adam and Worthmann 1989]. This is arguably the most important technique used to date for protecting original input data. In our scheme, we employ perturbation to inject uncertainty into mining output. The perturbation over output pattern significantly differs from that over input data. In input perturbation, the data utility is defined by the overall statistical characteristics of the dataset. The distorted data is fed as input into the following mining process. Typically no utility constraints are attached to individual data values. While in output perturbation, the perturbed results are directly presented to end-users, and the data utility is defined over each individual value.

There are typically two types of utility constraints for the perturbed results. First, each reported value should have enough accuracy; that is, the perturbed value should not deviate wildly from the actual value. Second, the semantic relationships among the results should be preserved to the maximum extent. There exist non-trivial trade-offs among these utility metrics. To our best knowledge, this work is the first to consider such multiple trade-offs in mining output perturbation.

Concretely, we consider two perturbation techniques, with their roots at statistics literature [Chin and Ozsoyoglu 1981; Adam and Worthmann 1989]. *Value distortion* perturbs the support by adding a random value drawn from certain probabilistic distribution. *Value bucketization* partitions the range of support into a set of disjoint, mutually exclusive intervals; instead of reporting the exact support, it returns the interval which the support belongs to.

Both techniques can be applied to output perturbation. However, value bucketization leads to fairly poor utility compared with value distortion, since all supports within an interval are modified to the same value, and any semantic constraints, for example, order or ratio, can hardly be enforced in this model. We thus focus on value distortion in the following discussion. Moreover, in order to guarantee the accuracy of individual frequent itemsets, we are more interested in probabilistic distributions with bounded intervals. We thus exemplify with discrete uniform distributions over integers, although our discussion is applicable for other distributions (details in Section 7).

4.3. Operation of BUTTERFLY*

On releasing the mining output of a stream window, we perturb the support of each frequent itemset X, $T(X)$[1] by adding a random variable r_X drawn from a discrete uniform distribution over the integers within an interval $[l_X, u_X]$. The sanitized support $T'(X) = T(X) + r_X$ is hence a random variable, which can be specified by its bias $\beta(X)$ and variance $\sigma^2(X)$. Intuitively, the bias indicates the difference of the expected value $E[T'(X)]$ and actual value $T(X)$, while the variance represents the average deviation of $T'(X)$ from $E[T'(X)]$. Note that compared with $T(X)$, r_X is nonsignificant, that is, $|r_X| \ll T(X)$.

[1]In what follows, without ambiguity, we omit the referred database $\mathcal{D}$ in the notations.

While this operation is simple, the setting of $\beta(X)$ and $\sigma^2(X)$ is nontrivial, in order to achieve sufficient privacy protection and utility guarantee simultaneously, which is the focus of our following discussion. Specifically, we will address the trade-off between privacy guarantee and output utility in Section 5, and the trade-offs among multiple utility metrics in Section 6.

5. BASIC BUTTERFLY*

We start with defining the metrics to quantitatively measure the precision of individual frequent itemsets, and the privacy protection for vulnerable patterns.

5.1. Precision Measure

The precision loss of a frequent itemset X incurred by perturbation can be measured by the *mean square error* (mse) of the perturbed support $T'(X)$:

$$\mathsf{mse}(X) = E[(T'(X) - T(X))^2] = \sigma^2(X) + \beta^2(X)$$

Intuitively, $\mathsf{mse}(X)$ measures the average deviation of perturbed support $T'(X)$ with respect to actual value $T(X)$. A smaller mse implies higher accuracy of the output. Also, it is conceivable that the precision loss should take account of the actual support. The same mse may indicate sufficient accuracy for an itmeset with large support, but may render the output of little value for another itemset with lower support. Therefore, we have the following precision metric:

Definition 5.1. (Precision Degradation) For each frequent itemset X, its precision degradation, denoted by $\mathsf{pred}(X)$, is defined as the relative mean squared error of $T'(X)$:

$$\mathsf{pred}(X) = \frac{E[(T'(X) - T(X))^2]}{T^2(X)} = \frac{\sigma^2(X) + \beta^2(X)}{T^2(X)}.$$

5.2. Privacy Measure

Distorting the original supports of frequent itemsets is only a part of the story, it is necessary to ensure that the distortion cannot be filtered out. Hence, one needs to consider the adversary's power in estimating the real supports of vulnerable patterns through the protection.

Without loss of generality, assume that the adversary desires to estimate the support of pattern P of the form $I \oplus (\overline{J \ominus I})$, and has full access to the sanitized support $T'(X)$ of all $X \in \mathcal{X}_I^J$. Let $T''(P)$ represent the adversary's estimation regarding $T(P)$. The privacy protection should be measured by the error of $T''(P)$. Following let us discuss such estimation from the adversary's perspective. Along the discussion, we will show how various prior knowledge possessed by the adversary may impact the estimation.

Recall that $T(p)$ is estimated following the inclusion-exclusion principle: $T(p) = \sum_{X \in \mathcal{X}_I^J} (-1)^{|X \ominus I|} T(X)$. From the adversary's view, each support $T(X)(X \in \mathcal{X}_I^J)$ is now a random variable; therefore $T(P)$ is also a random variable. Thus, the estimation accuracy of $T''(P)$ with respect to $T(P)$ (by the adversary) can be measured by the mean square error, defined as $\mathsf{mse}(P) = E[(T(P) - T''(P))^2]$. We consider the worst case (the best case for the adversary) wherein $\mathsf{mse}(P)$ is minimized, and define the privacy guarantee based on this lower bound. Intuitively, larger $\min \mathsf{mse}(P)$ indicates more significant estimation error by the adversary, that is, better privacy protection. Also it is noted that the privacy guarantee should account for actual support $T(P)$: if $T(P)$ is close to zero, trivial variance makes it hard for the adversary to infer if pattern P exists. Such "zero-indistinguishability" decreases as $T(P)$ grows. Therefore, we introduce the following privacy metric for vulnerable pattern P.

Definition 5.2. (PRIVACY GUARANTEE) For each vulnerable pattern P, its privacy guarantee, denoted by $\mathsf{prig}(P)$, is defined as its *minimum relative estimation error* (by the adversary):

$$\mathsf{prig}(P) = \frac{\min \mathsf{mse}(P)}{T^2(P)}.$$

In the following, we show how various assumptions regarding the adversary's prior knowledge impact this privacy guarantee. We start the analysis by considering each itemset independently, then take account of the interrelations among them.

Prior Knowledge 5.3. The adversary may have full knowledge regarding the applied perturbation, including its distribution and parameters.

In our case, the parameter of r_X specifies the interval $[l_X, r_X]$ from which the random variable r_X is drawn; therefore, from the adversary's view, of each $X \in \mathcal{X}_I^J$, its actual support $T(X) = T'(X) - r_X$, is a random variable following a discrete uniform distribution over interval $[l'_X, u'_X]$, where $l'_X = T'(X) - u_X$, $u'_X = T'(X) - l_X$ and has expectation $T'(X) - (l_X + u_X)/2$ and variance $\sigma^2(X)$. Recalling that $|r_X| \ll T(X)$, this is a bounded distribution over positive integers. Given the expectation of each $T(X)$, we have the following theorem that gives the lower bound of $\mathsf{mse}(P)$.

THEOREM 5.4. *Given the distribution $f(x)$ of a random variable x, the mean square error of an estimator e of x, $\mathsf{mse}(e) = \int_{-\infty}^{\infty}(x-e)^2 f(x)dx$ reaches its minimum value $Var[x]$, when $e = E[x]$.*

PROOF (THEOREM 5.4). We have the following derivation:

$$\begin{aligned}\mathsf{mse}(x) &= \int_{-\infty}^{\infty}(x-e)^2 f(x)\,\mathrm{d}x \\ &= E[x^2] + e^2 - 2e \cdot E[x] \\ &= (e - E[x])^2 + Var[x].\end{aligned}$$

Hence, $\mathsf{mse}(e)$ is minimized when $e = E[x]$. □

Therefore, $\mathsf{mse}(P)$ is minimized when $T''(P) = E[T(P)]$, which is the best guess that the adversary can achieve (note that the optimality is defined in terms of average estimation error, not the semantics, e.g., $E[T(P)]$ is possibly negative). In this case, the lowest estimation error is reached as $Var[T(P)]$.

In the case that each itemset is considered independently, the fact that $T(p)$ is a linear combination of all involved $T(X)$ implies that $Var[T(p)]$ can be approximated by the sum of the variance of all involved $T(X)$, that is, $\min \mathsf{mse}(p) = \sum_{X \in \mathcal{X}_I^J} \sigma^2(X)$.

Prior Knowledge 5.5. The supports of different frequent itemsets are interrelated by a set of inequalities, derived from the inclusion-exclusion principle.

Here, we take into consideration the dependency among the involved itemsets. As we have shown, each itemset X is associated with an interval $[l'_X, u'_X]$ containing its possible support. Given such itemset-interval pairs, the adversary may attempt to apply these inequalities to tighten the intervals, and thus obtaining better estimation regarding the support. Concretely, this idea can be formalized in the entailment problem [Calders 2004]:

Definition 5.6 (ENTAILMENT). A set of itemset-interval pairs $\mathcal{C}$ entail a constraint $T(X) \in [l_X, u_X]$, denoted by $\mathcal{C} \models T(X) \in [l_X, u_X]$ if every database $\mathcal{D}$ that satisfies $\mathcal{C}$, also satisfies $T(X) \in [l_X, u_X]$. The entailment is tight if for every smaller interval

$[l'_X, u'_X] \subset [l_X, u_X]$, $\mathcal{C} \not\models T(X) \in [l'_X, u'_X]$, that is, $[l_X, u_X]$ is the best interval that can be derived for $T(X)$ based on $\mathcal{C}$.

Clearly, the goal of the adversary is to identify the tight entailment for each $T(X)$ based on the rest; however, we have the following complexity result.

THEOREM 5.7. *Deciding whether $T(X) \in [l_X, u_X]$ is entailed by a set of itemset-interval pairs $\mathcal{C}$ is DP-Complete.*

PROOF (THEOREM 5.7-SKETCH). Deciding whether $\mathcal{C} \models T(X) \in [l_X, u_X]$ is equivalent to the entailment problem in the context of probabilistic logic programming with conditional constraints [Lukasiewicz 2001], which is proved to be DP-Complete. □

This theorem indicates that it is hard to leverage the dependency among the involved itemsets to improve the estimation regarding each individual itemset; therefore, we can approximately consider the supports of frequent itemsets as independent variables in measuring the adversary's power. The privacy guarantee $\mathsf{prig}(P)$ can thus be expressed as $\mathsf{prig}(P) = \sum_{X \in \mathcal{X}_I^J} \sigma^2(X) / T^2(P)$.

Prior Knowledge 5.8. The adversary may have access to other forms of prior knowledge, e.g., published statistics of the dataset, samples of a similar dataset, or supports of the top-k frequent itemsets, etc.

All these forms of prior knowledge can be captured by the notion of *knowledge point*: a knowledge point is a specific frequent itemset X, for which the adversary has estimation error less than $\sigma^2(X)$. Note that following Theorem 5.7, the introduction of knowledge points in general does not influence the estimation of other itemsets. Our definition of privacy guarantee can readily incorporate this notion. Concretely, let $\mathcal{K}_I^J$ denote the set of knowledge points in the set of $\mathcal{X}_I^J$, and $\kappa^2(X)$ be the average estimation error of $T(X)$ for $X \in \mathcal{K}_I^J$. We therefore have the refined definition of privacy guarantee.

$$\mathsf{prig}(P) = \frac{\sum_{X \in \mathcal{K}_I^J} \kappa^2(X) + \sum_{X \in \mathcal{X}_I^J \setminus \mathcal{K}_I^J} \sigma^2(X)}{T^2(P)}.$$

Another well-known uncertainty metric is entropy. Both variance and entropy are important and independent measures of privacy protection. However, as pointed out in Hore et al. [2004], variance is more appropriate in measuring individual-centric privacy wherein the adversary is interested in determining the precise value of a random variable. We therefore argue that variance is more suitable for our purpose, since we are aiming at protecting the exact supports of vulnerable patterns.

Prior Knowledge 5.9. The sanitized supports of the same frequent itemsets may be published in consecutive stream windows.

Since our protection is based on independent random perturbation, if the same support is repeatedly perturbed and published in multiple windows, the adversary can potentially improve the estimation by averaging the observed output (the law of large numbers). To block this type of attack, once the perturbed support of a frequent itemset is released, we keep publishing this sanitized value if the actual support remains unchanged in consecutive windows.

Discussion. In summary, the effectiveness of BUTTERFLY* is evaluated in terms of its resilience against both intrawindow and interwindow inference over stream mining output. We note three key implications.

First, the uncertainty of involved frequent itemsets is accumulated in the inferred vulnerable patterns. Moreover, more complicated inferencing attacks (i.e., harder to be detected) face higher uncertainty.

Second, the actual supports of vulnerable patterns are typically small (only a unique or less than K records match vulnerable patterns), and thus adding trivial uncertainty can make it hard to tell the existence of such patterns in the dataset.

Third, interwindow inference follows a two-stage strategy, that is, first deducing the transition between contingent windows, then inferring the vulnerable patterns. The uncertainty associated with both stages provides even stronger protection.

5.3. Trade-Off between Precision and Privacy

In our BUTTERFLY* framework, the trade-off between privacy protection and output utility can be flexibly adjusted by the setting of variance and bias for each frequent itemset. Specifically, variance controls the overall balance between privacy and utility, while bias gives a finer control over the balance between precision and other utility metrics, as we will show later. Here, we focus on the setting of variance. Intuitively, smaller variance leads to higher output precision, however also decreases the uncertainty of inferred vulnerable patterns, thus lower privacy guarantee.

To ease the discussion, we assume that all the frequent itemsets are associated with the same variance σ^2 and bias β. In Section 6 when semantic constraints are taken into account, we will lift this simplification, and consider more sophisticated settings.

Let C denote the minimum support for frequent itemsets. From the definition of precision metrics, it can be derived that for each frequent itemset X, its precision degradation $\mathsf{pred}(X) \leq (\sigma^2 + \beta^2)/C^2$, because $T(X) \geq C$. Let $P_1(C) = (\sigma^2 + \beta^2)/C^2$, i.e., the upper bound of precision loss for frequent itemsets. Meanwhile, for a vulnerable pattern $P = I(\overline{J \setminus I})$, it can be proved that its privacy guarantee $\mathsf{prig}(P) \geq (\sum_{X \in \mathcal{X}_I^J} \sigma^2)/K^2 \geq (2\sigma^2)/K^2$, because $T(P) \leq K$ and the inference involves at least two frequent itemsets. Let $P_2(C, K) = (2\sigma^2)/K^2$, i.e., the lower bound of privacy guarantee for inferred vulnerable patterns.

P_1 and P_2 provide convenient representation to control the trade-off. Specifically, setting an upper bound ϵ over P_1 guarantees sufficient accuracy of the reported frequent itemsets; while setting a lower bound δ over P_2 provides enough privacy protection for the vulnerable patterns. One can thus specify the precision-privacy requirement as a pair of parameters (ϵ, δ), where $\epsilon, \delta > 0$. That is, the setting of β and σ should satisfy $P_1(C) \leq \epsilon$ and $P_2(C, K) \geq \delta$, as

$$\sigma^2 + \beta^2 \leq \epsilon C^2 \tag{1}$$

$$\sigma^2 \geq \delta K^2/2. \tag{2}$$

To make both inequalities hold, it should be satisfied that $\epsilon/\delta \geq K^2/(2C^2)$. The term ϵ/δ is called the *precision-privacy ratio* (PPR). When precision is a major concern, one can set PPR as its minimum value $K^2/(2C^2)$ for given K and C, resulting in the minimum accuracy loss for given privacy requirement. The minimum PPR also implies that $\beta = 0$ and the two parameters ϵ and δ are coupled. We refer to the perturbation scheme with the minimum PPR as the basic BUTTERFLY*.

6. OPTIMIZED BUTTERFLY*

The basic BUTTERFLY* scheme attempts to minimize the accuracy loss of individual frequent itemsets, without taking account of their semantic relationships. Although easy to implement and resilient against attacks, this simple scheme may easily violate these semantic constraints directly related to the specific applications of the mining output, thus decreasing the overall utility of the results. In this section, we refine

this basic scheme by taking semantic constraints into our map and develop constraint-aware BUTTERFLY* schemes. For given precision and privacy requirement, the optimized scheme preserves the utility-relevant semantics to the maximum extent.

In this work, we specifically consider two types of constraints, *absolute ranking* and *relative frequency*. By absolute ranking, we refer to the order of frequent itemsets according to their supports. In certain applications, users pay special attention to the ranking of patterns, rather than their actual supports, for instance, querying the top-ten most popular purchase patterns. By relative frequency, we refer to the pairwise ratio of the supports of frequent itemsets. In certain applications, users care more about the ratio of two frequent patterns, instead of their absolute supports, for instance, computing the confidence of association rules.

To facilitate the presentation, we first introduce the concept of *frequency equivalent class* (FEC).

Definition 6.1. (FREQUENT EQUIVALENT CLASS). A frequent equivalent class (FEC) is a set of frequent itemsets that feature equivalent support. Two itemsets I, J belong to the same FEC if and only if $T(I) = T(J)$. The support of a FEC fec, $T(fec)$, is defined as the support of any of its member.

A set of frequent itemsets can be partitioned into a set of disjoint FECs, according to their supports. Also note that a set of FECs are a strictly ordered sequence: we define two FECs fec_i and fec_j as $fec_i < fec_j$ if $T(fec_i) < T(fec_j)$. Following we assume that the given set of FECs $\mathcal{FEC}$ are sorted according to their supports, i.e., $T(fec_i) < T(fec_j)$ for $i < j$.

Example 6.2. In our running example as shown in Fig. 3, given $C = 4$, there are three FECs, $\{cd\}$, $\{ac, bc\}$, $\{c\}$, with support 4, 5, and 8, respectively.

Apparently, to comply with the constraints of absolute ranking or relative frequency, the equivalence of itemsets in a FEC should be preserved to the maximum extent in the perturbed output. Thus, in our constraint-aware schemes, the perturbation is performed at the level of FECs, instead of specific itemsets.

We argue that this change does not affect the privacy guarantee as advertised, provided the fact that the inference of a vulnerable pattern involves at least two frequent itemsets with different supports, i.e., at least two FECs. Otherwise assuming that the involved frequent itemsets belong to the same FEC, the inferred vulnerable pattern would have support zero, which is a contradiction. Therefore, as long as each FEC is associated with uncertainty satisfying Equation (2), the privacy preservation is guaranteed to be above the advertised threshold.

6.1. Order Preservation

When the order of itemset support is an important concern, the perturbation of different FECs cannot be uniform, since that would easily invert the order of two itemsets, especially when their supports are close. Instead, we need to maximally separate the perturbed supports of different FECs, under the given constraints of Equation (1) and Equation (2). To capture this intuition, we first introduce the concept of *uncertainty region* of FEC.

Definition 6.3. (UNCERTAINTY REGION) The uncertainty region of FEC fec is the set of possible values of its perturbed support: $\{x | Pr(T'(fec) = x) > 0\}$.

For instance, when adding to FEC fec a random variable drawn from a discrete uniform distribution over interval $[a, b]$, the uncertainty region is all the integers within interval $[a+T(fec), b+T(fec)]$. To preserve the order of FECs with overlapping

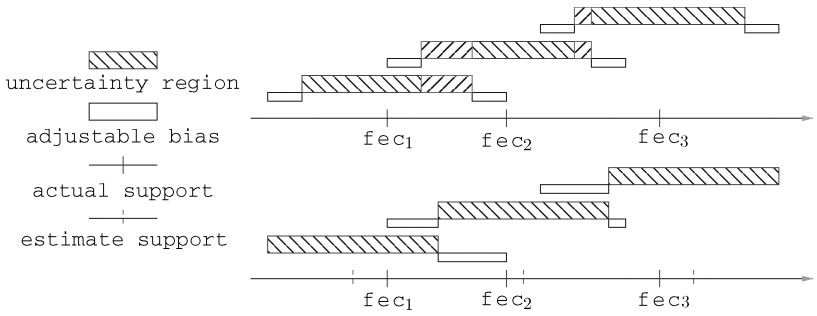

Fig. 4. Adjusting bias to minimize overlapping uncertainty regions.

uncertainty regions, we maximally reduce their intersection, by adjusting their bias setting.

Example 6.4. As shown in Figure 4, three FECs have intersected uncertainty regions, and their initial biases are all zero. After adjusting the biases properly, they share no overlapping uncertainty region; thus, the order of their supports is preserved in the perturbed output.

Note that the order is not guaranteed to be preserved if certain FECs still have overlapping regions after adjustment, due to the constraints of given precision and privacy parameters (ϵ, δ). We intend to achieve the maximum preservation under the given requirement.

Minimizing Overlapping Uncertainty Region. Below we formalize the problem of order preservation. Without loss of generality, consider two FECs fec_i, fec_j with $T(fec_i) < T(fec_j)$. To simplify the notation, we use the following short version: let $t_i = T(fec_i)$, $t_j = T(fec_j)$, t'_i and t'_j be their perturbed supports, and β_i and β_j the bias setting, respectively.

The order of fec_i and fec_j can be possibly inverted if their uncertainty regions intersect; that is, $Pr[t'_i \geq t'_j] > 0$. We attempt to minimize this inversion probability $Pr[t'_i \geq t'_j]$ by adjusting β_i and β_j. This adjustment is not arbitrary, constrained by the precision and privacy requirement. We thus introduce the concept of *maximum adjustable bias*:

Definition 6.5. (MAXIMUM ADJUSTABLE BIAS) For each FEC fec, its bias is allowed to be adjusted within the range of $[-\beta_{max}(fec), \beta_{max}(fec)]$, $\beta_{max}(fec)$ is called the maximum adjustable bias. For given ϵ and δ, it is defined as

$$\beta_{max}(fec) = \left\lfloor \sqrt{\epsilon T^2(fec) - \delta K^2/2} \right\rfloor$$

derived from Equation (1) and Equation (2).

Wrapping up the discussion above, the problem of preserving absolute ranking can be formalized as: given a set of FECs $\{fec_1, \ldots, fec_n\}$, find the optimal bias setting for each FEC fec within its maximum adjustable bias $[-\beta_{max}(fec), \beta_{max}(fec)]$ to minimize the sum of pairwise inversion probability: min $\sum_{i<j} Pr[t'_i \geq t'_j]$.

Exemplifying with discrete uniform distribution, we now show how to compute $Pr[t'_i \geq t'_j]$. Consider a discrete uniform distribution over interval $[a, b]$, with $\alpha = b - a$ as the interval length. The variance of this distribution is given by $\sigma^2 = [(\alpha+1)^2 - 1]/12$.

According to Equation (2), we have $\alpha = \lceil\sqrt{1+6\delta K^2}\rceil - 1$. Let d_{ij} be the distance of their estimators $e_i = t_i + \beta_i$ and $e_j = t_j + \beta_j$[2], that is, $d_{ij} = e_j - e_i$.

The intersection of uncertainty regions of fec_i and fec_j is a piece-wise function, with four possible types of relationships: 1) $e_i < e_j$, fec_i and fec_j do not overlap; 2) $e_i \leq e_j$, fec_i and fec_j intersect; 3) $e_i > e_j$, fec_i and fec_j intersect; 4) $e_i > e_j$, fec_i and fec_j do not overlap. Correspondingly, the inversion probability $Pr[t'_i \geq t'_j]$ is computed as follows:

$$Pr[t'_i \geq t'_j] = \begin{cases} 0 & d_{ij} \geq \alpha + 1 \\ \frac{(\alpha+1-d_{ij})^2}{2(\alpha+1)^2} & 0 < d_{ij} < \alpha + 1 \\ 1 - \frac{(\alpha+1+d_{ij})^2}{2(\alpha+1)^2} & -\alpha - 1 < d_{ij} \leq 0 \\ 1 & d_{ij} \leq -\alpha - 1. \end{cases}$$

Following we use C_{ij} (or C_{ji}) to denote $Pr[t'_i \geq t'_j]$, the cost function of the pair of fec_i and fec_j. The formulation of C_{ij} can be considerably simplified based on the next key observation: for any pair of fec_i and fec_j with $i < j$, the solution of the optimization problem contains no configuration of $d_{ij} < 0$, as proved in the next lemma.

LEMMA 6.6. *In the optimization solution of min* $\sum_{i<j} C_{ij}$*, any pair of FECs* fec_i *and* fec_j *with* $i < j$ *must have* $e_i \leq e_j$*, that is,* $d_{ij} \geq 0$*.*

PROOF (LEMMA 6.6). Assume that the estimators $\{e_1, \ldots, e_n\}$ corresponding to the optimal setting, and there exists a pair of FECs, fec_i and fec_j with $i < j$ and $e_i > e_j$. By switching their setting, that is, let e'_i (β'_i), and $e'_j(\beta'_j)$ be their new setting, and $e'_i = e_j$, and $e'_j = e_i$, the overall cost is reduced, because $\sum_{k\neq i,j} C_{ki} + C_{kj}$ remains the same, and C_{ij} is reduced, which is contradictory to the optimality assumption.

We need to prove that the new setting is feasible, that is $|\beta'_i| \leq \beta_{max}(fec_i)$ and $|\beta'_j| \leq \beta_{max}(fec_j)$. Here, we prove the feasibility of β'_i, and a similar proof applies to β'_j. First, according to the assumption, we know that

$$e_j = t_j + \beta_j < t_i + \beta_i = e_i \quad \text{and} \quad t_i < t_j;$$

therefore, we have the next fact

$$\beta'_i = \beta_j + t_j - t_i < \beta_i \leq \beta_{max}(fec_i).$$

We now just need to prove that $\beta'_i \geq -\beta_{max}(fec_i)$, equivalent to $\beta_j + t_j - t_i \geq -\beta_{max}(fec_i)$, which is satisfied if

$$t_j - t_i \geq \beta_{max}(fec_j) - \beta_{max}(fec_i).$$

By substituting the maximum adjustable bias with its definition, and considering the fact $\epsilon \leq 1$, this inequality can be derived. □

Therefore, it is sufficient to consider the case $d_{ij} \geq 0$ for every pair of fec_i and fec_j when computing $Pr[t'_i \geq t'_j]$. The optimization problem is thus simplified as: $\sum_{i<j}(\alpha + 1 - d_{ij})^2$.

One flaw of the discussion so far is that we treat all FECs uniformly without considering their characteristics, that is, the number of frequent itemsets within each FEC. The inversion of FECs containing more frequent itemsets is more serious than that of FECs with less members. Quantitatively, let s_i be the number of frequent itemsets in FEC fec_i, the inversion of two FECs fec_i and fec_j means the order of $(s_i + s_j)$ itemsets are disturbed.

[2]Following we will use the setting of bias and estimator exchangeably.

Therefore, our aim now is to solve the weighted optimization problem:

$$\begin{aligned} \min \quad & \sum_{i<j}(s_i+s_j)(\alpha+1-d_{ij})^2 \\ \text{s.t.} \quad & d_{ij}=\begin{cases}\alpha+1 & e_j-e_i\geq\alpha+1\\ e_j-e_i & e_j-e_i<\alpha+1\end{cases} \\ & \forall i<j, e_i\leq e_j \\ & \forall i, e_i\in\mathbb{Z}^+, |e_i-t_i|\leq\beta_{max}(fec_i). \end{aligned}$$

This is a quadratic integer programming (QIP) problem, with piece-wise cost function. In general, QIP is NP-Hard, even without integer constraints [Vavasis 1990]. This problem can be solved by first applying quadratic optimization techniques, such like simulated annealing, and then using random rounding techniques to impose the integer constraints. However, we are more interested in online algorithms that can flexibly trade between efficiency and accuracy. Following we present one such solution based on dynamic programming.

A Near-Optimal Solution. By relaxing the constraint that $\forall i<j, e_i\leq e_j$ to $e_i<e_j$, we obtain the following key properties: (i) the estimators of all the FECs are in strict ascending order, that is, $\forall i<j, e_i<e_j$; (ii) the uncertainty regions of all the FECs have the same length α. Each FEC can thus intersect with at most α of its previous ones. These properties lead to an *optimal substructure*, crucial for our solution.

LEMMA 6.7. *Given that the biases of the last α FECs $\{fec_{n-\alpha+1}:fec_n\}$[3] are fixed as $\{\beta_{n-\alpha+1}:\beta_n\}$, and $\{\beta_1:\beta_{n-\alpha}\}$ are optimal w.r.t. $\{fec_1:fec_n\}$, then for given $\{\beta_{n-\alpha}:\beta_{n-1}\}$, $\{\beta_1:\beta_{n-\alpha-1}\}$ must be optimal w.r.t. $\{fec_1:fec_{n-1}\}$.*

PROOF (LEMMA 6.7). Suppose that there exists a better setting $\{\beta'_1:\beta'_{n-\alpha-1}\}$ leading to lower cost w.r.t. $\{fec_1:fec_{n-1}\}$. Since fec_n does not intersect with any of $\{fec_1:fec_{n-\alpha-1}\}$, the setting of $\{\beta'_1:\beta'_{n-\alpha-1},\beta_{n-\alpha}:\beta_n\}$ leads to lower cost w.r.t. $\{fec_1:fec_n\}$, contradictory to our optimality assumption. □

Based on this optimal substructure, we propose a dynamic programming solution, which adds FECs sequentially according to their order. Let $C_{n-1}(\beta_{n-\alpha}:\beta_{n-1})$ represent the minimum cost that can be achieved by adjusting FECs $\{fec_1:fec_{n-\alpha-1}\}$ with the setting of the last α FECs fixed as $\{\beta_{n-\alpha}:\beta_n\}$. When adding fec_n, the minimum cost $C_n(\beta_{n-\alpha+1}:\beta_n)$ is computed using the rule:

$$C_n(\beta_{n-\alpha+1}:\beta_n)=\min_{\beta_{n-\alpha}}C_{n-1}(\beta_{n-\alpha}:\beta_{n-1})+\sum_{i=n-\alpha}^{n-1}(s_i+s_n)(\alpha+1-d_{in})^2$$

The optimal setting is the one with the minimum cost among all the combinations of $\{\beta_{n-\alpha+1}:\beta_n\}$.

Now, let us analyze the complexity of this scheme. Let β^*_{max} denote the upper bound of maximum adjustable biases of all FECs: $\beta^*_{max}=\max_i\beta_{max}(fec_i)$. For each fec, its bias can be chosen from at most $(2\beta^*_{max}+1)$ integers. Computing $C_n(\beta_{n-\alpha+1}:\beta_n)$ for each combination of $\{\beta_{n-\alpha+1}:\beta_n\}$ from $C_{n-1}(\beta_{n-\alpha}:\beta_{n-1})$ takes at most $(2\beta^*_{max}+1)$ steps, and the number of combinations is at most $(2\beta^*_{max}+1)^\alpha$. The time complexity of this scheme is thus bounded by $(2\beta^*_{max}+1)^{\alpha+1}n$, i.e., $O(n)$ where n is the total number of FECs. Meanwhile, the space complexity is also bounded by the number of cost function values needed to be recorded for each FEC, that is, $(2\beta^*_{max}+1)^\alpha$. In addition, at each step, we

[3]In the following we use $\{x_i:x_j\}$ as a short version of $\{x_i,x_{i+1},\ldots,x_j\}$.

Algorithm 1: Order-preserving bias setting

Input: $\{t_i, \beta_{max}(fec_i)\}$ for each $fec_i \in \mathcal{FEC}$, α, γ.
Output: β_i for each $fec_i \in \mathcal{FEC}$.
begin
  /*initialization*/;
  for $\beta_1 = -\beta_{max}(fec_1) : \beta_{max}(fec_1)$ **do**
    $C_1(\beta_1) = 0$;
  for $i = 2 : \gamma$ **do**
    for $\beta_i = -\beta_{max}(fec_i) : \beta_{max}(fec_i)$ **do**
      /*$e_i < e_j$*/;
      if $\beta_i + t_i > \beta_{i-1} + t_{i-1}$ **then**
        $C_i(\beta_1 : \beta_i) = C_{i-1}(\beta_1 : \beta_{i-1}) + \sum_{j=1}^{i-1}(s_j + s_i)(\alpha + 1 - d_{ji})^2$;
  /* dynamic programming */;
  for $i = \gamma + 1 : n$ **do**
    for $\beta_i = -\beta_{max}(fec_i) : \beta_{max}(fec_i)$ **do**
      if $\beta_i + t_i > \beta_{i-1} + t_{i-1}$ **then**
        $C_i(\beta_{i-\gamma+1} : \beta_i) = \min_{\beta_{i-\gamma}} C_{i-1}(\beta_{i-\gamma} : \beta_{i-1}) + \sum_{j=i-\gamma}^{i-1}(s_j + s_i)(\alpha + 1 - d_{ji})^2$;
  /*find the optimal setting*/;
  find the minimum $C_n(\beta_{n-\gamma+1} : \beta_n)$;
  backtrack and output β_i for each $fec_i \in \mathcal{FEC}$;
end

need to keep track of the bias setting for the added FECs so far for each combination, thus $(2\beta^*_{max} + 1)^\alpha(n - \alpha)$ in total.

In practice, the complexity is typically much lower than this bound, given that (i) under the constraint $\forall i < j, e_i < e_j$, a number of combinations are invalid, and (ii) β^*_{max} is an over-estimation of the average maximum adjustable bias.

It is noted that as α or β^*_{max} grows, the complexity increases sharply, even though it is linear in terms of the total number of FECs. In view of this, we develop an approximate scheme that allows trading between efficiency and accuracy. The basic idea is that on adding each FEC, we only consider its intersection with its previous γ FECs, instead of α ones ($\gamma < \alpha$). This approximation is tight when the distribution of FECs is not extremely dense, which is usually the case, as verified by our experiments. Formally,

$$C_n(\beta_{n-\gamma+1} : \beta_n) = \min_{\beta_{n-\gamma}} C_{n-1}(\beta_{n-\gamma} : \beta_{n-1}) + \sum_{i=n-\gamma}^{n-1}(s_i + s_n)(\alpha + 1 - d_{in})^2.$$

Now the complexity is bounded by $(2\beta^*_{max} + 1)^{\gamma+1}n$. By properly adjusting γ, one can control the balance between accuracy and efficiency.

The complete algorithm is sketched in Algorithm 1: one first initializes the cost function for the first γ FECs; then by running the dynamic programming procedure, one computes the cost function for each newly added FEC. The optimal configuration is the one with the global minimum value $C_n(\beta_{n-\gamma+1} : \beta_n)$.

6.2. Ratio Preservation

In certain applications, the relative frequency of the supports of two frequent itemsets carries important semantics, for example, the confidence of association rules. However, the random perturbation may easily render the ratio of the perturbed supports considerably deviate from the original value. Again, we achieve the maximum ratio

preservation by intelligently adjust the bias setting of FECs. First, we formalize the problem of ratio preservation.

Maximizing (k, $1/k$) Probability of Ratio. Consider two FECs fec_i and fec_j with $t_i < t_j$. To preserve the ratio of fec_i and fec_j, one is interested in making the ratio of perturbed supports t'_i/t'_j appear in the proximate area of original value t_i/t_j with high probability, for instance, interval $[k\frac{t_i}{t_j}, \frac{1}{k}\frac{t_i}{t_j}]$, where $k \in (0, 1)$ indicates the tightness of this interval. We therefore introduce the concept of $(k, 1/k)$ probability.

Definition 6.8. ($(k, 1/k)$ PROBABILITY) The $(k, 1/k)$ probability of the ratio of two random variables t'_i and t'_j, $Pr_{(k,1/k)}\left[\frac{t'_i}{t'_j}\right]$ is defined as

$$Pr_{(k,1/k)}\left[\frac{t'_i}{t'_j}\right] = Pr\left[k\frac{t_i}{t_j} \le \frac{t'_i}{t'_j} \le \frac{1}{k}\frac{t_i}{t_j}\right].$$

This $(k, 1/k)$ probability quantitatively describes the proximate region of original ratio t_i/t_j. A higher probability that t'_i/t'_j appears in this region indicates better ratio preservation. We are therefore interested in solving the following optimization problem:

$$\begin{aligned} \max \quad & \sum_{i<j} Pr_{(k,1/k)}\left[\frac{t'_i}{t'_j}\right] \\ \text{s.t} \quad & \forall i, e_i \in \mathbb{Z}^+, |e_i - t_i| \le \beta_{max}(fec_i). \end{aligned}$$

It is not hard to see that in the case of discrete uniform distribution, the $(k, 1/k)$ probability of the ratio of two random variables is a nonlinear piecewise function, that is, a nonlinear integer optimization problem. In general, nonlinear optimization problem is NP-Hard, even without integer constraints. Instead of applying off-the-shelf nonlinear optimization tools, we are more interested in efficient heuristics that can find near-optimal configurations with linear complexity in terms of the number of FECs. Following, we present one such scheme that performs well in practice.

A Near-Optimal Solution. We construct our bias setting scheme based on Markov's inequality. To maximize the $(k, 1/k)$ probability $Pr\left[k\frac{t_i}{t_j} \le \frac{t'_i}{t'_j} \le \frac{1}{k}\frac{t_i}{t_j}\right]$, we can alternatively minimize $Pr\left[\frac{t'_i}{t'_j} \ge \frac{1}{k}\frac{t_i}{t_j}\right] + Pr\left[\frac{t'_j}{t'_i} \ge \frac{1}{k}\frac{t_j}{t_i}\right]$. From Markov's inequality, we know that $Pr\left[\frac{t'_i}{t'_j} \ge \frac{1}{k}\frac{t_i}{t_j}\right]$ is bounded by

$$Pr\left[\frac{t'_i}{t'_j} \ge \frac{1}{k}\frac{t_i}{t_j}\right] \le \frac{E\left[\frac{t'_i}{t'_j}\right]}{\frac{1}{k}\frac{t_i}{t_j}} = k\frac{t_j}{t_i}E\left[\frac{t'_i}{t'_j}\right].$$

The maximization of the $(k, 1/k)$ probability of t'_i/t'_j is therefore simplified as the following expression (k is omitted since it does not affect the optimization result):

$$\min \frac{t_j}{t_i}E\left[\frac{t'_i}{t'_j}\right] + \frac{t_i}{t_j}E\left[\frac{t'_j}{t'_i}\right]. \tag{3}$$

The intuition here is that neither expectation $\frac{t_j}{t_i}E\left[\frac{t'_i}{t'_j}\right]$ nor $\frac{t_i}{t_j}E\left[\frac{t'_j}{t'_i}\right]$ should deviate wildly from one.

According to its definition, the expectation of $\frac{t'_i}{t'_j}$, $E\left[\frac{t'_i}{t'_j}\right]$, is computed as

$$E\left[\frac{t'_i}{t'_j}\right] = \frac{1}{(\alpha+1)^2}\sum_{e_j-\alpha/2}^{e_j+\alpha/2}\frac{1}{t'_j}\sum_{e_i-\alpha/2}^{e_i+\alpha/2} t'_i = \frac{e_i}{\alpha+1}\left(H_{e_j+\frac{\alpha}{2}} - H_{e_j-\frac{\alpha}{2}}\right),$$

where H_n is the nth harmonic number. It is known that $H_n = \ln n + \Theta(1)$; thus,

$$E\left[\frac{t'_i}{t'_j}\right] \approx \frac{e_i}{\alpha+1}\ln\frac{e_j+\alpha/2}{e_j-\alpha/2} = \frac{e_i}{\alpha+1}\ln\left(1+\frac{\alpha}{e_j-\alpha/2}\right). \tag{4}$$

This form is still not convenient for computation. We are therefore looking for a tight approximation for the logarithm part of the expression. It is known that $\forall x, y \in \mathbb{R}^+$, $(1+x/y)^{y+x/2}$ is a tight upper bound for e^x. We have the following approximation by applying this bound: $1+\alpha/(e_j-\alpha/2) = e^{\frac{\alpha}{e_j-\alpha/2+\alpha/2}} = e^{\frac{\alpha}{e_j}}$.

Applying this approximation to computing $E\left[\frac{t'_i}{t'_j}\right]$ in Equation (4), it is derived that

$$E\left[\frac{t'_i}{t'_j}\right] = \frac{e_i}{\alpha+1}\ln e^{\frac{\alpha}{e_j}} = \frac{\alpha}{\alpha+1}\frac{e_i}{e_j}.$$

The optimization of Equation (3) is thus simplified as

$$\min \frac{t_j}{t_i}\frac{e_i}{e_j} + \frac{t_i}{t_j}\frac{e_j}{e_i}. \tag{5}$$

Assuming that e_i is fixed, by differentiating Equation (5) w.r.t. e_j, and setting the derivative as 0, we get the solution of e_j as $e_j/e_i = t_j/t_i$, that is, $\beta_j/\beta_i = t_j/t_i$.

Following this solution is our bottom-up bias setting scheme: for each FEC fec_i, its bias β_i should be set in proportion to its support t_i. Note that the larger $(t_i+\beta_i)$ compared with α, the more accurate the applied approximation; hence, β_i should be set as its maximum possible value.

Algorithm 2: Ratio-preserving bias setting

Input: $\{t_i\}$ for each $fec_i \in FEC$, ϵ, δ, K.
Output: β_i for each $fec_i \in FEC$.
begin
  `/* setting of the minimum FEC */`;
  set $\beta_1 = \lfloor\sqrt{\epsilon t_1^2 - \delta K^2/2}\rfloor$;
  `/* bottom-up setting */`;
  for $i = 2 : n$ **do**
    set $\beta_i = \lfloor\beta_{i-1}\frac{t_i}{t_{i-1}}\rfloor$;
end

Algorithm 2 sketches the bias setting scheme: one first sets the bias of the minimum FEC fec_1 as its maximum $\beta_{max}(fec_1)$, and for each remaining FEC fec_i, its bias β_i is set in proportion to t_i/t_{i-1}. In this scheme, for any pair of fec_i and fec_j, their biases satisfy $\beta_i/\beta_j = t_i/t_j$. Further, we have the following lemma to prove the feasibility of this scheme. By feasibility, we mean that for each FEC fec_i, β_i falls within the allowed interval $[-\beta_{max}(fec_i), \beta_{max}(fec_i)]$.

LEMMA 6.9. *For two FECs fec_i and fec_j with $t_i < t_j$, if the setting of β_i is feasible for fec_i, then the setting $\beta_j = \beta_i \frac{t_j}{t_i}$ is feasible for fec_j.*

PROOF (LEMMA 6.9). Given that $0 < \beta_i \leq \beta_{max}(fec_i)$, then according to the definition of maximum adjustable bias, β_j has the following property

$$\beta_j = \beta_i \frac{t_j}{t_i} \leq \beta_{max}(fec_i)\frac{t_j}{t_i} = \left\lfloor \sqrt{\epsilon t_i^2 - \frac{\delta K^2}{2}} \right\rfloor \frac{t_j}{t_i}$$
$$= \left\lfloor \sqrt{\epsilon t_j^2 - \frac{\delta K^2}{2}\frac{t_j^2}{t_i^2}} \right\rfloor \leq \left\lfloor \sqrt{\epsilon t_j^2 - \frac{\delta K^2}{2}} \right\rfloor = \beta_{max}(fec_j) \qquad \square$$

Thus if β_1 is feasible for fec_1, β_i is feasible for any fec_i with $i > 1$, since $t_i > t_1$.

6.3. A Hybrid Scheme

While order-preserving and ratio-preserving bias settings achieve the maximum utility at their ends, in certain applications wherein both semantic relationships are important, it is desired to balance the two quality metrics in order to achieve the overall optimal quality.

We thus develop a hybrid bias setting scheme that takes advantage of the two schemes, and allows to flexibly adjust the trade-off between the two factors. Specifically, for each FEC fec, let $\beta_{op}(fec)$ and $\beta_{rp}(fec)$ denote its bias setting obtained by the order-preserving and frequency-preserving schemes, respectively. We have the following setting based on a linear combination:

$$\forall fec \in \mathcal{FEC} \quad \beta(fec) = \lambda\beta_{op}(fec) + (1-\lambda)\beta_{rp}(fec).$$

The parameter λ is a real number within the interval $[0, 1]$, which controls the trade-off between the two quality metrics. Intuitively, a larger λ tends to indicate more importance over order information, but less over ratio information, and vise versa. Particularly, the order-preserving and ratio-preserving schemes are the special cases when $\lambda = 1$ and 0, respectively.

7. EXTENSION TO OTHER DISTRIBUTION

In the section, we intend to study the impact of the perturbation distribution over the quality of privacy protection and (multi-)utility preservation. It will be revealed shortly that while uniform distribution leads to the best privacy protection, it may not be optimal in terms of other utility metrics.

7.1. Privacy and Precision

Recall that the precision degradation of a frequent itemset X is given by $\mathsf{pred}(X) = [\sigma^2(X) + \beta^2(X)]/T^2(X)$, while the privacy guarantee of a vulnerable pattern P of the form $I \oplus (\overline{J \ominus I})$ is given by $\mathsf{prig}(P) = \sum_{X \in \mathcal{X}_I^J} \sigma^2(X)/T^2(P)$. Clearly, if two perturbation distributions share the same bias and variance, they offer the same amount of precision preservation for X and privacy guarantee for P. Next we focus our discussion on order and ratio preservation.

7.2. Order Preservation

For ease of presentation, we assume that the perturbation added to the support of each FEC is drawn from a homogeneous distribution with probability density function (PDF) $f(\cdot)$, plus a FEC-specific bias. Following the development in Section 6, we attempt to minimize the sum of pair-wise inversion probability: $\min \sum_{i<j} Pr[t'_i \geq t'_j]$, by finding

the optimal bias setting for each FEC fec_i within its maximum adjustable bias β_i^{max}. Note that $\beta_i^{max} = \lfloor\sqrt{\epsilon t_i^2 - \delta K^2/2}\rfloor$ is solely determined by the precision and privacy requirement (ϵ, δ), irrespective of the underlying distribution $f(\cdot)$.

For general distribution $f(\cdot)$, the inversion probability $Pr[t_i' \geq t_j']$ is defined as:

$$\begin{aligned} Pr[t_i' \geq t_j'] &= \int_{-\infty}^{+\infty} f(x_j - \beta_j) \int_{x_j + t_j - t_i}^{+\infty} f(x_i - \beta_i)\,\mathrm{d}x_i\,\mathrm{d}x_j \\ &= \int_{-\infty}^{+\infty} f(x_j - \beta_j)(1 - F(x_j + t_j - t_i - \beta_i))\,\mathrm{d}x_j \\ &= \int_{-\infty}^{+\infty} F(x_j - \beta_j) f(x)\,\mathrm{d}x \\ &= \mathbb{E}[F(x - (t_j - t_i + \beta_j - \beta_i))] \triangleq \mathbb{E}[F(x - d_{ij})], \end{aligned}$$

where $F(\cdot)$ is the cumulative distribution function (CDF) of $f(\cdot)$, and d_{ij} denotes the distance of the estimators of t_i and t_j. Clearly, $Pr[t_i' \geq t_j']$ is the expectation of the CDF after shifting transformation, which is a continuous function of d_{ij} for unbounded distributions, for instance, normal distribution, and possibly piecewise function for discrete distributions, for instance, uniform distribution; thus, no closed form of $Pr[t_i' \geq t_j']$ is available for general $f(\cdot)$.

It is noted that Lemma 6.6 makes no specific assumption regarding the underlying distribution, and thus holding for any distribution $f(\cdot)$; therefore, under the optimal bias setting, for any $i < j$, it must hold that $d_{ij} \geq 0$. Furthermore, let s_i be the number of frequent itemsets within FEC fec_i. Taking into consideration the weight of each FEC, the optimization problem is formulated as:

$$\begin{aligned} \min \quad & \sum_{i<j} (s_i + s_j)\mathbb{E}[F(x - e_j + e_i)] \\ \text{s.t.} \quad & \forall i, |e_i - t_i| \leq \beta_i^{max} \\ & \forall i < j, e_i \leq e_j. \end{aligned}$$

This is in general a nonlinear programming problem, with the difficulty of optimization mainly depending on the concrete form of the underlying distribution. For example, in the case of uniform distribution, it becomes a quadratic programming problem (QIP); while in the case of Rademacher distribution, it becomes a piece wise minimization problem. Tailored optimization tools are therefore necessary for different distributions, which is beyond the scope of this work. Here, instead, we attempt to explore the interplay between the statistical properties of perturbation noise and the distribution of itemset supports.

From Figure 4, it is noticed that two contributing factors affect the inversion probability $Pr[t_i' \geq t_j']$, namely, the length of the uncertainty regions of fec_i and fec_j, and the average probability mass (per unit length) of fec_i and fec_j in the intersected region. Intuitively, if the uncertainty region length is large, the average probability mass distributed over the region tends to be small, but the possibility that two uncertainty regions intersect is high; meanwhile, if the uncertainty region length is small, they will have less chance to intersect, but the probability mass density in the intersected region will be large if they overlap. Here, we consider two representative distributions featuring small and large uncertainty regions for fixed variance σ^2.

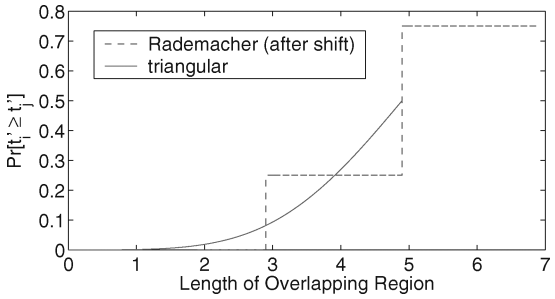

Fig. 5. Trade-off between uncertainty region length (intersection possibility) and probability mass density ($\sigma = 1$).

—Rademacher distribution. Its probability mass function $f(\cdot)$ is defined as:

$$f(x) = \begin{cases} 1/2 & x = -\sigma \text{ or } \sigma \\ 0 & \text{otherwise.} \end{cases}$$

The uncertainty region length is 2σ.

—triangular distribution. Its probability density function $f(\cdot)$ is given by:

$$f(x) = \begin{cases} (\sqrt{6}\sigma + x)/(6\sigma^2) & x \in [-\sqrt{6}\sigma, 0] \\ (\sqrt{6}\sigma - x)/(6\sigma^2) & x \in [0, \sqrt{6}\sigma] \\ 0 & \text{otherwise.} \end{cases}$$

The uncertainty region length is $2\sqrt{6}\sigma$.

Now, exemplifying with these two distributions, we attempt to explore the trade-off of uncertainty region length and probability mass density that contribute to the inversion probability. Figure 5 illustrates $Pr[t_i' \geq t_j']$ as a function of the intersection length of two uncertainty regions. To reflect the difference of uncertainty region length sof the two distributions, we horizontally shift the inversion probability of Rademacher distribution $2(\sqrt{6} - 1)\sigma$ units. It is noted that there is no clear winner over the entire interval; rather, each distribution demonstrates superiority over the other in certain regions. For example, when the intersection length is small, Rachemacher is better than triangular since its inversion probability is close to zero; when the intersection length reaches 3, there is a sharp increase in the inversion probability of Rachemacher, which makes triangular a better choice; after the intersection length exceeds 4, triangular dominates Rachemacher in terms of the inversion probability again.

From the analysis above, we have the following conclusions. 1) No single distribution is optimal for all possible distributions of supports in terms of order preservation; rather, the perturbation distribution needs to be carefully selected, adapted to the underlying support distribution. 2) Intuitively, when the underlying support distribution is relative sparse, that is, the gap between two consecutive supports is large, distributions with small uncertainty regions, for instance, Rachmacher, are preferable, which lead to less intersection possibility; when the support distribution is dense, distributions with less probability mass density, for instance, triangular, are more desirable. 3) The impact of the perturbation distribution over the quality of order preservation needs to empirically evaluated.

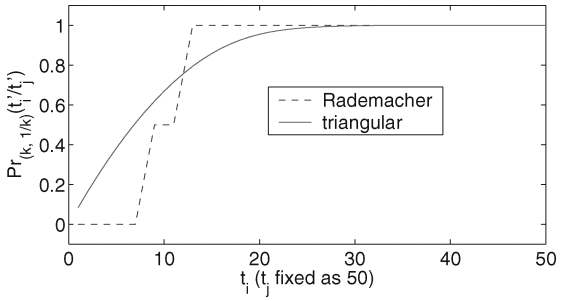

Fig. 6. Trade-off between uncertainty region length and probability mass density in ratio preservation, the parameter setting is: $\sigma = 1$, $\beta_i = 0$, $\beta_j = 0$, and $t_j = 50$.

7.2.1. Ratio Preservation. Next we study the impact of perturbation distribution over the quality of ratio preservation. We first reformulate the $(k, 1/k)$ probability under general distribution $f(\cdot)$. For ease of presentation, we assume that the perturbation distributions for all FECs are homogeneous, plus a FEC-specific bias.

Under general distribution $f(\cdot)$, the $(k, 1/k)$ probability is calculated as follows:

$$\begin{aligned} Pr_{(k,1/k)}[t'_i/t'_j] &= \int_{-\infty}^{\infty} f(x_j - \beta_j) \int_{k\frac{t_i}{t_j}(t_j+x_j)-t_i}^{\frac{1}{k}\frac{t_i}{t_j}(t_j+x_j)-t_i} f(x_i - \beta_i)\, \mathrm{d}x_i\, \mathrm{d}x_j \\ &= \mathbb{E}\Bigg[F\left(\frac{t_i}{t_j}\frac{x}{k} + \frac{t_i}{k} - t_i + \frac{t_i}{t_j}\frac{\beta_j}{k} - \beta_i\right) \\ &\quad -F\left(k\frac{t_i}{t_j}x + kt_i - t_i + k\frac{t_i}{t_j}\beta_j - \beta_i\right)\Bigg] \end{aligned}$$

Clearly, this quantity is the difference of the expectation of two CDFs after scaling and shifting transformation. Similar to the problem of order optimization, the difficulty of optimizing $\min \sum_{i<j} Pr_{(k,1/k)}[t'_i/t'_j]$ depends on the concrete form of the underlying perturbation distribution $f(\cdot)$, which needs to be investigated on a case-by-case basis, and is beyond the scope of this work. Here, we are interested in investigating the impact of uncertainty region length and probability mass density over the $(k, 1/k)$ probability.

Figure 6 illustrates the trade-off between uncertainty region length and probability mass density, with respect to varying ratio of t_i/t_j. For presentation purpose, we filter out the effect of bias setting (β_i and β_j are fixed as zero). We then fix t_j, and measure the $(k, 1/k)$ probability $Pr_{(k,1/k)}[t'_i/t'_j]$ of the two distributions, Rademacher and triangular, under varying t_i. Again, neither distribution demonstrates consistent superiority over the entire interval: for small ratio t_i/t_j, triangular is better than Rademacher given its larger $(k, 1/k)$ probability; as the ratio increases, Rademacher offers better quality of ratio preservation; while for large ratio (close to 1), the influence of both distributions is nonsignificant.

We can thus draw conclusions similar to the case of order preservation: no single distribution is optimal for all possible support distributions in terms of ratio preservation; rather, the perturbation distribution needs to be selected, based on the underlying support distribution. A rule of thumb is: when the underlying support distribution is sparse, that is, a large number of small ratios, distributions with small probability mass density, for instance, triangular, are more preferable; when the support distribution is relative dense, distributions with smaller uncertainty regions, for instance, Rademacher, are more preferable.

8. EXPERIMENTAL ANALYSIS

In this section, we investigate the efficacy of the proposed BUTTERFLY* approaches. Specifically, the experiments are designed to measure the following three properties: 1) privacy guarantee: the effectiveness against both intrawindow and interwindow inference; 2) result utility: the degradation of the output accuracy, the order and ratio preservation, and the trade-off among these utility metrics; 3) execution efficiency: the time taken to perform our approaches. We start with describing the datasets and the setup of the experiments.

8.1. Experimental Setting

We tested our solutions over both synthetic and real datasets. The synthetic dataset T20I4D50K is obtained by using the data generator as described in Agrawal and Srikant [1994], which mimics transactions from retail stores. The real datasets used include: 1) BMS-WebView-1, which contains a few months of clickstream data from an e-commerce web site; 2) BMS-POS, which contains several years of point-of-sale from a large number of electronic retailers; 3) Mushroom in UCI KDD archive,[4] which is used widely in machine learning research. All these datasets have been used in frequent pattern mining over streams [Chi et al. 2006].

We built our BUTTERFLY* prototype on top of *Moment* [Chi et al. 2006], a streaming frequent pattern mining framework, which finds closed frequent itemsets over a sliding window model. By default, the minimum support C and vulnerable support K are set as 25 and 5, respectively, and the window size is set as 2K. Note that the setting here is designed to test the effectiveness of our approaches with high ratio of vulnerable/minimum threshold (K/C). All the experiments were performed over a workstation with Intel Xeon 3.20GHz and 4GB main memory, running Red Hat Linux 9.0 operating system. The algorithm is implemented in C++ and compiled using g++ 3.4.

8.2. Experimental Results

To provide an in-depth understanding of our output-privacy protection schemes, we evaluated four different versions of BUTTERFLY*: the basic version, the optimized version with $\lambda = 0, 0.4$, and 1, respectively, over both synthetic and real datasets. Note that $\lambda = 0$ corresponds to the ratio-preserving scheme, while $\lambda = 1$ corresponds to the order-preserving one.

Privacy and Precision. To evaluate the effectiveness of BUTTERFLY* in terms of output-privacy protection, we need to find all potential privacy breaches in the mining output. This is done by running an analysis program over the results returned by the mining algorithm, and finding all possible vulnerable patterns that can be inferred through either intrawindow or interwindow inference.

Concretely, given a stream window, let $\mathcal{P}_{hv}$ denote all the hard vulnerable patterns that are inferable from the mining output. After the perturbation, we evaluate the relative deviation of the inferred value and the estimator for each pattern $P \in \mathcal{P}_{hv}$ for 100 continuous windows. we use the following average privacy (avg_prig) metric to measure the effectiveness of privacy preservation:

$$\text{avg\_prig} = \sum_{P \in \mathcal{P}_{hv}} \frac{(T'(P) - E[T'(P)])^2}{T^2(P)|\mathcal{P}_{hv}|}.$$

[4] http://kdd.ics.uci.edu/

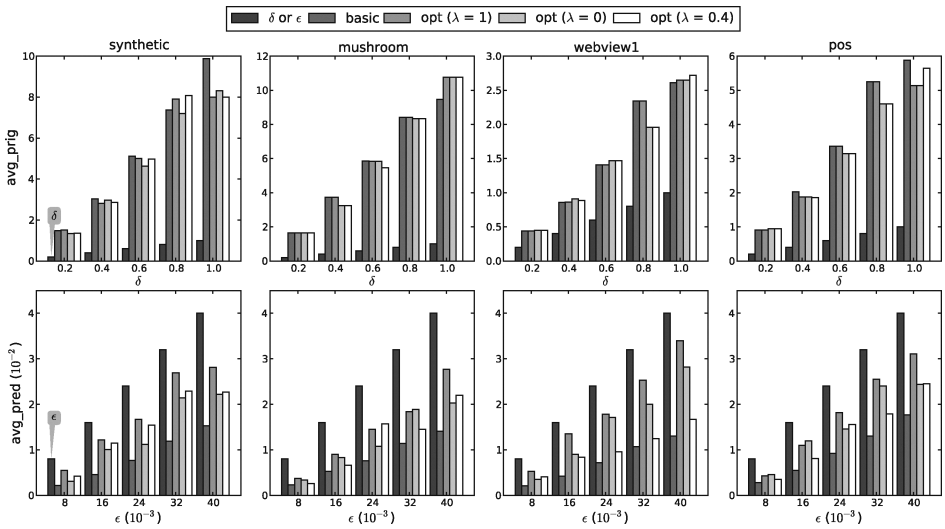

Fig. 7. Average privacy guarantee (*avg_prig*) and precision degradation (*avg_pred*).

The decrease of output precision is measured by the average precision degradation (avg_pred) of all frequent itemsets $\mathcal{I}$:

$$\text{avg\_pred} = \sum_{I \in \mathcal{I}} \frac{(T'(I) - T(I))^2}{T^2(I)|\mathcal{I}|}.$$

In this set of experiments, we fix the precision-privacy ratio as $\epsilon/\delta = 0.04$, and measure avg_prig and avg_pred for different settings of ϵ (δ).

Specifically, the four plots in the top tier of Figure 7 show that as the value of δ increases, all four versions of BUTTERFLY* provide similar amount of average privacy protection for all the datasets, far above the minimum privacy guarantee δ. The four plots in the lower tier show that as σ increases from 0 to 0.04, the output precision decreases; however, all four versions of BUTTERFLY* have average precision degradation below the system-supplied maximum threshold ϵ. Also note that among all the schemes, basic BUTTERFLY* achieves the minimum precision loss, for given privacy requirement. This can be explained by the fact that the basic scheme considers no semantic relationships, and sets all the biases as zero, while optimized BUTTERFLY* trades precision for other utility-related metrics. Although the basic scheme maximally preserves the precision, it may not be optimal in the sense of other utility metrics, as shown next.

Order and Ratio. For given privacy and precision requirement (ϵ, δ), we measure the effectiveness of BUTTERFLY* in preserving order and ratio of frequent itemsets.

The quality of order preservation is evaluated by the proportion of order-preserved pairs over all possible pairs, referred to as the rate of order preserved pairs (ropp):

$$\text{ropp} = \frac{\sum_{I,J \in \mathcal{I} \text{ and } T(I) \leq T(J)} \mathbf{1}_{T'(I) \leq T'(J)}}{C^2_{|\mathcal{I}|}},$$

where $\mathbf{1}_x$ is the indicator function, returning 1 if condition x holds, and 0 otherwise.

Analogously, the quality of ratio preservation is evaluated by the fraction of the number of $(k, 1/k)$ probability-preserved pairs over the number of possible pairs, referred

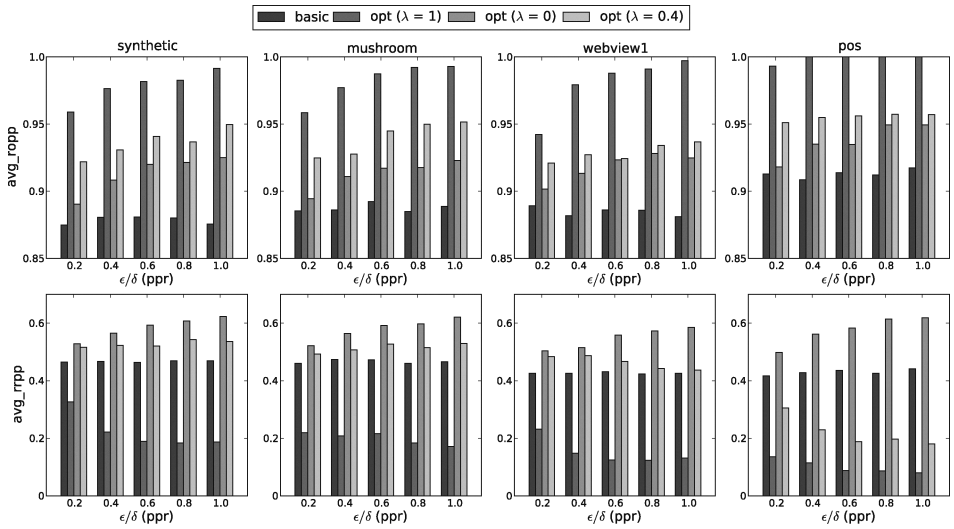

Fig. 8. Average order preservation (avg_ropp) and ratio preservation (avg_rrpp).

to as the rate of ratio preserved pairs (rrpp) (k is set 0.95 in all the experiments):

$$\text{rrpp} = \frac{\sum_{I,J\in\mathcal{I} \text{ and } T(I)\le T(J)} \mathbf{1}_{k\frac{T(I)}{T(J)} \le \frac{T'(I)}{T'(J)} \le \frac{1}{k}\frac{T(I)}{T(J)}}}{C^2_{|\mathcal{I}|}}.$$

In this set of experiments, we vary the precision-privacy ratio ϵ/δ for fixed $\delta = 0.4$, and measure the ropp and rrpp for four versions of BUTTERFLY* (the parameter $\gamma = 2$ in all the experiments), as shown in Figure 8.

As predicted by our theoretical analysis, the order-preserving ($\lambda = 1$) and ratio-preserving ($\lambda = 0$) bias settings are fairly effective, both outperform all other schemes at their ends. The ropp and rrpp increase as the ratio of ϵ/δ grows, because larger ϵ/δ offers more adjustable bias therefore leading to better quality of order or ratio preservation.

It is also noticed that the order-preserving scheme has the worst performance in terms of avg_rrpp, even worse than the basic approach. The explanation is that in order to distinguish overlapping FECs, the order-preserving scheme may significantly distort the ratio of pairs of FECs. In all these cases, the hybrid scheme $\lambda = 0.4$ achieves the second best in terms of avg_rrpp and avg_ropp, and an overall best quality when order and ratio information is equally important.

Tuning of Parameters γ and λ. Next we give a detailed discussion on the setting of the parameters γ and λ.

Specifically, γ controls the depth of dynamic programming in the order-preserving bias setting. Intuitively, a larger γ leads to better quality of order preservation, but also higher time and space complexity. We desire to characterize the gain of the quality of order preservation with respect to γ, and find the setting that balances the quality and efficiency.

For all four datasets, we measure the ropp with respect to the setting of γ, with result shown in Figure 9. It is noted that the quality of order preservation increases sharply at certain points $\gamma = 2$ or 3, and the trend becomes much flatter after that. This is explained by that in most real datasets, the distribution of FECs is not extremely dense;

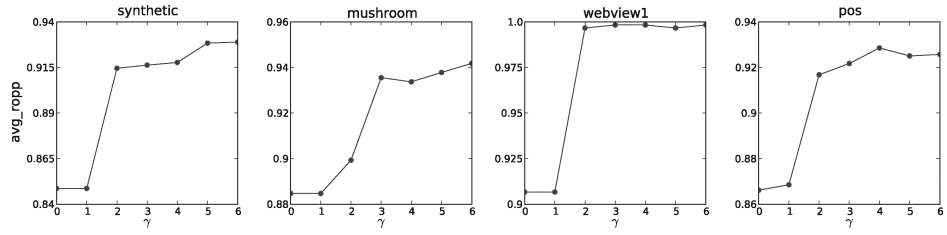

Fig. 9. Average rate of order-preserved pairs with respect to setting of γ.

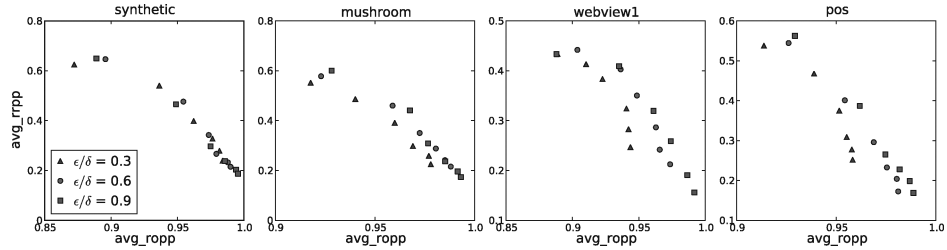

Fig. 10. Trade-off between order preservation and ratio preservation.

under proper setting of (ϵ, δ), a FEC can intersect with only 2 or 3 neighboring FECs on average. Therefore, the setting of small γ is usually sufficient for most datasets.

The setting of λ balances the quality of order and ratio preservation. For each dataset, we evaluate ropp and rrpp with different settings of λ (0.2, 0.4, 0.6, 0.8 and 1) and precision-privacy ratio ϵ/δ (0.3, 0.6 and 0.9), as shown in Figure 10.

These plots give good estimation of the gain of order preservation, for given cost of ratio preservation that one is willing to sacrifice. A larger ϵ/δ gives more room for this adjustment. In most cases, the setting of $\lambda = 0.4$ offers a good balance between the two metrics. The trade-off plots could be made more accurate by choosing more settings of λ and ϵ/δ to explore more points in the space.

8.2.1. Execution Efficiency. In the last set of experiments, we measure the computation overhead of BUTTERFLY* over the original mining algorithm for different settings of minimum support C. We divide the execution time into two parts contributed by the mining algorithm (mining algorithm) and BUTTERFLY* algorithm (butterfly), respectively. Note that we do not distinguish basic and optimized BUTTERFLY*, since basic BUTTERFLY* involves simple perturbation operations, with unnoticeable cost. The window size is set 5K for all four datasets.

The result plotted in Figure 11 shows clearly that the overhead of BUTTERFLY* is much less significant than the mining algorithm; therefore, it can be readily implemented atop existing stream mining algorithms. Further, while the current versions of BUTTERFLY* are window-based, it is expected that an incremental version of BUTTERFLY* can achieve even lower overhead.

It is noted that in most cases, the running time of both mining algorithm and BUTTERFLY* algorithm grows significantly as C decreases; however, the growth of the overhead of BUTTERFLY* is much less evident compared with the mining algorithm itself. This is expected since as the minimum support decreases, the number of frequent itemsets increases superlinearly, but the number of FECs has much lower growth rate, which is the most influential factor for the performance of BUTTERFLY*.

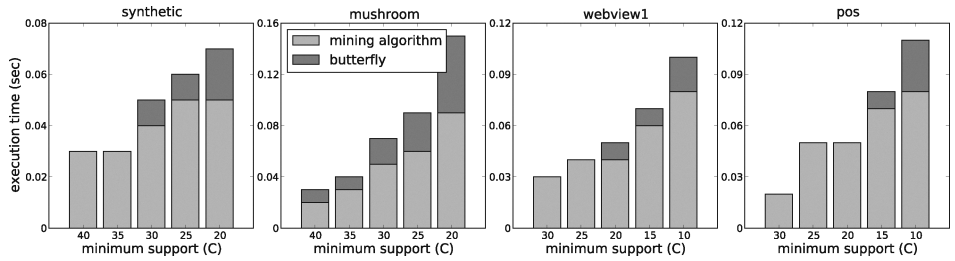

Fig. 11. Overhead of BUTTERFLY* algorithms in stream mining systems.

9. RELATED WORK

9.1. Disclosure Control in Statistical Database

The most straightforward solution to preserving output privacy is to detect and eliminate all potential privacy breaches, that is, the *detecting-then-removing* strategy, which stemmed from the inference control in statistical and census databases from 1970s. Motivated by the need of publishing census data, the statistics literature focuses mainly on identifying and protecting the privacy of sensitive data entries in contingency tables, or tables of counts corresponding to cross-classification of the microdata.

Extensive research has been done in statistical databases to provide statistical information without compromising sensitive information regarding individuals [Chin and Ozsoyoglu 1981; Shoshani 1982; Adam and Worthmann 1989]. The techniques, according to their application scenarios, can be broadly classified as *query restriction* and *data perturbation*. The query restriction family includes controlling the size of query results [Fellegi 1972], restricting the overlap between successive queries [Dobkin et al. 1979], suppressing the cells of small size [Cox 1980], and auditing queries to check privacy compromises [Chin and Ozsoyoglu 1981]; the data perturbation family includes sampling microdata [Denning 1980], swapping data entries between different cells [Dalenius and Reiss 1980], and adding noises to the microdata [Traub et al. 1984] or the query results [Denning 1980]. Data perturbation by adding statistical noise is an important method of enhancing privacy. The idea is to perturb the true value by a small amount ϵ where ϵ is a random variable with mean $= 0$ and a small variance $= \sigma^2$. While we adopt the method of perturbation from statistical literature, one of our key technical contributions is the generalization of the basic scheme by adjusting the mean to accommodate various semantic constraints in the applications of mining output.

9.2. Input Privacy Preservation

Intensive research efforts have been directed to addressing the input-privacy issues. The work of Agrawal and Srikant [2000] and Agrawal and Aggarwal [2001] paved the way for the rapidly expanding field of privacy-preserving data mining; they established the main theme of privacy-preserving data mining so as to provide sufficient privacy guarantee while minimizing the information loss in the mining output. Under this framework, a variety of techniques have been developed.

The work of Agrawal and Srikant [2000], Agrawal and Aggarwal [2001], Evfimievski et al. [2002] and Chen and Liu [2005] applied data perturbation, specifically random noise addition, to association rule mining, with the objective of maintaining sufficiently accurate estimation of frequent patterns while preventing disclosure of specific transactions (records). In the context of data dissemination and publication, group-based anonymization approaches have been considered. The existing work can be roughly classified as two categories: the first one aims at devising anonymization models and principles, as the criteria to measure the quality of privacy protection,

for example, k-anonymity [Sweeney 2002], l-diversity [Machanavajjhala et al. 2006], $(\epsilon, \delta)^k$-dissimilarity [Wang et al. 2009], etc.; the second category of work explores the possibility of fulfilling the proposed anonymization principles, meanwhile preserving the data utility to the maximum extent, [LeFevre et al. 2006; Park and Shim 2007]. Cryptographic tools have also been used to construct privacy-preserving data mining protocols, for example, secure multiparty computation [Lindell and Pinkas 2000; Vaidya and Clifton 2002]. Nevertheless, all these techniques focus on protecting input privacy for static datasets. Quite recently, Li et al. [2007] address the problem of preserving input privacy for streaming data, by online analysis of correlation structure of multivariate streams. Bu et al. [2007] distinguish the scenario of data custodian, where the data collector is entrusted, and proposes a perturbation scheme that guarantees no change in the mining output. In Kargupta et al. [2003] and Huang et al. [2005], it is shown that a hacker can potentially employ spectral analysis to separate the random noise from the real values for multiattribute data.

9.3. Output Privacy Preservation

Compared with the wealth of techniques developed for preserving input privacy, the attention given to protecting mining output is fairly limited. The existing literature can be broadly classified as two categories. The first category attempts to propose general frameworks for detecting potential privacy breaches. For example, Kantarcioğlu et al. [2004] propose an empirical testing scheme for evaluating if the constructed classifier violates the privacy constraint. The second category focuses on proposing algorithms to address the detected breaches for specific mining tasks. For instance, it is shown in Atzori et al. [2008] that the association rules can be exploited to infer information about individual transactions; while Wang et al. [2007] propose a scheme to block the inference of sensitive patterns satisfying user-specified templates by suppressing certain raw transactions.

This article is developed on the basis of Wang and Liu [2008] and it significantly extends the previous work from the following aspects. 1) In Section 3, we introduce a general inferencing and disclosure model that exploits the privacy breaches existing in current stream mining systems. 2) In Section 6, we propose an optimized order-preserving bias setting scheme that flexibly trades between computational efficiency and quality of order preservation. 3) In Section 6, we provide sound theoretical analysis to prove the efficacy of the order-preserving and ratio-preserving bias setting schemes, which compliments the experimental analysis in Wang and Liu [2008]. 4) In Section 7, we examine the impact of perturbation distribution over the quality of privacy protection and utility preservation. 5) In Section 8, we present more extensive experimental analysis over both synthetic and real datasets. 6) We provide a much more comprehensive comparison between our work and existing literature in Section 9.

10. CONCLUSIONS

In this work, we highlighted the importance of imposing privacy protection over (stream) mining output, a problem complimentary to conventional input privacy protection. We articulated a general framework of sanitizing sensitive patterns (models) to achieve output-privacy protection. We presented the inferencing and disclosure scenarios wherein the adversary performs attacks over the mining output. Motivated by the basis of the attack model, we proposed a lighted-weighted countermeasure, BUTTERFLY*. It counters the malicious inference by amplifying the uncertainty of vulnerable patterns, at the cost of trivial decrease of output accuracy; meanwhile, for given privacy and accuracy requirement, it maximally preserves the utility-relevant semantics in mining output, and thus achieving the optimal balance between privacy guarantee

and output quality. The efficacy of BUTTERFLY* is validated by extensive experiments on both synthetic and real datasets.

ACKNOWLEDGMENT

The authors would like to thank the ACM TODS editors and anonymous reviewers for their valuable constructive comments.

REFERENCES

ADAM, N. R. AND WORTHMANN, J. C. 1989. Security-control methods for statistical databases: a comparative study. *ACM Comput. Surv. 21,* 4, 515–556.

AGRAWAL, D. AND AGGARWAL, C. C. 2001. On the design and quantification of privacy preserving data mining algorithms. In *Proceedings of the 20th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS'01)*. ACM, New York, NY, 247–255.

AGRAWAL, R. AND SRIKANT, R. 1994. Fast algorithms for mining association rules in large databases. In *Proceedings of the 20th International Conference on Very Large Data Bases (VLDB'94)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, 487–499.

AGRAWAL, R. AND SRIKANT, R. 2000. Privacy-preserving data mining. *SIGMOD Rec. 29,* 2, 439–450.

ATZORI, M., BONCHI, F., GIANNOTTI, F., AND PEDRESCHI, D. 2008. Anonymity preserving pattern discovery. *VLDB J. 17,* 4, 703–727.

BABCOCK, B., BABU, S., DATAR, M., MOTWANI, R., AND WIDOM, J. 2002. Models and issues in data stream systems. In *Proceedings of the 21st ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS'02)*. ACM, New York, NY, 1–16.

BU, S., LAKSHMANAN, L. V. S., NG, R. T., AND RAMESH, G. 2007. Preservation of patterns and input-output privacy. In *Proceedings of the 23th IEEE International Conference on Data Mining (ICDE'07)*. IEEE Computer Society, Los Alamitos, CA, 696–705.

CALDERS, T. 2004. Computational complexity of itemset frequency satisfiability. In *Proceedings of the 23rd ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS'04)*. ACM, New York, NY, 143–154.

CALDERS, T. AND GOETHALS, B. 2002. Mining all non-derivable frequent itemsets. In *Proceedings of the 6th European Conference on Principles of Data Mining and Knowledge Discovery (PKDD'02)*. Springer-Verlag, Berlin, Germany, 74–85.

CHEN, K. AND LIU, L. 2005. Privacy preserving data classification with rotation perturbation. In *Proceedings of the 5th IEEE International Conference on Data Mining (ICDM'05)*. IEEE Computer Society, Los Alamitos, CA, 589–592.

CHI, Y., WANG, H., YU, P. S., AND MUNTZ, R. R. 2006. Catch the moment: maintaining closed frequent itemsets over a data stream sliding window. *Knowl. Inf. Syst. 10,* 3, 265–294.

CHIN, F. Y. AND OZSOYOGLU, G. 1981. Statistical database design. *ACM Trans. Datab. Syst. 6,* 1, 113–139.

COX, L. 1980. Suppression methodology and statistical disclosure control. *J. Amer. Stat. Asson. 75,* 370, 377–385.

DALENIUS, T. AND REISS, S. P. 1980. Data-swapping: A technique for disclosure control. *J. Statist. Plann. Inference 6*, 73–85.

DENNING, D. E. 1980. Secure statistical databases with random sample queries. *ACM Trans. Datab. Syst. 5,* 3, 291–315.

DOBKIN, D., JONES, A. K., AND LIPTON, R. J. 1979. Secure databases: protection against user influence. *ACM Trans. Datab. Syst. 4,* 1, 97–106.

EVFIMIEVSKI, A., SRIKANT, R., AGRAWAL, R., AND GEHRKE, J. 2002. Privacy preserving mining of association rules. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'02)*. ACM, New York, NY, 217–228.

FELLEGI, I. P. 1972. On the question of statistical confidentiality. *J. Amer. Stat. Asson. 67,* 337, 7–18.

HORE, B., MEHROTRA, S., AND TSUDIK, G. 2004. A privacy-preserving index for range queries. In *Proceedings of the 30th International Conference on Very Large Data Bases (VLDB'04)*. VLDB Endowment, Toronto, Canada, 720–731.

HUANG, Z., DU, W., AND CHEN, B. 2005. Deriving private information from randomized data. In *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD'05)*. ACM, New York, NY, 37–48.

KANTARCIOĞLU, M., JIN, J., AND CLIFTON, C. 2004. When do data mining results violate privacy? In *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data mining (KDD'04)*. ACM, New York, NY, 599–604.

KARGUPTA, H., DATTA, S., WANG, Q., AND SIVAKUMAR, K. 2003. On the privacy preserving properties of random data perturbation techniques. In *Proceedings of the 3rd IEEE International Conference on Data Mining (ICDM'03)*. IEEE Computer Society, Los Alamitos, CA, 99.

LEFEVRE, K., DEWITT, D. J., AND RAMAKRISHNAN, R. 2006. Mondrian multidimensional k-anonymity. In *Proceedings of the 22nd International Conference on Data Engineering (ICDE'06)*. IEEE Computer Society, Los Alamitos, CA, 25.

LI, F., SUN, J., PAPADIMITRIOU, S., MIHAILA, G. A., AND STANOI, I. 2007. Hiding in the crowd: Privacy preservation on evolving streams through correlation tracking. In *Proceedings of the 23th IEEE International Conference on Data Mining (ICDE'07)*. IEEE Computer Society, Los Alamitos, CA, 686–695.

LINDELL, Y. AND PINKAS, B. 2000. Privacy preserving data mining. In *Proceedings of the 20th Annual International Cryptology Conference on Advances in Cryptology (CRYPTO'00)*. Springer-Verlag, Berlin, Germany, 36–54.

LUKASIEWICZ, T. 2001. Probabilistic logic programming with conditional constraints. *ACM Trans. Comput. Logic 2,* 3, 289–339.

MACHANAVAJJHALA, A., GEHRKE, J., KIFER, D., AND VENKITASUBRAMANIAM, M. 2006. l-diversity: Privacy beyond k-anonymity. In *Proceedings of the 22th IEEE International Conference on Data Mining (ICDE'06)*. IEEE Computer Society, Los Alamitos, CA, 24.

O'CONNOR, L. 1993. The inclusion-exclusion principle and its applications to cryptography. *Cryptologia 17,* 1, 63–79.

PARK, H. AND SHIM, K. 2007. Approximate algorithms for k-anonymity. In *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD'07)*. ACM, New York, NY, 67–78.

SHOSHANI, A. 1982. Statistical databases: Characteristics, problems, and some solutions. In *Proceedings of the 8th International Conference on Very Large Data Bases (VLDB'82)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, 208–222.

SWEENEY, L. 2002. k-anonymity: a model for protecting privacy. *Int. J. Uncertain. Fuzziness Knowl.-Based Syst. 10,* 5, 557–570.

TRAUB, J. F., YEMINI, Y., AND WOŹNIAKOWSKI, H. 1984. The statistical security of a statistical database. *ACM Trans. Datab. Syst. 9,* 4, 672–679.

VAIDYA, J. AND CLIFTON, C. 2002. Privacy preserving association rule mining in vertically partitioned data. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'02)*. ACM, New York, NY, 639–644.

VAVASIS, S. A. 1990. Quadratic programming is in np. *Inf. Process. Lett. 36,* 2, 73–77.

WANG, K., FUNG, B. C. M., AND YU, P. S. 2007. Handicapping attacker's confidence: an alternative to k-anonymization. *Knowl. Inf. Syst. 11,* 3, 345–368.

WANG, T. AND LIU, L. 2008. Butterfly: Protecting output privacy in stream mining. In *Proceedings of the IEEE 24th International Conference on Data Engineering (ICDE'08)*. IEEE Computer Society, Los Alamitos, CA, 1170–1179.

WANG, T., MENG, S., BAMBA, B., LIU, L., AND PU, C. 2009. A general proximity privacy principle. In *Proceedings of the IEEE International Conference on Data Engineering (ICDE'09)*. IEEE Computer Society, Los Alamitos, CA, 1279–1282.

Received April 2009; revised January 2010; accepted February 2010