Estimating Process Conformance by Trace Sampling and Result Approximation

Martin Bauer, Han van der Aa, Matthias Weidlich

Department of Computer Science, Humboldt-Universität zu Berlin, Berlin, Germany
martin.bauer | han.van.der.aa | matthias.weidlich @hu-berlin.de

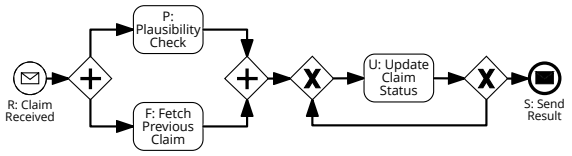
Abstract. The increasing volume of event data that is recorded by information systems during the execution of business processes creates manifold opportunities for process analytics. Specifically, conformance checking compares the behaviour as recorded by an information system to a model of desired behaviour. Unfortunately, state-of-the-art conformance checking algorithms scale exponentially in the size of both the event data and the model used as input. At the same time, event data used for analysis typically relates only to a certain interval of process execution, not the entire history. Given this inherent data incompleteness, we argue that an understanding of the overall conformance of process execution may be obtained by considering only a small fraction of a log. In this paper, we therefore present a statistical approach to ground conformance checking in trace sampling and conformance approximation. This approach reduces the runtime significantly, while still providing guarantees on the accuracy of the estimated conformance result. Comprehensive experiments with real-world and synthetic datasets illustrate that our approach speeds up state-of-the-art conformance checking algorithms by up to three orders of magnitude, while largely maintaining the analysis accuracy.

1 Introduction

Process-oriented information systems coordinate the execution of a set of actions to reach a business goal [15]. The behaviour of such systems is commonly described by process models that define a set of activities along with execution dependencies. However, once event data is recorded during runtime, the question of conformance emerges [9]: how do the modelled behaviour of a system and its recorded behaviour relate to each other? Answering this question is required to detect, interpret, and compensate deviations between a model of a process-oriented information system and its actual execution.

Driven by trends such as process automation, data sensing, and large-scale instrumentation of process-related resources, the volume of event data and the frequency at which it is generated is increasing in today's world: Event logs comprise up to billions of events [2]. Also, information systems are subject to frequent changes [34], so that analysis is often a continuous process, repeated when new event data becomes available.

Acknowledging the resulting need for efficient analysis, various angles have been followed to improve the runtime performance of state-of-the-art, alignment-based conformance checking algorithms [4], which suffer from an exponential worst-case complexity. Efficiency improvements have been obtained through the use of search-based methods [13, 25], planning algorithms [21], and distributed computing [16, 22]. Furthermore,



(a)

Case	Event Sequence
1	r p f f u s
2	r f p u f s
3	r p f u u u s
4	r p f f ⊥ s

(b)

Fig. 1: (a) Model of a claim handling process; (b) events of four process executions.

several authors suggested to compromise correctness and approximate conformance results to gain efficiency, e.g., by employing approximate alignments [29] or applying divide-and-conquer schemes in the computation of conformance results [3, 12, 20, 23]. However, these approaches primarily target the applied algorithms. Fundamentally, they still require the consideration of *all*, possibly billions, of recorded events.

In practice, an event log is recorded for a certain interval of process execution, not the entire history. Given this inherent incompleteness of event data, which is widely acknowledged [1, 17, 26], analysis often strives for a general understanding of the conformance of process execution. This may relate, e.g., to the overall fitness of recorded and modelled behaviour [4] or the activities that denote hotspots of non-conformance [3].

In this paper, we argue that for a general understanding of the overall conformance, it is sufficient to compute conformance results for only a small fraction of an event log. Since the latter per se provides an incomplete view, minor differences in the conformance results obtained for the whole log and a partial log may be attributed to the inherent uncertainty of the conformance checking setting. We illustrate this idea with a claim handling process in Fig. 1. Here, the events recorded for the first case indicate non-conformance, as a previous claim is fetched twice (f). Considering also the second case, the average amount of non-conformance (one deviation) and the set of non-conforming activities ($\{F\}$) are the same, though, despite the different sequence of events. Considering also the third and fourth case, new information on the overall conformance of process execution is obtained. Yet, the fourth case resembles the first one. Hence, its conformance (two deviations w.r.t. the model) may be approximated based on the result of the first case (one deviation) and the difference between both event sequences (one event differs).

To realise the above ideas, we follow two complimentary angles to avoid computation of conformance results for all available data. First, we contribute an incremental approach based on trace sampling, which, for each trace, assesses whether it yields new information on the overall conformance. Assuming the view of a series of binomial experiments, we establish bounds on the error of the conformance result derived from a partial event log. Second, we show how trace sampling is combined with result approximation that, instead of computing a conformance result for a trace at hand, relies on a worst-case approximation of its implications on the overall conformance. This way, we further reduce the amount of data for which conformance results are actually computed. We instantiate this framework for two types of conformance results as mentioned above, a numerical fitness measure and a distribution of conformance issues over all activities.

In the remainder, we first give preliminaries in Section 2. We then introduce our approach to sample-based (Section 3) and approximation-based (Section 4) conformance checking. Experimental results using real-world and synthetic event logs are presented in Section 5. We review related work in Section 6, before we conclude in Section 7.

2 Preliminaries

Events and event logs. We adopt an event model that builds upon a set of activities $\mathcal{A}$. An event recorded by an information system is assumed to be related to the execution of one of these activities. By $\mathcal{E}$, we denote the universe of all events. A single execution of a process, called 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, $L \subseteq 2^{\mathcal{E}^*}$. Our example in Fig. 1b defines four traces. While each event is unique, we represent them with small letters $\{r, p, f, u, s\}$, that indicate for which activity of the process model, denoted by respective capital letters $\{R, P, F, U, S\}$, the execution is signalled. Two distinct traces that indicate the same sequence of activity executions are of the same *trace variant*.

Process models. A process model defines the execution dependencies between the activities of a process. For our purposes, it is sufficient to abstract from specific process modelling languages and focus on the behaviour defined by a model. That is, a process model defines a set of execution sequences, $M \subseteq \mathcal{A}^*$, that capture sequences of activity executions that lead the process to its final state. For instance, the model in Fig. 1a defines the execution sequences $\langle R, P, F, U, S\rangle$ and $\langle R, F, P, U, S\rangle$, potentially including additional repetitions of U. We write $\mathcal{M}$ for the set of all process models.

Alignments. State-of-the-art techniques for conformance checking construct *alignments* between traces and execution sequences of a model to detect deviations [4, 23]. An alignment between a trace ξ and a model M, denoted by $\sigma(\xi, M)$ in the remainder, is a sequence of steps, each step comprising a pair of an event and an activity, or a *skip* symbol $\perp$, if an event or activity is without counterpart. For instance, for the non-conforming trace ξ_1 (case 1 from Fig. 1b), an alignment is constructed as follows:

Trace ξ_1	r	p	f	f	u	s
Execution sequence	R	P	F	$\perp$	U	S

Assigning costs to skip steps, a cost-optimal alignment (not necessarily unique) is constructed for a trace in relation to all execution sequences of a model [4]. An optimal alignment then enables the quantification of non-conformance. Specifically, the *fitness* of a log with respect to a given model is computed as follows:

$$\text{fitness}(L, M) = 1 - \frac{\sum_{\xi \in L} c(\xi, M)}{\sum_{\xi \in L} c(\xi, \emptyset) + |L| \times \min_{x \in M} c(\langle\rangle, \{x\})} \tag{1}$$

Here, $c(\xi, M)$ is the aggregated cost of an optimal alignment $\sigma(\xi, M)$. The denominator captures the maximum possible cost per trace, i.e., the sum of the costs of aligning a trace with an empty model, $c(\xi, \emptyset)$, and the minimal costs of aligning an empty trace with the model, $\min_{x \in M} c(\langle\rangle, \{x\})$. Using a standard cost function (all skip steps have equal costs), the fitness value of the example log $\{\xi_1, \xi_2, \xi_3, \xi_4\}$ in Fig. 1b is 0.9. Alignments further enable the detection of hotspots of non-conformance. To this end, the conformance result can be defined in terms of a *deviation distribution* that captures the relative frequency with which an activity (not to be confused with a task of a process model) is part of a conformance violation. For a log L and a model M,

this distribution follows from skip steps in the optimal alignments of all traces. It is formalised based on a bag of activities, $dev(L, M) : \mathcal{A} \to \mathbb{N}_0$ (note that multiple skip steps may relate to a single activity even in the alignment of one trace). The relative deviation frequency of an activity $a \in \mathcal{A}$ is then obtained by dividing the number of occurrences of a in the bag of deviations by the total number of deviations, i.e., $f_{dev(L,M)}(a) = dev(L, M)(a)/|dev(L, M)|$.

For our example in Fig. 1, assuming that skip steps relate to the highlighted trace positions, it holds that $dev(\{\xi_1, \xi_2, \xi_3, \xi_4\}, M) = [F^3, U]$ and $f_{dev(\{\xi_1,\ldots,\xi_4\},M)}(F) = 3/4$, so that the fetching of a previous claim (F) is identified as a hotspot of non-conformance.

3 Sample-Based Conformance Checking

This section describes how trace sampling can be used to improve the efficiency of conformance checking. The general idea is that it often suffices to only compute alignments for a subset of all trace variants to gain insights into the overall conformance of a log to a model. However, we randomly sample an event log trace-by-trace, not by trace variant, which avoids to load the entire log and step-wise reveals the distribution of traces among the variants. At some point, though, the sampled traces then do not provide new information on the overall conformance of the process.

In the remainder of this section, we first describe a general framework for sample-based conformance checking (Section 3.1), which we then instantiate for two types of conformance results: fitness (Section 3.2) and deviation distribution (Section 3.3).

3.1 Statistical Sampling Framework

To operationalise sample-based conformance checking, we regard it as a series of binomial experiments. In this, we follow a log sampling technique introduced in the context of process discovery [5, 7] and lift it to the setting of conformance checking.

Information novelty. When parsing a log trace-by-trace, some traces may turn out to provide information on the conformance of a log to a model that is similar or equivalent to the information provided by previously encountered traces. To assess whether this is the case, we capture the conformance information associated with a log by a *conformance function* $\psi : 2^{\mathcal{E}^*} \times \mathcal{M} \to \mathcal{X}$. That is, $\psi(L, M)$ is the conformance result (of some domain $\mathcal{X}$) between a log L and a model M. If we are interested in the distribution of deviations, ψ provides information on the model activities for which deviations are observed, whereas for fitness, it would return the fitness value.

Based thereon, we define a random Boolean predicate $\gamma(L', \xi, M)$ that captures whether a trace $\xi \in \mathcal{E}^*$ provides new information on the conformance with model M, i.e., whether it changes the result obtained already for a set of previously observed traces $L' \subseteq 2^{\mathcal{E}^*}$. Assuming that the distance between conformance results can be quantified by a function $d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_0^+$, we define a *new information predicate* as:

$$\gamma(L', \xi, M) \Leftrightarrow d(\psi(L', M), \psi(\{\xi\} \cup L', M)) > \epsilon. \tag{2}$$

Here, $\epsilon \in \mathbb{R}_0^+$ is a relaxation parameter. If incorporating trace ξ changes the conformance result by more than ϵ, then it adds new information over L'.

Framework. We exploit the notion of information novelty for hypothesis testing when sampling traces from an event log L. We determine when *enough* sampled traces have been included in a log $L' \subseteq L$ to derive an understanding of its overall conformance to a model M. Following the interpretation of log sampling as a series of binomial experiments [5], L' is regarded as sufficient if the algorithm consecutively draws a certain number of traces that did not contain new information. Specifically, with δ as a measure that bounds the probability of a newly sampled trace to provide new information over L', at a significance level α, a minimum sample size N is computed. Based on the normal approximation to the binomial distribution, the latter is given as $N \geq 1/2\delta \left(-2\delta^2 + z^2 + \sqrt{z}\right)$, where z corresponds to the realisation of a standardised normal random variable for $1 - \alpha$ (one-sided hypothesis test). As such, N is calculated given values for δ and α for the desired levels of similarity and significance, respectively.

Consider $\alpha = 0.01$ and $\delta = 0.05$, so that $N \geq 128$. Hence, after observing 128 traces without new information, sampling can be stopped knowing with 0.99 confidence that the probability of finding new information in the remaining log is less than 0.05.

Using the above formulation, our framework for sample-based conformance checking is presented in Alg. 1. The algorithm takes as input an event log L, a process model M, the number of trials that need to fail N, a predicate γ to determine whether a trace provides new information, and a conformance function ψ. Going through L trace-by-trace (lines 3–12), the algorithm conducts a series of binomial experiments that check, if a newly sampled trace provides new information according to the predicate γ (line 5). Once N consecutive traces without new information have been selected, the procedure stops and the conformance result is derived based on the sampled log L'.

Result re-use. Note that the algorithm provides a conceptual view, in the sense that checking the new information predicate $\gamma(L', \xi, M)$ in line 5 according to Eq. 2, requires the computation of $\psi(L', M)$ and $\psi(\{\xi\} \cup L', M)$ in each iteration. A technical realisation of this algorithm, of course, shall exploit that most types of conformance results can be computed incrementally. For instance, considering fitness and the deviation distribution, an alignment is computed only once per *trace variant*, i.e., per unique sequence of activity executions, and reused in the iterations of the algorithm. Also, the value of $\psi(L', M)$ for $\gamma(L', \xi, M)$ in line 5 is always known from the previous iteration, while the conformance result in line 13 is not actually computed at this stage, as the respective result is known from the last evaluation of $\gamma(L', \xi, M)$.

In the next sections, we discuss how to define γ when the conformance function assesses the fitness of a log to a model, or the observed deviation distribution.

3.2 Sample-Based Fitness

The overall conformance of a log to a model may be assessed by considering the log fitness (see Section 2) as a conformance function, $\psi_{fit}(L, M) = \text{fitness}(L, M)$. Then, determining whether a trace ξ provides new information over a log sample L' requires us to assess, if incorporating ξ leads to a difference in the overall fitness for the sampled log. Following Eq. 2, we capture this by computing the absolute difference between the fitness value for traces in the sample L' and the value of the sample plus the new trace:

$$d_{fit}(\psi_{fit}(L', M), \psi_{fit}(\{\xi\} \cup L', M)) = \left|\text{fitness}(L', M) - \text{fitness}(\{\xi\} \cup L', M)\right|. \quad (3)$$

Algorithm 1: Framework for Sample-Based Conformance Checking

input : L, an event log; M, a process model; N, a number of failed trials to observe;
γ, a predicate that holds true, if a trace provides new information;
ψ, a conformance function.
output : $\psi(L')$, the conformance results for sampled traces.

$L', \hat{L} \leftarrow \emptyset$; /* The sampled logs, overall and for current experiment series */
$i \leftarrow 0$; /* The number of current iterations without new information */
repeat /* Repeat until N traces without new information have been seen */
  $\xi \leftarrow select(L \setminus \hat{L})$; /* Sample a single trace */
  /* Check if ξ provides new information. Re-uses alignments of traces of the same variant and the previous conformance result $\psi(L', M)$. */
  if $\gamma(L', \xi, M)$ **then**
    $i \leftarrow 0$; /* Reset the counter */
    $\hat{L} \leftarrow \emptyset$; /* Reset log for current experiment series */
  else
    $i \leftarrow i + 1$; /* Increment the counter */
    $\hat{L} \leftarrow \hat{L} \cup \{\xi\}$; /* Add trace to log for current experiment series */
  $L' \leftarrow L' \cup \{\xi\}$; /* Add trace to overall sampled log */
until $i \geq N \vee L' = L$;
return $\psi(L')$; /* Return results based on the overall sampled log */

If this distance is smaller than the relaxation parameter ϵ, the change in the overall fitness value induced by trace ξ is considered to be negligible.

To illustrate this, consider a scenario with $\epsilon = 0.03$ and a sample consisting of the traces ξ_1 and ξ_3 of our running example (Fig. 1). Then, the log fitness for $\{\xi_1, \xi_3\}$ is 0.95. In this situation, if the next sampled trace is ξ_2, the distance function yields $|\text{fitness}(\{\xi_1, \xi_3\}, M) - \text{fitness}(\{\xi_1, \xi_2, \xi_3\}, M)| = 0.95 - 0.93 = 0.02$. In this case, since the distance is smaller than ϵ, we would conclude that the additional consideration of ξ_2 does not provide new information. By contrast, considering trace ξ_4 would yield a distance of $0.95 - 0.89 = 0.06$. This indicates that trace ξ_4 would imply a considerable change in the overall fitness value, i.e., it provides new information.

3.3 Sample-Based Deviation Distributions

Next, we instantiate the above framework for conformance checking based on the deviation distribution. As detailed in Section 2, this distribution captures the relative frequency with which activities are related to conformance issues.

To decide whether a trace ξ provides new information over a log sample L', we assess if the deviations obtained for ξ lead to a considerable difference in the overall deviation distribution. As such, the distance function for the predicate γ needs to quantify the difference between two discrete frequency distributions. This suggests to employ the L1-distance, also known as the *Manhattan distance*, as a measure:

$$d_{dev}(\psi_{dev}(L', M), \psi_{dev}(\{\xi\} \cup L', M)) = \sum_{a \in \mathcal{A}} \left| f_{dev(L', M)}(a) - f_{dev(\{\xi\} \cup L', M)}(a) \right| \quad (4)$$

Algorithm 2: Framework for Approximation-Based Conformance Checking

input : L', a log sample; M, a process model; ξ, a trace sampled of a yet unseen variant with $\xi \notin L'$; d, a distance function; ϵ, a relaxation parameter; k, a similarity threshold; $\hat{\psi}$, a partially approximating conformance function; $L'' \subseteq L'$, traces for which approximated results are used.

output : (v, L''), where $v \in \{true, false\}$ indicates whether ξ provides new information; L'' is the updated set of traces, for which approximated results are used.

$\xi_r \leftarrow \text{argmin}_{\xi' \in L' \setminus L''} sim(\xi, \xi')$; /* Select most similar trace */
if $sim(\xi, \xi_r) \leq k$ **then** /* Check if ξ and ξ_r are k-similar */
  $\Phi \leftarrow approx(\xi, \xi_r, L', M)$; /* Derive all possible approximations */
  if $\exists\, \phi \in \Phi : d(\hat{\psi}(L', L'', M), \phi) > \epsilon$ **then**
    /* Approx. indicates new information, use actual result */
    return $(true, L'')$;
  else /* Approx. does not indicate new information, use approx. result */
    return $(false, L'' \cup \{\xi\})$;
else /* No k-similar trace available, check for new information */
  $v \leftarrow d(\hat{\psi}(L', L'', M), \hat{\psi}(\{\xi\} \cup L', L'', M)) > \epsilon$;
  return (v, L'');

Taking up our example from Fig. 1, processing only the trace ξ_1, all deviations are related to the activity of fetching an earlier claim, i.e., $f_{dev(\{\xi_1\},M)}(F) = 1$. Notably, this does not change when incorporating traces ξ_2 and ξ_3, i.e., $f_{dev(\{\xi_1,\xi_2,\xi_3\},M)}(F) = 1$, as they do not provide new information in terms of the deviation distribution. If, after processing trace ξ_1, however, we sample ξ_4, we do observe such a difference: based on $f_{dev(\{\xi_1,\xi_4\},M)}(F) = 2/3$ and $f_{dev(\{\xi_1,\xi_4\},M)}(U) = 1/3$, we compute a Manhattan distance of $2/3$. With a relaxation parameter ϵ that is smaller than this value, we conclude that ξ_4 provides novel information.

Given the distance functions based on trace fitness and deviation distribution, it is interesting to note that these behave differently, as illustrated in our example: If the log is $\{\xi_1, \xi_2\}$ and trace ξ_3 is sampled next, the overall fitness changes. Yet, since ξ_3 is a conforming trace, it does not provide new information on the distribution of deviations.

4 Approximation-Based Conformance Checking

This section shows how conformance results can be approximated, further avoiding the need to compute a conformance result for certain traces. Our idea is to derive a worst-case approximation for traces that are similar to variants for which results have previously been derived. Approximation complements the sampling method of Section 3: Even when a trace of a yet unseen variant is sampled, we decide whether to compute an actual conformance result or whether to approximate it. As such, the decision on whether a trace provides new information may be taken either based on a computed or an approximated result.

Against this background, our technique for approximation-based conformance checking, formalised in Alg. 2, extends our procedure given in Alg. 1. In fact, it primarily provides a realisation of checking the new information predicate $\gamma(L', \xi, M)$, as done in

line 5 of Alg. 1. That is, whether the sampled trace ξ, of an unseen variant, provides new information is potentially decided based on the approximated, rather than computed impact of it on the overall conformance result. At the same time, however, the algorithm also needs to keep track of all sampled traces $L'' \subseteq L'$, for which the approximated results shall be used whenever a conformance result is computed. This leads to an adaptation of the conformance function ψ, i.e., we consider a *partially approximating conformance* function $\hat{\psi} : 2^{\mathcal{E}^*} \times 2^{\mathcal{E}^*} \times \mathcal{M} \to \mathcal{X}$. Given a log L' and a subset $L'' \subseteq L'$, this function approximates the conformance result $\psi(L', M)$ by computing solely $\psi(L' \setminus L'', M)$, i.e., the impact of traces L'' is not precisely computed. In the same way, to use the approximation technique as part of sample-based conformance checking, the use of the conformance function ψ in Alg. 1 also has to be adapted accordingly.

Turning to the details of Alg. 2, its input includes a log sample L', a sampled trace $\xi \notin L'$, and a process model M, i.e., the arguments of γ in line 5 of Alg. 1, as well as a distance function d and relaxation parameter ϵ from the definition of γ (Eq. 2). Moreover, there is a similarity threshold k to determine which traces may be used for approximation. Finally, the aforementioned set of traces L'' for which results shall be approximated and the respective adapted conformance function $\hat{\psi}$ are given as input.

From the sampled traces for which approximation is not applied (i.e., $L' \setminus L''$), the algorithm first selects the trace that is most similar to ξ, referred to as the reference trace ξ_r (line 1). Then, we assess whether this similarity is above the threshold k (line 2). If not, we check the trace for new information as done before, just using the adapted conformance function (lines 9–10). If ξ_r is sufficiently similar, however, we perform a worst-case approximation of the impact of ξ on the overall conformance result based on ξ_r (line 3). As part of that, we may obtain several different approximations Φ, each of which is checked whether it indicates new information over the current sample L' (line 4). Only if this is not the case, we conclude that ξ indeed does not provide new information and, by adding it to L'' make sure that its impact on the overall conformance will always only be approximated, but never precisely computed (line 7).

Next, we give details on the assessment of trace similarity (function sim, Section 4.1) and the conformance result approximation (function $approx$, Section 4.2).

4.1 Trace Similarity

Given a trace ξ and the part of the sample log for which approximation did not apply ($L' \setminus L''$), Alg. 2 requires us to identify a reference trace ξ_r that is most similar according to some function $sim : \mathcal{E}^* \times \mathcal{E}^* \to [0, 1]$. As we consider conformance results that are based on alignments, we define this similarity function based on the alignment cost of two traces. To this end, we consider a function c_t, which, in the spirit of function c discussed in Section 2, is the sum of the costs assigned to skip steps in an optimal alignment of two traces. To obtain a similarity measure, we normalise this aggregated cost by a maximal cost, which is obtained by aligning each trace with an empty trace. This normalisation resembles the one discussed for the fitness measure in Section 2. We define the similarity function for traces as $sim(\xi, \xi') = 1 - c_t(\xi, \xi')/(c_t(\xi, \langle\rangle) + c_t(\xi', \langle\rangle))$.

Considering trace $\xi_4 = \langle r, p, f, f, s \rangle$ of our running example, the most similar trace (assuming equal costs for all skip steps) is $\xi_1 = \langle r, p, f, f, u, s \rangle$, with $c_t(\xi_1, \xi_4) = 1$ and, thus, $sim(\xi_1, \xi_4) = 10/11$.

4.2 Conformance Result Approximation

In the approximation step of Alg. 2, we derive a set of worst-case approximations of the impact of the trace ξ on the overall conformance result, using the reference trace ξ_r (which is at least k-similar). Based thereon, it is decided whether ξ provides new information. The approximation, however, depends on the type of conformance result.

Fitness approximation. To approximate the impact of trace ξ on the overall fitness, we compute a single value, i.e., $approx(\xi, \xi_r, L', M)$ in line 3 of Alg. 2 yields a singleton set. This value is derived by reformulating Eq. 3, which captures the change in fitness induced by a sample trace. That is, we assess the difference between the current fitness, fitness(L', M), and an approximation of the fitness when incorporating ξ, i.e., fitness$(\{\xi\} \cup L', M)$. This approximation, denoted by $\widehat{fit}(\xi, \xi_r, L', M)$, is derived from (i) the change in fitness induced by the reference trace ξ_r, and (ii) the differences between ξ and ξ_r. The former is assessed using the aggregated alignment cost $c(\xi_r, M)$, whereas the latter leverages the aggregated cost of aligning the traces, $c_t(\xi, \xi_r)$. Normalising these costs, function $approx_{fit}(\xi, \xi_r, L', M)$ yields a worst-case approximation for the change in overall fitness imposed by ξ, as follows:

$$approx_{fit}(\xi, \xi_r, L', M) = \left\{ \left| \text{fitness}(L', M) - \widehat{fit}(\xi, \xi_r, L', M) \right| \right\} \tag{5}$$

$$\widehat{fit}(\xi, \xi_r, L', M) = 1 - \frac{\sum\limits_{\xi' \in L'} c(\xi', M) + c(\xi_r, M) + c_t(\xi, \xi_r)}{\sum\limits_{\xi' \in L'} c(\xi', \emptyset) + \max\limits_{\xi' \in \{\xi, \xi_r\}} c(\xi', \emptyset) + (|L'| + 1) \min\limits_{x \in M} c(\langle\rangle, \{x\})}$$

Turning to our running example, assume that we have sampled $\{\xi_1, \xi_2\}$ and computed the precise fitness value based on both traces, which is $1 - 2/(12 + 10) \approx 0.909$ using a standard cost function. If trace ξ_4 is sampled next, we approximate its impact using the similar trace ξ_1. To this end, we consider $c(\xi_1, M) = 1$ and $c_t(\xi_1, \xi_4) = 1$, which yields an approximated fitness value of $1 - (2 + 1 + 1)/(12 + 6 + 15) = 1 - 4/33 \approx 0.879$. This is close to the actual fitness value for $\{\xi_1, \xi_2, \xi_4\}$, which is $1 - 4/(17 + 15) \approx 0.875$. The minor difference stems from ξ_1 being slightly longer than ξ_4.

Deviation distribution approximation. To approximate the impact of trace ξ on the deviation distribution, we follow a similar approach as for fitness approximation. However, we note that the approximation function here yields a set of possible values, as there are multiple different distributions to be considered when measuring the Manhattan distance to the current distribution. The reason being that the difference between ξ and the reference trace ξ_r induces a set of possible changes of the distribution.

Specifically, we denote by $ed(\xi, \xi_r)$ the edit distance of the two traces, i.e., the pure number of skip steps in their alignment. This number gives an upper bound for the number of conformance issues that need to be incorporated in addition to those stemming from the alignment of the reference trace and the model, i.e., $dev(\{\xi_r\}, M)$. Yet, the exact activities are not known, so that we need to consider all bags of activities of size $ed(\xi, \xi_r)$, the set of which is denoted by $[\mathcal{A}]^{ed(\xi, \xi_r)}$. Each of these bags leads to a different approximation $\widehat{f_{dev}}(\beta, \xi_r, L', M)$ of the distribution $f_{dev(\{\xi\} \cup L', M)}$ that we are actually interested in. We compute those as follows:

$$approx_{dev}(\xi, \xi_r, L', M) = \bigcup_{\beta \in [\mathcal{A}]^{ed(\xi,\xi_r)}} \left\{ \sum_{a \in \mathcal{A}} \left| f_{dev(L',M)}(a) - \widehat{f_{dev}}(\beta, \xi_r, L', M)(a) \right| \right\} \tag{6}$$

$$\widehat{f_{dev}}(\beta, \xi_r, L', M)(a) = \frac{dev(L', M)(a) + dev(\{\xi_r\}, M)(a) + \beta(a)}{|dev(L', M)| + |dev(\{\xi_r\}, M)| + |\beta|}$$

Consider our example again: Based on $\{\xi_1, \xi_2\}$, we determine that $dev(\{\xi_1, \xi_2\}, M) = [F^2]$ and $f_{dev(\{\xi_1,\xi_2\},M)}(F) = 1$. If ξ_4 is then sampled, we obtain an approximation based on $dev(\{\xi_1\}, M) = [F]$ and $ed(\xi_4, \xi_1) = 1$. We therefore consider the change in the distribution incurred by approximating the deviations of ξ_4 as $[F] \uplus \beta$ with $\beta \in \{[R], [P], [F], [U], [S]\}$. For instance, $\widehat{f_{dev}}([R], \xi_4, \{\xi_1, \xi_2\}, M)$ yields a distribution assigning relative frequencies of $3/4$ and $1/4$ to activities F and R, respectively.

The above approximation may be tuned heuristically by narrowing the set of activities that are considered for β, i.e., the possible deviations incurred by the difference between ξ and ξ_r. While this means that $\widehat{f_{dev}}$ is no longer a worst-case approximation, it may steer the approximation in practice, hinting at which activities shall be considered for possible deviations. Such an approach is also beneficial for performance reasons: Since β may be any bag built of the respective activities, the exponential blow-up limits the applicability of the approximation to traces that are rather similar, i.e., for which $ed(\xi, \xi_r)$ is small.

Here, we describe one specific heuristic. First, we determine the overlap between ξ and ξ_r in terms of their maximal shared prefix and suffix of activities, for which the execution is signalled by their events. Next, we determine the events that are *not* part of the shared prefix and suffix, and derive the activities referenced by these events. Only these activities are then considered for the construction of β.

In our example, traces ξ_1 and ξ_4 share the prefix $\langle r, p, f, f \rangle$ and suffix $\langle s \rangle$. Thus, ξ_1 contains one event between the shared prefix and suffix, u, while there is none for ξ_4. Hence, we consider a single bag of deviations, $\beta = [U]$, and $\widehat{f_{dev}}([U], \xi_4, \{\xi_1, \xi_2\}, M)$ is the only distribution considered in the approximation.

5 Evaluation

This section reports on an experimental evaluation of the proposed techniques for sample-based and approximation-based conformance checking. Section 5.1 describes the three real-world and seven synthetic event logs used in the experimental setup, described in Section 5.2. The evaluation results demonstrate that our techniques achieve considerable efficiency gains, while still providing highly accurate conformance results (Section 5.3).

5.1 Datasets

We conducted our experiments based on three real-world and seven synthetic event logs, which are all publicly available.

Real-world data. The three real-world event logs differ considerably in terms of the number of unique traces they contain, as well as their average trace lengths, which represent key characteristics for our approach.

– *BPI-12* [31] is a log of a process for loan or overdraft applications at a Dutch financial institute that was part of the Business Process Intelligence (BPI) Challenge. The log contains 13,087 traces (4,366 variants), with 20.0 events per case (avg.).
– *BPI-14* [32] is the log of an ICT incident management process used in the BPI Challenge. For the experiments, we employed the event log of incidence activities, containing 46,616 traces (31,725 variants), with 7.3 events per case (avg.).
– *Traffic Fines* [11] is a log of an information system managing road traffic fines. The log contains 150,370 traces (231 variants), with 3.7 events per case (avg.).

We obtained accompanying process models for these logs using the inductive miner infrequent [19] with its default parameter settings (i.e., 20% noise filtering).

Table 1: Characteristics of the synthetic model-log pairs

Characteristic	**PrA**	**PrB**	**PrC**	**PrD**	**PrE**	**PrF**	**PrG**
Activities	363	317	317	429	275	299	335
Traces	1,200	1,200	500	1,200	1,200	1,200	1,200
Variants	1,049	1,126	500	1,200	1,200	1,200	1,200
Events per trace (avg.)	31.6	41.5	42.9	248.6	98.8	240.8	143.1

Synthetic data. To analyse the scalability of our techniques, we considered a synthetic dataset designed to stress-test conformance checking techniques [24]. It consists of seven process models and accompanying event logs. The models are considerably large and complex, as characterised in Table 1, which impacts the computation of alignments. Furthermore, the included event logs consist of a high number of variants (compared to the number of traces), which may affect the effectiveness of log sampling.

5.2 Experimental Setup

We employed the following measures and experimental setup to conduct the evaluation.

Measures. We measure the *efficiency* of our techniques by the fraction of traces from a log required to obtain our conformance results. This fraction indicates for how many traces the conformance computation was not needed due to the trace not being sampled, or the result being approximated. Simultaneously, we consider the fraction of the total trace variants for which our techniques actually had to establish alignments. As these fractions provide us with analytical measures of efficiency, we also assess the *runtime* of our techniques, based on a prototypical implementation. Again, this is compared to the runtime of the conformance checking over the complete log. Finally, we assess the impact of sampling and approximation on the *accuracy* of conformance results. We determine the accuracy by comparing the results, i.e., the fitness value or the deviation distribution, obtained using sampling and approximation, to the results for the total log.

All presented results are determined based on 20 experimental runs (i.e., replications) of which we report on the mean value, along with the 10th and 90th percentiles.

Environment. Our approach has been implemented as a plugin in ProM [33], which is publicly available.[1] For the computation of alignments, we rely on the ProM implementation of the search-based technique recently proposed in [13]. Runtime measurements have been obtained on a PC (Dual-Core, 2.5GHz, 8GB RAM) running Oracle Java 1.8.

[1] https://github.com/Martin-Bauer/Conformance_Sketching

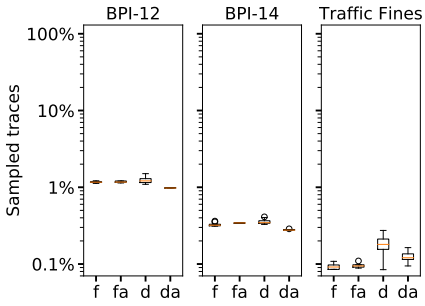


Fig. 2: Sample size efficiency

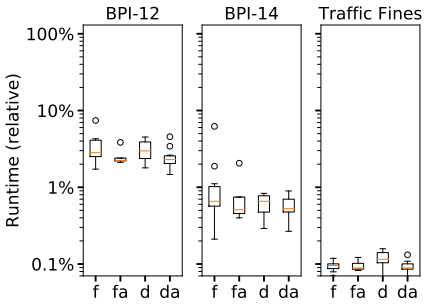


Fig. 3: Runtime efficiency

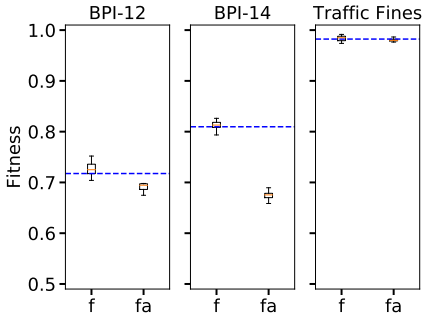


Fig. 4: Accuracy (fitness)

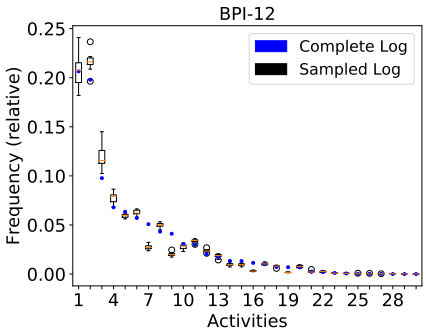


Fig. 5: Accuracy (deviation distribution)

5.3 Evaluation Results

This section first considers the overall efficiency and accuracy of our approach on the real-world event logs, using default parameter values ($\delta = 0.01$, $\alpha = 0.99$, $\epsilon = 0.01$, and $k = 1/3$), before conducting a sensitivity analysis in which these values are varied. Lastly, we demonstrate the scalability of our approaches by showing that their performance also applies to complex, synthetic datasets.

Efficiency. We first explore the efficiency of our approach in terms of sample size and runtime for four configurations: conformance in terms of fitness, without (f) and with approximation (fa), as well as for the deviation distribution without (d) and with approximation (da). Fig. 2 reveals that all configurations only need to consider a tiny fraction of the complete log. For instance, for BPI-14, the sample-based fitness computation (f) requires only 685 traces (on average) out of the total of 46,616 traces (i.e., 1.5% of the log). This sample included traces from 144 out of the total 31,725 variants (less than 0.5%), which means that the approach established just 144 alignments. As expected, these gains are propagated to the runtime of our approach, as shown in Fig. 3.

When looking at the overall efficiency results, we observe that the additional use of approximation generally does not lead to considerable improvements in comparison to just sampling. However, this is notably different for the deviation distribution of the Traffic Fines dataset. Here, without approximation, the sample size is 1323 on average (0.88%), whereas the sample size drops to 713 traces (0.47%) with approximation (da). Hence, approximation appears to be more important if sampling alone is not as effective.

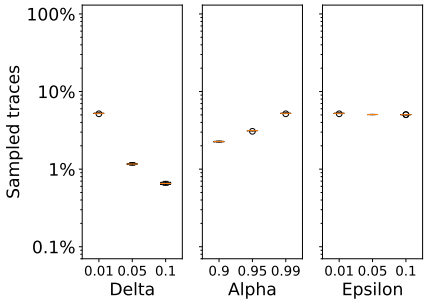


Fig. 6: Sample size sensitivity

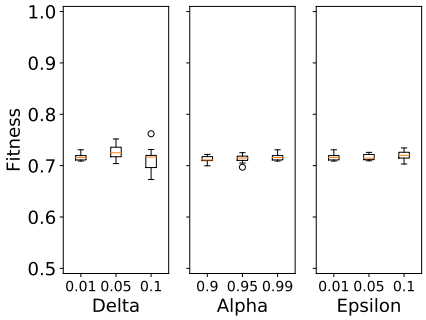


Fig. 7: Fitness sensitivity

Accuracy. The drastic gains in efficiency are obtained while maintaining highly accurate conformance results. According to Fig. 4, the fitness computed using sample-based conformance checking differs by less than 0.1% from the original fitness (indicated by the dashed, blue line). Since the accuracy in terms of deviation distribution is harder to capture in a single value, we use Fig. 5 to demonstrate that the deviation distribution obtained by our sample-based technique closely follows the distribution for the complete log. In decreasing order, Fig. 5 depicts the activities with their numbers of deviations observed in the complete log and in the sampled log. As shown, our technique clearly identifies which activities are most often affected by deviations, i.e., our technique correctly identifies the main hotspots of non-conformance. Although, for clarity, not depicted, our approach including approximation achieved comparable results.

Parameter sensitivity. We performed a parameter sensitivity analysis using sample-based conformance checking on the BPI-12 dataset. We explored how parameters δ (probability bound), α (significance value), and ϵ (relaxedness value), affect the performance of our approach in terms of efficiency (number of traces) and accuracy (fitness).[2] Fig. 6 shows that selection of δ and α have a considerable impact on the sample required for conformance checking. For instance, for $\delta = 0.01$ we require an average sample size of 684.9, whereas when we relax the bound to $\delta = 0.10$, the average size is reduced to 85.8 traces. For α, i.e., the confidence, we observe a range from 296.5 to 684.9 traces. By contrast, varying epsilon is shown to result in only marginal differences (ranging between 659.0 and 684.9). Still, for all these results, it should be considered that even the largest sample sizes represent only 5% of the traces in the original log.

Notably, as shown in Fig. 7, the average accuracy of our approach remains highly stable throughout this sensitivity analysis, ranging from 0.711 to 0.726. However, we do observe that the variance across replications differs for the parameter settings using smaller sample sizes, specifically for $\delta = 0.1$. Here, the obtained fitness values range between 0.67 and 0.76. This indicates that for such sample sizes, the selection of the particular sample may impact the obtained conformance result in some replications.

Scalability. The results obtained for the synthetic datasets confirm that our approaches are able to provide highly accurate conformance checking results in a small fraction of the runtime. Here, we reflect on experiments performed using our fitness-based sampling approach with $\delta = 0.05$, $\alpha = 0.99$, and $\epsilon = 0.01$. Fig. 8 shows that, for six out of

[2] While keeping the other parameter values stable at their respective defaults.

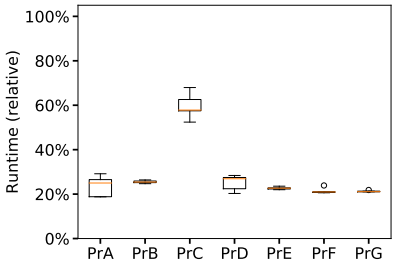


Fig. 8: Runtime efficiency, sample-based fitness computation (synthetic data)

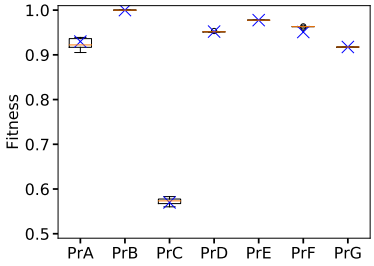


Fig. 9: Fitness (synthetic data). Blue crosses denote fitness of the total log.

seven cases, runtime is reduced to 21.2% to 25.5% of the time needed for the total log (sample sizes range from 10.7% to 12.2%). At the same time, for all cases, the obtained fitness results are virtually equivalent to those of the total log, see Fig. 9, where the fitness values of the total logs are given by blue crosses. When comparing these results to those of the real-world datasets, it should be noted that the synthetic logs hardly have any re-occurring trace variants, which makes it harder to generalize over the sampled results. This is particularly pronounced for process PrC: There is virtually no difference between the fitness obtained for the total log and the samples. Yet, the relatively low fitness value of 0.57 along with a comparatively small number of traces in the log (500 vs. 1,200) lead to a runtime of 59% with sampling. Still, overall, the results on the synthetic data demonstrate that our approach is beneficial in highly complex scenarios.

6 Related Work

Conformance checking can be grounded in various notions. Non-conformance may be detected based on a comparison of sets of binary relations defined over events of a log and activities of a model, respectively [35]. Other work suggests to 'replay' traces in a process model, thereby identifying whether events denote valid activity executions [27]. However, both of these streams have limitations with respect to completeness [9]. Therefore, alignment-based conformance checking techniques [4, 23], on which we focus in this work, are widely recognized as the state-of-the-art.

Acknowledging the complexity associated with the establishment of alignments, various approaches [13, 16, 25, 28], discussed in Section 1, have been developed to improve runtime efficiency. Other work also aims to achieve efficiency gains by approximating alignments, cf., [14, 30]. While these approaches can lead to efficiency gains, all of them fundamentally depend on the consideration of an entire event log. Moreover, our angle for approximation is different: We do not approximate alignments, but estimate a conformance result (fitness, deviation distribution) based on the distance between traces.

The sampling technique that we employ to avoid this is based on sampling used in sequence databases, i.e., datasets that contain traces. Sampling techniques for event logs have been previously applied for specification mining [10], for mining of Markov Chains [6], and for process discovery [5, 8]. However, we are the first to apply these sampling techniques to conformance checking, a use case in which computational efficiency is arguably even more important than in discovery scenarios.

7 Conclusion

In this paper, we argued that insights into the overall conformance of an event log with respect to a process model can be obtained without computing conformance results for *all* traces. Specifically, we presented two angles to achieve efficient conformance checking: First, through trace sampling, we achieve that only a small share of the traces of a log are considered in the first place. By phrasing this sampling as a series of random experiments, we are able to give guarantees on the introduced error in terms of a potential difference of the overall conformance result. Second, we introduced result approximation as a means to avoid the computation of conformance results even for some of the sampled traces. Exploiting similarities of two traces, we derive an upper bound for the conformance of one trace based on the conformance of another trace. Both techniques, trace sampling and result approximation, have been instantiated for two notions of conformance results, fitness as a numerical measure of overall conformance and the deviation distribution that highlights hotspots of non-conformance in terms of individual activities.

Our experiments highlight dramatic improvements in terms of conformance checking efficiency: Only 0.1% to 1% of the traces of real-world event logs (12% for synthetic data) need to be considered, which leads to corresponding speed-ups of the observed runtimes. At the same time, the obtained conformance results, whether defined as fitness or the deviation distribution, are virtually equivalent to those obtained for the total log.

In future work, we intend to lift our ideas to conformance checking that incorporates branching conditions in a process model or temporal deadlines. Moreover, we strive for an integration of divide-and-conquer schemes [18] in our approximation approach.

References

1. Van der Aalst, W.M.P.: Process Mining - Discovery, Conformance and Enhancement of Business Processes. Springer (2011)
2. Van der Aalst, W.M.P.: Data Scientist: The Engineer of the Future, pp. 13–26. Springer International Publishing, Cham (2014)
3. der Aalst, W.M.P.V., Verbeek, H.M.W.: Process discovery and conformance checking using passages. Fundam. Inform. 131(1), 103–138 (2014)
4. Adriansyah, A., van Dongen, B., Van der Aalst, W.M.P.: Conformance checking using cost-based fitness analysis. In: EDOC. pp. 55–64 (2011)
5. Bauer, M., Senderovich, A., Gal, A., Grunske, L., Weidlich, M.: How much event data is enough? A statistical framework for process discovery. In: CAiSE. pp. 239–256 (2018)
6. Biermann, A.W., Feldman, J.A.: On the synthesis of finite-state machines from samples of their behavior. IEEE Trans. Computers 100(6), 592–597 (1972)
7. Busany, N., Maoz, S.: Behavioral log analysis with statistical guarantees. In: ICSE. pp. 877–887. ACM (2016)
8. Carmona, J., Cortadella, J.: Process mining meets abstract interpretation. In: ECML/PKDD (1). Lecture Notes in Computer Science, vol. 6321, pp. 184–199. Springer (2010)
9. Carmona, J., van Dongen, B., Solti, A., Weidlich, M.: Conformance Checking – Relating Processes and Models. Springer (2018)
10. Cohen, H., Maoz, S.: Have we seen enough traces? In: ASE. pp. 93–103. IEEE (2015)
11. De Leoni, M.M., Mannhardt, F.F.: Road traffic fine management process (2015), https://doi.org/10.4121/uuid:270fd440-1057-4fb9-89a9-b699b47990f5

12. Dixit, P.M., Buijs, J.C.A.M., Verbeek, H.M.W., der Aalst, W.M.P.V.: Fast incremental conformance analysis for interactive process discovery. In: BIS. pp. 163–175. Springer (2018)
13. van Dongen, B.F.: Efficiently computing alignments - using the extended marking equation. In: Business Process Management. pp. 197–214 (2018)
14. van Dongen, B.F., Carmona, J., Chatain, T., Taymouri, F.: Aligning modeled and observed behavior: A compromise between computation complexity and quality. In: CAISE. vol. 10253, pp. 94–109 (2017)
15. Dumas, M., Rosa, M.L., Mendling, J., Reijers, H.A.: Fundamentals of Business Process Management, Second Edition. Springer (2018)
16. Evermann, J.: Scalable process discovery using map-reduce. IEEE TSC 9(3), 469–481 (2016)
17. van Hee, K.M., Liu, Z., Sidorova, N.: Is my event log complete? - A probabilistic approach to process mining. In: Int. Conf. on Research Challenges in Information Science. pp. 1–7 (2011)
18. Lee, W.L.J., Verbeek, H.M.W., Munoz-Gama, J., der Aalst, W.M.P.V., Sepúlveda, M.: Recomposing conformance: Closing the circle on decomposed alignment-based conformance checking in process mining. Inf. Sci. 466, 55–91 (2018)
19. Leemans, S.J.J., Fahland, D., der Aalst, W.M.P.V.: Discovering block-structured process models from event logs containing infrequent behaviour. In: BPM Workshops. LNBIP, vol. 171, pp. 66–78. Springer (2013)
20. Leemans, S.J.J., Fahland, D., der Aalst, W.M.P.V.: Scalable process discovery and conformance checking. Software and System Modeling (2), 599–631 (2018)
21. de Leoni, M., Marrella, A.: How planning techniques can help process mining: The conformance-checking case. In: Italian Symp. on Advanced Database Syst. p. 283 (2017)
22. Luo, C., He, F., Ghezzi, C.: Inferring software behavioral models with MapReduce. Sci. Comput. Program. 145, 13–36 (2017)
23. Munoz-Gama, J., Carmona, J., Aalst, W.v.d.: Single-entry single-exit decomposed conformance checking. Inf. Syst. 46, 102–122 (2014)
24. Munoz-Gama, J. (Jorge): Conformance checking in the large (dataset) (2013), https://data.4tu.nl/repository/uuid:44c32783-15d0-4dbd-af8a-78b97be3de49
25. Reißner, D., Conforti, R., Dumas, M., Rosa, M.L., Armas-Cervantes, A.: Scalable conformance checking of business processes. In: OTM to Meaningful Int. Syst. pp. 607–627 (2017)
26. Rogge-Solti, A., Senderovich, A., Weidlich, M., Mendling, J., Gal, A.: In log and model we trust? A generalized conformance checking framework. In: BPM. pp. 179–196 (2016)
27. Rozinat, A., Van der Aalst, W.M.P.: Conformance checking of processes based on monitoring real behavior. Information Systems 33(1), 64–95 (2008)
28. Taymouri, F., Carmona, J.: Model and event log reductions to boost the computation of alignments. In: SIMPDA. pp. 1–21 (2016)
29. Taymouri, F., Carmona, J.: A recursive paradigm for aligning observed behavior of large structured process models. In: Business Process Management. pp. 197–214. Springer (2016)
30. Taymouri, F., Carmona, J.: An evolutionary technique to approximate multiple optimal alignments. In: BPM. pp. 215–232 (2018)
31. Van Dongen, B.: BPI Challenge 2012 (2012), https://doi.org/10.4121/uuid:3926db30-f712-4394-aebc-75976070e91f
32. Van Dongen, B.: BPI Challenge 2014 (2014), https://doi.org/10.4121/uuid:c3e5d162-0cfd-4bb0-bd82-af5268819c35
33. Verbeek, E., Buijs, J.C.A.M., van Dongen, B.F., der Aalst, W.M.P.V.: Prom 6: The process mining toolkit. In: Business Process Management (Demonstration Track) (2010)
34. Weber, B., Reichert, M., Rinderle-Ma, S.: Change patterns and change support features - enhancing flexibility in process-aware information systems. DKE 66(3), 438–466 (2008)
35. Weidlich, M., Polyvyanyy, A., Desai, N., Mendling, J., Weske, M.: Process compliance analysis based on behavioural profiles. Inf. Syst. 36(7), 1009–1025 (2011)