CERIAS Tech Report 2013-7
Precision-bounded Access Control for Privacy-preserving Data Streams
by Zahid Pervaiz, Arif Ghafoor, Walid G. Aref
Center for Education and Research
Information Assurance and Security
Purdue University, West Lafayette, IN 47907-2086

Precision-bounded Access Control for Privacy-preserving Data Streams

Zahid Pervaiz, Arif Ghafoor *Fellow, IEEE*, Walid G. Aref *Senior Member, IEEE*

Abstract—Access control mechanisms and Privacy Protection Mechanisms (PPM) have been proposed for data streams. The access control for data stream allows roles access to tuples satisfying an authorized predicate sliding-window query. When the sensitive stream data is shared without a PPM the privacy can be compromised. The PPM meets privacy requirements, e.g., k-anonymity or l-diversity by generalization of stream data. This imprecision introduced by generalization can be reduced by delaying the publishing of stream data. However, the delay in sharing the stream tuples to achieve better accuracy can lead to false negatives if the tuples slide out of the window when a sliding-window query predicate is evaluated for access control mechanism.

To set a threshold on the loss of precision, access control mechanism defines the imprecision bound for each query. The challenge is to optimize the time duration for which the stream data is held by PPM so that the imprecision bounds for the maximum possible number of queries are met. In our formulation of the aforementioned problem we present the hardness results, provide an anonymization algorithm, and conduct experimental evaluation of the proposed algorithm. Experiments demonstrate that the proposed heuristic provides better precision as compared to the minimum or maximum delay heuristics.

Index Terms—Privacy, k-anonymity, Access Control, Data Stream.

1 Introduction

Data Stream Management Systems (DSMS) have been proposed to process transactional data, e.g., internet traffic, health monitoring, and sensor networks [1], [2]. Continuous queries on a data stream can be used to define real-time alerts for event detection [3]. Access control mechanisms for data streams ensure that only the authorized parts of the stream are available to each user or role [4], [5]. The objects to be protected under the access control mechanism are the queries or views of the data stream [5]. If the sensitive information in a data stream is not privacy protected, then the privacy of a person can be compromised even in the presence of access control. The well-known privacy preservation techniques of k-anonymity [6] and l-diversity [7] have also been used for privacy protection of data streams [8], [9]. However, to the best of our knowledge, precision-bounded access control along with privacy protection has not been investigated before for data streams, which is the focus of this paper.

The attribute values in the data stream tuples can be generalized to satisfy the given privacy requirements. Attribute data generalization introduces imprecision in the query results for access control mechanism. This imprecision can be reduced if the publishing of stream data is delayed. However, the delay introduces false negatives in the query results if the tuples satisfying the query predicate have not been made available to the access control mechanism at the instance of query evaluation.

Example 1 (Motivating Scenario)**.** Syndromic surveillance systems have been developed by state and federal agencies to detect and monitor public health emergencies [10], [11]. The emergency department data (age, gender, location, time of arrival, symptoms, etc.) from county hospitals is collected and is sent to the department of health at the state level after every hour [11]. The surveillance data stream is classified into syndrome categories and is anonymized by the state department of health and is then shared with departments of health at each county.

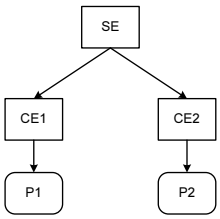
An role based access control policy is given in Figure 1. Role SE is above roles CE1 and CE2 in the role hierarchy and can execute all the permissions allowed to roles CE1 and CE2. This policy allows the users to execute the authorized queries on the data stream, e.g., Role CE1 can execute queries under Permission P1 over the stream data in a 24-hour window with a slide of 4 hours (i.e., updated every four hours). The Temporal Constraint (TC) T1 defines a sliding window (size = 24 hours, slide = 4 hours) of stream data upon which the query can execute. Permissions under an access control policy ensure that only the authorized part of the data stream is available to each user. The flu season starts in October, peaks around February, and lasts till April each year [12]. The epidemiologists at the state and county levels suggest community containment mea-

• *Z. Pervaiz and A. Ghafoor are with School of Electrical and Computer Engineering and W. Aref is with Department of Computer Science, Purdue University, IN, 47907.*
E-mail: zpervaiz@purdue.edu

sures, e.g., isolation or quarantine according to the number of persons infected in case of a flu outbreak. Depending upon the population density of a county, an epidemiologist can advise isolation if the number of persons reported with influenza are greater than 1000, or quarantine if the number is greater than 3000 during the last 24-hour window. The anonymization adds false positives to the query window and the precision can be improved by delaying the stream data. However, the delay adds false negatives for the queries (when a tuple satisfying the query has not been shared). The imprecision bound for each sliding-window query ensures that results are within the required tolerance at the time of query evaluation. The total imprecision for a query is the sum of false positives and false negatives. The high rate of false positives will cause unnecessary false alarms for the containment measures, while the high rate of false negatives will result in missed event detection.

Role	Designation
SE	State Epidemiologist
CE1	County 1 Epidemiologist
CE2	County 2 Epidemiologist

TC	Type	Definition
T1	Sliding Window	Size = 24 hours, Step = 4 hours



Permission	Authorized Query Predicate
P1	Location = County 1 ∧ Age = 15-65 ∧ Syndrome = Influenza ∧ T1
P2	Location = County 2 ∧ Age = 15-65 ∧ Syndrome = Influenza ∧ T1

Fig. 1. Access control policy

The contributions of the paper are as follows. First, we introduce the concept of precision-bounded access control for privacy-preserving data streams. Second, we formulate the Precision-bounded Access Control for privacy-preserving data strEams (PACE) problem and give hardness results. Third, we propose a heuristic for an approximate solution of the PACE problem and conduct empirical evaluation.

The rest of this paper proceeds as follows. In Section 2, relevant background is discussed. The problem formulation and definitions are presented in Section 3. Section 4 covers the proposed total imprecision minimization heuristic for multidimensional partitioning of stream data to satisfy imprecision bounds of predicate sliding-window queries. Experimental results are in Section 5. The related work is presented in Section 6 and Section 7 concludes the paper.

2 Background

Given a stream $T[i] = \{ID, TS, A_1, A_2, \ldots, A_d, SA\}$, where ID is an identity attribute, TS is a time-stamp attribute that represents the arrival time of a tuple, A_j is a Quasi Identifier (QI) attribute, SA is the sensitive attribute, d is the number of QI attributes, and, i is the current time-stamp. $T[i]$ represents all the data stream tuples that have arrived till the time instance i. The identity attribute (e.g., social security number) can uniquely identify an individual in a data stream. QI attributes (e.g., address, age) can be used with the background information to identify an individual even if the identity attribute has been suppressed [6]. If the sensitive attribute SA is associated with a unique individual, it will result in a privacy violation.

2.1 Anonymity Definitions

k-anonymity [6] for streaming data has been proposed by Cao et al. [8] and Zhou et al. [9]. It has been suggested to suppress the TS attribute in the anonymized stream for privacy protection. However, with respect to access control over streaming data, the TS attribute is required for the evaluation of sliding-window queries. We propose that the generalized time-stamp value for each equivalence class must be included in the anonymized stream. The time-stamp attribute is a quasi-identifier attribute as knowing the time-stamp value for a person in a relational stream data can allow to find the associated sensitive value and violate privacy of that person.

Definition 1 (Equivalence Class (EC))**.** An equivalence class is a set of tuples having the same QI attribute and time-stamp values.

Definition 2 (Stream k_s-anonymity Property [9])**.** A data stream $T^p[i]$ satisfies the k_s-anonymity property if each published equivalence class has k or more tuples and if $t_1.ID = t_2.ID$ then $EC(t_1) \neq EC(t_2)$ for any $t_1, t_2 \in T[i]$.

Here, $T^p[i]$ is the anonymized view of the stream data that has been published till time instance i. In $T^p[i]$, the identity attribute is suppressed, the QI and the time-stamp attribute values are generalized and the sensitive attribute is published. The time-stamp attribute gives the arrival time of a tuple $t \in T[i]$ and the delay in the publishing is equal to t.TS - t.PUB, where t.PUB is the time when a tuple is published (i.e., is added to $T^p[i]$). The second part of the stream k_s-anonymity definition sets the constraint that each equivalence class must have tuples with different ID's. In the case of a data stream, multiple tuples ($\geq k_s$) can be received from the same person/ID in a short span of time. Without the above constraint, the tuples with the same ID can be put into the same equivalence class without any generalization resulting in a high probability of privacy violation. l-diversity counters the homogeneity attacks possible against k-anonymity when all the tuples in an equivalence class have the same sensitive attribute value.

Definition 3 (Stream l_s-Diversity Property)**.** A data stream $T^p[i]$ satisfies the l_s-diversity property if each

equivalence class has at least l distinct values of the sensitive attribute and if $t_1.ID = t_2.ID$ then $EC(t_1) = EC(t_2)$ for any $t_1, t_2 \in T[i]$.

In the case of sensitive numeric attributes, if the numeric values in an equivalence class are close to each other, an l_s-diverse equivalence class can still leak private information. Variance diversity [13] and t-closeness [14] have been proposed to protect privacy against such a threat. The stream variance diversity is defined as follows:

Definition 4 (Stream Variance Diversity)**.** A data stream $T^p[i]$ is variance diverse if the variance V_s(EC) of each published equivalence class satisfies V_s(EC) $\geq v_s$, where v_s is the data stream variance diversity parameter and if $t_1.ID = t_2.ID$ then $EC(t_1) = EC(t_2)$ for any $t_1, t_2 \in T[i]$.

Definition 5 (Delay Constraint)**.** The delay constraint, denoted by δ, is the maximum delay before which a tuple is required to be published.

The delay constraint can be set according to the temporal guarantees required by the streaming data application on the availability of the anonymized tuples. The time constraint set by δ ensures that the delayed tuples eventually get published. The delay constraint can also be set based on storage limitations of the system anonymizing sensitive stream data.

$T^h[i]$ is the set of tuples from $T[i]$ that are put on hold and are still to be anonymized at time instance i. Figure 2(a) gives the data stream tuples $T[4]$ received till time instance 4, and Figure 2(b) gives the corresponding anonymized data stream $T^p[4]$. The data stream in Figure 2(a) does not satisfy k-anonymity because knowing the age and zip code of a person allows associating a disease to that person. The data stream $T^p[4]$ in Figure 2(b) is 2-anonymous and 2-diverse. The tuples $T^h[4]$ in Figure 2(c) are on hold and are still waiting to be published.

2.2 Stream Query Model

Predicate window queries have been proposed for streaming data management systems [15], [16]. Other types of queries on streaming data are the snapshot query [17] and the landmark query [18]. The sliding window query is defined by two parameters: 1) *Range* that defines the size of query window and 2) *Slide* that defines the step by which the window moves [15], [18], [19]. If the slide of the window is less than the range, then the sliding query windows overlap. Otherwise, if the slide is equal to the range, then the windows are non-overlapping and are also known as tumbling windows. The sliding-window query can be either tuple-count sliding-window or time-sliding-window [1]. In this paper, we consider time-sliding-window queries. The predicate sliding-window query is evaluated at the end of the window size and then the window slides by the step size.

		QI_1	QI_2	SA
ID	**TS**	**Age**	**Zip**	**Disease**
A	1	5	15	Flu
B	1	15	25	Fever
C	2	28	28	Diarrhea
D	2	25	15	Fever
E	3	22	28	Flu
F	3	32	35	Fever
G	4	38	32	Flu
H	4	35	25	Diarrhea
...	...	...	...	...

(a) Sensitive data stream $T[4]$

		QI_1	QI_2	SA	
ID	**TS**	**Age**	**Zip**	**Disease**	
A	1-2	5-25	15-15	Flu	
D	1-2	5-25	15-15	Fever	
C	2-3	22-28	28-28	Diarrhea	TS = 2
E	2-3	22-28	28-28	Flu	
F	3-4	30-40	20-40	Fever	TS = 3
G	3-4	30-40	20-40	Flu	
...	...	...	...	...	TS = 4

(b) 2-anonymous data stream $T^p[4]$

		QI_1	QI_2	SA
ID	**TS**	**Age**	**Zip**	**Disease**
B	1	15	25	Fever
H	4	35	25	Diarrhea

(c) Stream data on hold $T^h[4]$

Fig. 2. Generalization of streaming data for k-anonymity and l-diversity

2.3 Role Based Access Control (RBAC)

Definition 6 (RBAC Policy)**.** An RBAC policy ρ is a tuple $\langle U, R, P, UA, PA, RH \rangle$, where

U is a set of users, R is a set of Roles, P is a set of Permissions, RH is a Role Hierarchy that is a partial order on roles, UA is a user-to-role assignment relation, and RA is a role-to-permission assignment relation [20]. Role-based Access Control (RBAC) for data streams has been proposed by Carminati et al. [5], [21]. Permissions under P are the sliding-window query predicates that define the authorized view of the data stream.

3 Stream Anonymization with Imprecision Bounds

In this section, we give definitions for the imprecision, imprecision bound, average query bound violation, and formulate the problem of Precision-bounded Access Control for privacy-preserving data strEams (PACE).

3.1 Predicate Evaluation and Imprecision

A predicate sliding-window query is evaluated over a data stream, say $T[i]$, by including all stream tuples that satisfy the query predicate. We define predicate

evaluation over an anonymized data stream $T^p[i]$ as follows. We adhere to the same semantics suggested in [13], [22], i.e., include all tuples in equivalence classes that overlap the query predicate range. We will further refer to query evaluation under these semantics as the *Overlap* semantics. Access control using the overlap semantics provides access to more anonymized tuples than authorized. For strict access control enforcement, another semantics can be used for query evaluation, i.e, include all tuples in equivalence classes that are fully enclosed inside the query predicate range. This semantics will be further referred to as the *Enclosed* semantics. The definitions in the following paragraphs and the anonymization algorithm in Section 4 follow the Overlap semantics.

Definition 7 (False-Positive Tuple)**.** A tuple is a false-positive when it does not satisfy the sliding-window query predicate at the time instance of query evaluation but is included in the query result as the equivalence class in $T^p[i]$ that contains the tuple overlaps the query predicate.

The number of False-Positive (FP) tuples in the result of a predicate sliding-window query, say $Q_j[i]$, at any time instance i, is as follows:

$$FP_{Q_j[i]} = |Q_j(T^p[i])| - |Q_j(T[i] - T^h[i])|, \text{ where} \quad (1)$$

$$|Q_j(T^p[i])| = \sum_{EC \text{ overlaps } Q_j} |EC|$$

A published partition can add a false-positive tuple to a predicate sliding-window query due to a spatial overlap (QI attributes), temporal (time-stamp attribute) overlap, or both temporal and spatial overlaps.

Definition 8 (False-Negative Tuple)**.** A tuple is a false-negative when it satisfies the predicate sliding-window query at the time instance of query evaluation but is not included in the query result due to being on hold.

The number of False-Negative (FN) tuples for a query, say $Q_j[i]$, evaluated at time instance i, is as follows:

$$FN_{Q_j[i]} = |Q_j(T^h[i])| \quad (2)$$

If the publishing delay is increased, the false-positives will reduce because equivalence classes with less imprecision can be formed but the number of false-negatives will increase.

Definition 9 (Sliding-Window Query Imprecision)**.** Query imprecision is defined as the total sum of false-positives and false-negatives for a predicate sliding-window query evaluated on an anonymized stream $T^p[i]$ at any given time instance i. The imprecision for query $Q_j[i]$ evaluated at time instance i is denoted by $imp_{Q_j[i]}$ and is equal to the sum of false-positives and false-negatives.

$$imp_{Q_j[i]} = FP_{Q_j}[i] + FN_{Q_j}[i] \quad (3)$$

Here, Query $Q_j[i]$ is evaluated over $T^p[i]$ by including all the tuples in equivalence classes that overlap the query region and $T^h[i]$.

Example 2. The two-dimensional representation for the quasi-identifier attributes of the anonymized data stream in Figure 2 is given in Figure 3. The rectangles with solid lines represent Query Q_1 and Q_2. The rectangles with dotted lines represent partitions (equivalence class). Assume that Query $Q_2[4]$ is evaluated at time instance 4. There is one false-positive Tuple A for $Q_2[4]$ as Partition P_1 and P_2 overlap the query. The false-negative tuple for $Q_2[4]$ is Tuple H as that tuple is still to be published and has not joined any equivalence class.

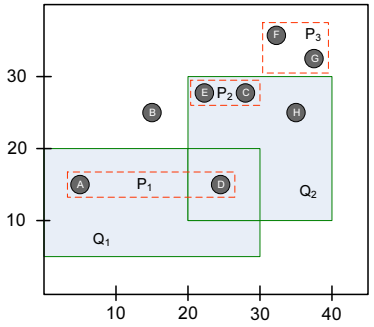


Fig. 3. Query evaluation over an anonymized data stream

Definition 10 (Query Imprecision Bound)**.** The query imprecision bound, denoted by $B_{Q_j[i]}$, is the total imprecision acceptable to a query predicate $Q_j[i]$ by the access control mechanism when the sliding-window query predicate is evaluated at time instance i.

A query violates the imprecision bound when, at the time of query evaluation, the total imprecision is more than the imprecision bound. For a sliding-window query, the query evaluation takes place at each step and hence we define the average query bound violation as follows:

Definition 11 (Average Query-bound Violation)**.** The Average Query-bound Violation (AQV) for a query Q_j is the average number of times the query imprecision bound is violated over a given time period.

$$AQV_{Q_j} = \frac{V_{Q_j}}{N_{Q_j}} \quad (4)$$

where N_{Q_j} is the number of steps Query Q_j has taken till the current time instance and V_{Q_j} is the

number of times the imprecision bound is violated on these steps.

Example 3. Consider a sliding-window query, say Q_1, that has taken 10 steps during a given time interval. At each query step, the query imprecision is evaluated and the imprecision value violates the imprecision bound at 4 of these steps. The average query bound violation for Q_1 is then 0.4 (4/10). Consider another query, say Q_2, that has taken 20 steps during the same time interval and the imprecision bound is violated at 6 of these steps. The average query bound violation for Q_2 is then 0.3 (6/20) and, on average, Q_2 has better accuracy than that of Q_1.

Definition 12 (Tuple Arrival Rate)**.** The data stream tuple arrival rate, denoted by λ, is the number of tuples received in a given time instance.

Intuitively, a higher tuple arrival rate translates into less imprecision as more tuples are available to form equivalence classes with fewer false-positives.

3.2 The PACE Problem

The Precision-bounded Access Control for privacy-preserving data strEams (PACE) problem is defined as follows:

Definition 13 (The PACE Problem)**.** Given a data stream $T[i]$, a set of predicate sliding-window queries Q, and privacy parameter k_s, the Precision-bounded Access Control for privacy-preserving data strEams (PACE) problem is to generate an anonymized stream $T^P[i]$ such that the average query bound violation sum for all queries $q \in Q$ is minimized.

The optimal k-anonymity problem has been shown to be NP-complete for generalization [23]. The hardness result for the PACE problem follows the construction of LeFevre et al. [24] that shows the hardness of k-anonymous multi-dimensional partitioning with the smallest average equivalence class size. The decision problem for k-anonymous partitioning while satisfying the query imprecision bounds for relational data has been shown to be NP-complete [22].

For the decision version of the problem, we consider a single time instance and a set of queries $q \in Q$. The data stream tuples received at that time instance can be transformed into an equivalent set of distinct $(tuple, count)$ pairs. All the queries are evaluated at this time instance. The constant qv defines an upper bound for the sum of the average query bound violation for all predicate sliding-window queries. The tuples received can either be published as partitions or can be put on hold. The decision version of the PACE problem is as follows:

Definition 14 (The Decisional PACE Problem)**.** Given a set $t \in T$ of unique $(tuple, count)$ pairs received at a given time instance and a set of sliding-window queries $q \in Q$ with imprecision bounds B_q, does there exist a multidimensional partitioning for T such that every published multidimensional region R_i in T^p, $\sum_{t \in R_i} count(t) \geq k_s$ and sum of average query bound violation for all queries is less than the positive constant qv?

Theorem 3.1. *Decisional PACE problem is NP-complete.*

Proof: Refer to Appendix A. □

3.3 Precision-bounded Access Control for Privacy-preserving Data Streams

A precision-bounded access control framework for privacy-preserving data streams is proposed(refer to Figure 4). The privacy protection mechanism ensures that the privacy and precision goals are met before the sensitive stream data is made available to the access control mechanism. The access control policy administrator defines sliding-window queries that define the authorized view of the data stream for each role. The Privacy Protection Mechanism (PPM) uses generalization of stream data tuples to anonymize and satisfy the given privacy requirement. The generalization adds uncertainty and precision of the authorized view is reduced. The uncertainty due to generalization can be reduced by delaying the stream tuples and forming equivalence classes with less imprecision. However, the delay introduces false-negatives if the stream tuples belonging to the authorized view are being held by PPM. The access control policy administrator provides the imprecision bound for each query and PPM is required to ensure that at the time of query evaluation the sum of false-negatives and false-positives is less than the imprecision bound.

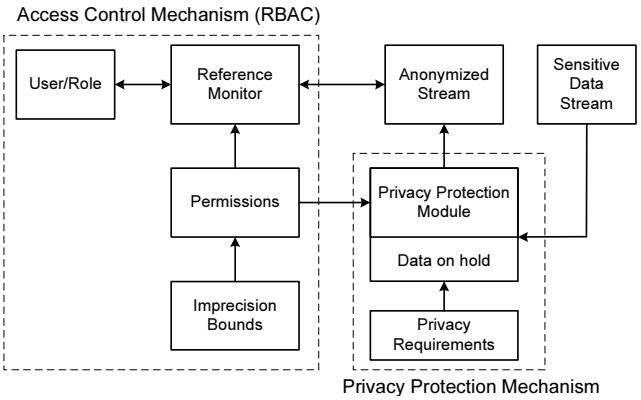


Fig. 4. Precision-bounded access control for privacy-preserving data streams

The purpose of access control is to ensure that each user accesses only authorized information. False-positives due to generalization under the overlap semantics mean that access is being provided to unauthorized tuples. False-negatives although deny

access to the authorized information but ensure that the access control policy is not being violated. The reference monitor can be set for relaxed access control enforcement using the overlap semantics or strict access control enforcement using the enclosed semantics. For the overlap semantics, the reference monitor may deny access to a permission if false-positives are more than a desired threshold.

3.4 Probabilistic Analysis for Query Bound Violation

A precision-bounded access control framework for privacy-preserving data streams has been presented in Section 3.3. The access control policy administrator sets the imprecision bound for each predicate sliding-window query and requires that the imprecision bound for the least number of queries is violated by the PPM. The policy administrator might revise the imprecision bounds for the queries if it is known that the probability of satisfying the bound of large number of queries at any time instance is very low. From this perspective, we are interested in answering the following questions:

- What is the probability that the number of queries violating imprecision bounds is less than a given threshold or is in a given range at any given time instance?
- How long it takes for the sum of average query-bound violation for all queries to reach a steady-state value?

Let $X_1[j], \ldots, X_n[j]$ be a set of independent random variables such that $Pr(X_i[j] = 1) = p_i$ and $Pr(X_i[j] = 0) = 1 - p_i$, where $0 \leq p_i \leq 1$. $X_i[j]$ is a random variable that is equal to 1 if a sliding-window query say, Q_i (size = w and step = 1) evaluated at time instance j violates the imprecision bound, otherwise is equal to 0. The step size for all n queries is 1. Thus, all queries are evaluated at each time instance. The number of queries violating imprecision bounds at time instance j is $X[j] = \sum_{i=1}^{n} X_i[j]$. $X_1[j], \ldots, X_n[j]$ at each time instance are called *Poisson trials* and follow a *Poisson binomial* distribution. The expected number of queries violating the imprecision bound at time instance j is $E[X[j]] = \mu = \sum_{i=1}^{n} p_i$ [25]. Dependency exists among the sliding-window queries evaluated at each time instance but for our analysis we assume that they are independent.

By the law of large numbers, the difference between the actual and expected values for a random process decreases as the number of trials increases. Formally, for a set of independent non-identically distributed random variables $X_1[j], \ldots, X_n[j]$ and $X[j] = \sum_{i=1}^{n} X_i[j]$, the sample average $\bar{\mu} = \frac{1}{j} \sum_{i=1}^{j} X[i]$ at time j, converges to the expected value $\mu = E[X[j]]$ as j approaches ∞ [26]. The sample average $\bar{\mu}$ for a large number of samples can be used to answer the first question: What is the probability that the number of sliding-window queries violating bounds is less than a given threshold? We use the Hoeffding/Chernoff bound [27] for the *Poisson trials* as given in Lemma 3.2.

Lemma 3.2. *Let $X_1[j], \ldots, X_n[j]$ be an independent Poisson trial at time instance j, then for $X[j] = \sum_{i=1}^{n} X_i[j]$, $\mu = E[X[j]]$, and $0 < \epsilon \leq 1$, we have*

$$Pr[X[j] < (1 - \epsilon)\mu] < e^{-\frac{\mu\epsilon^2}{2}} \tag{5}$$

The ϵ value is set according to the required threshold. However, in order to use Lemma 3.2 we would like to know the sample size that would give a high probability of $|\bar{\mu} - \mu|$ being smaller than some constant x. We give Theorem 3.3 to find S, the sample size, such that $|\bar{\mu} - \mu| < x$ for a given probability. The proof for Theorem 3.3 is similar to the proof of the generalized pairwise-independent sampling theorem [28].

Theorem 3.3. *Let $X_1[j], \ldots, X_n[j]$ be an independent Poisson trial at time instance j. $X_i[j]$ is a random variable that is 1 if a sliding-window query say, Q_i (with size = w and step = 1) that is evaluated at time instance j violates the imprecision bound, otherwise $X_i[j]$ is 0. The Var$[X_i] \leq b$ for $b \geq 0$, $X[j] = \sum_{i=1}^{n} X_i[j]$, $\bar{\mu} = \frac{1}{S} \sum_{i=1}^{S} X[i]$, $\mu = E[X[j]]$, and $x > 0$, we have*

$$S \geq \frac{bn}{Pr[|\bar{\mu} - \mu| > x]x^2} \tag{6}$$

Proof: Refer to Appendix A. □

Example 4. Suppose that for 500 predicate sliding-window queries (i.e., n=500) the tolerance for $\bar{\mu}$ is $x = 2.5$ and we want our estimate to be within tolerance with a 95% probability. The Var$[X_i]$ is equal to $p_i(1 - p_i)$ and the maximum value b of p_i in the interval $0 \leq p_i \leq 1$ occurs for $p_i = \frac{1}{2}$. From Equation 6, we get a sample size of 800, which means that when 800 time-stamps have elapsed, there is a 95% probability that $|\bar{\mu} - \mu| < 2.5$.

The *Poisson binomial* distribution for a large number of samples can be approximated by a normal distribution of sample mean $\bar{\mu}$ and standard deviation $\bar{\sigma}$ by the central limit theorem [29]. The cumulative distribution function of the approximate normal distribution can then be used to find the probability that the number of queries violating the imprecision bounds is in a given range at any time instance.

4 Algorithm for Precision-bounded Anonymization

Cao et al. [8] have proposed a clustering algorithm for anonymization of a data stream. Another approach proposed by Zhou et al. [9] uses an R-tree [30] based algorithm to anonymize. The stream tuples are added to leaf nodes in an R-tree with a constraint that each node should have between k to $2k$ tuples. When a leaf node is published, that node is removed from

the R-tree. The proposed heuristic listed in Algorithm 1 can be applied to both techniques for a given predicate sliding-window query workload. We follow the approach suggested by Zhou et al. but use an R^+-tree [31] instead of an R-tree. The R^+-tree-based anonymization algorithm for relational data has been proposed by Iwuchukwu et al. [32]. When the tuples are added to an R^+-tree, the leaf nodes and intermediate nodes are non-overlapping [32]. This condition can be maintained in case of data streams under the assumption that the stream tuple identity value is not repeated within the maximum delay. However, if the tuple identity value is repeated, then according to the data stream k_s-anonymity definition, the tuple cannot be put into the same leaf node. Adding that tuple to any other leaf node will create an overlapping leaf node. It is observed in the experiments in Section 5 that better accuracy can be achieved by maintaining non-overlapping leaf nodes and by setting the maximum delay such that the existing leaf nodes are published before any duplicate tuple is received.

An R^+-tree-based index is maintained at the PPM. The data stream tuples in T are first added to the active R^+-tree at each time instance. Then, the decision to publish each leaf node (equivalence class) is taken. For a leaf node in the active R^+-tree, the expected false-positives and the expected false-negatives are defined as follows.

Definition 15 (Expected False-Positives (EFP))**.** The Expected False-Positives for a leaf node partition P (EFP_P, for short) is defined as the sum of false-positives for all queries resulting from Partition P if the partition is published at the current time instance.

$$EFP_P = \sum_{Q_j \in Q} |P - Q_j| \tag{7}$$

In Equation 7 above, the minus sign denotes the set difference operation that gives the data stream tuples that are inside the partition but are outside the region defined by the predicate sliding-window query Q_j.

Definition 16 (Expected False-Negatives (EFN))**.** The Expected False-Negatives for a leaf node partition P (EFN_P, for short) is defined as the sum of false-negatives for all queries that will be evaluated at the next time instance resulting from Partition P if the partition is held by the PPM at the current time instance.

$$EFN_P = \sum_{Q_j \in Q} |Q_j(P)| \tag{8}$$

In Equation 8 above, only those sliding-window queries add false-negatives that will be evaluated in the next time instance.

A leaf-node in the R^+-tree can either contribute false-positives (if published) or false-negatives (if held) to the sliding-window queries. Therefore, we choose the option that contributes less imprecision for all queries from a partition. This means that a leaf-node partition can be held in the active R^+-tree till the time when EFP_P becomes smaller than EFN_P. We further define w_{FP} and w_{FN} as weights where $0 < w_{FP}, w_{FN} \leq 1$. The weight assignment should be done according to requirements of the application, e.g., w_{FP} can be set less than 1 for an application sensitive to false-negatives.

The Total Imprecision Minimization (TIM) algorithm is listed in Algorithm 1. The active R^+-tree is initialized in Line 1. In the **for** loop in Lines 2-5, at each time instance, the tuples arriving in data stream T are added to the active R^+-tree and the leaf nodes are split if the size of a leaf node is greater than $2k_s$. The leaf nodes are split along the median in the dimension having the least expected false-positives. The **for** loop in Lines 6-9 checks all the leaf nodes of the active R^+-tree. If the expected false-negatives by holding a leaf node are more than the false-positives by publishing the node, then the node is published as an equivalence class. For $w_{FP} = w_{FN}$, the sum of false-negatives for all queries will always be greater than total false-positives because until a partition is published, it only adds false-negatives. Access control enforcement under the overlap semantics can be adjusted to set a preference for lower false-positives ($w_{FP} > w_{FN}$) or lower false-negatives ($w_{FN} > w_{FP}$). The same algorithm can be used to satisfy the privacy requirements of l_s-diversity and variance diversity by publishing the leaf nodes only when they meet the privacy condition.

Algorithm 1: Total imprecision minimization

Input : T, k_s, Q, and B_{Q_j}
Output: $EC_1, EC_2, \ldots$

```
1 Initialize the active R+-tree
2 for (each t_m ∈ T arriving at time instant i) do
3     Add Tuple t_m to the active R+-tree
4     if (size of leaf node > 2k_s) then
5         Split the leaf node
6 for (all leaf nodes P in active R+-tree at time instant
  i) do
7     Update the imprecision cost of each leaf node
8     if (w_FN * EFN_P > EFP_P * w_FP) then
9         Publish the leaf node as EC and remove
          from active R+-tree
```

Example 5. Assume that eight tuples are received at some time instance as shown in Figure 5. The shaded rectangles with solid lines represent sliding-window queries while the rectangles with dotted lines represent partitions. The leaf-node Partitions P1 and P2 are added to the R^+-tree. The weight assignment is $w_{FP} = w_{FN} = 1$. The two sliding-window queries

Q_1 and Q_2 will be evaluated at the next time instance. The EFP for P1 is 1 and EFN is 3. Since EFP is less than EFN for P1, P1 is published. For P2, EFP is 3 and EFN is 1. Since EFN is less than EFP for P2, P2 is put on hold by the PPM. The total imprecision contributed by P1 and P2 at this time instance is 2. Note that if both P1 and P2 are published or held by PPM the total imprecision will be 4.

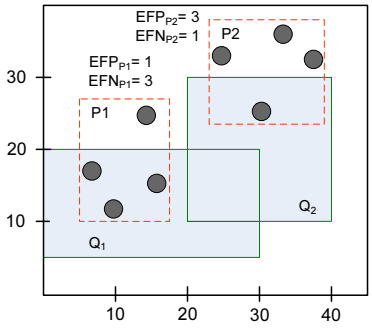


Fig. 5. EFP_P and EFN_P for leaf nodes $P1$ and $P2$

5 Experiments

The experiments for the empirical evaluation of the proposed algorithm have been carried out on two real datasets. The first dataset is the Adult dataset from UC Irvine Machine Learning Repository [33] having 45222 tuples and is the benchmark for k-anonymity research. The attributes in the Adult dataset are: Age, Work class, Marital status, Relationship, Race, Gender, Education, and, Occupation. The second dataset is the Census dataset [34] from IPUMS[1]. This dataset is extracted for the year 2001 using attributes: Age, Gender, Marital status, Race, Language, Education, Occupation, and Income. The size of this dataset is about 1.2 million tuples. For the k_s-anonymity experiments, we use the first six attributes as quasi-attributes from both datasets. To model the dataset as a data stream, we assume that 1000 tuples are received at each time instance. The maximum delay constraint δ is set to 5 time units for the Adult dataset and 25 time units for the Census dataset. It is assumed that the time interval between the two time instances is enough to update the R^+-tree and the query imprecision at each time instance. We also assume that the tuple ID is not repeated within the time duration of the maximum delay.

We use 100 and 300 queries as the workload/permissions for the Adult and Census datasets, respectively. The queries are generated randomly using the approach suggested by Iwuchukwu, et al. [32]. In this approach, two tuples are selected randomly from the tuple space and a query is formed by making a bounding box of these two tuples. The bounding box gives the predicates for the sliding-window query and then the window size and step are also selected randomly from a fixed range. For the Adult dataset, the range for the window size is 20-30 and for the step is 10-20, while for the Census dataset, the range for the window size is 100-200 and for the step is 50-100. The random query is then added to the workload if the sliding-window query meets the size constraint for the first step (8000 for the Adult dataset and 15000 for the Census dataset). The imprecision bounds for all sliding-window queries are set based on the query size at the time of query evaluation. An imprecision bound of 10% for a sliding-window query means that, at each step, when the query is evaluated, the imprecision should be less than 10% of the query size at that time instance.

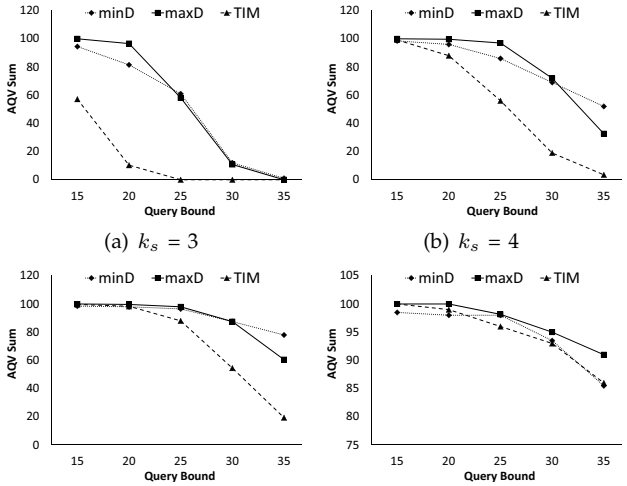
The approach proposed by Cao et al. [8] for data stream anonymization tries to minimize the error due to generalization with a constraint that tuples must be published before the maximum delay deadline. Zhou et al. [9] propose an R-tree-based approach to anonymize the data stream and propose a minimum-delay heuristic, where tuples are published as soon as they meet the privacy condition. They also propose a randomized algorithm to minimize a delay-based cost function and show that the accuracy can be further improved by taking the tuple distribution into account. In our experiments, we compare the proposed approach (i.e., Total Imprecision Minimization - TIM) with the maximum delay heuristic (denoted by maxD) and the minimum delay publishing (denoted by minD). w_{FP} and w_{FN} are set to 1 for TIM.

5.1 Varying Imprecision Bound

For the k_s-anonymity experiments, the value of k_s is fixed and the query imprecision bound is varied from 15% to 35% with increments of 5 and the sum of the average query-bound violation for all predicate sliding-window queries is evaluated. The results for k_s-anonymity are given in Figure 6 for the Adult dataset for k_s values of 3, 4, 5 and 6. The minimum delay heuristic is better than the maximum delay heuristic but TIM gives the best results for all values of k_s.

For the Census dataset results for k_s-anonymity are given in Figure 7 for k_s values of 3, 4, 5 and 6. The performance of TIM is better than those of minD and maxD for all values of k_s in the Census data. In this case, maxD performs better than minD as compared to the Adult dataset. The performance of maxD is dependent upon the maximum delay value. In the case of the Adult dataset, maxD only performs better than minD when the delay value is 2 as given in Figure 8(a) but the experiments were performed with $\delta = 5$. For the Census dataset with $\delta = 25$, observe that in Figure 9(a), the maxD heuristic is better than minD heuristic.

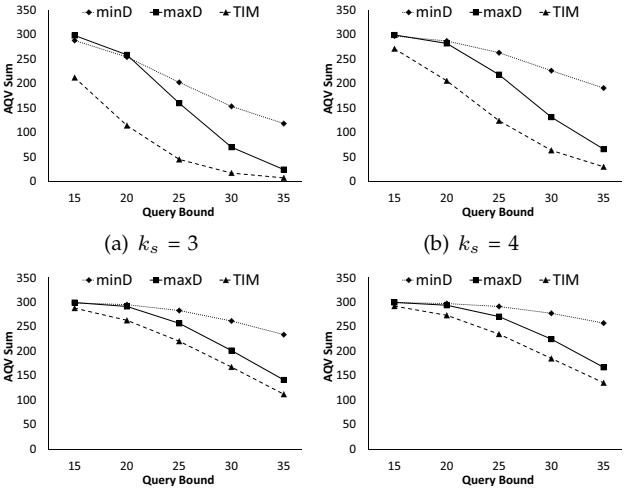
1. Available at http://usa.ipums.org/usa/



(a) $k_s = 3$ (b) $k_s = 4$

(c) $k_s = 5$ (d) $k_s = 6$

Fig. 6. The sum of the average query-bound violation for k-anonymity for the Adult dataset



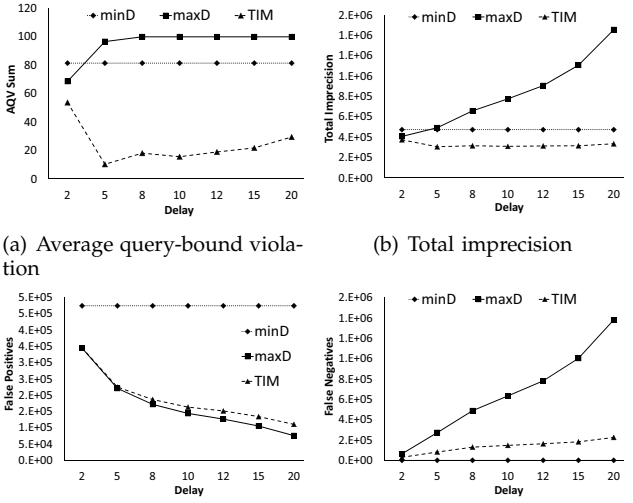
(a) $k_s = 3$ (b) $k_s = 4$

(c) $k_s = 5$ (d) $k_s = 6$

Fig. 7. The sum of the average query-bound violation for k-anonymity for the Census dataset

5.2 Varying the Maximum Delay (δ) Parameter

In the next experiment for the Adult dataset, k_s is set to 3 and the query imprecision bound is set to 20% of the sliding-window query size at the time of query evaluation. Then, the maximum delay value (δ) is varied from 2 time units to 20 time units as given in Figure 8. The total imprecision in Figure 8(b) is the sum of false-positives in Figure 8(c) and false-negatives in Figure 8(d). The total imprecision, false-positives, and false-negatives are calculated by adding, for all queries, the imprecision values at each query evaluation of the sliding-window query. The average query violation and the total imprecision for the minimum delay heuristic remain constant for all values of the δ as they are independent of the delay value. For the minimum delay heuristic, the false-negatives are zero as given in Figure 8(d) because the partitions are published without delay in the same time instance. TIM has the best performance in terms of the average query-bound violation and the total imprecision as given in Figure 8(a) and Figure 8(b). We can see in Figure 8(c) that as the δ value is increased, the total false-positives decrease for TIM as more data stream tuples are available to form partitions with less imprecision. Also, we can observe in Figure 8(d) that as the δ value is increased the false-negatives increase because more tuples are put on hold by the PPM.



(a) Average query-bound violation (b) Total imprecision

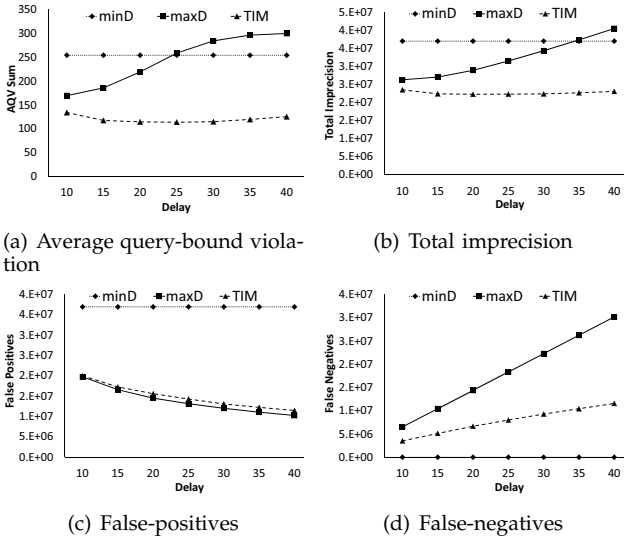
(c) False-positives (d) False-negatives

Fig. 8. Varying the maximum delay (δ) parameter for the Adult dataset

For the Census dataset, k_s is fixed at 3 and query imprecision bound is fixed at 20% of the sliding-window query size at the time of query evaluation. Then, the maximum delay (δ) value is varied from 10 time units to 40 time units in increments of 5 time units as given in Figure 9. Observe that TIM has the best performance in terms of the average query-bound violation and the total imprecision as given in Figures 9(a) and 9(b). For TIM, as the δ is increased, the false-positives decrease as given in Figure 9(c) because more data stream tuples are available to form partitions with fewer false-positives. On the other hand, the false-negatives increase as given in Figure 9(d) with the increase in the δ value as more tuples are held by PPM.

5.3 Varying the Rate of Tuple Arrival λ for TIM

The next experiment is performed to check the effect of the rate of tuple arrival on TIM. Intuitively, the higher rate of tuple arrivals should give better results because more data stream tuples are available at a given time instance to form partitions with fewer false-positives. For this experiment, we assume tuple arrival rates of 250, 500, 750, and 1000 tuples for Adult dataset at each time instance, a k_s value of 3 and an imprecision bound of 25%. We can see in Figure 10(a)

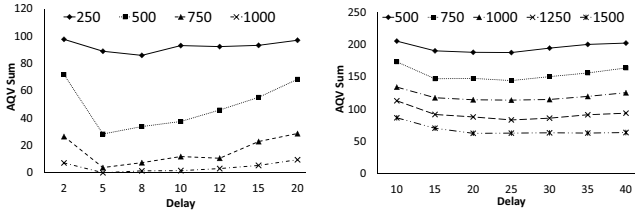


(a) Average query-bound violation (b) Total imprecision

(c) False-positives (d) False-negatives

Fig. 9. Varying the maximum delay parameter (δ) for the Census dataset

that, as the tuple arrival rate is increased, the sum of average query-bound violations decreases for all maximum delay values.



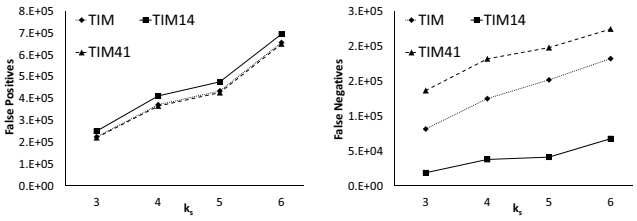
(a) Adult dataset (b) Census dataset

Fig. 10. Varying the rate of tuple arrival for TIM

For the Census dataset the same trend as that of the experiment on the Adult dataset is visible. The performance improves as the rate of tuple arrival increases. For this experiment, we assume tuple arrival rates of 500, 750, 1000, 1250, and 1500 tuples for each time instance, a k_s value of 3 and an imprecision bound of 20%. From Figure 10(b), observe that as the tuple arrival rate is increased the performance is improved because more data stream tuples are available to form partitions with fewer false-positives.

5.4 Varying Weights (w_{FN}, w_{FP}) for TIM

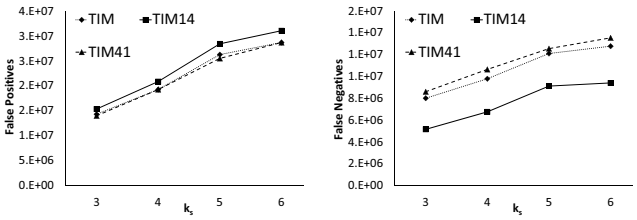
In TIM, a partition is published when the expected false-negatives are greater than the expected false-positives. In this experiment, weights are assigned to expected false-negative and false-positive values. By TIM14, it is meant that w_{FN} is four times as that of w_{FP} (and vice versa for TIM41). Observe that in Figure 11(a), for TIM14, there are more false-positives than for TIM and in Figure 11(b), there are less false-negatives as partitions are published early. Similarly, for TIM41 as compared to TIM, the false-positives are less and the false-negatives are more as partitions are held for a longer time by PPM.



(a) False-positives (b) False-negatives

Fig. 11. The imprecision for TIM using weights for the Adult dataset

We compare false-positives and false-negatives for TIM, TIM14, and, TIM41 for the Census dataset in this experiment as given in Figure 12. TIM14 denotes that the value of w_{FN} is four times as that of w_{FP}. We can see that in Figure 12(a) and Figure 12(b) as more weight is given to EFN in TIM14, the false-negatives decrease and the false-positives increase. Similarly, for TIM41 the false-positives decrease and false-negatives increase as compared to TIM.



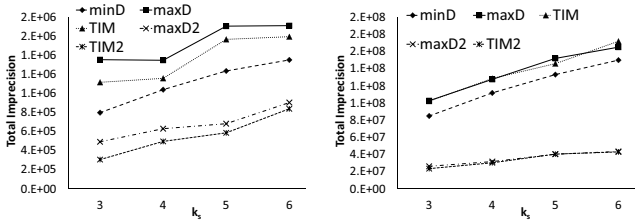
(a) False-positives (b) False-negatives

Fig. 12. The imprecision for TIM using weights for the Census dataset

5.5 Duplicate Tuples Received within the Maximum Delay (δ)

In the previous experiments, it has been assumed that the stream tuple-id values are not repeated. We now assume that a tuple-id can be repeated once within the maximum delay (δ) after time $\frac{\delta}{2}$. The δ is set to 10 and the total imprecision is plotted against k_s. The repetition of tuple-id forces PPM to create overlapping leaf nodes in the R-tree to satisfy the k_s-anonymity requirement. We can see in Figure 13(a) that maxD has the worst performance for all values of k_s because more delay allows more overlapping leaf nodes while minD has the best performance. For TIM2 and maxD2, knowing that a tuple-id is repeated after time $\frac{\delta}{2}$, we set the δ to 5 and use an R^+-tree with non-overlapping leaf nodes. This allows to reduce the total imprecision by an order of 3 as shown for TIM2 in contrast with minD.

In this experiment, for the Census dataset, we assume that a tuple-id can be repeated once within



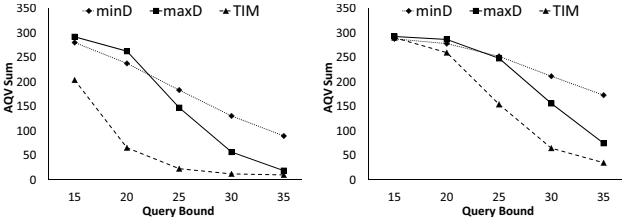
(a) Adult dataset (b) Census dataset

Fig. 13. Duplicate tuples received within the maximum delay (δ)

the maximum delay (δ) after time $\frac{\delta}{2}$. The δ value is set to 20 and the total imprecision is plotted against k_s. The repetition of tuple-id forces the creation of overlapping leaf nodes in the R-tree to satisfy the k_s-anonymity requirement. Notice that in Figure 13(b), the performance of minD is better than maxD and TIM due to fewer overlapping leaf nodes. For TIM2 and maxD2, knowing that the tuple-id is repeated after time $\frac{\delta}{2}$, the δ value is set to 10 and an R^+-tree is used with non-overlapping leaf nodes. This allows to reduce the imprecision by an order of 3 for TIM2.

5.6 l_s-diversity and Stream Variance Diversity

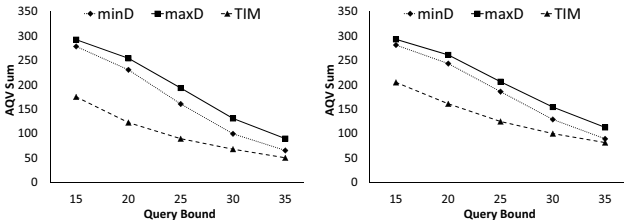
We use Attribute *occupation* as the sensitive attribute and the first six attributes as the QI attributes for the l_s-diversity experiments on data streams using the Census dataset. All the tuples having the occupation value as Not Applicable (0 in the dataset) in the Census dataset are removed leaving about 700k tuples. The proposed algorithm in Listing 1 is used for l_s-diversity with the constraint that each leaf node/equivalence class should be l_s-diverse after splitting and before publishing. The experiment is conducted for the l_s values of 3 and 4. For each value of l_s, we vary the query imprecision bounds from 15% to 35% with increments of 5 and find the sum of average query-bound violation for all sliding-window queries. The results are given in Figure 14 and demonstrate that TIM has the lowest average query-bound violation for l_s-diversity.



(a) l_s = 3 (b) l_s = 4

Fig. 14. l_s-diversity for the Census dataset

For the data stream variance diversity experiments, we use Attribute *income* as the sensitive attribute. All the tuples having the income value as Not Applicable (9999999 in the dataset) in the Census dataset are removed, which leaves about 950k tuples. The experiments are conducted for the variance values $\frac{V}{200}$ and $\frac{V}{100}$, where V is the variance of the sensitive attribute in the dataset. For a variance diversity value , the query imprecision bound is changed from 15% to 35% and the sum of the average query-bound violation for each publishing approach is calculated. Similar to k_s-anonymity and l_s-diversity, each leaf node in the active R^+-tree has to satisfy the variance diversity condition. The results for data stream variance diversity are given in Figure 15 and illustrate that TIM gives the best results as compared to minD and maxD.



(a) $v_s = \frac{V}{200}$ (b) $v_s = \frac{V}{100}$

Fig. 15. Stream variance diversity for the Census dataset

5.7 Sample Size for Sample Mean to Stabilize

In this experiment, the number of sliding-window queries violating bounds for TIM along with the sample mean (average query-bound violation) is given in Figure 16. We randomly select 500 queries (with window size = 25 and step = 1) of size $\geq$ 2500 in the first step and set imprecision bound to 30% of the query size. One of the important observations in this plot is the dependence of *Poisson trials* on query window size. The number of sliding-window queries violating bounds changes abruptly after intervals of about 25 time units. The dependence among trials will decrease if the query window size is reduced. Secondly, in Example 4, the sample size for tolerance $|\bar{\mu} - \mu| < 2.5$ with a 95% probability is found to be greater than 800. Observe that in Figure 16, after 800 time-stamps, $\bar{\mu}$ is almost stable.

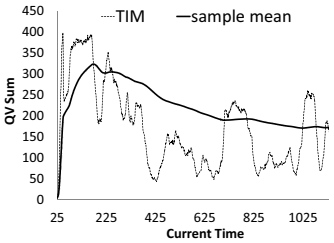


Fig. 16. Average query-bound violation versus Time for the Census dataset

5.8 Visual Representation of Heuristics

The visual representations of the published partitions resulting from the approaches minD, maxD, TIM with R-tree, and TIM with R^+-tree are given in Figure 17. 1200 tuples with two attributes are randomly selected (using a Normal distribution with μ = 50, σ = 10, and cardinality = 100) and it is assumed that rate of tuple arrival λ = 200. Five random predicate sliding-window queries are selected (the query size is greater than 400) with window size 5 and window step 2. The query imprecision bound is set to 15% of the sliding-window query size at the time of query evaluation. The maximum delay δ is set to 3 time-units. The rectangles with the blue (darker) lines are the queries while the rectangles with the red (lighter) lines are the partitions generated by the heuristics at k_s = 3. The partitions held by PPM are given in green (very light) color and the false-negative tuples inside these rectangle are marked by $*$ while the tuples outside the queries are marked by $\times$.

The summary of the comparisons for minD, maxD, TIM with R-tree, and TIM with R^+-tree is given in Table 1. In this table, AQV stands for the Average Query-bound Violation. Minimum delay publishing has zero false-negatives but the highest false-positives as visible in Figure 17(a). Observe that in Figure 17(b), maximum delay publishing allows to reduce the false-positives but adds a lot more false-negatives. In comparison, the partitions published by TIM given in Figure 17(c) have the lowest total imprecision and violates the bounds for the minimum number of predicate sliding-window queries. Figure 17(d) gives the partitions published by TIM using an R-tree instead of an R^+-tree. It is visible that the R-tree approach creates overlapping leaf nodes resulting in higher false-positives.

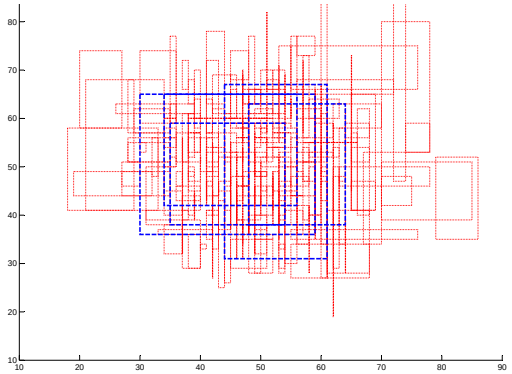
TABLE 1
Comparison of heuristics

	Total FP	Total FN	FP+FN	AQV
minD	403	0	403	3
maxD	126	595	721	5
TIM(R^+-tree)	189	12	201	0
TIM(R-tree)	328	10	338	4

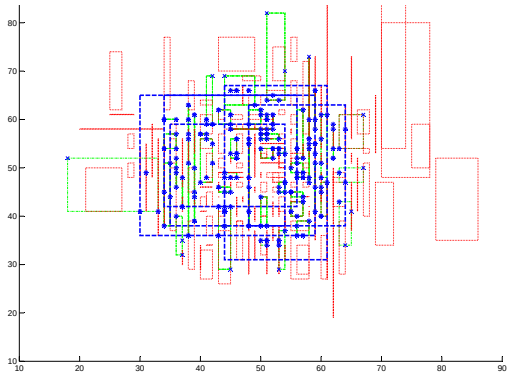
6 Related Work

In the related work, first the literature related to access control on data streams is reviewed and then research related to privacy preserving publishing of data streams is discussed. To the best our knowledge both the precision-bounded access control and privacy together for data streams has not been investigated before.

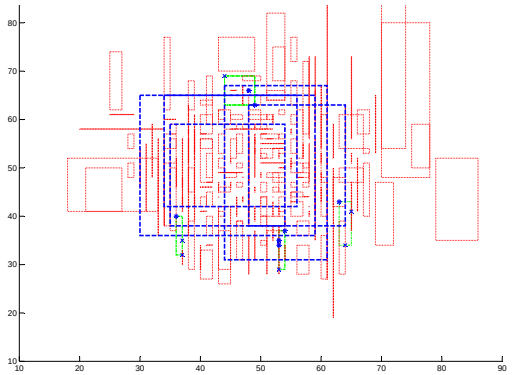
Nehme et al. propose security punctuation-based access control framework for data streams [4], [35]. A



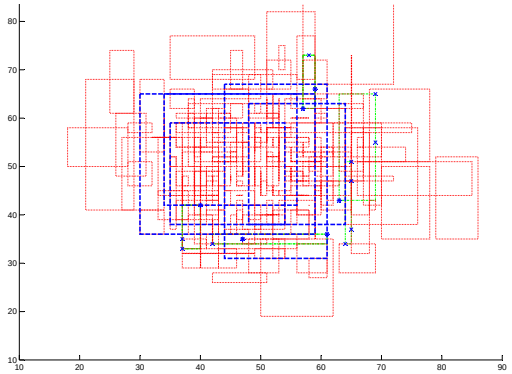
(a) Minimum delay



(b) Maximum delay



(c) TIM with R^+-tree



(d) TIM with R-tree

Fig. 17. Anonymization for two attributes with discrete normal distribution ($\mu = 50, \sigma = 10$)

security punctuation is a predicate that defines access to stream data and is created by the user generating stream data. The security punctuation tuples are then interleaved in the data stream. The subjects are assigned roles on the server and can execute authorized queries on the incoming data stream. The server allows the roles access to stream tuples according to the embedded security punctuation.

Role-based access control for data streams have been proposed by Carminati et al. [5], [36]. In their framework, there are two types of temporal constraints. First is the interval constraint during which the role can access stream data. Second is the window constraint that limits access to the data stream for each role according to the authorized view defined by the sliding-window query predicate. They consider two types of privileges over the authorized data that is read privilege for selection and projection operations and aggregate privilege for Min, Max, Count, Avg, and Sum operations. In the current paper, we follow the access control specification of Nehme et al. and Carminati et al. but further consider the privacy-preservation along with the precision-bounded access control.

Cao et al. have proposed CASTLE for continuously anonymizing data streams [8]. They extend the definition of k-anonymity for data streams and propose a clustering algorithm that publishes anonymized clusters before a given maximum delay deadline. The measure used to assess the quality of published clusters is the information loss metric [37] that does not consider the information loss due to delay in publishing. To overcome this shortcoming, Zhou et al. proposed a delay-based anonymization quality measure that increases the information loss as the publishing delay increases [9]. They propose a randomized-algorithm based on the R-tree. The data stream tuples are added to the active R-tree and the leaf nodes of the tree due at each time instance are published. The due time for each leaf node is evaluated randomly based on the information loss. They further use the distribution density of the data stream to improve the algorithm. Both Cao et al. and Zhou et al. suppress the time-stamp attribute in the anonymized stream. However, the time-stamp attribute is required to evaluate any predicate sliding-window query over the anonymized stream.

Dwork et al. have proposed *differential privacy* for data streams considering a single aggregate query [38]. Cao et al. further extend the model to sliding-window queries over data streams [39]. *Differential privacy* is achieved by adding random noise to original query results and offers better privacy guarantees than generalization, however syntactic anonymization techniques (e.g., generalization) provide better precision [40]. In the current paper our focus is on generalization and we further explore precision bounds for sliding-window queries over privacy-preserving data streams.

Access control and privacy techniques have been investigated for static relational data. LeFevre et al. [13], [24] and Iwuchukwu et al. [32] have proposed workload-aware anonymization for micro data publishing. Work has been done on micro data anonymization with accuracy and privacy constraints [22], [41], [42]. We have proposed the concept of imprecision bounds for accuracy-constrained access control on relational data [22]. However, the access control on data streams presents different challenges because of the temporal constraints defined by sliding-window query predicates. In the data stream literature, access control and privacy-preserving publishing have been considered in isolation. However, we propose a unified precision-bounded access control framework for privacy-preserving data streams.

7 Conclusions

Precision-bounded access control for privacy-preserving data streams has been proposed. The access control administrator defines the permitted view of the data stream along with the required precision. The privacy protection mechanism applies generalization to the stream data such that the privacy requirement is met and imprecision bound for the maximum number of sliding-window queries is satisfied. An algorithm has been proposed to minimize the total imprecision and experiments have been performed to compare the performance. In future work, we plan to extend the access control enforcement to *Enclosed semantics*. Also, in this paper, only sliding-window queries have been considered. We plan to add landmark, snapshot, and historical queries to the set of permitted queries.

Acknowledgments

The work reported in this paper has been partially supported by the National Science Foundation under Grants IIS-1117766 and IIS-0964639.

References

[1] L. Golab and M. Özsu, "Issues in data stream management," *ACM Sigmod Record*, vol. 32, no. 2, pp. 5–14, 2003.

[2] B. Babcock, S. Babu, M. Datar, R. Motwani, and J. Widom, "Models and issues in data stream systems," in *Proceedings of the twenty-first ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pp. 1–16, ACM, 2002.

[3] Y. Wei, S. Son, and J. Stankovic, "Rtstream: Real-time query processing for data streams," in *Ninth IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing*, pp. 10–pp, IEEE, 2006.

[4] R. Nehme, E. Rundensteiner, and E. Bertino, "A security punctuation framework for enforcing access control on streaming data," in *IEEE 24th International Conference on Data Engineering*, pp. 406–415, IEEE, 2008.

[5] B. Carminati, E. Ferrari, J. Cao, and K. Tan, "A framework to enforce access control over data streams," *ACM Transactions on Information and System Security (TISSEC)*, vol. 13, no. 3, p. 28, 2010.

[6] P. Samarati, "Protecting respondents' identities in microdata release," *IEEE Transactions on Knowledge and Data Engineering*, pp. 1010–1027, 2001.

[7] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam, "l-diversity: Privacy beyond k-anonymity," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 1, no. 1, pp. 3–es, 2007.

[8] J. Cao, B. Carminati, E. Ferrari, and K. Tan, "Castle: Continuously anonymizing data streams," *IEEE Transactions on Dependable and Secure Computing*, no. 99, pp. 1–1, 2008.

[9] B. Zhou, Y. Han, J. Pei, B. Jiang, Y. Tao, and Y. Jia, "Continuous privacy preserving publishing of data streams," in *Proceedings of the 12th International Conference on Extending Database Technology: Advances in Database Technology*, pp. 648–659, ACM, 2009.

[10] J. Buehler, A. Sonricker, M. Paladini, P. Soper, and F. Mostashari, "Syndromic surveillance practice in the united states: findings from a survey of state, territorial, and selected local health departments," *Advances in Disease Surveillance*, vol. 6, no. 3, pp. 1–20, 2008.

[11] S. Grannis, M. Wade, J. Gibson, and J. Overhage, "The indiana public health emergency surveillance system: Ongoing progress, early findings, and future directions," in *AMIA Annual Symposium proceedings*, vol. 2006, p. 304, American Medical Informatics Association, 2006.

[12] "The flu season." http://www.cdc.gov/flu/about/season/flu-season.htm.

[13] K. LeFevre, D. DeWitt, and R. Ramakrishnan, "Workload-aware anonymization techniques for large-scale datasets," *ACM Transactions on Database Systems (TODS)*, vol. 33, no. 3, pp. 1–47, 2008.

[14] N. Li, T. Li, and S. Venkatasubramanian, "Closeness: A new privacy measure for data publishing," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 7, pp. 943–956, 2010.

[15] T. Ghanem, A. Elmagarmid, P. Larson, and W. Aref, "Supporting views in data stream management systems," *ACM Transactions on Database Systems (TODS)*, vol. 35, no. 1, p. 1, 2010.

[16] T. Ghanem, W. Aref, and A. Elmagarmid, "Exploiting predicate-window semantics over data streams," *ACM SIGMOD Record*, vol. 35, no. 1, pp. 3–8, 2006.

[17] P. Bonnet, J. Gehrke, and P. Seshadri, "Towards sensor database systems," in *Mobile Data Management*, pp. 3–14, Springer, 2001.

[18] J. Gehrke, F. Korn, and D. Srivastava, "On computing correlated aggregates over continual data streams," *ACM SIGMOD Record*, vol. 30, no. 2, pp. 13–24, 2001.

[19] S. Chandrasekaran and M. Franklin, "Psoup: a system for streaming queries over streaming data," *The VLDB Journal*, vol. 12, no. 2, pp. 140–156, 2003.

[20] D. Ferraiolo, R. Sandhu, S. Gavrila, D. Kuhn, and R. Chandramouli, "Proposed NIST standard for role-based access control," *ACM Transactions on Information and System Security*, vol. 4, no. 3, pp. 224–274, 2001.

[21] B. Carminati, E. Ferrari, and K. Tan, "Specifying access control policies on data streams," *Advances in Databases: Concepts, Systems and Applications*, pp. 410–421, 2007.

[22] Z. Pervaiz, W. Aref, A. Ghafoor, and N. Prabhu, "Accuracy-constrained Privacy-preserving Access Control Mechanism for Relational Data," *IEEE Transactions on Knowledge and Data Engineering*, 2013.

[23] G. Aggarwal, T. Feder, K. Kenthapadi, R. Motwani, R. Panigrahy, D. Thomas, and A. Zhu, "Approximation algorithms for k-anonymity," *Journal of Privacy Technology*, vol. 2005112001, 2005.

[24] K. LeFevre, D. DeWitt, and R. Ramakrishnan, "Mondrian multidimensional k-anonymity," in *Proceedings of the 22nd International Conference on Data Engineering*, pp. 25–25, IEEE, 2006.

[25] W. Hoeffding, "On the distribution of the number of successes in independent trials," *The Annals of Mathematical Statistics*, vol. 27, no. 3, pp. 713–721, 1956.

[26] W. Feller, "The strong law of large numbers," *An Introduction to Probability Theory and Its Applications*, vol. 1, 1968.

[27] R. Motwani and P. Raghavan, *Randomized algorithms*. Cambridge university press, 1995.

[28] A. Meyer, "Deviation from the mean," Spring 2010. Mathematics for Computer Science, MIT OCW.

[29] W. Feller, "The fundamental limit theorems in probability," *Bulletin (New Series) of the American Mathematical Society*, vol. 51, no. 11, pp. 800–832, 1945.

[30] A. Guttman, "R-trees: a dynamic index structure for spatial searching," in *Readings in database systems (3rd ed.)*, pp. 90–100, Morgan Kaufmann Publishers Inc., 1998.

[31] T. Sellis, N. Roussopoulos, and C. Faloutsos, "The r+-tree: A dynamic index for multi-dimensional objects," *The VLDB Journal*, 1987.

[32] T. Iwuchukwu, *Anonymization techniques for large and dynamic data sets*. PhD thesis, The University of Wisconsin-Madison, 2008.

[33] A. Frank and A. Asuncion, "UCI Machine Learning Repository," 2010.

[34] B. Steven, A. Trent, G. Katie, G. Ronald, B. S. Matthew, and M. S., "Integrated Public Use Microdata Series: Version 5.0 [machine-readable database]," 2010.

[35] R. Nehme, H. Lim, and E. Bertino, "Fence: Continuous access control enforcement in dynamic data stream environments," in *IEEE 26th International Conference on Data Engineering (ICDE)*, pp. 940–943, IEEE, 2010.

[36] J. Cao, B. Carminati, E. Ferrari, and K. Tan, "Acstream: Enforcing access control over data streams," in *IEEE 25th International Conference on Data Engineering*, pp. 1495–1498, IEEE, 2009.

[37] G. Ghinita, P. Karras, P. Kalnis, and N. Mamoulis, "Fast data anonymization with low information loss," in *Proceedings of the 33rd international conference on Very large data bases*, pp. 758–769, VLDB Endowment, 2007.

[38] C. Dwork, M. Naor, T. Pitassi, and G. N. Rothblum, "Differential privacy under continual observation," in *Proceedings of the 42nd ACM symposium on Theory of computing*, pp. 715–724, ACM, 2010.

[39] J. Cao, Q. Xiao, G. Ghinita, N. Li, E. Bertino, and K.-L. Tan, "Efficient and accurate strategies for differentially-private sliding window queries," in *Proceedings of the 16th International Conference on Extending Database Technology*, pp. 191–202, ACM, 2013.

[40] C. Clifton and T. Tassa, "On syntactic anonymity and differential privacy," in *ICDE Workshop on Privacy-Preserving Data Publication and Analysis (PRIVDB)*, 2013.

[41] S. Chaudhuri, R. Kaushik, and R. Ramamurthy, "Database access control & privacy: Is there a common ground?," in *Proceedings of the 5th Biennial Conference on Innovative Data Systems Research (CIDR)*, pp. 96–103, 2011.

[42] G. Ghinita, P. Karras, P. Kalnis, and N. Mamoulis, "A framework for efficient data anonymization under privacy and accuracy constraints," *ACM Transactions on Database Systems (TODS)*, vol. 34, no. 2, p. 9, 2009.

Zahid Pervaiz is a PhD candidate at the School of Electrical and Computer Engineering, Purdue university. His research interests are in data privacy, distributed system security and access control.

Arif Ghafoor is a Professor in the School of Electrical and Computer Engineering, Purdue University. His research interests include database security, parallel and distributed computing, and multimedia information systems.

Walid G. Aref is a Professor of computer science at Purdue University. His research interests are in developing database technologies for emerging applications, e.g., spatial, multimedia, genomics, and sensor-based databases. He is also interested in indexing, data mining, scalable media servers, and geographic information systems (GIS).

APPENDIX A
PROOFS

Theorem A.1. *Decisional PACE problem is NP-complete.*

Proof: The proof is by reduction from Partition: **Partition** Given a finite set A and a size function $s(a_i) \in \mathbb{Z}^+$ for each $a_i \in A$. Does there exist a subset, $A^{'} \subseteq A$ such that

$$\sum_{a_i \in A^{'}} s(a_i) \quad = \quad \sum_{a_j \in A - A^{'}} s(a_j) \quad ?$$

For each $a_i \in A$, construct multiple tuples equal to $s(a_i)$. These tuples form a set of distinct pairs of the form $(tuple, count)$. This construct is similar to that in [24]. In each $(tuple, count)$ pair, the tuple is a point in the d-dimensional unit-hypercube defined by a vector $[0_1, \ldots, 0_{i-1}, 1_i, 0_{i+1}, \ldots, 0_d,]$ (i.e., the i^{th} co-ordinate is 1 and all others are 0). The union of all such pairs is the data stream tuples T received at the given time instance. The solution of partition problem will split the unit-hypercube into two regions. In this d-dimensional unit-hypercube, construct two sliding-window Queries q_1 and q_2 having both size and step equal to one and query predicates covering each of the partitions resulting from splitting the d-dimensional unit-hypercube.

The partition problem for A can be reduced to the following: *Let* $k_s = \sum \frac{s(a_i)}{2}$, B_{q_1} = 1, B_{q_2} = 1 *and* qv = 1. *Is there a* k_s*-anonymous multidimensional partitioning for* T *such that* $imp_{q_1} \leq B_{q_1}$ *and* $imp_{q_2} \leq B_{q_2}$ *?* We claim that there is a solution to the k_s-anonymous multidimensional partitioning for T satisfying the imprecision bound for the sliding window queries q_1 and q_2 if and only if there is a solution to the partition problem for A.

Suppose there exists a k_s-anonymous multidimensional partitioning for T satisfying the imprecision bound for the queries q_1 and q_2. The partitions will define two multidimensional regions R_1 and R_2 such that $\sum_{t \in R_1} count(t) = \sum_{t \in R_2} count(t) = k = \sum \frac{s(a_i)}{2}$. The $count(t)$ values in R_1 and R_2 will give the two disjoint subsets of A that define an equal partitioning of A.

In the other direction, suppose there is a solution to the partition problem for A. The solution will define two disjoint subsets A_1 and A_2. From these two subsets, we can find the multidimensional partitions R_1 and R_2 such that $\sum_{t \in R_1} count(t) = \sum_{s(a_i) \in A_1} s(a_i)$ and $\sum_{t \in R_2} count(t) = \sum_{s(a_i) \in A_2} s(a_i)$. The imprecision for the Query q_1 and q_2 is less than B_{q_1} and B_{q_2} as the false positives and false negatives are zero.

Finally, a given solution to the decisional k_s-anonymous multidimensional partitioning problem with imprecision bounds can be verified in polynomial time. All the multidimensional partitions are checked to see if they satisfy the k_s-anonymity requirement and that the imprecision bounds for the queries are satisfied.

□

Theorem A.2. *Let* $X_1[j], \ldots, X_n[j]$ *be an independent Poisson trial at time instance* j. $X_i[j]$ *is a random variable that is 1 if a sliding-window query, say* Q_i *(size =* w *and step = 1), evaluated at time instance* j *violates the imprecision bound otherwise is 0. The* $Var[X_i] \leq b$ *for* $b \geq 0$, $X[j] = \sum_{i=1}^{n} X_i[j]$, $\bar{\mu} = \frac{1}{S} \sum_{i=1}^{S} X[i]$, $\mu = E[X[j]]$, *and* $x > 0$, *we have:*

$$S \geq \frac{bn}{Pr[|\bar{\mu} - \mu| > x]x^2} \tag{9}$$

Proof: First, we find Var$[\bar{\mu}]$ and then use Chebyshev's inequality to find the lower bound on the sample size.

$$\begin{aligned} \text{Var}[\bar{\mu}] &= \frac{1}{S^2} \text{Var}\Big[\sum_{j=1}^{S} \sum_{i=1}^{n} X_i[j]\Big] && (10) \\ &= \frac{1}{S^2} \sum_{j=1}^{S} \sum_{i=1}^{n} \text{Var}\big[X_i[j]\big] \\ &\quad \text{(pairwise independence assumption)} \\ &\leq \frac{1}{S^2} \sum_{j=1}^{S} \sum_{i=1}^{n} b \qquad (\text{Var}[X_i] \leq b) \\ &\leq \frac{nb}{S} && (11) \end{aligned}$$

The Chebyshev's inequality states that

$$Pr[|\bar{\mu} - \mu| > x] \leq \frac{\text{Var}[\bar{\mu}]}{x^2} \tag{12}$$

Using the expression for Var$[\bar{\mu}]$ from Equation 11 and Chebyshev's inequality in Equation 12, we get

$$S \geq \frac{bn}{Pr[|\bar{\mu} - \mu|]x^2}$$

□