SaCoFa: Semantics-aware Control-flow Anonymization for Process Mining

Stephan A. Fahrenkog-Petersen* Martin Kabierski* Fabian Rösel* Han van der Aa† Matthias Weidlich*

*Humboldt-Universität zu Berlin
Berlin, Germany
{fahrenks,bauermart,roeselfa,weidlima}@hu-berlin.de

†University of Mannheim
Mannheim, Germany
han@informatik.uni-mannheim.de

***Abstract*—Privacy-preserving process mining enables the analysis of business processes using event logs, while giving guarantees on the protection of sensitive information on process stakeholders. To this end, existing approaches add noise to the results of queries that extract properties of an event log, such as the frequency distribution of trace variants, for analysis. Noise insertion neglects the semantics of the process, though, and may generate traces not present in the original log. This is problematic. It lowers the utility of the published data and makes noise easily identifiable, as some traces will violate well-known semantic constraints. In this paper, we therefore argue for privacy preservation that incorporates a process' semantics. For common trace-variant queries, we show how, based on the exponential mechanism, semantic constraints are incorporated to ensure differential privacy of the query result. Experiments demonstrate that our semantics-aware anonymization yields event logs of significantly higher utility than existing approaches.**

***Index Terms*—Privacy-preserving Process Mining, Differential Privacy, Anonymization**

I. Introduction

Process mining analyses business processes based on event logs recorded during their execution [1]. Event logs comprise sequences of events that reveal how a process is executed, by whom, and for whom. Since a log may include sensitive information about people involved in a process, its content is subject to privacy regulations, such as the GDPR [2]. Attempts to protect sensitive information through deletion or pseudonymisation of identifying information (e.g., names) are ineffective, given that the obscured information can often be re-identified by relating execution sequences to knowledge about the context of process execution [3]. To ensure that privacy regulations are still met, *privacy-preserving process mining* [4] strives to protect sensitive information in event logs and process mining results by ensuring that they provide well-known privacy guarantees, such as differential privacy [5].

Many process mining techniques analyse a process from an abstract control-flow perspective, in terms of its *trace-variant distribution*. They require information on the recorded sequences of activity executions, known as trace variants, and their occurrence frequencies. Recognising the importance of such distributions, trace-variant queries may be anonymised [6]. By inserting noise into the variant distribution of a log, differential privacy, a guarantee that bounds the impact of the data of one individual on the query result, is ensured.

The state-of-the-art approach to achieve differential privacy for trace-variant queries [6] has an important drawback, though. Employing a Laplacian mechanism, it inserts noise randomly, which neglects the semantics of the underlying process. The returned trace variants may then represent behaviour that was never observed or, more importantly, which is clearly impossible for the process at hand. For a treatment process in a hospital, for instance, the anonymized distribution may include trace variants in which a patient is *discharged* from the hospital before *arriving* there. Including such obviously incorrect sequences lowers the utility of the published data for process analysis, e.g., resulting in misleading models. At the same time, adversaries can easily recognize such trace variants as the result of the anonymization procedure, so that the assumed privacy guarantee no longer holds. Against this background, we target the question of *how to incorporate a process' semantics in control-flow anonymization*.

In this paper, we address the above research question with SaCoFa, an approach for semantics-aware control-flow anonymization. Our idea is to achieve differential privacy of trace-variant queries based on *exponential noise-insertion* techniques. Unlike noise insertion with the Laplacian mechanism, the exponential mechanism enables us to control the way noise is inserted, while providing the same degree of privacy [7].

Specifically, we present SaCoFa as a general algorithm to achieve controlled noise insertion using the exponential mechanism. That includes the definition of a score function to assess the loss induced by the insertion of noise, which enables us to incorporate a process' semantics. Given the exponential runtime complexity of SaCoFa, we further present semantics-aware optimizations for approximate privacy guarantees.

Compared to the state of the art, trace-variant distributions obtained with SaCoFa have a higher utility for process analysis and provide more robust privacy guarantees. We demonstrate these advantages in experiments with public datasets: Models discovered from the resulting distributions show higher F-scores and better generalization. Also, SaCoFa introduces less obvious noise, as classified by anomaly detection techniques.

Below, Section II first motivates the need for our work, before Section III introduces background information. Section IV presents SaCoFa, which we evaluate in Section V. We review related work in Section VI and conclude in Section VII.

TABLE I: Illustration of a trace-variant distribution, both original and privatized.

(a) Original trace-variant distribution.

Trace Variant	#
$\langle Register, Triage, Surg., Release\rangle$	20
$\langle Register, Triage, Surg., Antibio., Release\rangle$	12
$\langle Register, Triage, Antibio., Antibio..Release\rangle$	6
$\langle Register, Triage, Antibio., Surg., Release\rangle$	5
$\langle Register, Triage, Consul., Release\rangle$	2
$\langle Register, Triage, Consul., Surg., Release\rangle$	4

(b) Privatized trace-variant distribution.

Trace Variant	#
$\langle Register, Triage, Surg., Release\rangle$	18
$\langle Register, Triage, Antibio., Antibio., Release\rangle$	7
$\langle Release, Triage, Triage, Surg., Register\rangle$	4

II. Motivation

The state-of-the-art technique for the privatization of trace-variant distributions [6] constructs a prefix tree. It considers prefixes of trace variants of increasing lengths and obfuscates their occurrence counts using the Laplacian mechanism [7]. Due to the exponential growth of the set of possible prefixes for a set of activities, infrequent prefixes are pruned to achieve an acceptable runtime of the algorithm. While the resulting trace-variant distribution is differentially private, new behaviour may have been *introduced* to the distribution and behaviour from the original log may have been *removed*. We illustrate the resulting issues using the trace-variant distributions in Table I.

Behaviour insertion problems. The Laplacian mechanism introduces noise into a trace-variant distribution in a fully random manner. Any new trace variant is considered to be equally suitable or problematic, respectively. Depending on the underlying process, however, some trace variants may easily be identified as manipulated ones. For instance, the third trace variant in Table Ib contains a repetition of the $Triage$ activity. Similarly, although all traces in the original log start with the prefix $\langle Register, Triage\rangle$ and end with a $Release$ activity, the aforementioned variant in Table Ib violates these patterns. Even without detailed knowledge about the process, an adversary immediately identifies this variant as artificial behaviour and omits it during an attack, which effectively reduces the privacy guarantee associated with the published query result.

Behaviour removal problems. The pruning strategies employed when anonymizing a trace-variant distribution also lead to the removal of behaviour. In our example, the third, fourth, and fifth variants of Table Ia do not appear in Table Ib, i.e., they are assigned a count of zero. Since pruning is applied in the construction of the prefix tree, it may have far reaching consequences: Assigning the prefix $\langle Register, Triage, Consul.\rangle$ an occurrence frequency below the pruning threshold implies that *none* of the variants with this prefix will appear in the resulting distribution. In the worst case, this effect may materialize for the prefix $\langle Register, Triage\rangle$ in our example, which, arguably, would render the result useless for most process analyses.

Proposed approach. To alleviate the above issues, we argue that noise insertion shall be based on the exponential mechanism [8]. It enables us to assign scores to potential outputs, i.e., specific trace variants in our setting. This way, a prioritization of the trace variants that shall appear in the resulting distribution is achieved. In the remainder, we show how this idea enables us to incorporate the semantics of a process in the anonymization.

III. Background

Event model. Our work focuses on the control-flow perspective of business processes. Therefore, we use an event model that builds upon a set of activities $\mathcal{A}$. Each event in a log is assumed to correspond to one of these activities. Using $\mathcal{E}$ to denote the universe of all events, a single execution of a process, i.e., a *trace*, is modelled as a sequence of events $\xi \in \mathcal{E}^*$, such that no event can occur in more than one trace. An event log is a set of traces, $L \subseteq 2^{\mathcal{E}^*}$, with $\mathcal{L}$ as the universe of event logs. Distinct traces that indicate the same sequence of activity executions are said to be of the same *trace variant*, i.e., $\mathcal{A}^*$ is the universe of trace variants. The set of activities referenced by events in an event log L is denoted by $\mathcal{A}(L)$.

Trace-variant queries. A *trace-variant query* is a function $\tau(L) : \mathcal{L} \rightarrow \mathcal{A}^* \times \mathbb{N}$ that returns the trace-variant distribution of an event log L, i.e., it captures how often certain trace variants occur in L. Aside from $\tau(L)$ as the query over all trace variants, we define $\tau(L, v)$ as a query that returns the number of traces in L, for which the events correspond to the sequence of activities of trace variant v.

Differential privacy. The privacy of a query can be guaranteed by fulfilling a privacy guarantee [9]. A common guarantee is *differential privacy* [10], which has been adopted by companies such as Apple, SAP, and Google. The general idea behind differential privacy is to ensure that the inclusion of the data of one individual in a certain dataset will not significantly change the result returned by a query over this data. In the context of our work, this means that a trace-variant query τ is said to preserve differential privacy, if the trace-variant distribution returned by query $\tau(L)$ does not significantly differ from the distribution returned by a query over a *neighbouring* log, i.e., a log that contains one additional trace, $\tau(L \cup \{t\})$, for any trace $t \in \mathcal{E}^*$.

A trace-variant query τ that returns the actual frequency distribution, in general, cannot be expected to satisfy differential privacy. Hence, one relies on probabilistic queries $\hat{\tau}$ that approximate the true distribution, while satisfying the privacy guarantee. This leads to the following definition:

Definition 1 (Differential Privacy): Given a probabilistic trace-variant query $\hat{\tau}$ and privacy parameter $\epsilon \in \mathbb{R}$, query $\hat{\tau}$ provides *ϵ-differential privacy*, if for all neighbouring pairs of event logs $L_1, L_2 \in \mathcal{L}$ and for all sets of possible trace-variant distributions, $D \subseteq \mathcal{A}^* \times \mathbb{N}$, it holds that:

$$Pr[\hat{\tau}(L_1) \in D] \leq e^{\epsilon} \times Pr[\hat{\tau}(L_2) \in D]$$

where the probability is taken over the randomness introduced by the query $\hat{\tau}$.

The lower the value of ϵ, the stronger the privacy guarantee that is provided. In scenarios where an individual can be part of multiple traces, the privacy parameter ϵ shall be divided by the maximal number of traces related to an individual, in order to achieve the same degree of privacy.

To ensure differential privacy for a query, it is common to define a probabilistic query that inserts noise into the result of the original one. This noise-insertion mechanism is generally guided by a probability distribution.

Laplacian mechanism. The Laplacian mechanism inserts noise based on a Laplacian distribution. This mechanism was used in [6] to anonymize a trace-variant distribution. The impact of this mechanism generally depends on the strength of the privacy guarantee ϵ and the *sensitivity* Δf of some query q. A query $\hat{q}$ protected by the Laplace mechanism can formally be described as:

$$\hat{q} \leftarrow q + Lap(\frac{\Delta f}{\epsilon})$$

The sensitivity Δf depends on the maximum impact one individual can have on the result of query q. So, if q is a trace-variant query (τ, as introduced above) and one individual participates in at most one trace, the sensitivity is $\Delta f = 1$. If an individual can appear in multiple traces, the sensitivity is higher and more noise needs to be introduced to achieve ϵ-differential privacy. However, in such scenarios, the guarantee of ϵ-differential privacy may also be relaxed, which lowers the increase in sensitivity and still provides a relatively strong protection [11].

When used to insert noise into a trace-variant distribution, the Laplacian mechanism has considerable drawbacks, see Section II. That is, the probability that a certain anonymized trace-variant distribution is returned, only depends on the syntactic distance of this distribution to the actual one. This ignores that certain distributions are less desirable than others, even when they are syntactically just as different.

Exponential mechanism. The exponential mechanism enables a prioritization of certain query results by incorporating the notion of a score function into the noise insertion process. The score function s defines some results to be more desirable than others for the given dataset over which the query is evaluated, i.e., the higher $s(d, r)$, the more desirable is the query result r for the dataset d. Moreover, Δs is the sensitivity of the score function, i.e., the maximum differences between scores assigned to the possible results for two neighbouring datasets. Then, for some query q over dataset d and privacy parameter ϵ, the query $\hat{q}$ protected by the exponential mechanism is derived by choosing the result r with a probability proportional to $e^{(\epsilon s(d,r))/(2\Delta s)}$.

The mechanism may be lifted to our case of a trace variant query. For a given log L and a possible trace-variant distribution $D \in \mathcal{A}^* \times \mathbb{N}$, the score $s(L, D)$ shall capture whether D is desirable in terms of a process' semantics, as captured by L. Then, the mechanism returns a specific trace-variant distribution D with a probability proportional to $e^{(\epsilon s(L,D))/(2\Delta s)}$. This general idea will be exploited in our SaCoFa approach, as presented in the next section.

IV. Semantics-aware Control-flow Anonymization

This section introduces SaCoFa (semantics-aware control-flow anonymization) as an approach to retrieve the anonymized behaviour of an event log. Section IV-A presents the general algorithm based on the exponential mechanism. Section IV-B then defines the score function, needed to incorporate a process' semantics. Finally, Section IV-C discusses pruning strategies for SaCoFa, their computational necessity, and how the score function helps to decrease the negative effects of pruning.

A. The SaCoFa Algorithm

The idea of the SaCoFa algorithm is to construct a prefix tree of trace variants through step-wise expansion, where each step adds an activity or a dedicated end symbol to a branch in the tree. During this construction, prefixes are evaluated based on a *score function*, which reflects their compliance with the process' semantics, as captured in the original event log. Specifically, prefixes are categorized as *harmful* or *harmless*, depending on whether they violate semantic constraints and hence, threaten the utility of a trace-variant distribution.

While harmless prefixes are always added to the tree, some harmful prefixes typically also need to be incorporated, to achieve differential privacy. To this end, we leverage the exponential mechanism, which incorporates the score function to assign lower probabilities to prefixes that induce a stronger violation of a process' semantics. Hence, we are able to nudge the expansion of the tree to prefixes that are less harmful. In any case, all prefixes added to the tree are assigned noisy counts. To cope with the exponential growth of the prefix tree, we also prune the tree based on these noisy counts in each step of its expansion.

In Algorithm 1, we provide the pseudo-code for our algorithm. It takes as input an event log L and several parameters: the strength of the desired privacy guarantee ϵ, an upper bound on the trace-variant length k, and a pruning parameter p (or two pruning parameters $p_{harmless}$ and $p_{harmful}$, as detailed later). It returns $\tau'(L)$, i.e., an anonymized trace-variant distribution.

First, the algorithm initializes the prefix tree, represented as a set of prefixes T (line 1). Next, the trace-variant distribution d and the current prefix length n are initialized (lines 2-3). Then, the prefix tree is iteratively expanded, which will terminate when n reaches the maximal prefix length k (line 4).

Candidate generation. For each $n \leq k$, we expand the current tree by first generating a set of candidate prefixes. To obtain these candidates, we select each prefix $v \in T$ that is maximal, $|v| = n - 1$, and has not yet been ended, $v(|v|) \neq \perp$ (line 6). Note, the first iteration takes the empty prefix. Then, for each $a \in \mathcal{A}(L) \cup \{\perp\}$, i.e., for any activity or the end symbol $\perp$, we generate a new candidate by appending a to v and add it to the candidate set C (lines 7-8).

Tree expansion. The candidate prefixes in C are evaluated with a *score function* to classify them as harmless (C_{expand}) or harmful (C_{harm}) (lines 9-10). The definition of the score function depends on the incorporated notion of a process' semantics and will be discussed in Section IV-B. Here, we

Algorithm 1: The SaCoFa Algorithm

```
input : $L$, an event log; $\epsilon$, the privacy parameter; $k$, the max. prefix
        length; $p$ ($p_{harmless}, p_{harmful}$), the pruning parameter(s).
output: the result of $\tau'(L)$, an anonymized trace-variant distribution.
1  $T \leftarrow \{\langle\rangle\}$;                 /* Initalize the prefix tree */
2  $d \leftarrow \emptyset$;   /* Initalize the trace-variant distribution */
3  $n \leftarrow 1$;       /* Initialize the current prefix length */
4  while $n \leq k$ do    /* Consider prefixes up to length $k$ */
5      $C \leftarrow \emptyset$;               /* Initialize candidate set */
       /* Select candidate prefixes to expand                 */
6      foreach $v \in T \wedge |v| = n-1 \ \wedge v(|v|) \neq \perp$ do
           /* For each possible activity                      */
7          foreach $a \in \mathcal{A}(L) \cup \{\perp\}$ do
               /* Add expanded prefix to candidate set        */
8              $C \leftarrow C \cup \{v.\langle a\rangle\}$;
       /* Determine harmless prefix candidates                */
9      $C_{expand} \leftarrow \{c \in C \mid \mathbf{score}(L, c) = 1\}$ ;
       /* Determine harmfull prefix candidates                */
10     $C_{harm} \leftarrow C \setminus C_{expand}$ ;
       /* Select prefixes; harmful prefix candidates are
          selected using the exponential mechanism            */
11     $C_{expand} \leftarrow C_{expand} \cup Exp(C_{harm}, \mathbf{score}(L, .), \epsilon)$;
       /* Assign positive noisy count to prefixes             */
12     foreach $v \in C_{expand}$ do
13         $d(v) \leftarrow [\tau(L, v) + Lap(\frac{1}{\epsilon})]_{\geq 0}$ ;
14     $T \leftarrow \mathbf{prune}(T, d, p, C_{harm})$;   /* Prune prefix tree */
15     $n \leftarrow n + 1$;     /* Increase current prefix length */
   /* Return the distribution over all prefixes that are
      complete or of length $k$                               */
16 return $\{d(v) \mid v \in T \wedge (v(|v|) = \perp \vee |v| = k)\}$;
```

assume the score function to be applicable to prefixes, while the actual scoring (function s in Section III) refers to a distribution over prefixes with their frequencies all set to one.

Employing the exponential mechanism, we determine which of the harmful prefixes to add to the tree by random selection (line 11). Then, the selected harmful prefixes, together with the harmless ones, expand the prefix tree (line 12). Each of these prefixes is assigned a noisy count, based on its number of occurrences in the original log and random, Laplacian noise (line 13). The latter is configured by the privacy guarantee, while the noisy count is enforced to be a positive value, since the decision to include the prefix has already been taken as part of the exponential mechanism.

Tree pruning. After expanding the prefix tree, we prune it based on the noisy counts assigned to trace variants (line 14). A simple pruning strategy removes all prefixes from the tree, for which the noisy count is below a threshold set by parameter p. However, as we will discuss in Section IV-C, pruning may treat harmful and harmless prefixes differently (using two thresholds, $p_{harmless}$ and $p_{harmful}$). In general, we also favour pruning of harmful prefixes to avoid the removal of prefixes that conform to the semantics of the process at hand.

Result construction. Finally, the resulting trace-variant distribution is derived and returned (line 16). To this end, the counts of all prefixes that end with the symbol $\perp$ or that have a length of k are considered. Intuitively, prefixes of length k may represent variants of traces that have not yet finished execution.

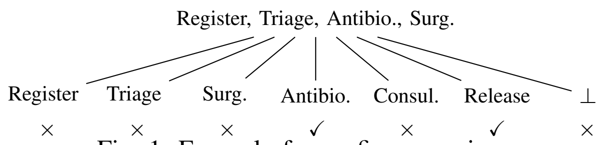


Fig. 1: Example for prefix expansion.

B. A Semantics-aware Score Function

The SaCoFa algorithm uses a score function to assess the utility loss associated with a prefix based on the process behaviour included in the original log L. The function is employed to distinguish harmless prefixes (C_{expand}) from harmful ones (C_{harm}) (line 9) and in the exponential mechanism (line 11). As such, the definition of the score function denotes a design choice that enables us to incorporate different notions of a process' semantics in the anonymization.

To exemplify this design choice, we propose a function that is based on a generalization of the behaviour in the original log. Specifically, we consider a behavioural abstraction that was proposed in the context of the *behavioural appropriateness* measure [12]. This behavioural abstraction defines rules between pairs of activities, reflecting their order and co-occurrence in a log. Specifically, given $a_1, a_2 \in \mathcal{A}(L)$, the rules capture if a_1 will always, never, or sometimes follow (or precede) an activity a_2, not necessarily directly. As such, the set of rules encodes hidden business logic, derived from a log without manual intervention.

We instantiate two score functions based on these rules, a binary and a continuous one. The binary instantiation classifies all prefixes that violate at least one rule as harmful, and all other prefixes as harmless. In contrast, the continuous instantiation counts the number of rule violations to quantify the harmfulness of a prefix. This degree of harmfulness is limited by a user-defined upper bound, since the sensitivity of the exponential mechanism considers the maximum impact that one trace can have on the score function.

As an illustration, consider the example given in Fig. 1, which depicts the expansion of prefix $\langle Register, Triage, Antibio., Surg.\rangle$, based on the example from Table I. In the original log, activity *Surg.* is always followed by activity *Release*, may be followed by activity *Antibio.*, and is never followed by the remaining activities. Respecting these behavioural rules, expansions based on the former two activities are considered harmless, while those in the latter are categorized as harmful.

As mentioned above, the score function may also be defined based on other behavioural models. In particular, it may be grounded in other sets of behavioural rules, such as those presented in [13], [14], which are then instantiated for the original log to capture the semantics of the underlying process. Moreover, rules may also originate from other sources, such as textual documents [15]. However, deriving the rules from the original log ensures that trace variants in the original log are more likely to be preserved.

C. Semantics-aware Pruning

To achieve differential privacy, the number of prefixes to be considered in the SaCoFa algorithm grows exponentially in the prefix length. Consequently, we incorporate pruning of trace variants to achieve tractability, as detailed below.

The need for generalization. Pruning comes with the risk of removing prefixes (and thus trace variants) that are common in the original log, which reduces the utility of the anonymized trace-variant distribution. However, unlike log anonymization with the Laplacian mechanism, our approach supports a differentiation between prefixes that are harmful and harmless for an anonymized distribution. Therefore, we can limit pruning to harmful prefixes. This way, the overall number of pruned prefixes is reduced, but harmless prefixes are always preserved, even when their noisy count is below the pruning parameter p.

A lower number of pruned prefixes, in general, also reduces privacy degradation. However, when pruning solely harmful prefixes, there is a risk to violate the required differential privacy guarantee. That is, if harmful prefixes are characterized based on their absence in the original log, the following may happen: For two neighbouring event logs, that differ by a trace of a variant that appears only in one of the logs, the anonymized variant-distributions may enable the identification of the respective trace. To avoid such situations, we employ a pruning strategy that incorporates behavioural generalization.

Rule-based pruning. By employing the abstraction underlying the behavioural appropriateness measure to identify harmful prefixes for pruning, we avoid to reveal the difference between two neighbouring event logs. Due to the implied behavioural generalization, a trace representing a difference between two logs may also induce a change in the respective rule sets. The changed rules potentially allow for more behaviour, i.e., they increase the set of harmless prefixes. Hence, the anonymized trace-variant distributions of neighbouring logs may differ by *multiple* trace variants, instead of just a single one.

For illustration, consider a log L_1 containing only traces that represent variants from Table Ia. Let $L_2 = L_1 \cup \{t\}$ be a neighbouring log, where t is a trace of the variant $\langle \mathit{Register}, \mathit{Antibio.}, \mathit{Release}\rangle$. Comparing the rule sets of both logs, trace t adds the rule that $\mathit{Register}$ is sometimes followed by $\mathit{Antibio}$. Hence, the SaCoFa algorithm would consider the prefix $\langle \mathit{Register}, \mathit{Antibio.}\rangle$ as harmless when anonymizing L_2, whereas it would be harmful regarding L_1. For L_2, further prefixes would then be derived and considered as harmless, e.g., $\langle \mathit{Register}, \mathit{Antibio.}, \mathit{Release}\rangle$ and $\langle \mathit{Register}, \mathit{Antibio.}, \mathit{Surg.}, \mathit{Release}\rangle$. Hence, the distributions derived for the logs will differ by more than one trace variant.

Therefore, pruning only harmful prefixes requires that a single trace either leads to multiple trace variants to be considered as harmless, or none at all. In practice, this may not be the case, which is why we relax the pruning strategy, as follows. We introduce $p_{harmless}$ and $p_{harmful}$ as separate pruning thresholds for harmless and harmful prefixes, respectively. By setting $1 < p_{harmless} < p_{harmful}$, we favour pruning of harmful prefixes. Yet, by pruning also some harmless prefixes, we ensure that information on the existence of a single trace variant is not disclosed, even if the above requirement is not met. Also, the two aforementioned extreme scenarios could be configured accordingly, i.e., pruning only harmful traces ($p_{harmless} = 1$ and $p_{harmful} > 1$) or pruning all prefixes that introduce new behaviour ($p_{harmless} = 1$ and $p_{harmful} = \infty$).

V. Evaluation

In this section, we investigate if control-flow anonymization with SaCoFa provides higher utility for process discovery than the state of the art. We first review the used datasets (Section V-A) and our experimental setup (Section V-B). We then present our experimental results (Section V-C), before we close with a qualitative discussion of the approach (Section V-D).

A. Dataset

We use three real-world event logs as a basis for our experiments, of which some characteristics are listed in Table II. We selected these logs since they differ in their size and complexity. The *Traffic Fines* log contains data on a very structured process, with just 231 variants over a total of 150,370 traces. In contrast, the *Sepsis* log captures an unstructured hospital-treatment process, containing 846 variants of which the vast majority occurred just once. Finally, the CoSeLoG event log provides a middle ground, with a semi-structured process that consists of 116 variants over 1,434 cases.

TABLE II: Descriptive statistics for the event logs.

Event Log	# Events	# Activities	# Cases	# Variants
CoSeLoG [16]	8,577	27	1,434	116
Sepsis [17]	15,214	16	1,050	846
Traffic Fines [18]	561,470	11	150,370	231

B. Experimental Setup

Baseline. We evaluate our approach against the state-of-the-art approach by Mannhardt et al. [6], which anonymizes the result of trace-variant queries based on the Laplacian mechanism.

Parameter settings. As specified in Section IV-A, SaCoFa takes four parameters: the strength of the desired privacy guarantee ϵ, an upper bound on the trace-variant length k, and the pruning parameters $p_{harmful}$ and $p_{harmless}$. Per event log, we set k so that roughly 80-90% of the original trace variants are covered. For each of the employed privacy guarantees, i.e., $\epsilon = \{1.0, 0.1, 0.01\}$, we explored pruning parameters starting at 2, 20, and 200, respectively, until a configuration was found such that the trace-variant query could be executed within several seconds. Overall, this approach resulted in the parameter settings given in Table III, which we employed for our approach and, if applicable, for the baseline.

Evaluation measures. To quantify the efficacy of our work, we assess the utility of process models discovered on the basis of the anonymized trace-variant distributions generated by SaCoFa and the baseline. For this discovery, we employ the Inductive Miner Infrequent [19] with the default noise threshold of 20%. Then, we determine the utility of a discovered model

TABLE III: Employed parameter settings.

Log	ϵ	k	$p_{harmful}$	$p_{harmless}$
CoSeLoG	1.0	10	3	/
	0.1	10	25	22
	0.01	10	220	200
Sepsis	1.0	23	4	/
	0.1	23	20	15
	0.01	23	190	150
Traffic Fines	1.0	9	2	/
	0.1	9	20	15
	0.01	9	150	120

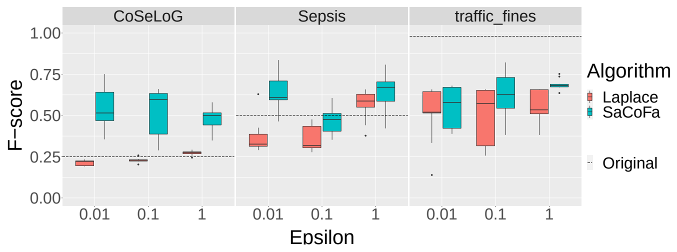


Fig. 2: F-score of discovered process models.

by measuring its F-score in relation to the original event log, i.e., the harmonic mean of the *fitness* [20] and *precision* [21]. Also, we evaluate the *generalization* [22] of the process models discovered from the anonymized event logs.

As a second evaluation dimension, we measure the fraction of easily-recognizable noise introduced into the anonymized event logs. To this end, we apply a standard anomaly detection technique, which employs isolation forests [23], to the anonymized event logs. We train the model on the original log, before using it to detect anomalous traces in the anonymized logs. As features in the learning process, we use a binary encoding of the activities, signalling if they are present in a trace. Moreover, we also encode the presence of directly-follows relations in a trace, with a binary encoding.

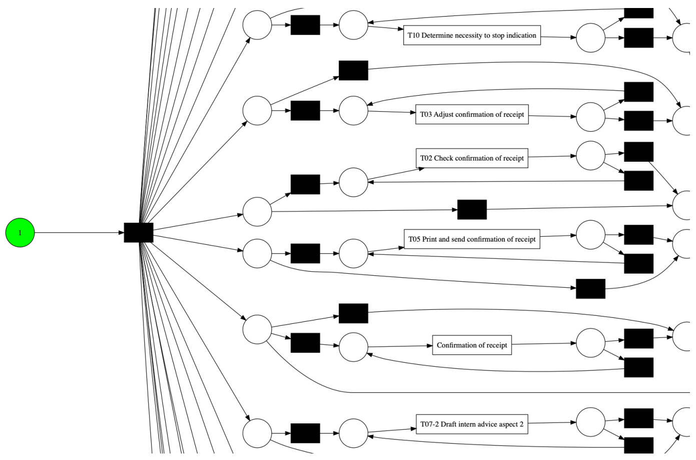
Implementation. To conduct our experiments, we implemented SaCoFa in Python. The source code is available on GitHub[1] under the MIT license. Furthermore, we used PM4Py's [24] implementation of the Inductive Miner and the evaluation measures. The implementation of the isolation forest is available in scikit-learn.[2]

Repetitions. To account for the non-deterministic nature of the algorithms, we perform 10 repetitions of all experiments. In the remainder, we report on the median and the bounds for the upper and lower quartile, using box plots.

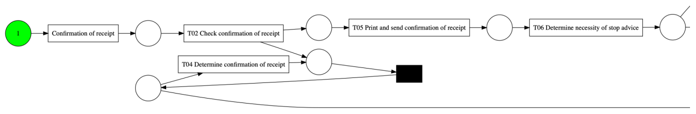
C. Results

Fig. 2 depicts the F-scores of the process models generated by SaCoFa, as well as the Laplacian baseline. As shown, SaCoFa outperforms the baseline considerably, being on par only for the setting with the strongest privacy guarantee ($\epsilon = 0.01$) and the most structured process (Traffic Fines).

[1]https://github.com/samadeusfp/SaCoFa
[2]https://scikit-learn.org/stable/


1
T10 Determine necessity to stop indication
T03 Adjust confirmation of receipt
T02 Check confirmation of receipt
T05 Print and send confirmation of receipt
Confirmation of receipt
T07-2 Draft intern advice aspect 2

(a) Laplacian baseline.


1
Confirmation of receipt
T02 Check confirmation of receipt
T05 Print and send confirmation of receipt
T06 Determine necessity of stop advice
T04 Determine confirmation of receipt

(b) SaCoFa approach.

Fig. 3: Process models obtained for anonymized versions of CoSeLoG ($\epsilon = 0.01$).

In particular, we observe major improvements for the two less-structured processes, which have more activities and longer traces, resulting in significantly higher F-scores obtained using SaCoFa, while providing the same privacy guarantee. Furthermore, our results also illustrate that, sometimes, the anonymized event logs lead to higher F-scores than the original log. The reason being that the Inductive Miner guarantees the generation of a fitting model, which may result in very low precision values. If an anonymized log contains less behaviour, above the threshold adopted by the discovery algorithm to filter noise, the model becomes more compact. It then shows higher precision and, therefore, also a higher F-score.

To further illustrate the above results, Fig. 3 shows excerpts of the process models obtained for the CoSeLoG process under the strongest privacy guarantee ($\epsilon = 0.01$). Here, Fig. 3a shows part of the model discovered from the log anonymized with the Laplacian baseline, while Fig. 3b is based on SaCoFa. As seen, the process model generated with SaCoFa is much more structured. It starts with a sequence of activities that, notably, is also the same in the process model generated from the original event log. In contrast, the model in Fig. 3a is very unstructured and strays far from the original process: nearly all activities can start a trace, be skipped, or executed multiple times.

Next, we turn to an assessment of the generalization of the obtained models. As illustrated in Fig. 4, the models generated based on the logs derived with SaCoFa are more general, i.e., they abstract more from the behaviour represented in the event log. Combined with the results for the F-score, we conclude that the logs anonymized with SaCoFa indeed have a higher utility for process discovery.

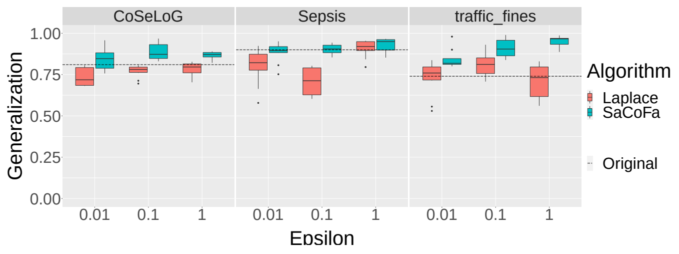


Fig. 4: Generalization of discovered process models.

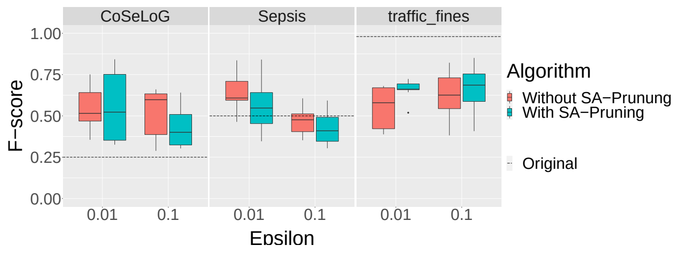


Fig. 6: F-score for configurations with and without pruning.

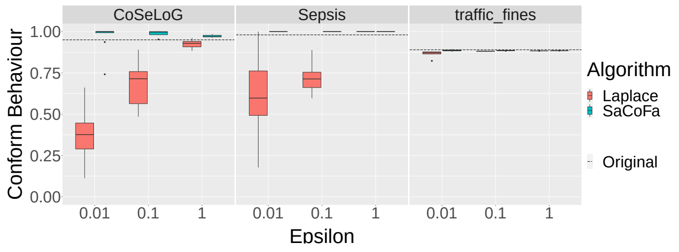


Fig. 5: Relative frequency of normal behaviour in event logs.

After investigating the utility for process discovery, we turn to our second evaluation perspective, the presence of easily-recognizable noise. In Fig. 5, we show the percentage of behaviour that is classified as normal behaviour by the aforementioned anomaly detection technique. While SaCoFa and the baseline both achieve good results for the Traffic Fines dataset, there is a clear trend for the other two logs: the baseline produces much more noise, directly recognizable as anomalous. Therefore, the traces introduced by SaCoFa are more in line with the original process' behaviour.

Finally, we investigate the effects of semantics-aware pruning, as all previously shown result have been obtained with regular pruning. Fig. 6 shows the F-score of models discovered from logs anonymized with and without pruning. Overall, semantics-aware pruning turns out to only be beneficial for the Traffic Fines log, which is the most structured one. For the less-structured logs, the F-score actually decreases in comparison to the approach without pruning. We attribute this observation to the significance of the rules used to separate harmful and harmless prefixes. Apparently, they are not always sophisticated enough to compensate for the additional variance introduced to the trace-variant distribution.

D. *Discussion*

Runtime aspects. As mentioned in Section V-B, parameter settings must be carefully selected in order for trace-variant queries to complete in a reasonable time. Specifically, we observed that the anonymization procedure either terminated within seconds, or not all, for both SaCoFa and the baseline. This reveals that there is a clear point when the prefix growth makes the trace-variant query intractable. So far this point is determined by step-wise altering the maximal variant length (k) and pruning parameters for a given ϵ. Nevertheless, we observe that SaCoFa can compute results for lower pruning thresholds than the Laplacian baseline. However, to ensure a fair comparison, we used the same pruning parameters for all mechanisms in our experiments.

Non-binary score functions. Beyond determining if a prefix is harmful or not, the behavioural appropriateness-based score can be used to quantify its degree of harmfulness. However, the sensitivity of the exponential mechanism depends on the maximal impact that a single case can have on the score function, i.e., on the function's maximal value (see Section III). Therefore, if we define a score function that quantifies harmfulness in, e.g., the range $[0, 3]$, the query's sensitivity would be $\Delta f = 3$, instead of $\Delta f = 1$ for a binary assessment. Since the exponential mechanism needs to insert more noise for a higher sensitivity, the benefit obtained from quantifying harmfulness in a non-binary manner, must outweigh the increased sensitivity that comes with it. While this did not appear to be the case in our current experiments, we believe that this approach could still work for more sophisticated score functions, tailored to the specifics of the process at hand. In any case, it is important to consider this trade-off and take it into account when choosing the right pruning parameter values.

VI. Related Work

We introduced SaCoFa as an approach for control-flow anonymization that answers trace-variant queries, while guaranteeing differential privacy. The state of the art to derive a trace-variant distribution, and the related directly-follows graph, under differential privacy, uses the Laplacian mechanism [6]. As discussed, this neglects a process' semantics, leading to potentially low data utility and noise that can be easily recognized. For trace-variant queries, Elkoumy et al. [25] further studied the relation between utility and risk, while the PRIPEL framework [26] uses trace-variant queries as a basis for privacy-preserving event log publishing.

Beyond differential privacy, the anonymization of event logs based on other privacy guarantees was studied. PRETSA [27] sanitizes event logs to ensure k-anonymity and t-closeness, which are guarantees based on the idea of grouping similar cases together. Furthermore, a process mining-specific extension of k-anonymity, called TLKC, was introduced in [28]. Previous work focused on improving the utility of these techniques through feature learning-based distance metrics [29]. Another group-based approach was introduced by Batista et al. [30], based on the uniformization of events within a group of individuals. The issue of continuously publishing anonymized event logs was studied in [31].

Approaching privacy preservation from the viewpoint of the analysis techniques used in process mining, it was shown how multi-party computation enables the construction of process models based on inter-organizational processes, without sharing the data between parties [32]. Other approaches target privacy-aware role mining [33] and the establishment of privacy-aware process performance indicators through the enforcement of differential privacy [34].

VII. Conclusion

Targeting control-flow anonymization for event logs, we introduced the SaCoFa approach to answer trace-variant queries. It is based on a prefix tree construction and the exponential mechanism. Unlike state-of-the-art techniques that leverage the Laplacian mechanism and, hence, introduce noise randomly to achieve differential privacy, SaCoFa incorporates the semantics of the underlying process when inserting noise, achieving the same privacy guarantee. Specifically, we introduced a score function that differentiates prefixes as being harmful or harmless, thereby incorporating a process' semantics in the anonymization. We further showed how an assessment of the harmfulness of prefixes may also guide pruning decisions in the prefix tree construction in order to achieve tractability.

Our evaluation experiments highlight that process models generated based on control-flow behaviour anonymized with SaCoFa have higher utility than those obtained with the state of the art. At the same time, the models are more general and, hence, abstract better from the behaviour represented in the event log. Moreover, we also showed that SaCoFa introduces less noise that is directly labelled as anomalous compared to the state of the art.

References

[1] W. Van Der Aalst, "Process mining: Overview and opportunities," *ACM Transactions on Management Information Systems (TMIS)*, vol. 3, no. 2, pp. 1–17, 2012.

[2] P. Voigt and A. Von dem Bussche, "The eu general data protection regulation (gdpr)," *A Practical Guide, 1st Ed., Cham: Springer International Publishing*, 2017.

[3] S. N. von Voigt, S. A. Fahrenkrog-Petersen, D. Janssen, A. Koschmider, F. Tschorsch, F. Mannhardt, O. Landsiedel, and M. Weidlich, "Quantifying the re-identification risk of event logs for process mining," in *International Conference on Advanced Information Systems Engineering*. Springer, 2020, pp. 252–267.

[4] G. Elkoumy, S. A. Fahrenkrog-Petersen, M. F. Sani, A. Koschmider, F. Mannhardt, S. N. von Voigt, M. Rafiei, and L. von Waldthausen, "Privacy and confidentiality in process mining–threats and research challenges," *arXiv preprint arXiv:2106.00388*, 2021.

[5] S. A. Fahrenkrog-Petersen, "Providing privacy guarantees in process mining." in *CAiSE (Doctoral Consortium)*, 2019, pp. 23–30.

[6] F. Mannhardt, A. Koschmider, N. Baracaldo, M. Weidlich, and J. Michael, "Privacy-preserving process mining," *Business & Information Systems Engineering*, vol. 61, no. 5, pp. 595–614, 2019.

[7] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Theory of cryptography conference*. Springer, 2006, pp. 265–284.

[8] F. McSherry and K. Talwar, "Mechanism design via differential privacy," in *FOCS*. IEEE, 2007, pp. 94–103.

[9] I. Wagner and D. Eckhoff, "Technical privacy metrics: a systematic survey," *ACM Computing Surveys (CSUR)*, vol. 51, no. 3, pp. 1–38, 2018.

[10] C. Dwork, "Differential privacy: A survey of results," in *International conference on theory and applications of models of computation*. Springer, 2008, pp. 1–19.

[11] H. B. Kartal, X. Liu, and X.-B. Li, "Differential privacy for the vast majority," *ACM TMIS*, vol. 10, no. 2, pp. 1–15, 2019.

[12] A. Rozinat and W. M. P. van der Aalst, "Conformance checking of processes based on monitoring real behavior," *Inf. Syst.*, vol. 33, no. 1, pp. 64–95, 2008.

[13] M. Weidlich and J. M. E. M. van der Werf, "On profiles and footprints - relational semantics for petri nets," in *Application and Theory of Petri Nets - 33rd International Conference, PETRI NETS 2012, Hamburg, Germany, June 25-29, 2012. Proceedings*, ser. Lecture Notes in Computer Science, S. Haddad and L. Pomello, Eds., vol. 7347. Springer, 2012, pp. 148–167.

[14] A. Polyvyanyy, M. Weidlich, R. Conforti, M. L. Rosa, and A. H. M. ter Hofstede, "The 4c spectrum of fundamental behavioral relations for concurrent systems," in *PETRI NETS*, G. Ciardo and E. Kindler, Eds., vol. 8489. Springer, 2014, pp. 210–232.

[15] H. van der Aa, H. Leopold, and H. A. Reijers, "Checking process compliance against natural language specifications using behavioral spaces," *Inf. Syst.*, vol. 78, pp. 83–95, 2018.

[16] J. Buijs, "Receipt phase of an environmental permit application process ('WABO'), CoSeLoG project," 8 2014. [Online]. Available: https://data.4tu.nl/articles/dataset/Receipt_phase_of_an_environmental_permit_application_process_WABO_CoSeLoG_project/12709127

[17] F. Mannhardt, "Sepsis cases-event log," *Eindhoven University of Technology. Dataset*, pp. 227–228, 2016.

[18] M. M. De Leoni and F. F. Mannhardt, "Road traffic fine management process," 2015, https://doi.org/10.4121/uuid:270fd440-1057-4fb9-89a9-b699b47990f5.

[19] S. J. J. Leemans, D. Fahland, and W. M. P. van der Aalst, "Discovering block-structured process models from event logs containing infrequent behaviour," in *BPM Workshops*, 2013, pp. 66–78.

[20] A. Berti and W. M. van der Aalst, "Reviving token-based replay: Increasing speed while improving diagnostics." in *ATAED@ Petri Nets/ACSD*, 2019, pp. 87–103.

[21] J. Munoz-Gama and J. Carmona, "A fresh look at precision in process conformance," in *BPM*. Springer, 2010, pp. 211–226.

[22] J. C. Buijs, B. F. van Dongen, and W. M. van der Aalst, "Quality dimensions in process discovery: The importance of fitness, precision, generalization and simplicity," *International Journal of Cooperative Information Systems*, vol. 23, no. 01, p. 1440001, 2014.

[23] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in *2008 eighth ieee international conference on data mining*. IEEE, 2008, pp. 413–422.

[24] A. Berti, S. J. van Zelst, and W. van der Aalst, "Process mining for python (pm4py): bridging the gap between process-and data science," *arXiv preprint arXiv:1905.06169*, 2019.

[25] G. Elkoumy, A. Pankova, and M. Dumas, "Privacy-preserving directly-follows graphs: Balancing risk and utility in process mining," *arXiv preprint arXiv:2012.01119*, 2020.

[26] S. A. Fahrenkrog-Petersen, H. van der Aa, and M. Weidlich, "Pripel: Privacy-preserving event log publishing including contextual information," in *BPM*. Springer, 2020, pp. 111–128.

[27] ——, "Pretsa: event log sanitization for privacy-aware process discovery," in *2019 International Conference on Process Mining (ICPM)*. IEEE, 2019, pp. 1–8.

[28] M. Rafiei and W. M. van der Aalst, "Group-based privacy preservation techniques for process mining." Elsevier, 2021, p. 101908.

[29] F. Rösel, S. A. Fahrenkrog-Petersen, H. van der Aa, and M. Weidlich, "A distance measure for privacy-preserving process mining based on feature learning," *arXiv preprint arXiv:2107.06578*, 2021.

[30] E. Batista and A. Solanas, "A uniformization-based approach to preserve individuals' privacy during process mining analyses," *Peer-to-Peer Networking and Applications*, pp. 1–20, 2021.

[31] M. Rafiei and W. M. van der Aalst, "Privacy-preserving continuous event data publishing," *arXiv preprint arXiv:2105.11991*, 2021.

[32] G. Elkoumy, S. A. Fahrenkrog-Petersen, M. Dumas, P. Laud, A. Pankova, and M. Weidlich, "Secure multi-party computation for inter-organizational process mining," in *BPMDS*. Springer, 2020, pp. 166–181.

[33] M. Rafiei and W. M. van der Aalst, "Mining roles from event logs while preserving privacy," in *BPM Workshops*. Springer, 2019, pp. 676–689.

[34] M. Kabierski, S. A. Fahrenkrog-Petersen, and M. Weidlich, "Privacy-aware process performance indicators: Framework and release mechanisms," in *CAiSE*. Springer, 2021, pp. 19–36.