# Hiding in the Forest: Privacy-preserving Process Performance Indicators

Martin Kabierski[a], Stephan A. Fahrenkrog-Petersen[a], Matthias Weidlich[a]

[a]*Department of Computer Science, Humboldt-Universität zu Berlin, Berlin, Germany*

**Abstract**

Event logs recorded during the execution of business processes provide a valuable starting point for operational monitoring, analysis, and improvement. Specifically, measures that quantify any deviation between the recorded operations and organizational goals enable the identification of operational issues. The data to compute such process-specific measures, commonly referred to as process performance indicators (PPIs), may contain personal data of individuals, though, which implies an inevitable risk of privacy intrusion that must be addressed.

In this article, we target the privacy-aware computation of process performance indicators. To this end, we adopt tree-based definitions of PPIs according to the well-established PPINOT meta-model. For such a PPI, we design data release mechanisms for the functions in a PPI tree. Using a probabilistic formulation of the expected result of a privatized PPI, we further show how to determine the combination of release mechanisms that inflicts the least loss in utility. Moreover, given a set of PPIs, we provide an algorithmic framework to manage an inherent trade-off: Privatization may strive for maximal utility of each *single* PPI or for maximal reuse of privatized functions among *all* PPIs to use a privacy budget most effectively. Results from experiments with synthetic as well as real-world data indicate the general feasibility of privacy-aware PPIs and shed light on the trade-offs once a set of them is considered.

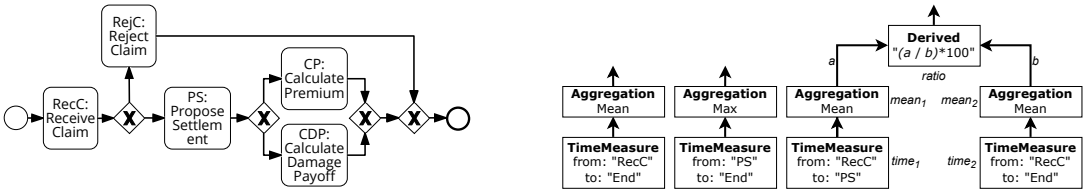
*Keywords:* Performance Indicators, Process Monitoring, Differential Privacy

## 1. Introduction

Once business processes are supported by information systems, it is often possible to extract event data on the execution of individual activities as part of a specific case [1]. Such data, which are commonly captured in the form of event logs, provides a valuable starting point for operational monitoring, analysis, and improvement. In particular, event logs enable conclusions on the extent to which qualitative as well as quantitative properties of process execution are in line with organizational goals. An example for a qualitative assessment is the verification of the conformance of the recorded process behaviour against a behavioural specification [2]. A quantitative assessment, in turn, may refer to the question whether a process shows desired performance characteristics [3].

Monitoring, analysis, and improvement of quantitative properties of a business process are typically driven by the definition of process performance indicators (PPIs), i.e., quantifiable metrics that allow the evaluation of a process efficiency and effectiveness, calculated over the individual cases of a process [4]. We illustrate the notion of a PPI with a simple claim handling process. As illustrated using the BPMN diagram in Fig. 1a, such a process may involve the receipt of a claim (`RecC`), which is either rejected (`RejC`) or processed. In the latter case, a settlement is proposed (`PS`), before the premium (`CP`) or the damage payoff (`CDP`) are calculated. For this process, Fig. 1b illustrates three PPIs. They follow the PPINOT meta-model [4], which defines PPIs as trees of functions. Specifically, the first PPI assesses the time required from the receipt of a claim (`RecC`) to the end of the process (`End`), and is defined by the mean of the observed values. The second PPI considers the maximal time between the settlement (`PS`) and the process end (`End`). The third PPI, in turn, is based on two temporal measures, the average time from the claim receipt (`RecC`) to the settlement (`PS`) and the process end (`End`), respectively. It then defines

*Email addresses:* martin.kabierski@hu-berlin.de (Martin Kabierski), stephan.fahrenkrog-petersen@hu-berlin.de (Stephan A. Fahrenkrog-Petersen), matthias.weidlich@hu-berlin.de (Matthias Weidlich)




(a) A simple claim handling process.

(b) A forest consisting of three PPIs.

Figure 1: Example scenario for the evaluation of privacy-preserving process performance indicators.

the ratio, in percent, of the former duration.

Given that event logs contain information about the individuals involved in process execution, the computation of PPIs implies an inevitable risk of privacy intrusion. In the above example, an event log may contain personal data of claimants as well as the knowledge workers handling the claims. Such sensitive data may take the form of financial information such as income, health-related data such as an HIV diagnosis, or personal information such as relationship status, depending on the process associated with the data. Thus, if an adversary obtains this data utilizing multiple queries on the recorded data, it may gain access to such sensitive information that can be exploited and inflict harm on the target individual [5]. Yet, the data can be relevant for the ongoing optimization of the processes at large, as it gives insight into the general trends of the recorded process executions. Addressing the privacy risk is not merely an ethical consideration, but also enforced by legal frameworks, such as the GDPR [6] and the CCPA [7]. They restrict the use of personal data without consent, especially for secondary use of the data, i.e. purposes other than those for which the data were originally recorded. Yet, these regulations allow the usage of the data for said purposes, given that the data is properly anonymized. It has been shown, that the mere removal of personal identifiers in process logs does not offer sufficient protection against privacy attacks [8]. Therefore, sophisticated privacy protection is necessary [9]

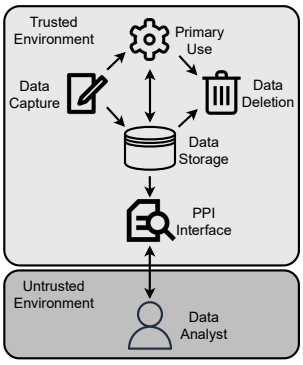



Figure 2: Architecture for privacy-preserving PPIs.

Against this background, we argued for an architecture as the foundation for privacy-preserving PPIs [5], which is illustrated in Fig. 2. It separates the trusted environment of the process owner, who captures and stores event data for the primary purpose of supporting the execution of the process, from the untrusted environment of the process analyst, who intends to compute PPIs over the event data for some secondary purpose involving process monitoring, analysis, and improvement. To avoid that the privacy of an individual is compromised, i.e., to avoid that their individual contribution to a PPI is revealed, access from the untrusted environment to the event data is guarded by a query interface. Given a PPI definition, the interface realizes a data release mechanisms that computes the result and adds noise to it, thereby guaranteeing differential privacy. This way, the evaluation of a PPI reduces a privacy budget that represents the strength of the obtained guarantee.

In this paper, we provide a comprehensive instantiation of the architecture, which is derived by answering the following research questions:

Q1: *How to privatize the functions of a PPI?* Data privatization typically induces a trade-off between the strength of a privacy guarantee and a loss in data utility. As such, the latter shall be minimized for a desired strength of the guarantee. However, privatization models for traditional aggregates [6, 7] ignore the structure of process-related data, and of the PPIs defined over them. Hence, we devise dedicated release mechanisms for the functions of PPIs that follow the PPINOT meta-model.

Q2: *How to best privatize a PPI?* While the above release mechanisms target individual functions, different combinations of these mechanisms may be employed to privatize a complete PPI. Using a probabilistic formulation of the expected PPI result, we show how to determine the combination of release mechanisms that inflicts the least loss in data utility.

Q3: *How to orchestrate the release mechanisms for a set of PPIs?* Once a set of PPIs needs to be evaluated, we face a trade-off between optimizing the utility of each *single* PPI or the reuse of privatized func-

tions among *all* PPIs to use a privacy budget most effectively. We therefore present an algorithmic framework to manage this trade-off.

This paper builds upon an earlier conference paper [5], which introduced the general architecture outlined in Fig. 2 and proposed the release mechanisms for the functions of a PPI (Q1). Here, we extend these results with an approach to use these release mechanisms to privatize a PPI with minimal loss in utility (Q2). Moreover, we lift the approach from a single PPI to a set of PPIs (Q3).

We evaluate our approach with synthetic and real-world data. Specifically, in a series of controlled experiments, we assess the sensitivity of the release mechanisms for various properties of the considered event log. Moreover, in a case study with a public event log in the healthcare domain, we demonstrate feasibility of our approach and shed light on the interplay of the utility of individual PPIs and their joint reduction of a given privacy budget.

In the remainder, we clarify preliminaries for our work in Section 2. We then introduce our solutions to the aforementioned research questions on the privatization of PPI functions (Section 3), on the optimal combination of them for a single PPI (Section 4), and on the orchestration of the trade-off imposed by a set of PPIs (Section 5). The results of our experimental evaluation are presented in Section 6, before we review related work in Section 7 and conclude in Section 8.

## 2. Preliminaries

To provide the foundations for our work, this section first introduces a model for event logs (Section 2.1). We then review the definition of process performance indicators according to the PPINOT meta-model(Section 2.2). In terms of stochastic foundations, we further discuss random variables (Section 2.3), before turning to the concept of differential privacy (Section 2.4).

### 2.1. Notions and Notations for Event Logs

We consider ordered, finite datasets, each being a set of elements $X = \{x_1, \ldots, x_n\}$ that carry a numeric value and are partially ordered by $\leq$. To simplify the presentation, we will consider datasets as sets of integers or real numbers. Yet, in practice, a dataset may contain multiple elements referring to the same numeric value.

The cardinality of the dataset is denoted as $|X| = n$. For one of the (potentially many) elements of $X$ that are minimal and maximal according to $\leq$, we write $\underline{X}$ and $\overline{X}$, respectively. An interval of the dataset is defined by $I = (x_{lower}, x_{upper})$ with $x_{lower}, x_{upper} \in X$ and $x_{lower} \leq x_{upper}$. Lifting the notation for minima and maxima to $I$, we define $\underline{I} = x_{lower}$ and $\overline{I} = x_{upper}$.

Our notion of an event log is based on a relational event model [8]. That is, an *event schema* is defined by a tuple of attributes $A = (A_1, \ldots, A_n)$, so that an *event* is an instance of the schema, i.e., a tuple of attribute values $e = (a_1, \ldots, a_n)$. An event schema consists of at least three attributes: The *case* that identifies the process instance to which an event belongs, the *timestamp* for the point in time an event has been recorded, and the *activity*, for which the execution is signalled by an event. The *timestamp*-ordered list of events corresponding to a single *case* is called a *trace*. Such a trace represents the execution of a single process instance. An *event log* is a set of traces.

### 2.2. Process Performance Indicators

A *key performance indicator (KPI)* is a metric that quantifies, to which extent the goals set for an organisation are fulfilled. A *process performance indicator (PPI)* is a KPI, which is related to a single business process and which is evaluated solely based on the traces recorded during process execution. The *Process Performance Indicator Notation (PPINOT)* [**?** ] is a meta-model for the definition and evaluation of PPIs. At its core, the PPINOT model relies on the composition of measures, i.e., simple, well-defined functions that enable the definition and automated evaluation of more complex PPIs:

*Base measures* concern a single instance of a process and include event counts (e.g., to count activity executions), timestamp differences between events, the satisfaction of conditions, or aggregations over the events' attribute values.

*Aggregation measures* are multi-instance measures that combine values from multiple process instances into a single value. PPINOT includes aggregation measures to calculate the minimum, maximum, mean, and sum of a set of input values.

*Derived measures* are user-defined functions of arbitrary form, applied to a single process instance, or a set thereof.

A PPI defined using the PPI meta-model is a function composition tree, so that a set of PPIs is a forest.

Consider the third PPI of our example in Fig. 1b. Here, the leaf nodes represent base measures, $time_1$ and $time_2$, that assess the timestamp difference between events signalling the receipt of a claim (`RecC`), and events indicating the settlement (`PS`) or the process end (`End`), respectively. In either case, the mean is adopted as an

aggregation measure, $mean_1$ and $mean_2$, before computing the *ratio*, $(a/b) \cdot 100$, which is a derived measure.

### 2.3. Random Variables

To reason about the computation of PPIs, we later need a model of random variables and functions of them. Let $\Omega$ be a sample space of possible outcomes $\omega \in \Omega$ of a random experiment. A random variable is a function $X : \Omega \rightarrow R$, mapping each point in the sample space to a real number $r \in R$. We denote as $X = x$, that random variable $X$ realizes as value $x \in R$, i.e $X$ takes on the value $x$. If $X$ takes on a finite set or countably infinite set of realizations, we call it discrete; and continuous otherwise.

A probability function $f : X \rightarrow [0, 1]$ of a random variable $X$ maps each possible realization of $X$ to a value $p$, s.t. $\Sigma_{x \in \Omega} = 1$. We denote as $P(X{=}x) = f(x)$ the probability of the realization $x$ of $X$. If $X$ is discrete, $f(x)$ is referred to as a probability mass function; otherwise, $f(x)$ is referred to as a probability density function. Furthermore, we define $\mathrm{range}(X)$ as the set of possible realizations of $X$.

For illustration purposes, let $\Omega$ be the result of two coin tosses, i.e., $\Omega = \{H, H\}, \{H, T\}, \{T, H\}, \{T, T\}$. Now, let random variable $X$ be the number of heads of two coin tosses, i.e., $X(\{H, H\}) = 2, X(\{H, T\}) = X(\{T, H\}) = 1, X(\{T, T\}) = 0$. Assuming fair coin tosses, the probabilities of the realizations of $X$ are given as $P(X{=}2) = 0.25$, $P(X{=}1) = 0.5$, $P(X{=}0) = 0.25$.

Next, we consider functions of random variables. Let $X$ be a discrete random variable with probability mass function $f$. Let $g$ be a function. Then, $Y = g(X)$ is also a random variable with $P(Y{=}y) = \Sigma_{x:g(x)=y} f(x)$.

Let $\Omega = \{-1, 0, 1, 2\}$ and let $X$ be a random variable with $P(X{=}-1) = 0.1$, $P(X{=}0) = 0.2$, $P(X{=}1) = 0.3$, and $P(X{=}2) = 0.4$. Now, for $Y(x) = x^2$, we get $P(Y{=}0) = 0.2$, $P(Y{=}1) = 0.4$, and $P(Y{=}4) = 0.4$.

Likewise, the concept can be extended to functions of multiple random variables. Assume that $X_1, X_2, \ldots, X_n$ are discrete random variables with probability mass functions $f_1, f_2, \ldots, f_n$. Let $Y = g(X_1, X_2, \ldots, X_n)$ be a function of multiple discrete random variables. Then, $Y$ is a discrete random variable with $P(Y{=}y) = \Sigma_{x_1,x_2,\ldots,x_n:g(x_1,x_2,\ldots,x_n)=y}\ f_1(x_1) \cdot f_2(x_2) \cdot \ldots \cdot f_n(x_n)$.

### 2.4. Differential Privacy

Differential privacy [9] is a privacy guarantee that limits the impact a single element may have on the output of a function $f$ that is computed over a set of elements. Therefore, it limits an adversary to conclude on the set of used input elements (or the presence of a certain input element) from the result of the function. This obfuscation is usually achieved by adding noise, for which the magnitude depends on the sensitivity $\Delta f$ of function $f$, i.e., the $\Delta f$ denotes the maximal impact any element $x \in X$ may have on $f(X)$.

A randomized mechanism $K$ is a randomized function that can be applied to a dataset, with $\mathrm{range}(K)$ as the set of possible results. Let $D_1, D_2$ be two neighbouring datasets, i.e., they differ in exactly one element. The randomized mechanism $K$ provides $(\epsilon, \delta)$-differential privacy, if the following inequality holds for the probabilities of the function result falling into a sub-range of all possible results:

$$\forall\, S \subseteq \mathrm{range}(K) : \\ P(K(D_1) \in S) \;\leq\; e^{\epsilon} P(K(D_2) \in S) + \delta.$$

Differential privacy enforces an upper bound on the difference in result probabilities of neighbouring datasets. If $\delta = 0$, $K$ is said to be $\epsilon$-differentially private (and the $\delta$ may be omitted altogether). We note that larger $\epsilon$ values imply weaker privacy, whereas smaller values yield a stronger privacy guarantee.

In the remainder, we rely on different mechanisms to achieve differential privacy. First, we incorporate the Laplace mechanism [9], which adds noise sampled from a Laplace distribution onto $f(X)$. The distribution is considered with a location parameter $\mu = 0$ and a scale parameter $b = \Delta f/\epsilon$, i.e., the amount of noise is derived from the aforementioned sensitivity of function $f$. Since the Laplace distribution is symmetric, the location is set to zero, and there is an exponential falloff on either side of the distribution, the results of $f(X)$ can be expected to be close to the original values after noise insertion.

However, the symmetric and monotonous falloff of the Laplace distribution also has a downside. In particular, it yields undesirable results in scenarios where result values close to the true result have a disproportionally negative effect on the utility. To avoid this issue, the *exponential mechanism* [10] has been proposed. It constructs a probability space based on a function $q(D, r)$, which assigns a score to all possible results $r \in \mathrm{range}(K)$ based on the input dataset $D$. Here, a higher score is assigned to more desirable results, which implies an exponentially higher result probability. The mechanism then chooses a result $r \in \mathrm{range}(K)$ with a probability proportional to $e^{(\epsilon q(D,r))/(2\Delta q)}$, where $\Delta q$ is the sensitivity of the scoring function, i.e., the maximum change in assigned scores possible for two neighbouring datasets.

Both above mechanisms, the Laplace mechanism and

the exponential mechanism, assume that $\Delta f$ and $\Delta q$ are known beforehand. In applications where these values cannot be determined, the *sample-and-aggregate framework* [11] drops this assumption. It samples subsets of the input set and evaluates the given function per sample. The obtained results are then combined using a known differentially private aggregator. If function $f$ can be approximated well on small sub-samples, then the results per sample are close to $f(X)$. By aggregating these approximated results using a differentially private mean, i.e., by computing the mean $m$ and adding noise drawn from a Laplacian calibrated with $\Delta(m)$, one achieves a differentially private result for $f(X)$, even though the sensitivity of the function $f$ is unknown.

While a differentially private mechanism achieves privacy by systematically obfuscating the true value, repeatedly querying the mechanism for the same data increases the risk of information leakage. Each privatized value known to an attacker allows the approximation of the obfuscated value, through reconstruction of the probability mass function of the mechanism. Thus, to limit the knowledge obtainable by an attacker, a pre-defined privacy budget $\epsilon_{max}$ is introduced. Whenever the mechanism is queried, the privacy budget is reduced, thereby limiting the number of possible queries for the same data. Once the budget is depleted, no further access is allowed. Here, the reduction of the budget depends on $\epsilon$, i.e. larger values for $\epsilon$ exhaust larger amounts of the budget, since there is an increased risk of de-anonymization.

## 3. Privatizing Process Performance Indicators

To realize the architecture for privacy-preserving PPIs in Fig. 2, this section first outlines how the structure of PPIs in terms of function composition trees can be exploited for privacy protection in Section 3.1. Based thereon, Section 3.2 introduces a set of data release mechanism that guarantee $\epsilon$-differential privacy.

### *3.1. Using Function Composition Trees for Privacy Protection*

Our idea is to exploit the compositional nature of PPIs defined in the PPINOT meta-model for privacy protection. Instead of adding noise to the final query result, we introduce noise, with smaller magnitude, at the inner functions of a PPI. Such a compositional approach still guarantees $\epsilon$-differential privacy of the result. At the same time, it enables us to minimize the overall introduced error. Hence, data utility is preserved to a higher degree, which leads to more useful process analysis, under the same privacy guarantees.

We aim to protect the privacy of individuals, of whom personal data is materialized in a trace. Hence, the results of base measures shall be protected. However, common PPIs assess the general performance of process execution by aggregating these results in multi-instance measures (aggregation or multi-instance derived measures), so that guarantees in terms of differential privacy may be given for these measures. This raises the question of selecting a subset of the multi-instance measures for privatization. On the one hand, this selection shall ensure that the results of *all* aforementioned single-instance measures are protected. On the other hand, the selection shall be *minimal* to keep the introduced noise to the absolutely necessary magnitude.

We capture the above intuition with the notion of an admissible set of measures of a PPI. Let $(F, \rho)$ be the function composition tree of a PPI, with $F$ as the set of measures and $\rho : F \rightarrow 2^F$ as the function assigning child measures to measures. With $\rho^*$ as the transitive closure of $\rho$, a set of measures $F' \subseteq F$ is admissible, if it satisfies the following conditions:

- it contains only multi-instance measures:
  $f \in F'$ implies that $f \in \mathrm{dom}(\rho)$;
- it covers all trace-based measures:
  $\forall\, f \in (F \setminus \mathrm{dom}(\rho)) : \exists\, f' \in F' : f \in \rho^*(f')$;
- it is minimal:
  $\forall\, F'' \subset F' : \exists\, f \in (F \setminus \mathrm{dom}(\rho)) : \forall\, f'' \in F'' :$
  $f \notin \rho^*(f'')$.

The first condition of an admissible set applies, as differential privacy may only be used for the aggregation of multiple inputs, thus single-instance measures cannot be privatized with the given privacy framework. The second condition ensures, that the selected set of functions privatizes all base measures, that directly access trace information. Finally, the third condition ensures, that only the minimum amount of noise to achieve $\epsilon$-differential privacy is added onto the intermediate results.

The function composition tree of the third PPI in Fig. 1b has two sets of admissible measures, one including only the derived measure, $\{ratio\}$, and one including both aggregation measures, $\{mean_1, mean_2\}$. Either way, both single-instance measures that refer to timestamp differences, $time_1$ and $time_2$, are covered. In contrast, the set $\{mean_1\}$ is not admissible, as it would leave the single-instance measure $time_2$ uncovered (second condition). Likewise, selecting both base measures, $\{time_1, time_2\}$, or the set $\{mean_1, mean_2, ratio\}$ would not be admissible either. The former would violate the first condition on the inclusion of multi-instance measures, while the latter would violate the third condition on the minimality.

A set of measures that is admissible induces one specific way to privatize a single PPI by incorporating a release mechanism for each of the measures. While we later discuss how to chose among different admissible sets of measures, also in the light of multiple PPIs, we first turn to the definition of the release mechanisms.

### *3.2. Release Mechanisms for Multi-Instance Measures*

The design of a release mechanism for a specific multi-instance measure is influenced by (i) the ability to assess the domain of input values over which the measure is evaluated, and (ii) the ability to assess the sensitivity of the measure. As for the first aspect, we can rely on an estimation of the respective domain. Here, a simple estimation is based on the minimal and maximal values, $\underline{X}$ and $\overline{X}$, of the dataset $X$ used as input for the measure (i.e., the result of the child measures). The bounds may be extended by constant offsets to account for the fact that the dataset $X$ is merely a sample of an unknown domain. The sensitivity of the measure, in turn, depends on the semantics of the measure. While for the aggregation functions of PPINOT, this sensitivity may be estimated, it is unknown in the general case of derived measures.

Below, we first introduces three release mechanisms for aggregation measures: an instantiation of the Laplace mechanism; an interval-based mechanism based on the exponential mechanism; and a threshold-sensitive mechanism that extends the interval-based one to preserve the significance of a measure related to a threshold. Finally, we discuss how derived measures, in the absence of an estimate of their sensitivity, can be privatized using a sample-and-aggregate strategy.

**Laplace Mechanism for Aggregation Measures.** Privatization of an aggregation measure can be based on the addition of Laplace noise to the actual result. As mentioned, this requires to estimate the sensitivity $\Delta f$ of the given aggregation function, i.e., the maximal impact any element $x \in X$ may have on $f(X)$. For the aggregation functions of the PPINOT meta-model, the sensitivity is derived as $\Delta(min) = \Delta(max) = |\overline{X} - \underline{X}|$, $\Delta(sum) = \overline{X}$ and $\Delta(mean) = |\overline{X} - \underline{X}|/|X|$. Based thereon, noise from a Laplacian (with parameters $\mu = 0$ and $b = \Delta f/\epsilon$, see Section 2.4) is added to $f(X)$.

Since the sensitivity $\Delta f$ directly influences the magnitude of added noise, for $mean$ measures, this mechanism potentially leaks information about the number $|X|$ of process instances (and hence, individuals) within the given scope. An adversary may conclude on the difference $|\overline{X} - \underline{X}|$ based on the magnitude of noise from another PPI incorporating a $min$ or $max$ measure and, based thereon, derive $|X|$ from the magnitude of noise in a PPI with a $mean$ measure. However, in practice, $|X|$ may be revealed explicitly to enable a process analyst to assess the statistical reliability of the PPI result.

**Interval-based Mechanism for Aggregation Measures.** The drawback of the Laplace mechanism is the inherently high sensitivity, which scales linearly with the domain of input values. Our idea, therefore, is to group similar result values into intervals and score them using the exponential mechanism. This way, we obtain a release mechanism with a score function sensitivity $\Delta q = 1$, which ultimately leads to a smaller magnitude of noise for large domains of input values.

To realize this idea, our interval-based release mechanisms consists of three phases:

(1) *Interval creation:* We partition the domain and the range of the aggregation function into intervals.

(2) *Interval probability construction:* Scores are assigned to these intervals, which are then converted to result probabilities.

(3) *Result sampling:* Using these probabilities, an interval is chosen as the output interval, from which the result value is sampled.

The *interval creation* is based on the range of the aggregation function, given as $\text{range}(f(X)) = (\underline{X}, \overline{X})$ for $f \in \{min, max, mean\}$ and $\text{range}(f(X)) = (\underline{X} \cdot |X|, \overline{X} \cdot |X|)$ for $f = sum$. This range is split into non-overlapping intervals $I = \{I_0, \ldots, I_n\}$, with $I_0 \cap \ldots \cap I_n = \emptyset$ and $I_0 \cup \ldots \cup I_n = \text{range}(f(X))$. Let $\tau(x_i, x_j) = (x_i + x_j)/2$ be the mean of $x_i, x_j$ and let $I_f$ be the interval containing the result value, i.e. $f(X) \in I_f$. For $mean$ and $sum$, the range of $f(X)$ is divided into evenly spaced intervals of size $\Delta f$, so that $f(X) = \tau(\underline{I_f}, \overline{I_f})$ is the mean of its containing interval. For $min$ and $max$, the range of $f(X)$ is divided into $n$ intervals of different size, for which the boundaries are the means of neighbouring values $\tau(x_i, x_{i+1})$ with $x_i, x_{i+1} \in X$.

Fig. 3 exemplifies the intervals for a dataset $X = \{2, 3, 7, 8, 10\}$. For $min$ and $max$, the interval boundaries are 2.5, 5, 7.5, and 9. For $mean$ and $sum$, the intervals have size $\Delta f = 1.6$ and $\Delta f = 10$, and are centred around $mean(X) = 6$ and $sum(X) = 30$.

The *interval probability construction* relies on a scoring function that assigns higher scores to intervals that are closer to the interval containing $f(X)$. Let $I_1, \ldots, I_n$ be the intervals in the order induced by $\leq$ over their

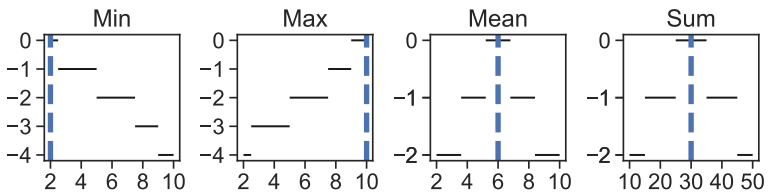



Figure 3: Intervals and scores for the aggregation functions for dataset $X = \{2, 3, 7, 8, 10\}$.

boundaries, and let $1 \leq k \leq n$ be the index of interval $I_f$ containing the result value. Then, the score for each interval $I_i$ is defined as $q(i) = -|k - i|$, as illustrated in Fig. 3 for the example. Here, intervals, that lie closer to $f(X)$, denoted by the blue dashed lines, are scored higher, than those further away. Since each interval $I_i$ corresponds to a set of potential result values, we incorporate the size of this set in the probability computation. Hence, the probability for $I_i$ is defined as:

$$P(I_i) = \frac{|I_i| \cdot e^{(\epsilon \cdot q(i)/2 \cdot \Delta q)}}{\Sigma_{1 \leq j \leq n} |I_j| \cdot e^{(\epsilon \cdot q(j)/2 \cdot \Delta q)}}$$

*Result sampling* chooses one interval based on their probabilities. From this interval one specific value is drawn based on a uniform distribution over all interval values.

**Threshold-Sensitive Mechanism for Aggregation Measures.** The interval-based mechanism is problematic, if a PPI is tested against a threshold, as often done in practice. Consider the dataset $X$ and assume that the *sum* function is the root of a PPI's function composition tree, i.e., $f(X) = 30$ as shown in Fig. 3. Assume that it is important whether the PPI is less or equal than 30. Then, adding noise may change the actual interpretation of the PPI, since the release mechanism will sometimes publish values larger than 30.

To mitigate this effect, we present a threshold-sensitive release mechanism that extends the *interval creation* and *interval probability construction* of the above mechanism. Let $\chi$ be a Boolean function formalizing a threshold, e.g., $\chi(x) = x \leq 30$. Then, the Boolean predicate $\phi(x, f(X), \chi) \Leftrightarrow \chi(x) \equiv \chi(f(X))$ describes, whether the possible result value $x \in \text{range}(f(X))$ leads to the same outcome of $\chi$ as the true result $f(X)$. For our example, $\phi(20, 30, \chi)$ holds true ($20 \leq 30$ and $30 \leq 30$), whereas $\phi(40, 30, \chi)$ is false ($40 \not\leq 30$, but $30 \leq 30$).

Using this predicate, we adapt the intervals $I = I_1, \ldots, I_n$ obtained during *interval creation*, so that interval boundaries coincide with changes in $\phi$. Let $B(\phi)$ be the boundary values of $\phi$, i.e., the values $x \in \text{range}(f(X))$ with $\lim_{y<x, y\to x} \phi(y, f(X), \chi) \neq \lim_{y>x, y\to x} \phi(y, f(X), \chi)$. For our example, we arrive at $B(\phi) = \{30\}$. Based thereon, we split each interval $I_i$ containing a boundary value $b \in B(\phi)$ into two new intervals $(\underline{I_i}, b), (b, \overline{I_i})$. Hence, each interval contains only values that share the outcome of the Boolean function $\chi$. In our example, the interval $(25, 35)$ is split into $(25, 30)$ and $(30, 35)$, as shown in Fig. 4.

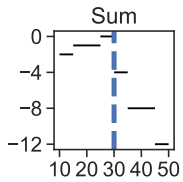



Figure 4: Adapted intervals and scores.

Finally, the scoring function used for *interval probability construction* is adapted. Let $d(i)$ be the minimal inter-interval-distance of interval $I_i$ to any other interval $I_j$ with $\phi(x, f(X), \chi) \neq \phi(y, f(X), \chi)$ for all $\underline{I_i} \leq x \leq \overline{I_i}$ and $\underline{I_j} \leq y \leq \overline{I_j}$. As before, let $k$ be the index of interval $\overline{I_f}$ containing the result value. Then, scores assigned to intervals that preserve the outcome of the Boolean function $\chi$ remain unchanged. For all other intervals $I_i$, the score is reduced by $\xi \cdot d(i)$, i.e., by the distance to the closest interval preserving the outcome multiplied by a falloff factor $\xi \in \mathbb{N}$. The adapted scoring function is defined as:

$$q(i) = \begin{cases} -|k - i| & \text{if } \phi(x, f(X), \chi) \\ & \text{for all } \underline{I_i} \leq x \leq \overline{I_i}, \\ -|k - i| - \xi \cdot d(i) & \text{otherwise.} \end{cases}$$

Fig. 4 illustrates the adapted scores for our running example, using $\xi = 3$. The scores of the right-most three intervals are reduced, as all of their values lead to a different outcome compared to the true result, $f(X) = 30$, when testing against $\chi(x) = x \leq 30$.

We obtain $d(4) = 1, d(5) = 2$, and $d(6) = 3$ for those intervals, given that the third interval $(25, 30)$ is the closest one retaining $\phi$ to any of those three. Thus, we arrive at $q(4) = -4, q(5) = -8$, and $q(6) = -12$. As the largest possible change in scores assigned to a possible result value in neighbouring input sets is never larger than $\xi$ and as the interval sizes are determined based on $\Delta f$, we conclude that $\Delta q = \xi$.

**Sample-and-aggregate Mechanism for Derived Measures.** Since the sensitivity of a derived multi-instance

measures is unknown in the general case, the above mechanisms are not applicable. However, many derived measures may be approximated using small samples, since their range is often independent of the domain of their input values. Functions that compute a normalized result are an example of this class of measures. For instance, the derived measure that denotes the root of the function composition tree of the third PPI in Fig. 1b yields a percentage, i.e., it is normalized to 0% to 100%. For such measures, the sample-and-aggregate-framework mentioned in Section 2.4 may be instantiated. That is, the actual result $f(X)$ is computed on $n$ partitions of $X$. The obtained results per sample are then aggregated using a differentially private mean function to achieve privatization of the derived measure.

## 4. Optimal Privatization of a PPI

So far, we introduced the notion of an admissible function set for the privatization of a PPI, along with several data release mechanisms that may be applied to each function in such a set. As such, we outlined a design space for each PPI in terms of the selection of an admissible function set and the respective release mechanism. In this section, we show how to guide this selection in order to achieve optimal privatization of a PPI that minimizes the loss in utility.

In Section 4.1, we first give an overview of the general framework to find an optimal instantiation of an admissible function set. We then describe the involved steps in detail: Section 4.2 outlines how to identify possible function set instantiations; Section 4.3 discusses the generation of output random variables for a function set instantiation; Section 4.4 discusses how to compute a utility value per instantiation; and Section 4.5 elaborates on an optimal selection among the instantiations.

### *4.1. Framework*

Given a PPI, an admissible set of functions needs to be selected for privatization and a release mechanism needs to be instantiated for each selected function. Ideally, this shall be done such that the utility of the PPI result is preserved as much as possible, under the given privacy parameter $\epsilon$. Yet, having a user take this selection would require access to the recorded trace data to be analysed, which is undesirable. At the same time, the estimation of the expected utility loss is challenging, given that the expected outcome of a specific mechanism may be influenced by many factors, such as number of inputs or their distribution, which may also change for each time scope for which a PPI is to be evaluated. We will later demonstrate the impact of these factors for the presented data release mechanisms as part of our experimental evaluation.

Against this background, we present the framework shown in Fig. 5, which, as part of the PPI interface, determines a admissible function set and the instantiations of release mechanism. To this end, the PPI interface considers the utility score of the output distribution for each possible instantiation of the mechanisms for each potential admissible function set. More specifically, given a PPI in the form of a function tree defined in PPINOT and the desired privacy parameter $\epsilon$, this involves four steps:

(1) The interface determines all admissible function sets and all instantiations of the respective functions with differentially private release mechanisms, jointly referred to as *admissible function set instantiations*. Here, parameters of the mechanisms (e.g., the falloff factor $\xi$ for the threshold-sensitive mechanism for aggregation measures) may be incorporated and lead to separate instantiations.

(2) The output random variable describing the probabilities of each possible result of an admissible function set instantiation is calculated.

(3) For each of the output variables, a utility score is calculated. It takes into account the pre-calculated true value of the PPI and quantifies the goodness of a value sampled from its distribution.

(4) Finally, the interface selects the admissible function set instantiation, for which the output random variable achieves the highest utility score. Then, the PPI is evaluated accordingly, which yields a privatized result.

It is worth to note that this general approach is independent of the specific set of considered release mechanisms. While we later apply with the mechanisms introduced in Section 3.2, it may be adopted also for other sets of differentially private release mechanisms.

### *4.2. Identifying Admissible Function Set Instantiations*

First, we identify all sets of functions of a PPI, given as a function composition tree, that are admissible, i.e., that satisfy the conditions detailed in Section 3.1. To illustrate this, we turn to the third PPI provided in Fig. 1b. Here, as mentioned, two admissible function sets exist, $\{ratio\}$ and $\{mean_1, mean_2\}$. Next, all instantiations of these functions with a set of pre-defined, differentially private release mechanisms for each type of measure are determined. For illustration, we consider

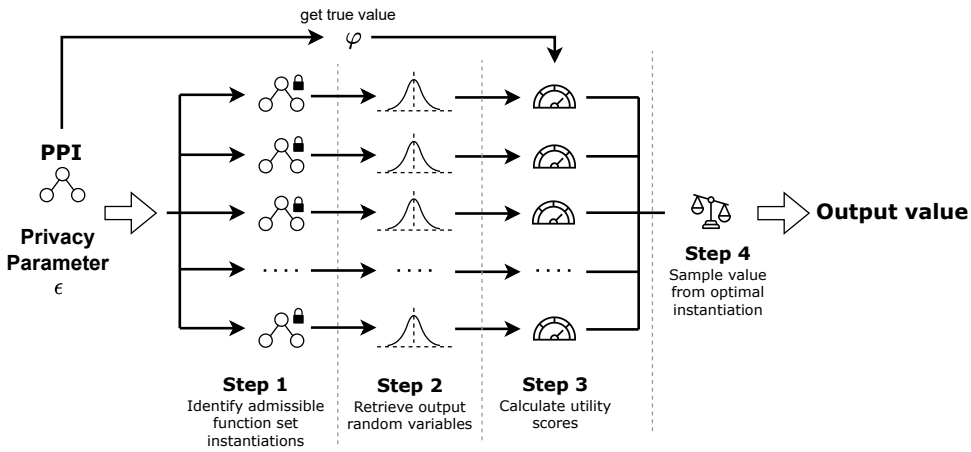



Figure 5: Overview of the framework for optimal privatization of a PPI.

the mechanism proposed in Section 3, i.e., the Laplace mechanism ($L$) and the interval mechanism ($I$) for aggregated measures, and the sample-and-aggregate mechanism ($S$) for multi-instance derived measures. To keep the example simple, we neglect the parameters of the mechanisms. Then, the set of admissible function set instantiations for the example includes the following elements: $\{ratio : S\}$, $\{mean_1 : L, mean_2 : L\}$, $\{mean_1 : L, mean_2 : I\}$, $\{mean_1 : I, mean_2 : L\}$, and $\{mean_1 : I, mean_2 : I\}$.

### *4.3. Generating Output Random Variables*

We now turn to the evaluation of PPIs under the assumption that inputs are not singular values, but random variables describing the probabilities of all potential inputs. While generally, these random variables can be either continuous, discrete or a mix-thereof, we assume all variables to be discrete. This assumption is motivated by the fact that the evaluation of continuous random variables relies on the integration of probability density functions, which are, in general, difficult to evaluate and often approximated numerically. We therefore rely on transformations of probability density functions into probability mass functions by means of equi-width histograms. Moreover, we recall from Section 2.3 that a function $g$ over a set of random variables $X_1, X_2, \ldots, X_n$ yields again a random variable, describing the output probabilities of all possible outputs of this function, given the provided input random variables.

We illustrate this procedure with the example provided in Fig. 6 depicting the function tree instantiation $\{mean_1 : L, mean_2 : L\}$ for the third PPI in Fig. 1b. Here, the time frame to consider consists of the three traces, $t_1$, $t_2$, $t_3$, for which the two base measures in

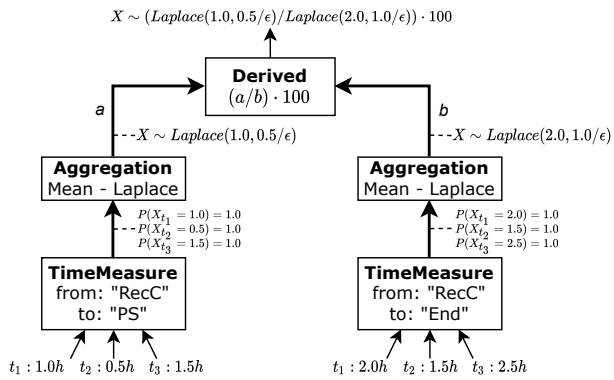



Figure 6: For each time frame consisting of a set of traces of the log, we calculate the random variable describing the probabilities of each possible output value.

the leaf nodes directly retrieve the values $1.0h$, $0.5h$, $1.5h$; and $2.0h$, $1.5h$, $2.5h$, respectively. The values are subsequently transformed into random variables with a probability of $1.0$ for their respective values. Next, these random variables are used as input for the aggregation measures, which rely on the respective function applied to a set of random variables instead of singular values. For the left aggregation measure, the resulting random variable is a Laplacian, with location parameter $\mu = 1.0$ (since $1.0 = (1.0 + 0.5 + 1.5)/3$) and scale parameter $b = 0.5/\epsilon$ (since $\Delta_{mean}(1.0, 0.5, 1.5) = (1.0 + 0.5 + 1.5)/3 = 0.5$). Similarly, the resulting random variable of the second aggregation measure is a Laplacian with parameters $\mu = 1.0$ and $b = 1.0/\epsilon$. The random variables obtained from the aggregation measures are used as inputs for the derived measure. The resulting random variable and its probability mass func-

tion describe the probability of each possible output of the chosen instantiation $\{mean_1 : L, mean_2 : L\}$.

To realize the procedure in the general case, we handle the different types of measures of a PPI, as follows:

**Base Measures.** Base measures do not assume a random variable as input, since they are computed directly over the given traces. However, conceptually, their result is not a single value, but a distribution for a random variable denoting the result. The only possible realization of the variable is the retrieved value. This way, we achieve compositionality since all functions building on base measures may now assume random variables as input.

**Single-Instance Derived Measures.** These measures can be regarded as functions defined over a single random variable. As such, they evaluate the respective function over all realizations of the input random variable. For instance, let the function $g(X)$ defined by the measure be a separation of the inputs in two classes, *0* and *1*, depending on whether the input is below a threshold of *10*. Now, assume the input to be random variable $X$ with realizations 2, 8, 12 and the respective probabilities to be $P(X{=}2) = 0.3$, $P(X{=}8) = 0.4$, $P(X{=}12) = 0.3$. Then, the output random variable $Y = g(X)$ consists of two possible realization 0 and 1, with probabilities $P(Y{=}0) = 0.7$ and $P(Y{=}1) = 0.3$.

**Aggregation Measures and Multi-instance Derived Measures.** These measures evaluate a function $g(X_1, X_2, \ldots, X_n)$ on a set of input random variables $X_1, X_2, \ldots, X_n$ with realizations $\text{range}(X_1) = \{x_{11}, x_{12}, \ldots, x_{1m}\}$, $\text{range}(X_2) = \{x_{21}, x_{22}, \ldots, x_{2m}\}, \ldots, \text{range}(X_n) = \{x_{n1}, x_{n2}, \ldots, x_{nm}\}$. For aggregation measures, these functions are $mean$, $sum$, $min$, and $max$, while for derived measures these functions are user-defined.

To evaluate the measure and obtain the output random variable $Y$ the measure thus needs to evaluate $g(X_1, X_2, \ldots, X_n)$ on all combinations of realizations of input variables, i.e., the set $X_{joined} = \text{range}(X_1) \times \text{range}(X_2) \times \ldots \times \text{range}(X_n)$, of which each element $\{(X_1{=}x_{1i}), (X_2{=}x_{2j}), \ldots, (X_n k{=}x_{nk})\}$ has probability $P(X_1{=}x_{1i}) \cdot P(X_2{=}x_{2j}) \cdot \ldots \cdot P(X_n{=}x_{nk})$. In cases where the measure shall not be privatized, i.e., evaluated without a differentially private release mechanism, the output random variable $Y$ is defined with $\text{range}(Y) = \bigcup g(X_{joined})$ and corresponding probabilities are $P(Y{=}y_i) = \Sigma_{(x_1,x_2,\ldots,x_n)\in X_{joined}:g(x_1,x_2,\ldots,x_n)=y_i} P(X_1 = x_1) \cdot P(X_2 = x_2) \cdot \ldots \cdot P(X_n = x_n)$. For instance, let the input variables for an aggregation measure evaluating the maximum of inputs be $\text{range}(X_1) = \{0, 2\}$ and $\text{range}(X_2) = \{1, 4\}$, and let the probabilities be given as $P(X_1{=}0) = 0.5$, $P(X_1{=}2) = 0.5$ and $P(X_2{=}1) = 0.8$, $P(X_2{=}4) = 0.2$. Then, $X_{joined} = (\{0, 1\}, \{0, 4\}, \{2, 1\}, \{2, 4\})$ and for the output random variable $Y$, $\text{range}(Y) = \{1, 2, 4\}$. The corresponding probabilities for each possible realization of $Y$ are given as $P(Y{=}1) = 0.5 \cdot 0.8 = 0.4$, $P(Y{=}2) = (0.5 \cdot 0.8) + (0.5 \cdot 0.2) = 0.5$, and $P(Y{=}4) = 0.5 \cdot 0.2 = 0.1$.

If privatization is applied, the outcome of evaluating the measure for a given combination of realizations is not a singular value, but a random variable itself. Let $Y_1, \ldots, Y_n$ be the output random variables of evaluating $g$ using a differentially private release mechanism on each combination of possible realizations in $X_{joined}$. Then, for the output random variable $Y$, $\text{range}(Y) = \bigcup_{1\leq i\leq n} \text{range}(Y_i)$ and $P(Y{=}y_i) = \frac{1}{n}\Sigma_{1\leq j\leq n} P(Y_j{=}y_i)$. As mentioned, if the output random variable is continuous, we discretize it using equi-width histograms, such that the number of possible realizations is equal to a pre-defined count.

### *4.4. Computing a Utility Score per Instantiation*

Once the random variable $Y$ associated with one admissible function set instantiation has been derived, we proceed with the computation of a utility score. It describes the expected information loss, when sampling a result from this particular instantiation. It is computed based on the difference of the true value of the PPI derived for the given traces and the value obtained by sampling the distribution of the random variable denoting the result.

Formally, let $\varphi$ be the true output value and $\Phi(Y, \varphi)$ be a utility function. As for the latter, we consider the average distance of a random variable to a particular value in terms of the *Rooted Mean Squared Error (RMSE)*:

$$RMSE(Y, \varphi) = \sqrt{\Sigma_{y\in\text{range}(Y)} P(Y{=}y) \cdot (\varphi - y)^2}.$$

Output random variables that have higher variance or that lie further away from $\varphi$ are assigned a higher value.

### *4.5. Selecting an Optimal Function Set Instantiation*

Once all output random variables $Y_1, Y_2, \ldots, Y_n$ for each of the instantiations of admissible function sets $f_1, f_2, \ldots, f_n$ have been retrieved, we select an instantiation that yields the largest utility regarding the true output value. That is, we select a variable as:

$$Y^* = \text{argmax}_{Y\in\{Y_1,Y_2,\ldots,Y_n\}} \Phi(Y, \varphi).$$

When using the *RMSE* as utility function, variable $Y^*$ that minimizes the *RMSE* corresponds to the function

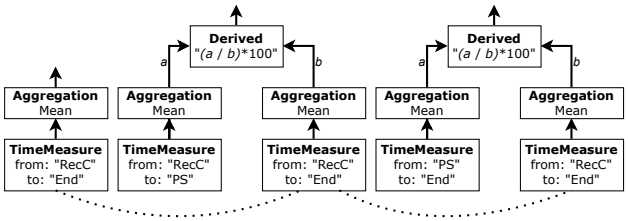



Figure 7: A set of PPIs that share a subtree.

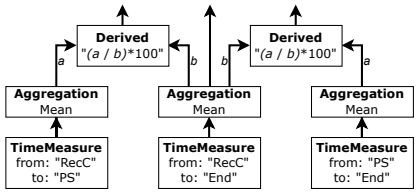



Figure 8: PPIs reusing results of the shared subtree.

set instantiation $f^*$ for which the distribution of possible output values yields the highest utility.

As a final step, variable $Y^*$ is used to sample a value that denotes the result of the privacy-preserving PPI.

## 5. Privatization of Multiple PPIs

The previous sections introduced a generic approach to privatize a single PPI. However, in many application scenarios, multiple PPIs need to be evaluated over the event data of a single process, such as illustrated with the three PPIs in Fig. 1b for the claim handling process in Fig. 1a. This is problematic, though: Multiple base measures of different PPIs may access the recorded event data, such that each of them reduces the privacy budget assigned to the data by a value proportional to the privacy parameter $\epsilon$. In the worst case, this may render it impossible to compute some of the PPIs, as for certain data, the privacy budget $\epsilon_{max}$ may have been exhausted. To avoid such situations, this section shows how shared subtrees of PPIs may be exploited to limit the reduction of the privacy budget when evaluating the PPIs.

Below, we first give the intuition of our approach to exploit shared subtrees of PPIs with an example (Section 5.1), before we present its operationalization based on constrained admissible function sets (Section 5.2).

### 5.1. Intuition

As mentioned above, base measures of different PPIs may access the same event data, thereby reducing the respective privacy budget considerably, up to the point that not all measures can be computed. Yet, some of the PPIs may share a set of subtrees, i.e., the PPIs are partly based on equivalent measures. In Fig. 7, this is exemplified by three PPIs defined for our running example in Fig. 1a. These PPIs all share a subtree that first retrieves the time from the claim receipt (`RecC`) until the process end (`End`) and then computes the mean average of the obtained values.

Naively evaluating the PPIs in isolation deducts three times from the privacy budget $\epsilon_{max}$, as the read values are privatized independently for each PPI, which increases the risk that the obfuscated value may be deduced. However, as all PPIs access the same value, it is sensible to use a single, privatized output of the subtree to evaluate all three PPIs, as illustrated in Fig. 8. This way, the privacy budget is reduced only once.

Conceptually, the three PPIs have been merged, and the value of the shared subtree is used to evaluate all three PPIs. While this limits the reduction of the privacy budget, it potentially comes with a loss in utility. If each PPI is privatized individually, an optimal admissible function set instantiation may be based on measures that are not shared with other PPIs. As such, to foster reuse of the results of some shared measures, it may be required to select instantiations that are not optimal for some PPIs.

### 5.2. Constrained Admissible Function Sets

In order to realize the above idea, we constrain the construction of the admissible function sets per PPI. In particular, the constraints enforce the privatization of root nodes of shared subtrees, thereby enabling the reuse of these results. Assuming a set of PPIs $P = \{(F_1, \rho_1), (F_2, \rho_2), \ldots, (F_n, \rho_n)\}$, this set of root nodes of shared subtrees $S \subset \bigcup_{1 \leq i \leq n} F_i$ is identified prior to step 1 of our proposed framework (Fig. 5).

The set $S$ can be obtained either through manual inspection during the definition of PPIs or automatically. It may be constructed automatically by enumerating, for each multi-instance measure in a function composition tree, content information of all children in a pre-defined order of elements. If any two trees contain equivalent enumerations, then they share at least one sub-tree. However, this enumeration considers all shared subtrees, and not only the maximal shared subtrees. Therefore, the enumeration needs to be filtered, such that only maximal shared subtrees remain. This can be achieved by traversing a function tree in a breadth-first manner. Should at any time during this traversal, an enumerated node be reached, it is added to the final set of root nodes of shared subtrees, whereas its child nodes are skipped.

The output of this retrieval step is the set $S$ of root nodes of shared subtrees. To manage the trade-off be-

tween the utility loss and the preservation of $\epsilon_{max}$, we introduce a constraint-parameter $0 < k < |S|$. This parameter constrains the selection framework to consider function set instantiations, such that exactly $k$ shared subtrees are privatized. Formally let $S_k = \{S' \in S \mid |S'| = k\}$ be all sets of root nodes of shared subtrees of size $k$. Then, for a PPI $(F_i, \rho_i)$ and a set $S'_k \in S_k$, we consider an admissible function set $F'_i \subseteq F_i$ only, if it also satisfies the following condition:

$$\forall\, f \in F_i : f \in S'_k \Rightarrow f \in F'_i.$$

Thus, for one set of shared subtrees $S'_k$, the framework only considers function set instantiations that explicitly privatize the selected subtrees and subsequently selects that instantiation obtaining the best utility for these sets. As above, let $\Phi$ be a utility function to evaluate the error introduced by a specific instantiation with respect to the true value. Then, with $Y_i(S')$ as the random variable of the root node of the PPI $(F_i, \rho_i)$ for which privatization was achieved based on the shared subtrees $S' \in S_k$, and $\varphi$ as the true output value, our approach is to select an instantiation as follows:

$$S^*_i = \text{argmax}_{S' \in S_k}\, \Phi(Y_i(S'), \varphi).$$

Consider, again, the PPIs in Fig. 7. Assume that the PPIs are given as $P = \{(F_1, \rho_1), (F_2, \rho_2), (F_3, \rho_3)\}$ with three shared subtrees $S = \{f_1, f_2, f_3\}$, as mentioned before. Now, let $k = 2$. Then, $S_2 = \{(f_1, f_2), (f_2, f_3), (f_3, f_1)\}$. For each of these sets, the framework selects the optimal function sets instantiations for the PPIs in $P$ in terms of the utility function. Assume that the latter may be given by the $RMSE$ with values $10, 8, 2$ for $(f_1, f_2)$, $6, 1, 3$ for $(f_2, f_3)$ and $2, 3, 2$ for $(f_3, f_1)$. Then, we would obtain $RMSE_{(f_1,f_2)} = 20$, $RMSE_{(f_2,f_3)} = 10$, $RMSE_{(f_3,f_1)} = 7$ as the aggregated utility loss, so that the optimal function set instantiations associated with $(f_3, f_1)$ are chosen for the privatized evaluation of the PPIs.

## 6. Evaluation

To assess the feasibility and utility of the proposed approach, we realized the PPI interface on top of an existing Java-based PPINOT implementation[1]. The implementation extends the PPI definition syntax of the PPINOT implementation to include privacy protection. While this has not been done for this work, this implementation can be extended to simplify the definition for end users of PPIs using L-pattern or graphical models, as has been proposed for the original PPINOT metamodel [12, 13]. We conducted controlled experiments using synthetic data (Section 6.1). Furthermore, we evaluated the selection of an optimal function set instantiation for PPIs defined on the *Sepsis Cases* log (Section 6.2). Finally, we report on a case study using the PPIs defined on *Sepsis Cases*, comparing the framework with the direct evaluation of PPIs on logs that have been anonymized a-priori using the PRIPEL framework [14] (Section 6.3). While the framework introduces an overhead regarding the evaluation of PPIs, we deemed this overhead to be negligible as the overall runtime for evaluating each PPI was in the range of seconds. As such, we deem the runtime aspects of our framework as reasonable for application in practice. Our implementation and evaluation scripts are publicly available[2].

### 6.1. Controlled Experiments

In a first series of experiments, we assessed the impact of different properties of the dataset $X$ used as input. Specifically, we considered the impact of the estimation of the domain of input values, its size and underlying value distribution, and the privacy parameter $\epsilon$. We sampled sets of 10, 50, 100, and 200 random values from a Gaussian distribution, a Pareto distribution, and a Poisson distribution. We chose these distributions, as they are often observed in event data recorded by business processes. To account for the randomness of our mechanisms, we performed 200 runs per experiment. Unless noted otherwise, the input domain is estimated using the minimal and maximal element of $X$, the dataset comprises 200 values drawn from a Gaussian distribution, and the privacy parameter is set as $\epsilon = 0.1$.

**Input Boundary Estimation.** First, we compare the boundary estimation using the minimal and maximal elements in $X$ with extensions of these boundaries by $15\%$ and $30\%$ at either boundary. The results for the interval-based mechanism, see Fig. 9, show that an extension of the domain increases the introduced magnitude of noise for all functions due to an increase in sensitivity. These observations are confirmed for the Laplace mechanism. Yet, for $min$ and $max$, there is a shift of the expected result towards the true result $f(X)$ (denoted by the blue line). The reason is that, without the extension, $f(X)$ coincides with boundary values of $X$. The extension increases the size of the interval containing $f(X)$, which increases the probability of this interval to be chosen.

[1] https://mvnrepository.com/artifact/es.us.isa.ppinot/ppinot-model

[2] https://github.com/MartinKabierski/privacy-aware-ppinot

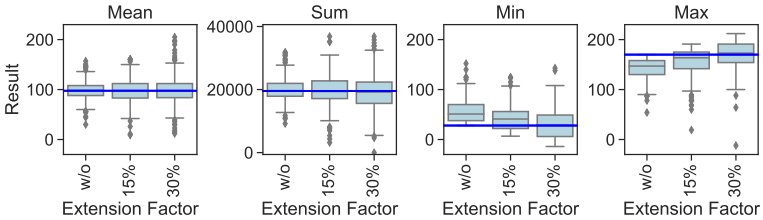



Figure 9: Impact of the input boundary estimation on the results.

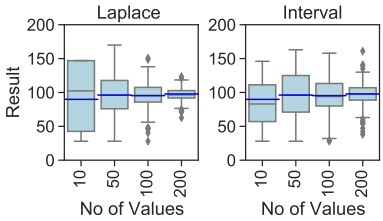
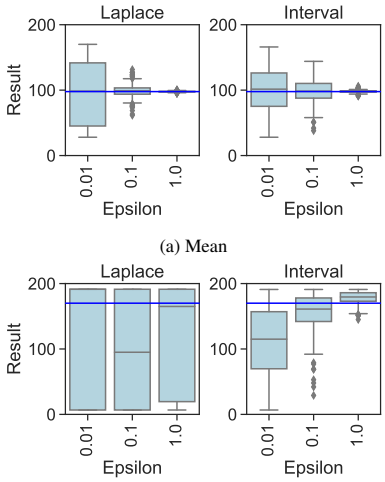



Figure 10: Impact of the dataset cardinality on the results for $mean$.




(a) Mean

(b) Max

Figure 11: Sensitivity of mean and maximum function towards $\epsilon$.

**Input Size and Distribution.** For the Laplace and interval-based mechanisms, we identify a dependency of $\Delta f$ on the input size for $mean$ functions. This dependency coincides with smaller noise magnitudes for larger input sizes, as illustrated in Fig. 10.

These trends were confirmed for the interval-based mechanism for $min$ and $max$. Here, the increased number of intervals and a more fine-grained differentiation between result values leads to higher utility, i.e., the expected result is close to the actual one. Yet, the trends are only visible for distributions with small inter-value distances, such as the Gaussian. For the Pareto- and Poisson-distributions, there was a significant reduction in utility for larger inputs using $max$. These distributions preserve most of their probability mass on the smaller values, Larger values are therefore more scarce, which inadvertently results in the creation of disproportionally large intervals and the same output probability for large portions of the output space.

**Epsilon.** The results obtained when changing the privacy parameter $\epsilon$ are shown in Fig. 11a for $mean$. Both the Laplace and interval-based mechanism show a similar increase in the introduced noise. The Laplace mechanism yields better results for larger $\epsilon$.

For $min$ and $max$, however, the interval-based mechanism clearly outperforms the Laplace mechanism for all values of $\epsilon$, see Fig. 11b for the maximum function. Here, the large sensitivity for the Laplace mechanism completely obfuscates the actual result $f(X)$, rendering the mechanism inappropriate for these functions.

**Threshold-sensitive Mechanism.** For the extension of the interval-based mechanism that aims to preserve the significance for thresholds, the general trends remain unaffected. However, the threshold-sensitive mechanism shifts large portions of the probability mass of the output space, as shown in Fig. 12. Here, the threshold to preserve is $\phi(x) : x < f(X) \pm y$, with $y$ being 100 for $sum$ and 10 for the other aggregation functions. For comparison, the results for the interval-based mechanism without threshold preservation are also given. There is a clear shift in output probabilities, depending on which values preserve the same properties as $f(X)$. Note that the results should not be interpreted in absolute terms, but serve as a binary indicator regarding the threshold.

**Derived Measures.** The sample-and-aggregate mechanism for derived measures mirrored the trends of the Laplace mechanism for $mean$. This is expected since the mechanism is based on the privatized mean. Yet, due to the use of $m$ buckets of size $n$, the magnitude of noise is larger. The mechanism requires $m$ times as many values in $X$ to achieve the same sensitivity as the mechanism for the $mean$. Since the mean is computed using $n$ values per bucket, the result estimation is accurate only for large datasets.

### 6.2. *Selection of optimal function set instantiations*

To investigate the differences in expected utilities of the different instantiations of admissible function sets,

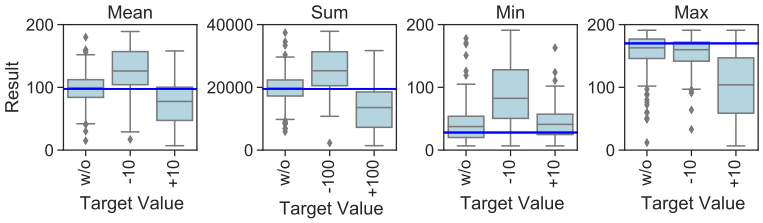



Figure 12: Results for the threshold-sensitive mechanism using differing result thresholds.

Table 1: PPIs defined for the Sepsis Cases log.

| ID | Measure | Target Values |
|---|---|---|
| 1: | Avg waiting time until admission | <24 hours |
| 2: | Avg length of stay | <30 days |
| 3: | Max length of stay | <35 days |
| 4: | Returning patient within 28 days | <5% |
| 5: | Antibiotics within one hour | >95% |
| 6: | Lactic acid test within three hours | >95% |

we evaluated six PPIs defined over the publicly available *Sepsis Cases* log using the proposed framework for the selection of optimal function set instantiations. *Sepsis Cases* is an event log consisting of 1050 process instances, recorded during a 16 month-period (from 11-2013 until 03-2015), of patients in emergency department suspected to Sepsis conditions. The PPIs used were created based on criteria and guidelines presented in [15, 16] and are listed in Table 1. Some concern the lengths of stays and treatments for patients (PPI 1-4), wile others target the adherence to treatment guidelines (PPI 5-6). PPI 1 to 3 could be defined using an aggregation measure using the specified aggregation function (*mean* for 1 and 2, *max* for 3). PPI 4 to 6 on the other hand were modeled using a multi instance derived measure, defined using the function $(a/b) \cdot 100$, where both $a$ and $b$ are *sum*-instanced aggregation measures, with $a$ summing up the counted values, and $b$ counting the number of trace instances in the current time scope. For the sake of interpretability, we set the time scopes in which to evaluate the PPIs to 4 months, consisting of 105, 309, 303, 263 and 25 process instances respectively. We set $\epsilon = 0.1$ and considered the Laplace mechanism ($L$) and Interval mechanism ($I$) for aggregation measures, and the sample-and-aggregate mechanism ($S$) using 10 partitions for derived multi-instance measures. For each time scope and considered function set instantiation, we recorded the *RMSE* and the probability mass function of the resulting random variable.

**Maximizing utility.** We report the *RMSE* values for PPI 1,2 and 3 in Table 2, and for PPI 4,5 and 6 in Table 3. For PPI 1 and 2, the *RMSE* of instantiations using the Laplace mechanism is consistently smaller for all time scopes. For PPI 1 the probability mass functions of the selected random variables for each time scope are illustrated in Fig. 13. Here, the number of instances per time scope seems to have the biggest influence on the *RMSE* of the final random variables, which is in line with the findings in Section 6.1. Contrary, for PPI 3, the Interval mechanism outperforms the Laplace mechanism, which also confirms the findings in Section 6.1. For illustration purposes, Fig. 14 shows the probability mass functions of both instantiations for the third time scope. Due to the pessimistic sensitivity estimation of the Laplace mechanism, the resulting random variable assign similar

| ID | Time Scope | RMSE | |
|---|---|---|---|
| | | *L* | *I* |
| 1 | 1 | 68.5 | 135.7 |
| | 2 | 31.5 | 66.4 |
| | 3 | 59.4 | 126.5 |
| | 4 | 57.9 | 122.7 |
| | 5 | 19.8 | 24.1 |
| 2 | 1 | 3.6 | 6.2 |
| | 2 | 2.4 | 4.3 |
| | 3 | 2.1 | 3.8 |
| | 4 | 3.5 | 6.8 |
| | 5 | 3.4 | 3.8 |
| 3 | 1 | 14.5 | 11.4 |
| | 2 | 20.2 | 12.9 |
| | 3 | 18.4 | 11.2 |
| | 4 | 29.5 | 22.8 |
| | 5 | 7.7 | 6.7 |

Table 2: Utility scores for PPIs 1 to 3

| ID | Time Scope | RMSE | | | | |
|---|---|---|---|---|---|---|
| | | *L,L* | *L,I* | *I,L* | *I,I* | *S* |
| 4 | 1 | 7.8 | 7.8 | 15.2 | 15.2 | 5.5 |
| | 2 | 4.3 | 4.3 | 7.6 | 7.6 | 2.8 |
| | 3 | 4.3 | 4.3 | 7.7 | 7.7 | 6.4 |
| | 4 | 4.6 | 4.6 | 8.7 | 8.7 | 3.3 |
| | 5 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 5 | 1 | 9.2 | 9.2 | 15.8 | 15.8 | 14.6 |
| | 2 | 4.9 | 4.6 | 8.7 | 8.7 | 6.2 |
| | 3 | 4.6 | 4.6 | 8.7 | 8.7 | 8.8 |
| | 4 | 5.3 | 5.3 | 9.9 | 9.9 | 10.5 |
| | 5 | 28.1 | 28.1 | 33.4 | 33.4 | 34.2 |
| 6 | 1 | 9.1 | 9.1 | 15.7 | 15.7 | 12.7 |
| | 2 | 4.6 | 4.6 | 8.9 | 8.9 | 7.9 |
| | 3 | 4.7 | 4.7 | 9.2 | 9.2 | 13.1 |
| | 4 | 5.4 | 5.4 | 10.2 | 10.2 | 7.5 |
| | 5 | 24.7 | 24.7 | 27.2 | 27.2 | 27.7 |

Table 3: Utility scores for PPIs 4 to 6

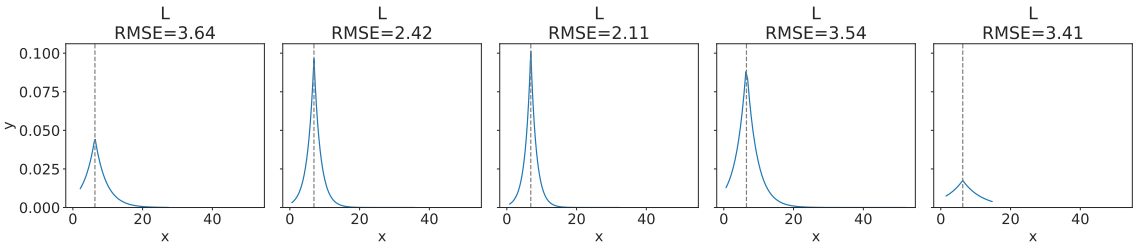



Figure 13: Probability mass functions of the random variables minimizing the *RMSE* for PPI 1.

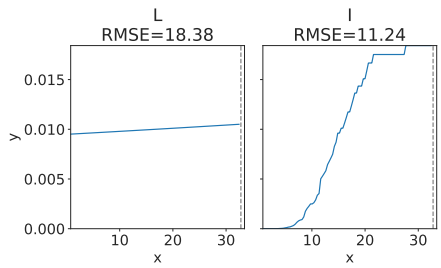



Figure 14: Probability mass functions for both admissible function set instantiations of the third time scope for PPI 3.

probabilities to all possible outputs with a slight skew towards larger values, while the Interval mechanism shifts large portions of its probability mass toward the true maximum value.

For PPIs 4,5 and 6, that have a larger set of possible function set instantiations, we see more variety, not only between PPIs, but also between time scopes. For instance, for PPI 4, in three out of the five time scope, the sample-and-aggregate mechanism is optimal, while for the third scope, the instantiations using the Laplace mechanism for counting the returning patients are optimal. In general, for all PPIs, no distinctions are made between different mechanism instantiations of the subtree counting the number of process instances. Here, since each process instance returns the same value of 1, the boundary estimation is minimal for all instantiations. In many cases we report large differences between the *RMSE* values of the possible instantiations, with the largest in the reported values being between the interval mechanism-based summation and the Laplace-based summation of the first time scope of PPI 4, whose probability mass functions are shown in Fig. 15. Here, we denote a difference in maximal difference in the *RMSE* of 9.7% (5.5 for *S*, 15.2 for *I,L* and *I,I*), signifying the gain of an optimal selection to preserve utility loss.

**Minimizing budget loss.** When evaluating the PPIs using optimal instantiations, while minimizing the loss in privacy budget, the initial tree analysis, identifies one shared subtree between PPIs 4,5, and 6. The shared subtree is the aggregation measure, counting the process instances, which is used throughout all three PPIs to relate the information of interest to the number of instances. Thus, the admissible function sets after considering this subtree, are constrained to those instantiations, that privatize on both aggregated measures. Therefore, for all PPIs, suboptimal instantiations need to be selected. For PPI 4, the optimal instantiation for time scope 1,2 and 4 change to *L,L* or *L,I*, respectively. The aggregated loss in utility associated with these instantiations is 5.1 . For both PPI 5 and 6, since the optimal instantiations are *L,L* and *L,I*, no loss in utility is induced in these cases.

The obtained results emphasize, that selecting the optimal function set instantiation manually proves to be troublesome. This holds especially for more compelx PPIs consisting of multiple privatizable measure. As all mechanisms are sensitive to a different set of factors (see Section 6.1) in differing magnitudes, and as the realizations of these factors may change over the different time scopes in the same log, it is not possible to reliably select an instantiation that performs optimally for all cases. Likewise, selecting a suboptimal instantiation may result in a large utility loss, should a given time scope favor function set instantiations other than the selected. Thus, rather than a manual selection, a selection should be taken on a per-input set basis, in the protected environment. Not only does this remove the need for the process analyst to make an educated, uninformed guess, but also guarantees, that the output result has the highest utility possible under the used utility function and induced constraints.

### *6.3. Case Study: Process for Sepsis Cases*

To explore how the presented mechanisms and the selection of optimized function isntantiations perform in a real-world application, we evaluated the previously defined PPIs on a monthly time scope, comparing our approach to a state-of-the-art privatization approach for

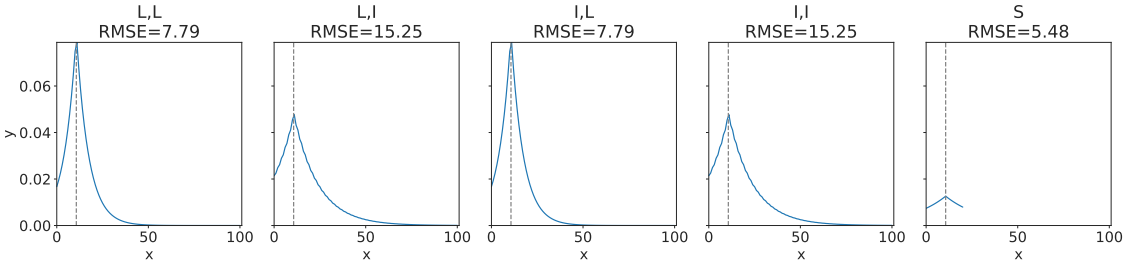



Figure 15: Probability mass functions for all admissible function set instantiations of the first time scope for PPI 4.

event logs. That is, we evaluated the same PPIs non-anonymously using logs that have been anonymized using PRIPEL [14]. Again, we evaluated each PPI 10 times using $\epsilon = 0.1$, reporting aggregate values. While results for all PPIs are available online[3],

due to space constraints, we here focus on PPI 1 and PPI 6, see Fig. 16 and Fig. 17, respectively.

For PPIs 1 to 3, we were able to reconstruct the general trends of the non-privatized analysis (exemplified for PPI 1 in Fig. 16). Yet, we also observed specific months with high result variances. For PPI 1 and 2 ($mean$ functions), the variance stems from the large domain of input values in these months, resulting in larger sensitivities. For PPI 3 ($max$ function), variances were relatively small, yet the framework consistently underestimates the maximum. This is due to large amounts of the probability mass being distributed below the maximum value.

The results obtained with our framework are in sharp contrast to those achieved when privatizing the event log with PRIPEL before computing the PPIs in a regular manner. As shown in Fig. 16 (right), the latter approach accumulates an error over the recorded time period. The steadily increasing deviation from the true value is caused by traces that represent outlier behaviour, which was artificially created by PRIPEL.

For PPIs 4 to 6, the results follow the general trends of the true values, see Fig. 17 for PPI 6. The result obtained using PRIPEL on the other hand, undererstimate the true values, while also not following trends in the data. In months, in which few traces are selected for a PPI, e.g., at the beginning and end of the covered time period, the variance is notably larger for our proposed framework, an effect that is avoided by the approach based on event log privatization.

Our results provide evidence that the proposed framework enables the computation of privacy-aware PPIs that mirror the general trends of their true values. Only for time periods, in which the PPI computation is based solely on a few traces, our framework does not yield sensible results. Thus, given a sufficiently large number of traces as the basis for the evaluation of PPIs, we can expect our framework to retain the trends, while selecting the best possible instantiations of release mechanisms from all of the available options.

[3] https://github.com/MartinKabierski/privacy-aware-ppinot

## 7. Related Work

To define PPIs, it was suggested to rely on ontology-based systems [17] or to resort to predicate logic to enable formal verification [18]. In this work, we followed the PPINOT meta-model, which is expressive due to its compositional approach. The compositionality is also the reason why we opted for the adoption of differential privacy in our approach. Other privacy models include k-anonymity [19] and its derivatives [20, 21], which statically mask recorded data points. Yet, since the evaluation of PPIs is driven by queries and processes continuously record data, these techniques are not suitable.

In the context of data-driven business process analysis, i.e., process mining, the re-identification risk related to event data was highlighted empirically in [22] as one of many privacy risk that were identified in the literature [23]. To mitigate this risk, different directions have been followed, namely the anonymization of event data or the direct anonymization of the process analysis artefact [24]. Moreover, the literature on privacy-preserving process mining can be divided into two sub-areas, based on the adopted privacy guarantees. Some techniques focus on providing differential privacy, whereas others employ group-based guarantees, such as the aforementioned $k$-anonymity.

Anonymization techniques that focus on differential privacy either focus on control-flow queries [25, 26, 27] or the anonymization of complete event logs [14]. Our work fills the gap for queries that are not purely control-flow related, but instead measure process performance.

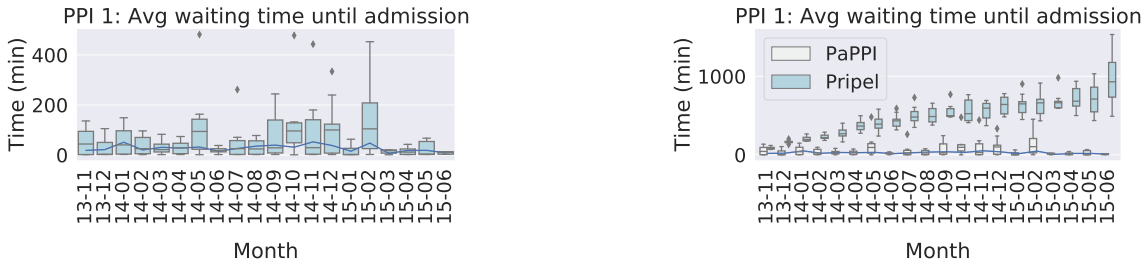



Figure 16: Evaluation Results for PPI1, privacy-aware PPI (left), and Privacy-aware PPI & PRIPEL (right).

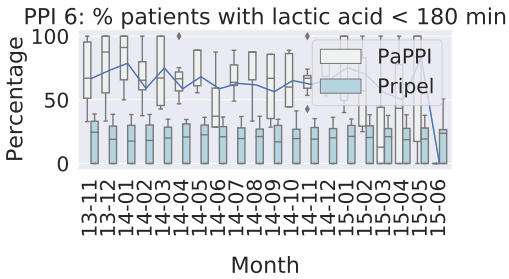



Figure 17: Evaluation Results of PPI6, Privacy-aware PPI & Pripel

In the realm of group-based privacy guarantees, the general aim is to strip the event data from information that could hurt the privacy either by changing the distribution of the respective data [28, 29] or suppressing it [30]. An important concern of all these techniques is the aim to preserve the utility of the data for the analysis purpose. To this end, semantics-aware distance functions [31] may be integrated within the aforementioned techniques.

Turning to the protection of privacy based on process mining artefacts, several research directions have been followed. In particular, the analysis of the control-flow of a process in a distributed setting was addressed, either under the assumption of a trusted third-party [32] or without that assumption, through the adoption of secure multi-party computation [33]. Also, a hardware-based approach that leverages trusted execution environments has been proposed for this setting [34]. Furthermore, privacy-preserving role mining was studied in [35].

The above techniques for privacy-preserving process mining have been integrated in several tools to make them easily applicable [36, 37, 38].

In general, we conclude that most work in the area of privacy-preserving process mining has focused on the control-flow perspective of processes. Our work complements these techniques with an approach to achieve privacy-preserving evaluation of process performance indicators.

## 8. Conclusion

In this work, we proposed an approach to the privacy-aware evaluation of process performance indicators based on event logs recorded during the execution of business processes. We presented a generic framework that includes an explicit interface to serve as the single point of access for the evaluation of PPIs. For this framework, we introduce a set of release mechanisms, that ensure $\epsilon$-differential privacy and a structured methodology of instantiating these release mechanisms for PPIs defined using the PPINOT meta-model. To cope with the problem of selecting an optimal instantiation of release mechanisms for a PPI, we presented a probabilistic formulation of the evaluation of a PPI, which enables us to minimize the loss according to some utility function. Moreover, we discussed the trade-off between minimizing the loss in utility for each single PPI and maximizing the reuse of privatized functions among multiple PPIs to use a privacy budget most effectively; and presented means to handle this trade-off. We evaluated our approach on both synthetic data and PPIs defined over a public event log. The results highlight the feasibility of our approach, as well as the advantages of the automated instantiation of the release mechanisms, compared to a manual instantiation. In future work, we intend to define additional utility functions that take into account whether output values comply with the true result on the fulfilment of target values of the PPI. Furthermore, we aim at incorporating additional release mechanisms, thereby broadening the design space for the privatization of PPIs.

## Acknowledgment

This work was partially funded by the German Research Foundation (DFG), project 421921612.

## References

[1] W. M. P. van der Aalst, Process Mining - Data Science in Action, Second Edition, Springer, 2016. doi:10.1007/

978-3-662-49851-4.
URL https://doi.org/10.1007/978-3-662-49851-4

[2] J. Carmona, B. F. van Dongen, A. Solti, M. Weidlich, Conformance Checking - Relating Processes and Models, Springer, 2018. doi:10.1007/978-3-319-99414-7.
URL https://doi.org/10.1007/978-3-319-99414-7

[3] A. del-Río-Ortega, M. Resinas, A. Ruiz-Cortés, Business process performance measurement, in: S. Sakr, A. Y. Zomaya (Eds.), Encyclopedia of Big Data Technologies, Springer, 2019. doi:10.1007/978-3-319-63962-8_99-1.
URL https://doi.org/10.1007/978-3-319-63962-8_99-1

[4] A. del-Río-Ortega, M. Resinas, C. Cabanillas, A. R. Cortés, On the definition and design-time analysis of process performance indicators, Inf. Syst. 38 (4) (2013) 470–490. doi:10.1016/j.is.2012.11.004.
URL https://doi.org/10.1016/j.is.2012.11.004

[5] M. Kabierski, S. A. Fahrenkrog-Petersen, M. Weidlich, Privacy-aware process performance indicators: Framework and release mechanisms, in: M. L. Rosa, S. W. Sadiq, E. Teniente (Eds.), Advanced Information Systems Engineering - 33rd International Conference, CAiSE 2021, Melbourne, VIC, Australia, June 28 - July 2, 2021, Proceedings, Vol. 12751 of Lecture Notes in Computer Science, Springer, 2021, pp. 19–36. doi:10.1007/978-3-030-79382-1_2.
URL https://doi.org/10.1007/978-3-030-79382-1_2

[6] Y. S.Aldeen, M. Salleh, M. A. Razzaque, A comprehensive review on privacy preserving data mining, SpringerPlus 4, 694 (2015). doi:10.1186/s40064-015-1481-x.

[7] R. Mendes, J. Vilela, Privacy-preserving data mining: Methods, metrics and applications, IEEE Access PP (2017) 1–1. doi:10.1109/ACCESS.2017.2706947.

[8] A. Arasu, B. Babcock, S. Babu, J. Cieslewicz, M. Datar, K. Ito, R. Motwani, U. Srivastava, J. Widom, STREAM: the stanford data stream management system, in: M. N. Garofalakis, J. Gehrke, R. Rastogi (Eds.), Data Stream Management - Processing High-Speed Data Streams, Data-Centric Systems and Applications, Springer, 2016, pp. 317–336. doi:10.1007/978-3-540-28608-0_16.
URL https://doi.org/10.1007/978-3-540-28608-0_16

[9] C. Dwork, Differential privacy, Automata, languages and programming (2006) 1–12.

[10] F. McSherry, K. Talwar, Mechanism design via differential privacy, in: 48th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2007), October 20-23, 2007, Providence, RI, USA, Proceedings, IEEE Computer Society, 2007, pp. 94–103. doi:10.1109/FOCS.2007.41.
URL https://doi.org/10.1109/FOCS.2007.41

[11] K. Nissim, S. Raskhodnikova, A. Smith, Smooth sensitivity and sampling in private data analysis, in: Proceedings of the Thirty-Ninth Annual ACM Symposium on Theory of Computing, STOC '07, Association for Computing Machinery, New York, NY, USA, 2007, p. 75–84. doi:10.1145/1250790.1250803.
URL https://doi.org/10.1145/1250790.1250803

[12] A. del Río-Ortega, M. Resinas, A. Durán, A. Ruiz-Cortés, Using templates and linguistic patterns to define process performance indicators, Enterprise Information Systems 10 (2) (2016) 159–192. doi:10.1080/17517575.2013.867543.

[13] A. del Río-Ortega, M. Resinas, A. Durán Toro, B. Bernárdez, A. Ruiz-Cortés, M. Toro, Visual ppinot: A graphical notation for process performance indicators, Business and Information Systems Engineering online (2017) 1–25. doi:10.1007/s12599-017-0483-3.

[14] S. A. Fahrenkrog-Petersen, H. van der Aa, M. Weidlich, PRIPEL: privacy-preserving event log publishing including contextual information, in: D. Fahland, C. Ghidini, J. Becker, M. Dumas (Eds.), Business Process Management - 18th International Conference, BPM 2020, Seville, Spain, September 13-18, 2020, Proceedings, Vol. 12168 of Lecture Notes in Computer Science, Springer, 2020, pp. 111–128. doi:10.1007/978-3-030-58666-9_7.
URL https://doi.org/10.1007/978-3-030-58666-9_7

[15] F. Mannhardt, D. Blinde, Analyzing the trajectories of patients with sepsis using process mining, in: J. Gulden, S. Nurcan, I. Reinhartz-Berger, W. Guédria, P. Bera, S. Guerreiro, M. Fellmann, M. Weidlich (Eds.), Joint Proceedings of the Radar tracks at the 18th International Working Conference on Business Process Modeling, Development and Support (BPMDS), and the 22nd International Working Conference on Evaluation and Modeling Methods for Systems Analysis and Development (EMMSAD), and the 8th International Workshop on Enterprise Modeling and Information Systems Architectures (EMISA) co-located with the 29th International Conference on Advanced Information Systems Engineering 2017 (CAiSE 2017), Essen, Germany, June 12-13, 2017, Vol. 1859 of CEUR Workshop Proceedings, CEUR-WS.org, 2017, pp. 72–80.
URL http://ceur-ws.org/Vol-1859/bpmds-08-paper.pdf

[16] A. Stefanini, D. Aloini, E. Benevento, R. Dulmin, V. Mininno, Performance analysis in emergency departments: a data-driven approach, Measuring Business Excellence 22(2) (2018) 130–145. doi:10.1108/MBE-07-2017-0040.

[17] B. Wetzstein, Z. Ma, F. Leymann, Towards measuring key performance indicators of semantic business processes, in: W. Abramowicz, D. Fensel (Eds.), Business Information Systems, 11th International Conference, BIS 2008, Innsbruck, Austria, May 5-7, 2008. Proceedings, Vol. 7 of Lecture Notes in Business Information Processing, Springer, 2008, pp. 227–238. doi:10.1007/978-3-540-79396-0_20.
URL https://doi.org/10.1007/978-3-540-79396-0_20

[18] V. Popova, A. Sharpanskykh, Modeling organizational performance indicators, Inf. Syst. 35 (4) (2010) 505–527. doi:10.1016/j.is.2009.12.001.
URL https://doi.org/10.1016/j.is.2009.12.001

[19] L. Sweeney, k-anonymity: A model for protecting privacy, IEEE Security and Privacy 10 (2002) 1–14.

[20] A. Machanavajjhala, D. Kifer, J. Gehrke, M. Venkitasubramaniam, *L*-diversity: Privacy beyond *k*-anonymity, ACM Trans. Knowl. Discov. Data 1 (1) (2007) 3. doi:10.1145/1217299.1217302.
URL https://doi.org/10.1145/1217299.1217302

[21] N. Li, T. Li, S. Venkatasubramanian, t-closeness: Privacy beyond k-anonymity and l-diversity, in: R. Chirkova, A. Dogac, M. T. Özsu, T. K. Sellis (Eds.), Proceedings of the 23rd International Conference on Data Engineering, ICDE 2007, The Marmara Hotel, Istanbul, Turkey, April 15-20, 2007, IEEE Computer Society, 2007, pp. 106–115. doi:10.1109/ICDE.2007.367856.
URL https://doi.org/10.1109/ICDE.2007.367856

[22] S. N. von Voigt, S. A. Fahrenkrog-Petersen, D. Janssen, A. Koschmider, F. Tschorsch, F. Mannhardt, O. Landsiedel, M. Weidlich, Quantifying the re-identification risk of event logs for process mining - empiricial evaluation paper, in: S. Dustdar, E. Yu, C. Salinesi, D. Rieu, V. Pant (Eds.), Advanced Information Systems Engineering - 32nd International Conference, CAiSE 2020, Grenoble, France, June 8-12, 2020, Proceedings, Vol. 12127 of Lecture Notes in Computer Science, Springer, 2020, pp. 252–267. doi:10.1007/978-3-030-49435-3_16.
URL https://doi.org/10.1007/978-3-030-49435-3_

16

[23] G. Elkoumy, S. A. Fahrenkrog-Petersen, M. F. Sani, A. Koschmider, F. Mannhardt, S. N. von Voigt, M. Rafiei, L. von Waldthausen, Privacy and confidentiality in process mining: Threats and research challenges, ACM Trans. Manag. Inf. Syst. 13 (1) (2022) 11:1–11:17. doi:10.1145/3468877.
URL https://doi.org/10.1145/3468877

[24] S. A. Fahrenkrog-Petersen, Providing privacy guarantees in process mining, in: M. L. Rosa, P. Plebani, M. Reichert (Eds.), Proceedings of the Doctoral Consortium Papers Presented at the 31st International Conference on Advanced Information Systems Engineering (CAiSE 2019), Rome, Italy, June 3-7, 2019, Vol. 2370 of CEUR Workshop Proceedings, CEUR-WS.org, 2019, pp. 23–30.
URL http://ceur-ws.org/Vol-2370/paper-03.pdf

[25] F. Mannhardt, A. Koschmider, N. Baracaldo, M. Weidlich, J. Michael, Privacy-preserving process mining - differential privacy for event logs, Bus. Inf. Syst. Eng. 61 (5) (2019) 595–614. doi:10.1007/s12599-019-00613-3.
URL https://doi.org/10.1007/s12599-019-00613-3

[26] S. A. Fahrenkrog-Petersen, M. Kabierski, F. Rösel, H. van der Aa, M. Weidlich, Sacofa: Semantics-aware control-flow anonymization for process mining, in: C. D. Ciccio, C. D. Francescomarino, P. Soffer (Eds.), 3rd International Conference on Process Mining, ICPM 2021, Eindhoven, Netherlands, October 31 - Nov. 4, 2021, IEEE, 2021, pp. 72–79. doi:10.1109/ICPM53251.2021.9576857.
URL https://doi.org/10.1109/ICPM53251.2021.9576857

[27] G. Elkoumy, A. Pankova, M. Dumas, Mine me but don't single me out: Differentially private event logs for process mining, in: C. D. Ciccio, C. D. Francescomarino, P. Soffer (Eds.), 3rd International Conference on Process Mining, ICPM 2021, Eindhoven, Netherlands, October 31 - Nov. 4, 2021, IEEE, 2021, pp. 80–87. doi:10.1109/ICPM53251.2021.9576852.
URL https://doi.org/10.1109/ICPM53251.2021.9576852

[28] S. A. Fahrenkrog-Petersen, H. van der Aa, M. Weidlich, PRETSA: event log sanitization for privacy-aware process discovery, in: International Conference on Process Mining, ICPM 2019, Aachen, Germany, June 24-26, 2019, IEEE, 2019, pp. 1–8. doi:10.1109/ICPM.2019.00012.
URL https://doi.org/10.1109/ICPM.2019.00012

[29] E. Batista, A. Solanas, A uniformization-based approach to preserve individuals' privacy during process mining analyses, Peer-to-Peer Networking and Applications 14 (3) (2021) 1500–1519.

[30] M. Rafiei, W. M. P. van der Aalst, Group-based privacy preservation techniques for process mining, Data Knowl. Eng. 134 (2021) 101908. doi:10.1016/j.datak.2021.101908.
URL https://doi.org/10.1016/j.datak.2021.101908

[31] F. Rösel, S. A. Fahrenkog-Petersen, H. v. d. Aa, M. Weidlich, A distance measure for privacy-preserving process mining based on feature learning, in: International Conference on Business Process Management, Springer, 2021, pp. 73–85.

[32] C. Liu, H. Duan, Q. Zeng, M. Zhou, F. Lu, J. Cheng, Towards comprehensive support for privacy preservation cross-organization business process mining, IEEE Transactions on Services Computing 12 (4) (2016) 639–653.

[33] G. Elkoumy, S. A. Fahrenkrog-Petersen, M. Dumas, P. Laud, A. Pankova, M. Weidlich, Secure multi-party computation for inter-organizational process mining, in: S. Nurcan, I. Reinhartz-Berger, P. Soffer, J. Zdravkovic (Eds.), Enterprise, Business-Process and Information Systems Modeling - 21st International Conference, BPMDS 2020, 25th International Conference, EMMSAD 2020, Held at CAiSE 2020, Grenoble, France, June 8-9, 2020, Proceedings, Vol. 387 of Lecture Notes in Business Information Processing, Springer, 2020, pp. 166–181. doi:10.1007/978-3-030-49418-6_11.
URL https://doi.org/10.1007/978-3-030-49418-6_11

[34] M. Müller, A. Simonet-Boulogne, S. Sengupta, O. Beige, Process mining in trusted execution environments: Towards hardware guarantees for trust-aware inter-organizational process analysis.

[35] M. Rafiei, W. M. P. van der Aalst, Mining roles from event logs while preserving privacy, in: C. D. Francescomarino, R. M. Dijkman, U. Zdun (Eds.), Business Process Management Workshops - BPM 2019 International Workshops, Vienna, Austria, September 1-6, 2019, Revised Selected Papers, Vol. 362 of Lecture Notes in Business Information Processing, Springer, 2019, pp. 676–689. doi:10.1007/978-3-030-37453-2_54.
URL https://doi.org/10.1007/978-3-030-37453-2_54

[36] M. Bauer, S. A. Fahrenkrog-Petersen, A. Koschmider, F. Mannhardt, H. van der Aa, M. Weidlich, Elpaas: Event log privacy as a service, in: B. Depaire, J. D. Smedt, M. Dumas, D. Fahland, A. Kumar, H. Leopold, M. Reichert, S. Rinderle-Ma, S. Schulte, S. Seidel, W. M. P. van der Aalst (Eds.), Proceedings of the Dissertation Award, Doctoral Consortium, and Demonstration Track at BPM 2019 co-located with 17th International Conference on Business Process Management, BPM 2019, Vienna, Austria, September 1-6, 2019, Vol. 2420 of CEUR Workshop Proceedings, CEUR-WS.org, 2019, pp. 159–163.
URL http://ceur-ws.org/Vol-2420/paperDT9.pdf

[37] G. Elkoumy, S. A. Fahrenkrog-Petersen, M. Dumas, P. Laud, A. Pankova, M. Weidlich, Shareprom: A tool for privacy-preserving inter-organizational process mining, in: W. M. P. van der Aalst, J. vom Brocke, M. Comuzzi, C. D. Ciccio, F. García, A. Kumar, J. Mendling, B. T. Pentland, L. Pufahl, M. Reichert, M. Weske (Eds.), Proceedings of the Best Dissertation Award, Doctoral Consortium, and Demonstration & Resources Track at BPM 2020 co-located with the 18th International Conference on Business Process Management (BPM 2020), Sevilla, Spain, September 13-18, 2020, Vol. 2673 of CEUR Workshop Proceedings, CEUR-WS.org, 2020, pp. 72–76.
URL http://ceur-ws.org/Vol-2673/paperDR02.pdf

[38] M. Rafiei, A. Schnitzler, W. M. P. van der Aalst, PC4PM: A tool for privacy/confidentiality preservation in process mining, in: W. M. P. van der Aalst, R. M. Dijkman, A. Kumar, F. Leotta, F. M. Maggi, J. Mendling, B. T. Pentland, A. Senderovich, M. Sepúlveda, E. S. Asensio, M. Weske (Eds.), Proceedings of the Best Dissertation Award, Doctoral Consortium, and Demonstration & Resources Track at BPM 2021 co-located with 19th International Conference on Business Process Management (BPM 2021), Rome, Italy, September 6th - to - 10th, 2021, Vol. 2973 of CEUR Workshop Proceedings, CEUR-WS.org, 2021, pp. 106–110.
URL http://ceur-ws.org/Vol-2973/paper_268.pdf