Stochastic Alignments: Matching an Observed Trace to Stochastic Process Models

Tian Li[1,2], Artem Polyvyanyy[2], and Sander J.J. Leemans[1,3]

[1] RWTH Aachen University, Germany
t.li,s.leemans@bpm.rwth-aachen.de
[2] The University of Melbourne, Australia
artem.polyvyanyy@unimelb.edu.au
[3] Fraunhofer, Germany

Abstract. Process mining leverages event data extracted from IT systems to generate insights into the business processes of organizations. Such insights benefit from explicitly considering the frequency of behavior in business processes, which is captured by stochastic process models. Given an observed trace and a stochastic process model, conventional alignment-based conformance checking techniques face a fundamental limitation: They prioritize matching the trace to a model path with minimal deviations, which may, however, lead to selecting an unlikely path. In this paper, we study the problem of matching an observed trace to a stochastic process model by identifying a likely model path with a low edit distance to the trace. We phrase this as an optimization problem and develop a heuristic-guided path-finding algorithm to solve it. Our open-source implementation demonstrates the feasibility of the approach and shows that it can provide new, useful diagnostic insights for analysts.

Keywords: Process mining · Stochastic process mining · Stochastic alignment · Conformance checking · Stochastic conformance checking

1 Introduction

A process is a series of coordinated steps. In modern organizations, information systems record processes executed by employees, managers, and customers as event data. These data can be extracted into event logs comprising sequences of executed events, called *traces*, triggered by different historical process instances. Organizations typically establish detailed guidelines for process execution to help achieve their objectives [16]. These guidelines are often represented as *process models* that reflect standards or regulations [1] for executing the processes.

Process mining is a data analysis technique that aims to deliver meaningful insights from event data. In particular, techniques that explicitly consider the frequency or likelihood of process behavior are categorized as stochastic process mining approaches. For instance, to avoid the risk that effort is spent on rare behavior and leads to misinformed decisions, it is vital to distinguish between frequent and rare process behavior [23]. Stochastic process mining techniques achieve this by treating an event log as a finite sample drawn from a probability distribution over traces, while modeling stochastic processes as probability distributions over traces [4].

Conformance checking techniques such as alignments enable analysts to find commonalities and discrepancies between the modeled behavior and the observed behavior [11]. Conventional alignments-based techniques [2,15] identify a path allowed by the model with as few deviations as possible from an observed trace. However, when considering a stochastic perspective, if the selected path is unlikely according to the stochastic process model, it may not be the most likely explanation of the path through the model. As a consequence, further diagnostics are based on process behavior that is less relevant to be followed by design.

For instance, a path with a probability of 10% according to the model and an edit distance of 3 to the trace may be a better match than a path with a probability of 0.1% and an edit distance of 2. The scenario highlights two possibly competing objectives when matching the trace to the stochastic model of the process: the probability of the selected path allowed by the model and its edit distance to the trace.

In this paper, we propose a *stochastic alignment* technique, which matches a single trace to a stochastic process model and produces an alignment that balances the importance of the normative behavior (the probability of the model path) and alignment cost (the edit distance between model path and trace). Business analysts can explicitly weigh the trade-off with a user-defined parameter. For a specific type of stochastic model, stochastic labeled Petri net, we formulate the search for the model path as an optimization problem solved by the a-star algorithm [18]. Our technique has been implemented and is publicly available. The experiments demonstrate that the technique is feasible for real-life event data. Moreover, the case study shows that it can generate new and useful insights when explaining deviations in observed traces.

The remainder of the paper proceeds as follows. We first introduce preliminaries in Section 2. In Section 3, we present our technique and then evaluate it in Section 4. Then, we discuss related work in Section 5. Finally, Section 6 draws conclusions and sketches future work.

2 Preliminaries

In this section, we present concepts used in the subsequent sections.

Given a set of elements E, a multiset $X : E \to \mathbb{N}$ maps the elements of E to the natural numbers, such that X allows for multiple instances for each element. Multisets are shown within [ ] with superscript frequencies. For example, $X = [b^4, c^5]$ is a multiset with four b's and five c's. Multiset union $X_u = X_1 \uplus X_2$ denotes that $\forall_{e \in E}\ X_u(e) = \max(X_2(e), X_1(e))$. Multiset subset $X_1 \mathrel{\underline{\Subset}} X_2$ denotes $\forall_{e \in E} X_2(e) \geq X_1(e)$. If $X_1 \mathrel{\underline{\Subset}} X_2$, then $X_3 = X_2 \setminus X_1$ is the multiset difference, such that $\forall_{e \in E} X_3(e) = X_2(e) - X_1(e)$. The set of all multisets on E is denoted by $\mathcal{B}(E)$. Given an alphabet $\Sigma = \{a_1, \ldots, a_n\}$, the Parikh vector of a sequence

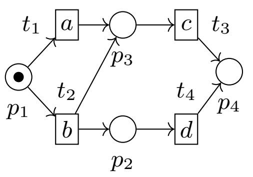


Fig. 1: Labeled Petri net N_1.

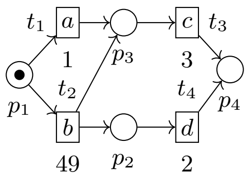


Fig. 2: Stochastic labeled Petri net N_2.

over Σ is a function $\vec{\cdot}\colon \Sigma^* \to \mathbb{N}^{|\Sigma|}$ that results in a column vector such that $\vec{\sigma} = [|\sigma|_{a_1}, \ldots, |\sigma|_{a_n}]$, where $|\sigma|_{a_i}$ denotes the number of a_i in σ.

An *event log* is a collection of *traces*. Each trace is a finite sequence of *activities*. An activity is a description of the event that is occurring. The control-flow perspective of processes can be captured using labeled Petri nets.

Definition 1 (Labeled Petri Nets). *A* labeled Petri net *(LPN) is a tuple (P, T, F, A, ρ, M_0) where P is a finite set of places, T is a finite set of transitions such that $P \cap T = \emptyset$, $F \Subset \mathcal{B}((P \times T) \cup (T \times P))$ is a flow relation, A is a finite set of activities, $\rho\colon T \to A \cup \{\tau\}$ with $\tau \notin A$ is a labeling function, and $M_0 \in \mathcal{B}(P)$ is the initial marking.*

We adopt the dot notation to refer to components in tuples. For example, given an LPN N, its transitions are indicated by $N.T$. A transition t with $\rho(t) = \tau$ is *silent*. ${}^\bullet t = \{p \in P \mid (p,t) \Subset F\}$ and $t^\bullet = \{p \in P \mid (t,p) \Subset F\}$ denote the *preset* and *post-set* of t. A state in LPN is called *marking M* that marks certain places (represented by circles) with tokens (represented by black dots), such that $M \in \mathcal{B}(P)$. By T_M, we denote the set of all transitions enabled at marking M, that is, $T_M = \{t \in T \mid {}^\bullet t \Subset M\}$. A *deadlock* marking does not enable any transition. Firing an enabled transition $t \in T_M$ results in a new marking $M' = (M \setminus {}^\bullet t) \uplus t^\bullet$. A *model path* is a sequence of transitions $\eta = \langle t_1, \ldots, t_n \rangle$ that brings the LPN from its M_0 to a deadlock marking, that is, there is a sequence of markings $\langle M_0, \ldots, M_n \rangle$ for which it holds that $\forall_{1 \le i \le n} {}^\bullet t_i \Subset M_{i-1} \wedge M_i = (M_{i-1} \setminus {}^\bullet t_i) \uplus t_i^\bullet$ and M_n is a deadlock marking. The projection of a model path by ρ on the labels of its non-silent transitions is a *trace*.

For instance, Fig. 1 illustrates an LPN N_1. The initial marking of N_1 is $[p_1^1]$, and there are two reachable deadlock markings, i.e., $[p_4^1]$ and $[p_4^2]$. $\langle t_1, t_3 \rangle$ is a model path that brings N_1 from $[p_1^1]$ to $[p_4^1]$, and it can be projected to trace $\langle a, c \rangle$. $\langle t_2, t_3, t_4 \rangle$ is another model path that brings N_1 from $[p_1^1]$ to $[p_4^2]$, and it can be projected to trace $\langle b, c, d \rangle$.

The trace net of a trace is an LPN without choices, such that each place and transition has at most one incoming and at most one outgoing arc. In Eq. (1), *consumption matrix $\mathcal{C}_N$* of $|N.P|$ rows and $|N.T|$ columns specifies the necessary tokens for firing transitions in LPN N. In Eq. (2), *incidence matrix $\mathcal{I}_N$* of $|N.P|$ rows and $|N.T|$ columns reflects the flow relation in N.

$$\mathcal{C}_N(p,t) = \begin{cases} -1 & \textit{if (p,t)} \in N.F \textit{ and (t,p)} \notin N.F\textit{, and} \\ 0 & \textit{otherwise.} \end{cases} \tag{1}$$

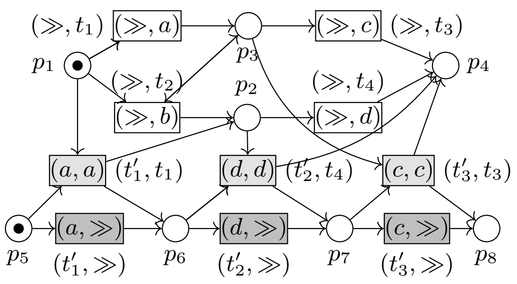


Fig. 3: The synchronous product net of $\langle a, d, c\rangle$ and N_2.

γ_1:

σ	a	$\gg$	d	c	$\gg$
η_1	$\gg$	t_2	$\gg$	t_3	t_4

γ_2:

σ	$\gg$	a	d	c
η_2	t_2	$\gg$	t_4	t_3

γ_3:

σ	a	d	c
η_3	t_1	$\gg$	t_3

Fig. 4: Example alignments for $\langle a, d, c\rangle$ and N_2.

$$\mathcal{I}_N(p,t) = \begin{cases} 1 & \textit{if } (t,p) \in N.F \textit{ and } (p,t) \notin N.F, \textit{ and} \\ C_N(p,t) & \textit{otherwise.} \end{cases} \tag{2}$$

A variant of LPNs with stochastic information is stochastic labeled Petri nets.

Definition 2 (Stochastic Labeled Petri Nets). *A* stochastic labeled Petri net *(SLPN) is a tuple $(P, T, F, A, \rho, M_0, w)$, where (P, T, F, A, ρ, M_0) is an LPN, and $w\colon T \to \mathbb{R}^+$ is a weight function.*

Given an SLPN, a transition t enabled at marking M can fire with probability $\mathrm{p}(t \mid M) = {}^{w(t)}/_{\Sigma_{t' \in T_M} w(t')}$. The probability $\mathrm{p}(\eta)$ of observing path η is equal to $\prod_{1 \le i \le n} \mathrm{p}(t_i \mid M_{i-1})$. For example, Fig. 2 depicts an SLPN N_2, in which transitions t_1 and t_2 are enabled at initial marking M_0. Transition t_2 has a weight of 49, and a firing probability of ${}^{49}/_{(1+49)} = {}^{49}/_{50}$ at M_0.

Using SLPN and the trace net, we define the synchronous product model as their combination with an additional set of synchronous transitions, while the transitions of the original SLPN and the trace net are represented by pairing them with the new symbol $\gg$.

Definition 3 (Synchronous Product Nets). *Let $N^\sigma = (P^\sigma, T^\sigma, F^\sigma, A^\sigma, \rho^\sigma, M_0^\sigma)$ be the trace net of trace σ, and $N^s = (P^s, T^s, F^s, A^s, \rho^s, M_0, w^s)$ be an SLPN. Their synchronous product net $N^\otimes = (P^\otimes, T^\otimes, F^\otimes, A^\otimes, \rho^\otimes, M_0^\otimes)$ is a tuple with:*

- *$P^\otimes = P^s \cup P^\sigma$ is the set of places,*
- *$T^\otimes = \{(t^\sigma, t^s) \in (T^\sigma \cup \{\gg\}) \times (T^s \cup \{\gg\}) \mid t^\sigma \neq \gg \vee t^s \neq \gg \vee \rho^\sigma(t^\sigma) = \rho^s(t^s)\}$ is the set of original transitions and the synchronous transitions,*
- *$F^\otimes = \{((t^\sigma, t^s), p) \in T^\otimes \times P^\otimes \mid (t^s, p^s) \in F^s \vee (t^\sigma, p^\sigma) \in F^\sigma\} \cup \{(p, (t^\sigma, t^s)) \in P^\otimes \times T^\otimes \mid (p^s, t^s) \in F^s \vee (p^\sigma, t^\sigma) \in F^\sigma\}$ is a flow relation,*
- *$A^\otimes = A^\sigma \cup A^s$,*
- *$\rho^\otimes\colon T^\otimes \to A^\sigma \cup A^s \cup \{\tau\}$ is a labeling function,*
- *$M_0^\otimes = M_0^s \uplus M_0^\sigma$ is the initial marking.*

For example, Fig. 3 illustrates a synchronous product net for N_2 and a trace $\langle a, d, c\rangle$. We adopt a similar notation as [2], which categorize each transition $t \in N^\otimes.T$ as: 1) a non-silent model move $(\gg, t^s)$ with $\rho^s(t^s) \neq \tau$; 2) a silent

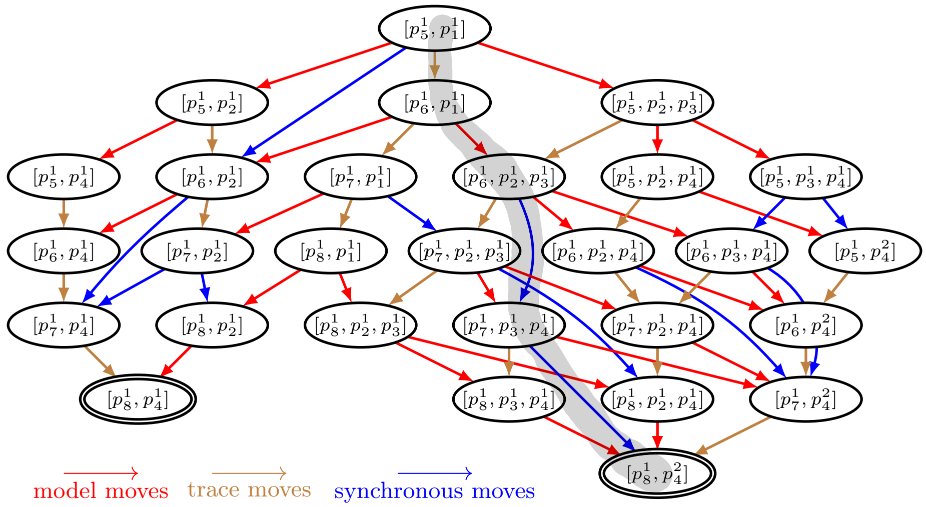


Fig. 5: The state space of the synchronous product net in Fig. 3.

model move $(\gg, t^s)$ with $\rho^s(t^s) = \tau$; 3) a trace move $(t^\sigma, \gg)$ with $\rho^\sigma(t^\sigma) \neq \gg$; 4) a synchronous move (t^σ, t^s) with $\rho^s(t^s) = \rho^\sigma(t^\sigma) \wedge \rho^s(t^s) \neq \gg \wedge \rho^\sigma(t^\sigma) \neq \gg$.

Given a trace and an SLPN, an alignment is a sequence of moves in their synchronous product net.

Definition 4 (Alignments). *Let N^σ be a trace net of σ, N^s be an SLPN, and $N^\otimes$ be their synchronous product net. A model path γ from the initial marking to a deadlock marking in $N^\otimes$ is an alignment between σ and N^s.*

Fig. 4 illustrates three alignments, γ_1, γ_2 and γ_3. By stripping the $\gg$ symbol, the upper row can be projected to the trace, while the lower row can be projected to a model path in the SLPN. Typically, transitions of type synchronous moves and silent model moves in the synchronous product net are assigned a cost of zero [2]. For other types of transition, we assign a cost of one since they represent deletion or insertion. Thus, γ_1, γ_2 and γ_3 are alignments with a cost of 4, 2, and 1, respectively. The total move cost is comparable to the Levenshtein Edit Distance [18] between the corresponding model path η in SLPN and the trace σ, denoted as $d(\sigma, \eta)$[4].

Fig. 5 presents the state space of the synchronous product net in Fig. 3. The highlighted gray path in Fig. 5 corresponds to alignment γ_2, which consists of one trace move $(t'_1, \gg)$, one model move $(\gg, t_2)$, and two synchronous moves (t'_2, t_4) and (t'_3, t_3).

Finally, to establish a direct connection between transitions in the original SLPN and the model moves and synchronous moves in the synchronous product net, we introduce two reverse functions as follows.

[4] In this paper, we use edit distance and move cost interchangeably.

Definition 5 (Reverse Functions). *Let $N^\sigma = (P^\sigma, T^\sigma, F^\sigma, A^\sigma, \rho^\sigma, M_0^\sigma)$ be the trace net of trace σ, and $N^s = (P^s, T^s, F^s, A^m, \rho^s, M_0^s, w^s)$ be an SLPN, and $N^\otimes = (P^\otimes, T^\otimes, F^\otimes, A^\otimes, \rho^\otimes, M_0^\otimes)$ be their synchronous product net. $r_m \colon \mathcal{B}(P^\otimes) \to \mathcal{B}(P^s)$ is a function that removes the multiset of places of trace net from marking in $N^\otimes$, and $r_t \colon T^\otimes \backslash \{\{\gg\} \cup T^\sigma\} \to T^s$ is a function that maps the model moves and synchronous moves back to transitions in N^s.*

For instance, the initial marking of the synchronous product net $[p_1^1, p_5^1]$ can be mapped to the initial marking $[p_1^1]$ in SLPN using r_m. The synchronous move (t_1, t_1') and model move $(t_1, \gg)$ can be mapped to transition t_1 in SLPN with r_t.

As the alignment explicitly identifies a model path through the original SLPN, we can associate a model move or a synchronous move with the probability of the model path. To do so, we introduce the concept of probability gain based on the reverse functions.

Definition 6 (Probability Gain). *Let $N^\otimes = (P^\otimes, T^\otimes, F^\otimes, A^\otimes, \rho^\otimes, M_0^\otimes)$ be the synchronous product net of an SLPN $N^s = (P^s, T^s, F^s, A^s, \rho^s, M_0^s, w^s)$ and a trace net N^σ, r_m and r_t be the reverse functions. Then, the probability gain is a function that takes a marking $m^\otimes$ of $N^\otimes$ and an enabled transition $t^\otimes \in T_{m^\otimes}$:*

$$\mathrm{p}(m^\otimes, t^\otimes) = \begin{cases} 1 & \text{if } t^\otimes \text{ is a trace move} \\ \frac{w^s(r_t(t^\otimes))}{\Sigma_{t \in T^s_{r_m(m^\otimes)}} w^s(t)} & \text{otherwise.} \end{cases} \tag{3}$$

The probability gain function in Eq. (3) associates a model move or a synchronous move with the probability of the corresponding model transition, and associates a trace move with probability 1. For instance, in the synchronous product net shown in Fig. 3, $(\gg, t_1)$, $(\gg, t_2)$, $(t_1', \gg)$ and (t_1', t_1) are enabled at the initial marking $[p_5^1, p_1^1]$. The probability gain of trace move $(t_1', \gg)$ is 1. For $(\gg, t_1)$, $(\gg, t_2)$, and (t_1', t_1), their probability gains are $1/100$, $99/100$ and $1/100$, respectively.

The probability gain of an alignment is computed as the multiplication of the probability gain of its sequence of moves, which corresponds to the probability of the underlying model path from SLPN. Although we assign a probability gain of 1 to trace moves, this does not introduce bias favoring moves on the trace over moves on the model (whose probability depends on relative model weights), since alignment selection is determined by the probability of the model path.

In the following section, we discuss the way to use probability gain to compute stochastic alignment.

3 Stochastic Alignment

In this section, we introduce our stochastic alignment technique, which matches an observed trace to a stochastic process model. The problem is solved by translating it to a shortest-path search problem that can be addressed by the a-star algorithm. First, we describe the Pareto front that exists for the probability

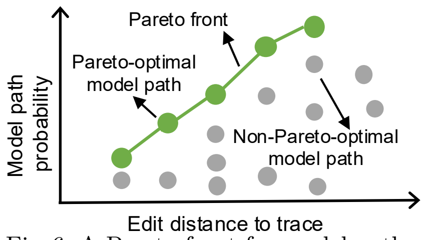


Fig. 6: A Pareto front for model paths.

Table 1: The probability of model paths in Fig. 2, and their edit distance to $\langle a, d, c \rangle$.

model path	model trace	probability	edit distance
$\langle t_1, t_3 \rangle$	$\langle a, c \rangle$	$^5/_{500}$	1
$\langle t_2, t_3, t_4 \rangle$	$\langle b, c, d \rangle$	$^{297}/_{500}$	4
$\langle t_2, t_4, t_3 \rangle$	$\langle b, d, c \rangle$	$^{198}/_{500}$	2

of model paths and their edit distance to the trace. Second, we introduce a loss function with a user-defined balance factor, which enables business analysts to weigh the trade-off and compute an alignment that satisfies the Pareto-optimality. Third, we translate our problem to a shortest path problem using a standard a-star algorithm [18] and provide a heuristic for SLPNs.

A model path with a low edit distance to a given trace effectively explains where the model and the log trace disagree. However, suppose such a model path has a marginal probability according to the stochastic model, then it may be deemed as less relevant to match the trace. This scenario illustrates that the probability of a selected model path and its edit distance to the trace are two potentially competing objectives when explaining deviations. This trade-off naturally leads to a Pareto front, which we illustrate in Figure 6.

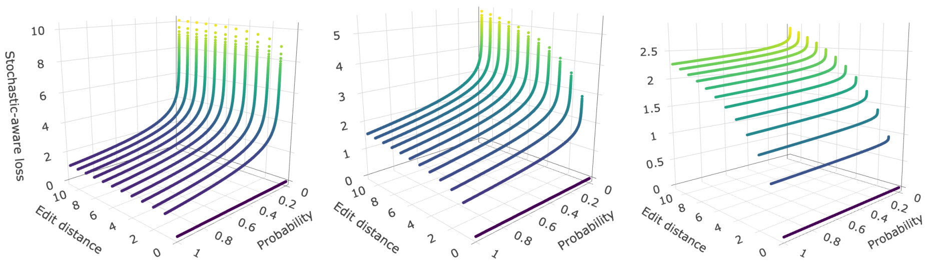
In this work, a Pareto front consists of model paths representing an optimal trade-off: any reduction in edit distance would necessarily decrease the model path probability, and any increase in model path probability would necessarily increase its edit distance to the trace. For instance, as illustrated in Table 1, given the observed trace $\langle a, d, c \rangle$, there are three Pareto-optimal model paths allowed by N_2, namely $\langle t_1, t_3 \rangle$, $\langle t_2, t_3, t_4 \rangle$, and $\langle t_2, t_4, t_3 \rangle$.

3.1 Loss Function

Selecting an alignment from the Pareto front requires balancing two competing objectives: the model path probability p and its edit distance d to the trace. This can be quantified through a function $f_1 \colon d \to \mathbb{R}_{\geq 0}$ for edit distance d and a function $f_2 \colon p \to \mathbb{R}_{\geq 0}$ for probability p. As demonstrated previously, no single model path in SLPN may be optimal for both f_1 and f_2 simultaneously. Thus, a stochastic alignment should be measured by a combined function $loss \colon (d, p) \to \mathbb{R}_{\geq 0}$ that addresses this trade-off, so that it satisfies the following properties.

Property 1. Given a trace, if two model paths have the same edit distance to it, the more likely model path yields a smaller value for the loss function. That is, let d be the edit distance with $d > 0$, for all $p_1 > p_2 \Rightarrow loss(d, p_1) < loss(d, p_2)$.

Property 2. Given a trace, if two model paths are equally likely according to the stochastic model, the model path with a smaller edit distance to the trace yields



(a) α=0.1. (b) α=0.5. (c) α=0.9.

Fig. 7: The value of loss function with different user-defined balance factor α.

a smaller value for the loss function. That is, let p be the probability, for all $d_1 < d_2 \Rightarrow loss(d_1, p) < loss(d_2, p)$.

Property 3. Given a trace, the more likely model path with a smaller edit distance to the trace yields a smaller value for the loss function. That is, for all $0 \leq d_1 < d_2 \wedge p_1 > p_2 > 0 \Rightarrow loss(d_1, p) < loss(d_2, p)$.

To account for these properties, we propose a loss function to measure Pareto optimality as follows.

$$loss(d,p) = \begin{cases} (\ln(d+1))^{\alpha} & \text{if } \alpha = 1 \\ (1 - \ln p)^{1-\alpha} & \text{if } \alpha = 0 \\ (\ln(d+1))^{\alpha} \cdot (1 - \ln p)^{1-\alpha} & \text{if } \alpha \in (0,1) \end{cases} \tag{4}$$

In Eq. (4), $\ln(1 + d)$ is the logarithmic edit distance, in which we add one to avoid the negative-infinite $\ln 0$, and $(1 - \ln p)$ represents the inverse of the logarithm of the path's probability.

The loss function is controlled by a user-defined balance factor $\alpha \in [0, 1]$, which assigns a weight to the logarithmic probability function and logarithmic edit distance function, so business analysts can balance between these two perspectives. It is trivial to see that the function adheres to Properties 1, 2, and 3 when α is not equal to 0 or 1. For the special case of $\alpha = 1$, the model path with the least edit distance to the trace has the least loss. This corresponds to conventional alignment that only minimizes the deviations between the model path and the trace. In another special case of $\alpha = 0$, the most likely model path has the least loss.

Fig. 7 illustrates the influence of α. The loss values for different α are not directly comparable, as a lower α leads to a generally higher loss for the same edit distance and model path probability. If α is 0.5, the trade-off between the edit distance and the probability of the model path scales relatively evenly in both directions, as shown in Fig. 7b. For example, a model path with a probability of 0.003 and an edit distance of 4 to the trace has a stochastic-aware loss smaller than a model path with a probability of 0.001 and an edit distance of 3. When α

Table 2: The value of loss function for running example with different balance factors.

path	α=0	α=0.25	α=0.5	α=0.75	α=1
$\langle t_1, t_3\rangle$	3.000	1.688	0.950	**0.535**	**0.301**
$\langle t_2, t_3, t_4\rangle$	**1.226**	**1.065**	0.926	0.804	0.699
$\langle t_2, t_4, t_3\rangle$	1.402	1.071	**0.818**	0.625	0.477

is set to 0.1, the value of the loss function is based mainly on the probability of the path of the model, as a marginal increase in the probability would lead to a smaller loss from the probability range of 0 to 0.2. For example, a model path with a probability of 0.0013 and an edit distance of 3 to the trace has a loss that is smaller than a model path with a probability of 0.001 and an edit distance of 2. When α approaches 1, a model path with a small edit distance can hardly be overtaken by another model path with a larger edit distance.

Table 2 presents the values of the loss function using different α for our running example. Different model paths are favored under different α, as highlighted in bold. The example illustrates that, depending on the user-defined α, the stochastic alignment computed changes accordingly.

3.2 Computing a Stochastic Alignment

Given a trace and an SLPN of the process, we introduce a shortest path search to compute a stochastic alignment that minimizes the loss function.

Applying a-star. We solve our problem by identifying a sequence of moves in the synchronous product net that minimizes the loss function. This optimization can be formulated as a shortest path problem [2]. To guide the a-star search, we introduce a dual-component heuristic: one estimates the minimum remaining edit distance to reach a deadlock marking, while the other estimates the maximum remaining probability gain to a deadlock marking. During search iteration, the $f(M)$ of a marking M visited can be formulated as:

$$f(M) = \begin{cases} (\ln(g_d + h_d + 1))^{\alpha} & \text{if } \alpha = 1 \\ (1 - \ln(g_p * h_p))^{1-\alpha} & \text{if } \alpha = 0 \\ (\ln(g_d + h_d + 1))^{\alpha} \cdot (1 - \ln(g_p * h_p))^{1-\alpha} & \text{if } \alpha \in (0, 1) \end{cases} \tag{5}$$

where g_d and g_p are the actual edit distance and probability gain from the initial marking to M, h_d and h_p are the estimated remaining edit distance and probability gain from M to a deadlock marking. We first introduce the heuristic that underestimates the edit distance with Mixed-Integer Linear Programming (MILP).

MILP-based Edit Distance Heuristic. Let $N^{\otimes} = (P^{\otimes}, T^{\otimes}, F^{\otimes}, A^{\otimes}, \rho^{\otimes}, M_0^{\otimes})$ be the synchronous product net of a trace and an SLPN, with incidence matrix $\mathcal{I}$ and consumption matrix $\mathcal{C}$. Then, we solve the following edit distance heuristic.

$$
\begin{aligned}
\text{minimize } & h_d = \vec{d}^\intercal \cdot \vec{x} \\
\text{such that } & \vec{x}(t) \in \mathbb{N} \wedge && (6)\\
& \overrightarrow{M_d} = \vec{M} + \mathcal{I} \cdot \vec{x} \wedge && (7)\\
& \forall_{t \in T^\otimes} \exists_{p \in P^\otimes} \ \overrightarrow{M_d}(p) < \mathcal{C}(p,t) && (8)
\end{aligned}
$$

In which $\vec{x}$ is a Parikh vector that describes the number of times each transition should be fired to reach a deadlock marking from M, and $\vec{d}$ is a $|N^\otimes.T|$-sized vector for the transition-based edit distance function. The heuristic for edit distance h_d is the result of the minimization.

Constraint (6) specifies that each element in the solution vector $\vec{x}$ must be non-negative. Constraints (7) and (8) specify that M_d should be a deadlock marking that does not enable any transition. Due to possibly (infinitely) many deadlock markings in an SLPN, the synchronous net can also have multiple deadlock markings. Thus, the MILP-based edit distance heuristics differentiate from existing (I)LP-based heuristics [2,15] by not explicitly specifying a deadlock marking. As no transition is enabled at M_d, at least one place in the preset of each transition does not contain enough tokens. We use binary decision variables and apply the big-M method [17] to construct constraints (8). Therefore, the heuristic is computed by solving an MILP.

Lemma 1 (h_d provides a lower bound on the edit distance to the deadlock marking). *Let $N^\otimes = (P^\otimes, T^\otimes, F^\otimes, A^\otimes, \rho^\otimes, M_0^\otimes)$ be a synchronous product net, and $d : T^\otimes \to \mathcal{R}_{\geq 0}$ be an edit distance function. Given a marking $M \in \mathcal{B}(P^\otimes)$, and a firing sequence $\gamma \in {T^\otimes}^*$ such that $M[\gamma\rangle M_d$ and $T^\otimes_{M_d} = \emptyset$, it holds that $h_d \leq d(\gamma)$.*

This lemma follows from the fact that $h_d = \vec{d}^\intercal \cdot \vec{x} \leq d(\gamma) = \vec{d}^\intercal \cdot \vec{\gamma}$, otherwise $\vec{x}$ is not minimizing. Likewise, we define another heuristic that overestimates the remaining probability of reaching the deadlock marking.

MILP-based Probability Gain Heuristic. Let $N^\otimes = (P^\otimes$, $T^\otimes$, $F^\otimes$, $A^\otimes$, $\rho^\otimes$, $M_0^\otimes)$ be the synchronous product net of a trace and an SLPN, with incidence matrix $\mathcal{I}$ and consumption matrix $\mathcal{C}$. Then, we solve the following probability gain heuristic.

$$
\begin{aligned}
\text{maximize } & h_p = \vec{p}^\intercal \cdot \vec{x} \\
\text{such that } & \vec{x}(t) \in \mathbb{N} \wedge && (9)\\
& \overrightarrow{M_d} = \vec{M} + \mathcal{I} \cdot \vec{x} \wedge && (10)\\
& \forall_{t \in T^\otimes} \exists_{p \in P^\otimes} \ \overrightarrow{M_d}(p) < \mathcal{C}(p,t) && (11)
\end{aligned}
$$

In which $\vec{x}$ is a Parikh vector, $\vec{p}$ is a $|N^\otimes.T|$-sized vector for the transition-based probability gain function, and h_p is the maximized estimation of the remaining probability gain.

Lemma 2 (h_p provides an upper bound on the probability gain to the deadlock marking). *Let $N^{\otimes} = (P^{\otimes}, T^{\otimes}, F^{\otimes}, A^{\otimes}, \rho^{\otimes}, M_0^{\otimes})$ be a synchronous product net, and $p\colon T^{\otimes} \to \mathcal{R}_{\geq 0}$ be a probability gain function. Given a marking $M \in \mathcal{B}(P^{\otimes})$, and a firing sequence $\gamma \in {T^{\otimes}}^{*}$ such that $M[\gamma\rangle M_d$ and $T^{\otimes}_{M_d} = \emptyset$, it holds that $h_p \geq p(\gamma)$.*

This lemma follows from the fact that $h_p = \vec{p}^{\mathsf{T}} \cdot \vec{x} \geq p(\gamma) = \vec{d}^{\mathsf{T}} \cdot \vec{\gamma}$, otherwise $\vec{x}$ is not maximizing. As the sub-heuristic for edit distance is minimizing and the sub-heuristic for probability gain is maximizing, the dual-heuristic for a-star is admissible.

Lemma 3 (The dual-heuristic is admissible). *Let $N^{\otimes} = (P^{\otimes}, T^{\otimes}, F^{\otimes}, A^{\otimes}, \rho^{\otimes}, M_0^{\otimes})$ be the synchronous product net, g_d and g_p be the actual edit distance and probability gain for the path from $M_0^{\otimes}$ to marking M, h_d and h_p be the heuristic of edit distance and probability gain for M. For all firing sequence $\gamma \in {T^{\otimes}}^{*}$ such that $M[\gamma\rangle M_d$ and $T^{\otimes}_{M_d} = \emptyset$, it holds that $h_d \leq d(\gamma)$ and $h_p \geq p(\gamma)$. With the loss function Eq.* (4)*, we have:*

$$\begin{aligned} \ln(g_d + h_d + 1)^{\alpha} &\leq \ln(g_d + d(\gamma) + 1)^{\alpha}, \ \textit{if } \alpha = 0 \\ (1 - \ln(g_p * h_p))^{(1-\alpha)} &\leq (1 - \ln(g_p * p(\gamma)))^{(1-\alpha)}, \ \textit{if } \alpha = 1 \end{aligned}$$

The dual-heuristic for $\alpha \in (0, 1)$ also lead to an underestimation of loss for M.

Algorithmic description Using the heuristic for edit distance and probability gain, we propose a modified version of the a-star search algorithm. The input of our algorithm is the synchronous product net of a trace and SLPN, and the loss function defined by the user in Eq. (4). In each iteration, the marking with the minimized loss is chosen for expansion. If it is not a deadlock marking, the search expands over all subsequent markings. The edit distance and the probability gain of the path when reaching a marking, combined with the heuristic, determine the loss for the expanded marking. The algorithm terminates once a deadlock marking is found and returns an alignment.

4 Evaluation

Our technique for computing the stochastic alignment has been implemented and is publicly available.[5] We first examine the scalability of the technique using stochastic models mined from real-life event logs. Subsequently, we conduct a case study to illustrate actionable insights gained from our approach. All experiments were performed on a MacBook Pro, with an M2 Pro processor, 32 GB memory, and macOS Sequoia 15.

[5] https://github.com/BPM-Research-Group/Ebi

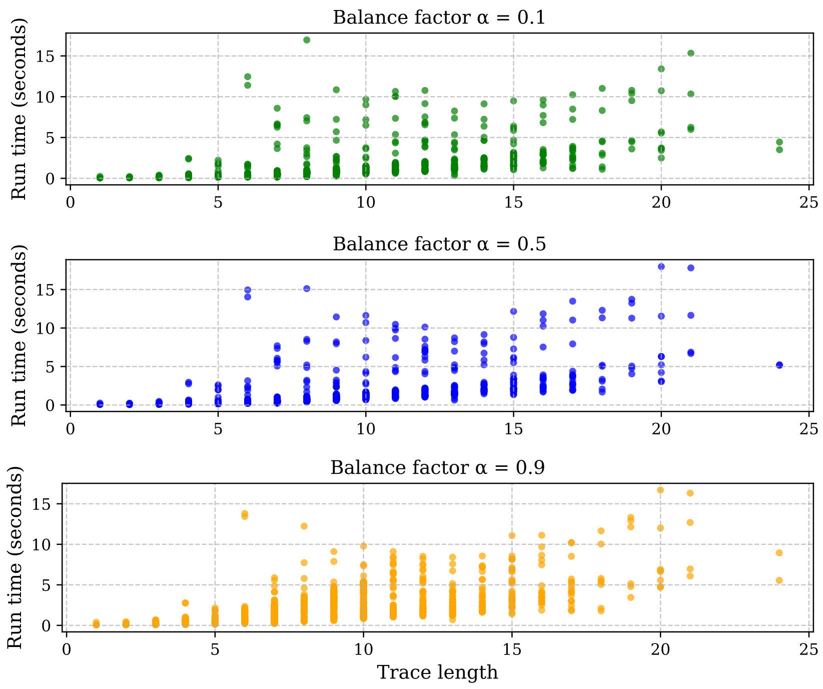


Fig. 8: Running time versus trace length for different balance factors.

4.1 Scalability

To evaluate the scalability of our technique, we used four event logs: road traffic fine management [26], loan application [13], payment request [14], and domestic declaration [14]. For each event log, we used Inductive Miner [7] and Direct Follow Miner [24] to discover two control-flow models and then applied the alignment weight estimator [10] to obtain SLPNs. We then computed our stochastic alignment for each SLPN and individual trace from 1040 model-trace pairs, for several values of α (0.1, 0.5, and 0.9), and measured the trace length and run time of our technique. Each such run was repeated three times to reduce random effects, and the reported times are the averages over three runs. Figure 8 shows the results.

Most alignments are computed in less than 10 seconds, with a few exceptions, though no computation took more than 20 seconds. Our technique computes a shortest path in the reachability graph of the synchronous product net of the trace net and SLPN. The size of this reachability graph grows exponentially with the number of places in the product net, while the shortest path, without heuristics, can, in the worst case, be computed using Dijkstra's algorithm that relies on the Fibonacci heap priority queue in $O(E + V logV)$, where V and E are the numbers of vertices and edges in the reachability graph. The few outliers in Fig. 8 show scenarios where the a-star performs poorly, as the solution vector

$\vec{x}$ for the MILP does not necessarily correspond to an actual path, and thus $\vec{x}$ needs to be recomputed often. For different values of α, we see a slight reduction in run time for lower α, i.e., the model path probability having a larger influence, which may be due to alignments being computationally more expensive than finding the most likely model path. In general, the results show that our technique is feasible for observed traces from real-life event logs and SLPNs discovered from these logs.

4.2 Applicability

To evaluate the applicability of our approach, we applied our technique to analyze an observed trace from a travel permit process [14]. The process usually starts with a travel permit submitted by an employee. After submission, the permit request is sent to the travel administration for approval. If approved, the request is forwarded to the budget owner and then to the supervisor. In some cases, the director also needs to approve the request. The process ends with the payment being requested and handled.

The experiment involves an auditor who has been equipped with the normative SLPN and has selected a trace observed outside the training log, which stood out because it was nonconforming. The goal is to analyze where the trace likely deviated from the model and to investigate whether any rules were violated.

To obtain the normative stochastic process model, we split a training log consisting of the traces from 2018 onward. Then, we apply Split Miner [7] with default settings to discover a BPMN model, which is subsequently converted to an LPN, to which alignment weight estimation is applied [10].

The observed trace selected by the auditor is $\langle$*Permit submitted by employee, Permit final approved by supervisor, Start trip, Declaration submitted by employee, Declaration final approved by supervisor, Request payment, End trip, Payment handled*$\rangle$ ($\langle PS, PF, ST, DS, DF, RP, ET, PH \rangle$). If the auditor only considers the edit distance, that is, applies our technique with $\alpha = 1$ that outputs a conventional alignment, a closest model path is $m_1 = \langle$*Permit submitted by employee, Permit approved by administration, Permit final approved by supervisor, Start trip, Declaration submitted by employee, Declaration approved by administration, Declaration final approved by supervisor, Request payment, Payment handled*$\rangle$ ($\langle PS, PA, PF, ST, DS, DA, DF, RP, PH \rangle$) with an edit distance of 3. In particular, the model path m_1 contains two steps in which the administration is involved, while the trace does not contain these steps. This may indicate a policy violation. Moreover, this model path m_1 has a probability of 0.32%.

The auditor then applies our technique with $\alpha = 0.5$ to obtain a model path that balances the likelihood of the model path and the edit distance. This model path is $m_{0.5} = \langle$*Permit submitted by employee, Permit approved by administration, Permit final approved by supervisor, Start trip, End trip, Declaration submitted by employee, Declaration approved by administration, Declaration final approved by supervisor, Request payment, Payment handled*$\rangle$ ($\langle PS, PA, PF, ST, ET, DS, DA, DF, RP, PH \rangle$). $m_{0.5}$ has a probability of 7.23% according to the model, which is 22 times higher than m_1.

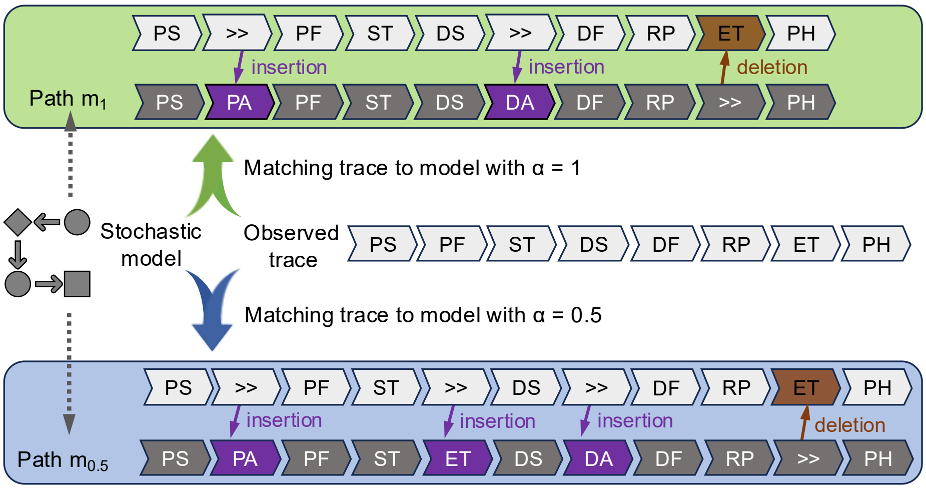


Fig. 9: Matching the observed trace to the model with a different balance factor.

In Fig. 9, we illustrate two alignment results returned to the auditor. The green region above corresponds to the conventional alignment with the least deviations, while the blue region below shows an alternative stochastic alignment that balances between deviation severity and path probability. Compared to m_1, $m_{0.5}$ adds a deviation to the additional activity *End trip* before *Declaration submitted by the employee*. As the model is normative, the auditor concludes that *Permit approved by administration* and *Declaration approved by administration* are missing from the observed trace (e.g., the execution of the activity was not performed following the formal procedure). At the same time, the more likely model path suggests that the payment declaration should have been submitted after the end of the trip, rather than after the start of the trip. The example shows that stochastic alignment generates new useful diagnostic insights for analysts in comparison to conventional alignment.

5 Related Work

Conformance checking provides mechanisms to relate modeled and observed process behavior. The techniques in non-stochastic settings, such as token-replay [30] and alignments [2,15,28], have been extensively discussed.

The stochastic perspective of process behavior has enabled several new conformance checking techniques. Entropic Relevance [5] computes the average number of bits to compress the event log using the information of the likelihood of trace in a stochastic process model. Unit Earth Movers' Stochastic Conformance (uEMSC) [25] and Earth Movers' Stochastic Conformance (EMSC) [21] measure the effort to transform the distribution of traces in the log to the distribution described in the stochastic model. [29] extends EMSC by considering partial trace mismatches, which is solved using Markovian abstraction [6] for SLPNs. Li et al. [27] use the Jensen-Shannon Distance to compare two probability distributions over traces with their average probability distribution. [19] applies the software performance engineering of Markovian processes from logs and compares it with model using uEMSC. Although these techniques compute a numerical value between a stochastic model and an event log, the traces in the log are not explicitly matched to the model. Moreover, they are not applicable if an aggregated event log is not available.

Bogdanov et al. [9] compute an alignment for a stochastically known trace with a probability function. Alizadeh et al. [3] extend the alignment cost functions by accounting for the probabilities of activities based on the process history, while Koorneef et al. [20] proposed to calculate the most probable alignment between a model and a log based on the probabilities of the observed process behavior in the event log. However, the models used in these techniques are not stochastic. A work that uses stochastic models for alignment computation is the work by Bergami et al. [8], which matches a trace to a stochastic workflow net and returns a ranked list of approximated alignments.

The technique proposed in this paper is not directly comparable to existing techniques, as the input event data (individual trace vs. aggregated log [2,3,20]) and models (stochastic vs. non-stochastic [2,20,9]) differ fundamentally. Furthermore, the output is one single alignment rather than a numerical value [5,21,27] or a list of alignments [8].

6 Conclusion

In this paper, we propose a stochastic alignment technique that matches a trace with a likely path through a stochastic process model. We introduce a user-defined parameter α that ranges from zero to one to allow for the trade-off between alignment cost and normative behavior. For stochastic labeled Petri nets (SLPNs), we formalize this as an optimization problem and solve it by finding a shortest path in the synchronous product net using the a-star algorithm. To navigate the search from the initial marking to potentially infinitely many deadlock markings, we use a dual-component heuristic.

Although the proposed stochastic alignment shares conceptual similarity with conventional alignment [2] by matching a trace to a model path, its stochastic perspective introduces additional constraints. The first difference lies in how the technique handles stochastic models containing loops composed entirely of silent model moves. A stochastic alignment ($\alpha < 1$) that identifies a path resulting from silent loops will incur a higher loss compared to one without such loops, as additional silent model moves represent more decision points, while the probability gains associated with silent model moves are constrained to be less than or equal to one. Moreover, unless the balance factor is equal to 1, one cannot map all the traces to the most likely path of the model, as it is extremely unlikely that *all* traces of the log took the most likely path and not a single trace took other less likely paths. Enforcing all traces in a log to follow the "most likely" model paths would contradict real-world variability, as a subset of traces inherently follows less probable paths.

The technique has been implemented and is publicly available [22]. Our evaluation shows that the technique is feasible on real-life event data. In a case study on a travel permit approval process, the technique identified deviations while quantifying the likelihood of the model path, offering auditors new and interesting insight into the conformance of the process.

Accounting for multiple traces (or a complete event log) simultaneously is left for future work. When matching a group of traces with a stochastic process model, instead of optimizing the match for each trace, one can search for a globally optimal match in which some likely model paths do not get overutilized by multiple traces using, for instance, ideas similar to those applied in Earth Mover's Distance [21]. We also plan to investigate a way to compute all Pareto-optimal model paths for a given trace. Currently, the proposed technique is guided by a user-defined parameter to search for an alignment that identifies a Pareto-optimal model path. We also plan to adopt other loss functions, such as the work in decision theory [12] that uses aggregation functions to balance multiple objectives.

References

1. van der Aalst, W.M.P.: Process Mining - Data Science in Action, Second Edition. Springer (2016)
2. Adriansyah, A.: Aligning observed and modeled behavior (2014)
3. Alizadeh, M., de Leoni, M., Zannone, N.: History-based construction of alignments for conformance checking: Formalization and implementation. In: SIMPDA (Revised Selected Papers). Lecture Notes in Business Information Processing, vol. 237, pp. 58–78. Springer (2014)
4. Alkhammash, H., Polyvyanyy, A., Moffat, A.: Stochastic directly-follows process discovery using grammatical inference. In: CAiSE. LNCS, vol. 14663, pp. 87–103. Springer (2024)
5. Alkhammash, H., Polyvyanyy, A., Moffat, A., García-Bañuelos, L.: Entropic relevance: A mechanism for measuring stochastic process models discovered from event data. Inf. Syst. **107**, 101922 (2022)
6. Augusto, A., Armas-Cervantes, A., Conforti, R., Dumas, M., Rosa, M.L.: Measuring fitness and precision of automatically discovered process models: A principled and scalable approach. IEEE Trans. Knowl. Data Eng. **34**(4), 1870–1888 (2022)
7. Augusto, A., Conforti, R., Dumas, M., Rosa, M.L., Polyvyanyy, A.: Split miner: automated discovery of accurate and simple business process models from event logs. Knowl. Inf. Syst. **59**(2), 251–284 (2019)
8. Bergami, G., Maggi, F.M., Montali, M., Peñaloza, R.: Probabilistic trace alignment. In: ICPM. pp. 9–16. IEEE (2021)
9. Bogdanov, E., Cohen, I., Gal, A.: Conformance checking over stochastically known logs. In: BPM (Forum). Lecture Notes in Business Information Processing, vol. 458, pp. 105–119. Springer (2022)
10. Burke, A., Leemans, S.J.J., Wynn, M.T.: Stochastic process discovery by weight estimation. In: ICPM Workshops. LNBIP, vol. 406, pp. 260–272. Springer (2020)

11. Carmona, J., van Dongen, B.F., Weidlich, M.: Conformance checking: Foundations, milestones and challenges. In: Process Mining Handbook, Lecture Notes in Business Information Processing, vol. 448, pp. 155–190. Springer (2022)
12. Donadello, I., de Souza, R.L., Hunter, A., Dragoni, M.: Compromises in dialogical argumentation: Aggregated policies for biparty decision theory. In: ECAI. Frontiers in Artificial Intelligence and Applications, vol. 392, pp. 3437–3444. IOS Press (2024)
13. van Dongen, B.: Bpi challenge 2017 (2017)
14. van Dongen, B.: Bpi challenge 2020 (2020)
15. van Dongen, B.F.: Efficiently computing alignments - using the extended marking equation. In: BPM. Lecture Notes in Computer Science, vol. 11080, pp. 197–214. Springer (2018)
16. Dumas, M., van der Aalst, W.M.P., ter Hofstede, A.H.M. (eds.): Process-Aware Information Systems: Bridging People and Software Through Process Technology. Wiley (2005)
17. Griva, I., Nash, S.G., Sofer, A.: Linear and Nonlinear Optimization. SIAM (2008)
18. Hart, P.E., Nilsson, N.J., Raphael, B.: A formal basis for the heuristic determination of minimum cost paths. IEEE Trans. Syst. Sci. Cybern. **4**(2), 100–107 (1968)
19. Incerto, E., Vandin, A., Ahrabi, S.S.: Stochastic conformance checking based on variable-length markov chains. Inf. Syst. **133**, 102561 (2025)
20. Koorneef, M., Solti, A., Leopold, H., Reijers, H.A.: Automatic root cause identification using most probable alignments. In: Business Process Management Workshops. Lecture Notes in Business Information Processing, vol. 308, pp. 204–215. Springer (2017)
21. Leemans, S.J.J., van der Aalst, W.M.P., Brockhoff, T., Polyvyanyy, A.: Stochastic process mining: Earth movers' stochastic conformance. Inf. Syst. **102**, 101724 (2021)
22. Leemans, S.J.J., Li, T., van Detten, J.N.: Ebi - a stochastic process mining framework. In: ICPM Doctoral Consortium / Demo. CEUR Workshop Proceedings, vol. 3783. CEUR-WS.org (2024)
23. Leemans, S.J.J., Polyvyanyy, A.: Stochastic-aware precision and recall measures for conformance checking in process mining. Inf. Syst. **115**, 102197 (2023)
24. Leemans, S.J.J., Poppe, E., Wynn, M.T.: Directly follows-based process mining: Exploration & a case study. In: ICPM. IEEE (2019)
25. Leemans, S.J.J., Syring, A.F., van der Aalst, W.M.P.: Earth movers' stochastic conformance checking. In: BPM Forum. Lecture Notes in Business Information Processing, vol. 360, pp. 127–143. Springer (2019)
26. de Leoni, M.M., Mannhardt, F.: Road traffic fine management process (2015)
27. Li, T., Leemans, S.J.J., Polyvyanyy, A.: The jensen-shannon distance metric for stochastic conformance checking. In: ICPM workshops, volume to appear of LNBIP (2024)
28. Li, T., van Zelst, S.J.: Cache enhanced split-point-based alignment calculation. In: ICPM. pp. 120–127. IEEE (2022)
29. Rocha, E.G., Leemans, S.J.J., van der Aalst, W.M.P.: Stochastic conformance checking based on expected subtrace frequency. In: ICPM. pp. 73–80. IEEE (2024)
30. Rozinat, A., van der Aalst, W.M.P.: Conformance checking of processes based on monitoring real behavior. Inf. Syst. **33**(1), 64–95 (2008)