

# Evaluating Conformance Measures in Process Mining using Conformance Propositions (Extended Version)

Anja F. Syring<sup>1</sup>, Niek Tax<sup>2</sup>, and Wil M.P. van der Aalst<sup>1,2,3</sup>

<sup>1</sup> Process and Data Science (Informatik 9),  
RWTH Aachen University, D-52056 Aachen, Germany

<sup>2</sup> Architecture of Information Systems,  
Eindhoven University of Technology, Eindhoven, The Netherlands

<sup>3</sup> Fraunhofer Institute for Applied Information Technology FIT,  
Sankt Augustin, Germany

**Abstract.** Process mining sheds new light on the relationship between process models and real-life processes. Process discovery can be used to learn process models from event logs. Conformance checking is concerned with quantifying the *quality* of a business process model in relation to event data that was logged during the execution of the business process. There exist different categories of conformance measures. *Recall*, also called fitness, is concerned with quantifying how much of the behavior that was observed in the event log fits the process model. *Precision* is concerned with quantifying how much behavior a process model allows for that was never observed in the event log. *Generalization* is concerned with quantifying how well a process model generalizes to behavior that is possible in the business process but was never observed in the event log. Many recall, precision, and generalization measures have been developed throughout the years, but they are often defined in an ad-hoc manner without formally defining the desired properties up front. To address these problems, we formulate *2l conformance propositions* and we use these propositions to evaluate current and existing conformance measures. The goal is to trigger a discussion by clearly formulating the challenges and requirements (rather than proposing new measures). Additionally, this paper serves as an overview of the conformance checking measures that are available in the process mining area.

**Keywords:** Process mining · Conformance checking · Evaluation measures

## 1 Introduction

Process mining [1] is a fast growing discipline that focuses on the analysis of event data that is logged during the execution of a business process. Events in such an event log contain information on what was done, by whom, for whom, where, when, etc. Such event data are often readily available from information systems that support the execution of the business process, such as ERP, CRM, or BPM systems. *Process discovery*, the task of automatically generating a process model that accurately describes a business process based on such event data, plays a prominent role in process mining. Throughout

the years, many process discovery algorithms have been developed, producing process models in various forms, such as Petri nets, process trees, and BPMN.

Event logs are often incomplete, i.e., they only contain a sample of all possible behavior in the business process. This not only makes process discovery challenging; it is also difficult to assess the quality of the process model in relation to the log. Process discovery algorithms take an event log as input and aim to output a process model that satisfies certain properties, which are often referred to as the four quality dimensions [1] of process mining: (1) *recall*: the discovered model should allow for the behavior seen in the event log (avoiding “non-fitting” behavior), (2) *precision*: the discovered model should not allow for behavior completely unrelated to what was seen in the event log (avoiding “underfitting”), (3) *generalization*: the discovered model should generalize the example behavior seen in the event log (avoiding “overfitting”), and (4) *simplicity*: the discovered model should not be unnecessarily complex. The simplicity dimension refers to Occam’s Razor: “one should not increase, beyond what is necessary, the number of entities required to explain anything”. In the context of process mining, this is often operationalized by quantifying the complexity of the model (number of nodes, number of arcs, understandability, etc.). We do *not* consider the simplicity dimension in this paper, since we focus on *behavior* and abstract from the actual model representation. Recall is often referred to as *fitness* in process mining literature. Sometimes fitness refers to a combination of the four quality dimensions. To avoid later confusion, we use the term recall which is commonly used in pattern recognition, information retrieval, and (binary) classification. Many conformance measures have been proposed throughout the years, e.g., [1,3,5,12,13,14,15,24,25,27,31,32].

So far it remains an open question whether existing measures for recall, precision, and generalization measure what they are aiming to measure. This motivates the need for a formal framework for conformance measures. Users of existing conformance measures should be aware of seemingly obvious quality issues of existing approaches and researchers and developers that aim to create new measures should be clear on what conformance characteristics they aim to support. To address this open question, this paper evaluates state-of-the-art conformance measures based on 21 propositions introduced in [2].

The remainder is organized as follows. Section 2 discusses related work. Section 3 introduces basic concepts and notations. The rest of the paper is split into two parts where the first one discusses the topics of recall and precision (Section 4) and the second part is dedicated to generalization (Section 5). In both parts, we introduce the corresponding conformance propositions and provide an overview of existing conformance measures. Furthermore, we discuss our findings of validating existing these measures on the propositions. Additionally, Section 4 demonstrates the importance of the propositions on several baseline conformance measures, while Section 5 includes a discussion about the different points of view on generalization. Section 6 concludes the paper.

## 2 Related work

In early years, when process mining started to gain in popularity and the community around it grew, many process discovery algorithms were developed. But at that time

there was no standard method to evaluate the results of these algorithms and to compare them to the performance of other algorithms. Based on this, Rozinat et al. [28] called on the process mining community to develop a standard framework to evaluate process discovery algorithms. This led to a variety of fitness/recall, precision, generalization and simplicity notions [1]. These notions can be quantified in different ways and there are often trade-offs between the different quality dimensions. As shown using generic algorithms assigning weights to the different quality dimensions [10], one quickly gets degenerate models when leaving out one or two dimensions. For example, it is very easy to create a simple model with perfect recall (i.e., all observed behavior fits perfectly) that has poor precision and provides no insights.

Throughout the years, several conformance measures have been developed for each quality dimension. However, it is unclear whether these measures actually measure what they are supposed to. An initial step to address the need for a framework to evaluate conformance measures was made in [29]. Five so-called *axioms* for precision measures were defined that characterize the desired properties of such measures. Additionally, [29] showed that none of the existing precision measures satisfied all of the formulated axioms. In comparison to [29] Janssenswillen et al. [19] did not rely on qualitative criteria, but quantitatively compared existing recall, precision and generalization measures under the aspect of feasibility, validity and sensitivity. The results showed that all recall and precision measures tend to behave in a similar way, while generalization measures seemed to differ greatly from each other. In [2] van der Aalst made a follow-up step to [29] by formalizing recall and generalization in addition to precision and by extending the precision requirements, resulting in a list of 21 conformance propositions. Furthermore, [2] showed the importance of probabilistic conformance measures that also take into account trace probabilities in process models. Beyond that, [29] and [2] motivated the process mining community to develop new precision measures, taking the axioms and propositions as a design criterion, resulting in the measures among others the measures that are proposed in [26] and in [7]. Using the 21 propositions of [2] we evaluate state-of-the-art recall (e.g. [4,26,3,16,23,27,33]), precision (e.g. [3,16,17,23,13,26,27,30]) and generalization (e.g. [3,13,16]) measures.

This paper uses the mainstream view that there are at least four quality dimensions: fitness/recall, precision, generalization, and simplicity [1]. We deliberately do not consider simplicity, since we focus on behavior only (i.e., not the model representation). Moreover, we treat generalization separately. In a controlled experiment one can assume the existence of a so-called “system model”. This model can be simulated to create a synthetic event log used for discovery. In this setting, conformance checking can be reduced to measuring the similarity between the discovered model and the system model [9,20]. In terms of the well-known confusion matrix, one can then reason about true positives, false positives, true negatives, and false negatives. However, without a system model and just an event log, it is not possible to find false positives (traces possible in the model but not in reality). Hence, precision cannot be determined in the traditional way. Janssenswillen and Depaire [18] conclude in their evaluation of state-of-the-art conformance measures that none of the existing approaches reliably measures this similarity. However, in this paper, we follow the traditional view on the quality dimensions and exclude the concept of the system from our work.

Whereas there are many fitness/recall and precision measures there are fewer generalization measures. Generalization deals with future cases that were not yet observed. There is no consensus on how to define generalization and in [19] it was shown that there is no agreement between existing generalization metrics. Therefore, we cover generalization in a separate section (Section 5). However, as discussed in [1] and demonstrated through experimentation [10], one cannot leave out the generalization dimension. The model that simply enumerates all the traces in the log has perfect fitness/recall and precision. However, event logs cannot be assumed to be complete, thus proving that a generalization dimension is needed.

### 3 Preliminaries

A *multiset* over a set  $X$  is a function  $B : X \rightarrow \mathbb{N}$  which we write as  $[a_1^{w_1}, a_2^{w_2}, \dots, a_n^{w_n}]$  where for all  $i \in [1, n]$  we have  $a_i \in X$  and  $w_i \in \mathbb{N}^*$ .  $\mathbb{B}(X)$  denotes the set of all multisets over set  $X$ . For example,  $[a^3, b, c^2]$  is a multiset over set  $X = \{a, b, c\}$  that contains three  $a$  elements, one  $b$  element and two  $c$  elements.  $|B|$  is the number of elements in multiset  $B$  and  $B(x)$  denotes the number of  $x$  elements in  $B$ .  $B_1 \uplus B_2$  is the sum of two multisets:  $(B_1 \uplus B_2)(x) = B_1(x) + B_2(x)$ .  $B_1 \setminus B_2$  is the difference containing all elements from  $B_1$  that do not occur in  $B_2$ . Thus,  $(B_1 \setminus B_2)(x) = \max \{B_1(x) - B_2(x), 0\}$ .  $B_1 \cap B_2$  is the intersection of two multisets. Hence,  $(B_1 \cap B_2)(x) = \min \{B_1(x), B_2(x)\}$ .  $[x \in B \mid b(x)]$  is the multiset of all elements in  $B$  that satisfy some condition  $b$ .  $B_1 \subseteq B_2$  denotes that  $B_1$  is contained in  $B_2$ , e.g.,  $[a^2, b] \subseteq [a^2, b^2, c]$ , but  $[a^2, b^3] \not\subseteq [a^2, b^2, c^2]$  and  $[a^2, b^2, c] \not\subseteq [a^3, b^3]$ .

Process mining techniques focus on the relationship between observed behavior and modeled behavior. Therefore, we first formalize event logs (i.e., observed behavior) and process models (i.e., modeled behavior). To do this, we consider a very simple setting where we only focus on the control-flow, i.e., sequences of activities.

#### 3.1 Event Logs

The starting point for process mining is an event log. Each *event* in such a log refers to an *activity* possibly executed by a *resource* at a particular *time* and for a particular *case*. An event may have many more attributes, e.g., transactional information, costs, customer, location, and unit. Here, we focus on control-flow. Therefore, we only consider activity labels and the ordering of events within cases.

**Definition 1 (Traces).**  $\mathcal{A}$  is the universe of activities. A trace  $t \in \mathcal{A}^*$  is a sequence of activities.  $\mathcal{T} = \mathcal{A}^*$  is the universe of traces.

Trace  $t = \langle a, b, c, d, a \rangle$  refers to 5 events belonging to the same case (i.e.,  $|t| = 5$ ). An event log is a collection of cases each represented by a trace.

**Definition 2 (Event Log).**  $\mathcal{L} = \mathbb{B}(\mathcal{T})$  is the universe of event logs. An event log  $l \in \mathcal{L}$  is a finite multiset of observed traces.  $\tau(l) = \{t \in l\} \subseteq \mathcal{T}$  is the set of traces appearing in  $l \in \mathcal{L}$ .  $\bar{\tau}(l) = \mathcal{T} \setminus \tau(l)$  is the complement of the set of non-observed traces.

Event log  $l = [ \langle a, b, c \rangle^5, \langle b, a, d \rangle^3, \langle a, b, d \rangle^2 ]$  refers to 10 cases (i.e.,  $|l| = 10$ ). Five cases are represented by the trace  $\langle a, b, c \rangle$ , three cases are represented by the trace  $\langle b, a, d \rangle$ , and two cases are represented by the trace  $\langle a, b, d \rangle$ . Hence,  $l(\langle a, b, d \rangle) = 2$ .

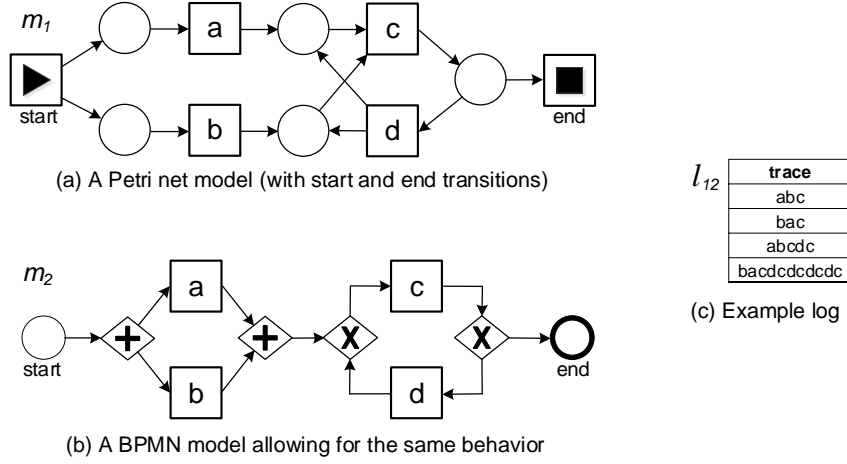


Fig. 1: Two process models  $m_1$  and  $m_2$  allowing for the same set of traces ( $\tau(m_1) = \tau(m_2)$ ) with an example log  $l_{12}$  (c).

### 3.2 Process Models

The behavior of a process model  $m$  is simply the set of traces allowed by  $m$ . In our definition, we will abstract from the actual representation (e.g. Petri nets or BPMN).

**Definition 3 (Process Model).**  $\mathcal{M}$  is the set of process models. A process model  $m \in \mathcal{M}$  allows for a set of traces  $\tau(m) \subseteq \mathcal{T}$ .  $\bar{\tau}(m) = \mathcal{T} \setminus \tau(m)$  is the complement of the set of traces allowed by model  $m \in \mathcal{M}$ .

A process model  $m \in \mathcal{M}$  may abstract from the real process and leave out unlikely behavior. Furthermore, this abstraction can result in  $\tau(m)$  allowing for traces that cannot happen (e.g., particular interleavings or loops).

We distinguish between *representation* and *behavior* of a model. Process model  $m \in \mathcal{M}$  can be represented using a plethora of modeling languages, e.g., Petri nets, BPMN models, UML activity diagrams, automata, and process trees. Here, we abstract from the actual representation and focus on behavioral characteristics  $\tau(m) \subseteq \mathcal{T}$ .

Figure 1 (a) and (b) show two process models that have the same behavior:  $\tau(m_1) = \tau(m_2) = \{\langle a, b, c \rangle, \langle a, c, b \rangle, \langle a, b, c, d, c \rangle, \langle b, a, c, d, c \rangle, \dots\}$ . Figure 1(c) shows a possible event log generated by one of these models  $l_{12} = [\langle a, b, c \rangle^3, \langle b, a, c \rangle^5, \langle a, b, c, d, c \rangle^2, \langle b, a, c, d, c, d, c, d, c, d, c \rangle^2]$ .

The behavior  $\tau(m)$  of a process model  $m \in \mathcal{M}$  can be of infinite size. We use Figure 1 to illustrate this. There is a “race” between  $a$  and  $b$ . After  $a$  and  $b$ , activity  $c$  will occur. Then there is a probability that the process ends or  $d$  can occur. Let  $t_{a,k} = \langle a, b \rangle \cdot (\langle c, d \rangle)^k \cdot \langle c \rangle$  be the trace that starts with  $a$  and where  $d$  is executed  $k$  times.  $t_{b,k} = \langle b, a \rangle \cdot (\langle c, d \rangle)^k \cdot \langle c \rangle$  is the trace that starts with  $b$  and where  $d$  is executed  $k$  times.  $\tau(m_1) = \tau(m_2) = \bigcup_{k \geq 0} \{t_{a,k}, t_{b,k}\}$ . Some examples are given in Figure 1(c).

Since any log contains only a finite number of traces, one can never observe all traces possible in  $m_1$  or  $m_2$ .

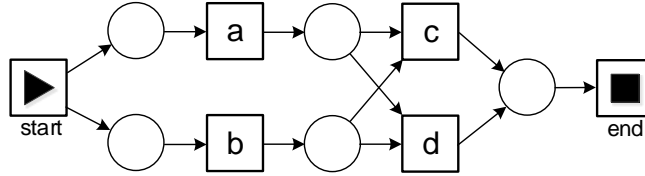


Fig. 2: A process model  $m_3$  discovered based on log  $l_3 = [\langle a, b, c \rangle^5, \langle b, a, d \rangle^3, \langle a, b, d \rangle^2]$ .

### 3.3 Process Discovery

A discovery algorithm takes an event log as input and returns a process model. For example, the model  $m_3$  in Figure 2 could have been discovered based on event log  $l_3 = [\langle a, b, c \rangle^5, \langle b, a, d \rangle^3, \langle a, b, d \rangle^2]$ . Ideally, the process model should capture the (dominant) behavior observed but it should also generalize without becoming too imprecise. For example, the model allows for trace  $t = \langle b, a, c \rangle$  although this was never observed.

**Definition 4 (Discovery Algorithm).** A discovery algorithm can be described as a function  $disc \in \mathcal{L} \rightarrow \mathcal{M}$  mapping event logs onto process models.

We abstract from concrete discovery algorithms. Over 100 discovery algorithms have been proposed in literature [1]. Merely as a reference to explain basic notions, we define three simple, but extreme, algorithms:  $disc_{ofit}$ ,  $disc_{ufit}$ , and  $disc_{nfit}$ . Let  $l \in \mathcal{L}$  be a log.  $disc_{ofit}(l) = m_o$  such that  $\tau(m_o) = \tau(l)$  produces an overfitting model that allows only for the behavior seen in the log.  $disc_{ufit}(l) = m_u$  such that  $\tau(m_u) = \mathcal{T}$  produces an underfitting model that allows for any behavior.  $disc_{nfit}(l) = m_n$  such that  $\tau(m_n) = \bar{\tau}(l)$  produces a non-fitting model that allows for all behavior *not* seen in the log.

## 4 Recall and Precision

Many recall measures have been proposed in literature [1,3,5,12,13,14,15,24,25,27,31,32]. In recent years, also several precision measures have been proposed [6,29]. Only few generalization measures have been proposed [3]. The goal of this paper is to evaluate these quality measures. To achieve this, in the following the propositions introduced in [2] are applied to existing conformance measures.

The notion of recall and precision are well established in the process mining community. Definitions are in place and there is an agreement on what these two measures are supposed to measure. However, this is not the case for generalization. There exist different points of view on what generalization is supposed to measure. Depending on these, existing generalization measures might greatly differ from each other.

To account for the different levels of maturity in recall, precision and generalization and to address the controversy in the generalization area, the following section will solely handle recall and precision while Section 5 focuses on generalization. Both

sections establish baseline measures, introduce the corresponding propositions of [2], present existing conformance measures and evaluate them using the propositions.

#### 4.1 Baseline Recall and Precision measures

We assume the existence of two functions:  $rec()$  and  $prec()$  respectively denoting recall and precision. Both take a log and model as input and return a value between 0 and 1. The higher the value, the better.

**Definition 5 (Recall).** A recall measure  $rec \in \mathcal{L} \times \mathcal{M} \rightarrow [0, 1]$  aims to quantify the fraction of observed behavior that is allowed by the model.

**Definition 6 (Precision).** A precision measure  $prec \in \mathcal{L} \times \mathcal{M} \rightarrow [0, 1]$  aims to quantify the fraction of behavior allowed by the model that was actually observed.

If we ignore frequencies of traces, we can simply count fractions of traces yielding the following two simple measures.

**Definition 7 (Trace-Based L2M Precision and Recall).** Let  $l \in \mathcal{L}$  and  $m \in \mathcal{M}$  be an event log and a process model. Trace-based L2M precision and recall are defined as follows:

$$rec_{TB}(l, m) = \frac{|\tau(l) \cap \tau(m)|}{|\tau(l)|} \quad prec_{TB}(l, m) = \frac{|\tau(l) \cap \tau(m)|}{|\tau(m)|} \quad (1)$$

Since  $|\tau(l)|$  is bounded by the size of the log,  $rec_{TB}(l, m)$  is well-defined. However,  $prec_{TB}(l, m)$  is undefined when  $\tau(m)$  is unbounded (e.g., in case of loops).

One can argue, that the frequency of traces should be taken into account when evaluating conformance which yields the following measure. Note that it is not possible to define frequency-based precision based on a process model that does not define the probability of its traces. Since probabilities are specifically excluded from the scope of this paper, the following approach only defines frequency-based recall.

**Definition 8 (Frequency-Based L2M Recall).** Let  $l \in \mathcal{L}$  and  $m \in \mathcal{M}$  be an event log and a process model. Frequency-based L2M recall is defined as follows:

$$rec_{FB}(l, m) = \frac{||[t \in l \mid t \in \tau(m)]||}{|l|} \quad (2)$$

#### 4.2 A Collection of Conformance Propositions

In [2], 21 *conformance propositions* covering the different conformance dimensions (except simplicity) were given. In this section, we focus on the general, recall and precision propositions introduced in [2]. We discuss the generalization propositions separately, because they reason about unseen cases not yet recorded in the event log. Most of the conformance propositions have broad support from the community, i.e., there is broad consensus that these propositions should hold. These are marked with a “+”. More controversial propositions are marked with a “0” (rather than a “+”).

**General Propositions** The first two propositions are commonly accepted; the computation of a quality measure should be deterministic (**DetPro**<sup>+</sup>) and only depend on behavioral aspects (**BehPro**<sup>+</sup>). The latter is a design choice. We deliberately exclude simplicity notions.

**Proposition 1 (DetPro**<sup>+</sup>). *rec(), prec(), gen() are deterministic functions, i.e., the measures rec(l, m), prec(l, m), gen(l, m) are fully determined by l ∈ ℒ and m ∈ ℳ.*

**Proposition 2 (BehPro**<sup>+</sup>). *For any l ∈ ℒ and m<sub>1</sub>, m<sub>2</sub> ∈ ℳ such that τ(m<sub>1</sub>) = τ(m<sub>2</sub>): rec(l, m<sub>1</sub>) = rec(l, m<sub>2</sub>), prec(l, m<sub>1</sub>) = prec(l, m<sub>2</sub>), and gen(l, m<sub>1</sub>) = gen(l, m<sub>2</sub>), i.e., the measures are fully determined by the behavior observed and the behavior described by the model (representation does not matter).*

**Recall Propositions** In this subsection, we consider a few *recall propositions*. *rec* ∈ ℒ × ℳ → [0, 1] aims to quantify the fraction of observed behavior that is allowed by the model. Proposition **RecPro1**<sup>+</sup> states that extending the model to allow for more behavior can never result in a lower recall. From the definition follows, that this proposition implies **BehPro**<sup>+</sup>. Recall measures violating **BehPro**<sup>+</sup> also violate **RecPro1**<sup>+</sup> which is demonstrated as follows:

For two models m<sub>1</sub>, m<sub>2</sub> with τ(m<sub>1</sub>) = τ(m<sub>2</sub>) it follows from **RecPro1**<sup>+</sup> that rec(l, m<sub>1</sub>) ≤ rec(l, m<sub>2</sub>) because τ(m<sub>1</sub>) ⊆ τ(m<sub>2</sub>). From **RecPro1**<sup>+</sup> follows that rec(l, m<sub>2</sub>) ≤ rec(l, m<sub>1</sub>) because τ(m<sub>2</sub>) ⊆ τ(m<sub>1</sub>). Combined, rec(l, m<sub>2</sub>) ≤ rec(l, m<sub>1</sub>) and rec(l, m<sub>1</sub>) ≤ rec(l, m<sub>2</sub>) gives rec(l, m<sub>2</sub>) = rec(l, m<sub>1</sub>), thus, recall measures that fulfill **RecPro1**<sup>+</sup> are fully determined by the behavior observed and the behavior described by the model, i.e., representation does not matter.

**Proposition 3 (RecPro1**<sup>+</sup>). *For any l ∈ ℒ and m<sub>1</sub>, m<sub>2</sub> ∈ ℳ such that τ(m<sub>1</sub>) ⊆ τ(m<sub>2</sub>): rec(l, m<sub>1</sub>) ≤ rec(l, m<sub>2</sub>).*

Similarly to **RecPro1**<sup>+</sup>, it cannot be the case that adding fitting behavior to the event logs, lowers recall (**RecPro2**<sup>+</sup>).

**Proposition 4 (RecPro2**<sup>+</sup>). *For any l<sub>1</sub>, l<sub>2</sub>, l<sub>3</sub> ∈ ℒ and m ∈ ℳ such that l<sub>2</sub> = l<sub>1</sub> ⊔ l<sub>3</sub> and τ(l<sub>3</sub>) ⊆ τ(m): rec(l<sub>1</sub>, m) ≤ rec(l<sub>2</sub>, m).*

Similarly to **RecPro2**<sup>+</sup>, one can argue that adding non-fitting behavior to event logs should not be able to increase recall (**RecPro3**<sup>0</sup>). However, one could also argue that recall should not be measured on a trace-level, but should instead distinguish between non-fitting traces by measuring the *degree* in which a non-fitting trace is still fitting. Therefore, unlike the previous propositions, this requirement is debatable as is indicated by the “0” tag.

**Proposition 5 (RecPro3**<sup>0</sup>). *For any l<sub>1</sub>, l<sub>2</sub>, l<sub>3</sub> ∈ ℒ and m ∈ ℳ such that l<sub>2</sub> = l<sub>1</sub> ⊔ l<sub>3</sub> and τ(l<sub>3</sub>) ⊆ τ(m): rec(l<sub>1</sub>, m) ≥ rec(l<sub>2</sub>, m).*

For any k ∈ ℕ: l<sup>k</sup>(t) = k · l(t), e.g., if l = [⟨a, b⟩<sup>3</sup>, ⟨c⟩<sup>2</sup>], then l<sup>4</sup> = [⟨a, b⟩<sup>12</sup>, ⟨c⟩<sup>8</sup>]. We use this notation to enlarge event logs without changing the original distribution. One could argue that this should not influence recall (**RecPro4**<sup>0</sup>), e.g., rec([⟨a, b⟩<sup>3</sup>,



$\langle c \rangle^2], m) = \text{rec}([\langle a, b \rangle^{12}, \langle c \rangle^8], m)$ . On the other hand, larger logs can provide more confidence that the log is indeed a representative sample of the possible behavior. Therefore, it is debatable whether the size of the event log should have influence on recall as indicated by the “0” tag.

**Proposition 6 (RecPro4<sup>0</sup>).** *For any  $l \in \mathcal{L}$ ,  $m \in \mathcal{M}$ , and  $k \geq 1$ :  $\text{rec}(l^k, m) = \text{rec}(l, m)$ .*

Finally, we provide a proposition stating that recall should be 1 if all traces in the log fit the model (**RecPro5<sup>+</sup>**). As a result, the empty log has recall 1 for any model. Based on this proposition,  $\text{rec}(l, \text{disc}_{\text{ofit}}(l)) = \text{rec}(l, \text{disc}_{\text{ufit}}(l)) = 1$  for any log  $l$ .

**Proposition 7 (RecPro5<sup>+</sup>).** *For any  $l \in \mathcal{L}$  and  $m \in \mathcal{M}$  such that  $\tau(l) \subseteq \tau(m)$ :  $\text{rec}(l, m) = 1$ .*

**Precision Propositions** Precision ( $\text{prec} \in \mathcal{L} \times \mathcal{M} \rightarrow [0, 1]$ ) aims to quantify the fraction of behavior allowed by the model that was actually observed. Initial work in the area of checking requirements of conformance checking measures started with [29], where five axioms for precision measures were introduced. The precision propositions that we state below partly overlap with these axioms, but some have been added and some have been strengthened. Axiom 1 of [29] specifies **DetPro<sup>+</sup>** for the case of precision, while we have generalized it to the recall and generalization dimension. Furthermore, **BehPro<sup>+</sup>** generalizes axiom 4 of [29] from its initial focus on precision to also cover recall and generalization. **PrecPro1<sup>+</sup>** states that removing behavior from a model that does not happen in the event log cannot lead to a lower precision. From the definition follows, that this proposition implies **BehPro<sup>+</sup>**. Precision measures violating **BehPro<sup>+</sup>** also violate **PrecPro1<sup>+</sup>**. Adding fitting traces to the event log can also not lower precision (**PrecPro2<sup>+</sup>**). However, adding non-fitting traces to the event log should not change precision (**PrecPro3<sup>0</sup>**).

**Proposition 8 (PrecPro1<sup>+</sup>).** *For any  $l \in \mathcal{L}$  and  $m_1, m_2 \in \mathcal{M}$  such that  $\tau(m_1) \subseteq \tau(m_2)$  and  $\tau(l) \cap (\tau(m_2) \setminus \tau(m_1)) = \emptyset$ :  $\text{prec}(l, m_1) \geq \text{prec}(l, m_2)$ .*

This proposition captures the same idea as axiom 2 in [29], but it is more general. Axiom 2 only put this requirement on precision when  $\tau(l) \subseteq \tau(m_1)$ , while **PrecPro1<sup>+</sup>** also concerns the situation where this does not hold.

**Proposition 9 (PrecPro2<sup>+</sup>).** *For any  $l_1, l_2, l_3 \in \mathcal{L}$  and  $m \in \mathcal{M}$  such that  $l_2 = l_1 \uplus l_3$  and  $\tau(l_3) \subseteq \tau(m)$ :  $\text{prec}(l_1, m) \leq \text{prec}(l_2, m)$ .*

This proposition is identical to axiom 5 in [29].

**Proposition 10 (PrecPro3<sup>0</sup>).** *For any  $l_1, l_2, l_3 \in \mathcal{L}$  and  $m \in \mathcal{M}$  such that  $l_2 = l_1 \uplus l_3$  and  $\tau(l_3) \subseteq \bar{\tau}(m)$ :  $\text{prec}(l_1, m) = \text{prec}(l_2, m)$ .*

One could also argue that duplicating the event log should not influence precision because the distribution remains the same (**PrecPro4<sup>0</sup>**), e.g.,  $\text{prec}([\langle a, b \rangle^{20}, \langle c \rangle^{20}], m) = \text{prec}([\langle a, b \rangle^{40}, \langle c \rangle^{40}], m)$ . Similar to (**RecPro3<sup>0</sup>**) and (**RecPro4<sup>0</sup>**), the equivalents on the precision side are tagged with “0”.

**Proposition 11 (PrecPro4<sup>0</sup>).** For any  $l \in \mathcal{L}$ ,  $m \in \mathcal{M}$ , and  $k \geq 1$ :  $prec(l^k, m) = prec(l, m)$ .

If the model allows for the behavior observed and nothing more, precision should be maximal (**PrecPro5<sup>+</sup>**). One could also argue that if all modeled behavior was observed, precision should also be 1 (**PrecPro6<sup>0</sup>**). The latter proposition is debatable because it implies that the non-fitting behavior cannot influence perfect precision, as indicated by the “0” tag. Consider for example extreme cases where the model covers just a small fraction of all observed behavior (or even more extreme situations like  $\tau(m) = \emptyset$ ). According to **PrecPro5<sup>+</sup>** and **PrecPro6<sup>0</sup>**,  $rec(l, disc_{ofit}(l)) = 1$  for any log  $l$ .

**Proposition 12 (PrecPro5<sup>+</sup>).** For any  $l \in \mathcal{L}$  and  $m \in \mathcal{M}$  such that  $\tau(m) = \tau(l)$ :  $prec(l, m) = 1$ .

**Proposition 13 (PrecPro6<sup>0</sup>).** For any  $l \in \mathcal{L}$  and  $m \in \mathcal{M}$  such that  $\tau(m) \subseteq \tau(l)$ :  $prec(l, m) = 1$ .

### 4.3 Evaluation of Baseline Conformance Measures

To illustrate the presented propositions and justify their formulation, we evaluate the conformance measures defined as baselines in Section 4.1. Note that these 3 baseline measures were introduced to provide simple examples that can be used to discuss the propositions. We conduct this evaluation under the assumption that  $l \neq []$ ,  $\tau(m) \neq \emptyset$  and  $\langle \rangle \notin \tau(m)$ . Table 5 in Appendix A.1 summarizes the evaluation.

**General Propositions.** Based on the definition of  $rec_{TB}$  and  $rec_{FB}$  it is clear that all measures can be fully determined by the log and the model. Consequently, **DetPro<sup>+</sup>** hold for these two baseline conformance measures. However,  $prec_{TB}$  is undefined when  $\tau(m)$  is unbound and, therefore, non-deterministic.

The behavior of the model is defined as sets of traces  $\tau(m)$ , which abstracts from the representation of the process model itself. Therefore, all recall and precision baseline conformance measures fulfill **BehPro<sup>+</sup>**.

**Recall Propositions.** Considering measure  $rec_{TB}$ , it is obvious that **RecPro1<sup>+</sup>** holds if  $\tau(m_1) \subseteq \tau(m_2)$ , because the intersection between  $\tau(m_2)$  and  $\tau(l)$  will always be equal or bigger to the intersection of  $\tau(m_1)$  and  $\tau(l)$ . The **RecPro2<sup>+</sup>** proposition holds for  $rec_{TB}$ , if  $l_2 = l_1 \uplus l_3$  and  $\tau(l_3) \subseteq \tau(m)$ , because the additional fitting behavior is added to the nominator as well as the denominator of the formula:  $|\tau(l_1) \cap \tau(m)| + |\tau(l_3)| / (|\tau(l_1)| + |\tau(l_3)|)$ . This can never decrease recall. Furthermore, **RecPro3<sup>0</sup>** propositions holds for  $rec_{TB}$  since adding unfitting behavior cannot increase the intersection between traces of the model and the log if  $l_2 = l_1 \uplus l_3$  and  $\tau(l_3) \subseteq \bar{\tau}(m)$ . Consequently, only the denominator of the formula grows, which decreases recall. Similarly, we can show that these two proposition hold for  $rec_{FB}$ .

Duplication of the event log cannot affect  $rec_{TB}$ , since it is defined based on the set of traces and not the multiset. The proposition also holds for  $rec_{FB}$  since nominator and denominator of the formula will grow in proportion. Hence, **RecPro4<sup>0</sup>** holds for both

baseline measures. Considering  $rec_{TB}$ , **RecPro5<sup>+</sup>** holds, since  $\tau(l) \cap \tau(m) = \tau(l)$  if  $\tau(l) \subseteq \tau(m)$  and consequently  $|\tau(l) \cap \tau(m)| / |\tau(l)| = |\tau(l)| / |\tau(l)| = 1$ . The same conclusions can be drawn for  $rec_{FB}$ .

**Precision Propositions.** Consider proposition **PrecPro1<sup>+</sup>** together with  $prec_{TB}$ . The proposition holds, since removing behavior from the model that does not happen in the event log will not affect the intersection between the traces of the model and the log:  $\tau(l) \cap \tau(m_2) = \tau(l) \cap \tau(m_1)$  if  $\tau(m_1) \subseteq \tau(m_2)$  and  $\tau(l) \cap (\tau(m_2) \setminus \tau(m_1)) = \emptyset$ . At the same time the denominator of the formula decreases, which can never decrease precision itself. **PrecPro2<sup>+</sup>** also holds for  $prec_{TB}$ , since the fitting behavior increases the intersection between traces of the model and the log, while the denominator of the formula stays the same. Furthermore, **PrecPro3<sup>0</sup>** holds for  $prec_{TB}$ , since unfitting behavior cannot affect the intersection between traces of the model and the log.

Duplication of the event log cannot affect  $prec_{TB}$ , since it is defined based on the set of traces and not the multiset, i.e. **PrecPro4<sup>0</sup>** holds.

Considering  $prec_{TB}$ , **PrecPro5<sup>+</sup>** holds, since  $\tau(l) \cap \tau(m) = \tau(m)$  if  $\tau(m) = \tau(l)$  and consequently  $|\tau(l) \cap \tau(m)| / |\tau(m)| = |\tau(m)| / |\tau(m)| = 1$ . Similarly, **PrecPro6<sup>0</sup>** holds for  $prec_{TB}$ .

#### 4.4 Existing Recall Measures

The previous evaluation of the simple baseline measures shows that the recall measures fulfill all propositions and the baseline precision measure only violates one proposition. However, the work presented in [29] demonstrated for precision, that most of the existing approaches violate seemingly obvious requirements. This is surprising compared to the results of our simple baseline measure. Inspired by [29], this paper takes a broad look at existing conformance measures with respect to the previously presented propositions. In the following section, existing recall and precision measures are introduced, before they will be evaluated in Section 4.6.

**Causal footprint recall ( $rec_A$ ).** Van der Aalst et al. [4] introduce the concept of the footprint matrix, which captures the relations between the different activities in the log. The technique relies on the principle that if activity  $a$  is followed by  $b$  but  $b$  is never followed by  $a$ , then there is a causal dependency between  $a$  and  $b$ . The log can be described using four different relations types. In [1] it is stated that a footprint matrix can also be derived for a process model by generating a complete event log from it. Recall can be measured by counting the mismatches between both matrices. Note that this approach assumes an event log which is complete with respect to the directly follows relations.

**Token replay recall ( $rec_B$ ).** Token replay measures recall by replaying the log on the model and counting mismatches in the form of missing and remaining tokens. This approach was proposed by Rozinat and van der Aalst [27]. During replay, four types of tokens are distinguished:  $p$  the number of *produced* tokens,  $c$  the number of *consumed* tokens,  $m$  the number of *missing* tokens that had to be added because a transition was

not enabled during replay and  $r$  the number of *remaining* tokens that are left in the model after replay. In the beginning, a token is produced in the initial place. Similarly, the approach ends by consuming a token from the final place. The more missing and remaining tokens are counted during replay the lower recall:  $rec_B = \frac{1}{2}(1 - \frac{m}{c}) + \frac{1}{2}(1 - \frac{r}{p})$ . Note that the approach assumes a relaxed sound workflow net, but it allows for duplicate and silent transitions.

**Alignment recall ( $rec_C$ ).** Another approach to determine recall was proposed by van der Aalst et al. [3]. It calculates recall based on alignments, which detect process deviations by mapping the steps taken in the event log to the ones of the process model. This map can contain three types of steps (so-called moves): *synchronous* moves when event log and model agree, *log* moves if the event was recorded in the event log but should not have happened according to the model and *model* moves if the event should have happened according to the model but did not in the event log. The approach uses a function that assigns costs to log moves and model moves. This function is used to compute the optimal alignment for each trace in the log (i.e. the alignment with the least cost associated).

To compute recall, the total alignment cost of the log is normalized with respect to the cost of the worst-case scenario where there are only moves in the log and in the model but never together. Note, that the approach assumes an accepting Petri net with an initial and final state. However, it allows for duplicate and silent transitions in the process model.

**Behavioral recall ( $rec_D$ ).** Goedertier et al. [16] define recall according to its definition in the data mining field using true positive (TP) and false negative (FN) counters.  $TP(l, m)$  denotes the number of true positives, i.e., the number of events in the log that can be parsed correctly in model  $m$  by firing a corresponding enabled transition.  $FN(l, m)$  denotes the number of false negatives, i.e., the number of events in the log for which the corresponding transition that was needed to mimic the event was not enabled and needed to be force-fired. The recall measure is defined as follows:

$$rec_D(l, m) = \frac{TP(l, m)}{TP(l, m) + FN(l, m)}.$$

**Projected recall ( $rec_E$ ).** Leemans et al. [23] developed a conformance checking approach that is also able to handle big event logs. This is achieved by projecting the event log as well as the model on all possible subsets of activities of size  $k$ . The behavior of a projected log and projected model is translated into the minimal deterministic finite automata (DFA)<sup>4</sup>. Recall is calculated by checking the fraction of the behavior that is allowed for by the minimal log-automaton that is also allowed for by the minimal model-automaton for each projection and by averaging the recall over each projection.

**Continued parsing measure ( $rec_F$ ).** This continued parsing measure was developed in the context of the heuristic miner by Weijters et al. [33]. It abstracts from the representation of the process model by translating the Petri net into a causal matrix. This

<sup>4</sup> Every regular language has a unique minimal DFA according to the Myhill–Nerode theorem.

matrix defines input and output expressions for each activity, which describe possible in- and output behavior. When replaying the event log on the causal matrix, one has to check whether the corresponding input and output expressions are activated and therefore enable the execution of the activity. To calculate the continued parsing measure the number of events  $e$  in the event log, as well as the number of missing activated input expressions  $m$  and remaining activated output expressions  $r$  are counted. Note, that the approach allows for silent transitions in the process model but excludes duplicate transitions.

**Eigenvalue recall ( $rec_G$ ).** Polyvyanyy et al. [26] introduce a framework for the definition of language quotients that guarantee several properties similar to the propositions introduced in [2]. To illustrate this framework, they apply it in the process mining context and define a recall measure. Hereby they rely on the relation between the language of a deterministic finite automaton (DFA) that describes the behavior of the model and the language of the log. In principle, recall is defined as in Definition 7. However, the measure is undefined if the language of the model or the log are infinite. Therefore, instead of using the cardinality of the languages and their intersection, the measure computes their corresponding eigenvalues and sets them in relation. To compute these eigenvalues, the languages have to be irreducible. Since this is not the case for the language of event logs, Polyvyanyy et al. [26] introduce a short-circuit measure over languages and proved that it is a deterministic measure over any arbitrary regular language.

#### 4.5 Existing Precision Measures

**Soundness ( $prec_H$ ).** The notion of soundness as defined by Greco et al. [17] states that a model is precise if all possible enactments of the process have been observed in the event log. Therefore, it divides the number of unique traces in the log compliant with the process model by the number of unique traces through the model. Note, that this approach assumes the process model in the shape of a workflow net. Furthermore, it is equivalent to the baseline precision measure  $prec_TB$ .

**Simple behavioral appropriateness ( $prec_I$ ).** Rozinat and van der Aalst [27] introduce simple behavioral appropriateness to measure the precision of process models. The approach assumes that imprecise models enable a lot of transitions during replay. Therefore, the approach computes the mean number of enabled transitions  $x_i$  for each unique trace  $i$  and puts it in relation to the visible transitions  $T_V$  in the process model. Note, that the approach assumes a sound workflow net. However, it allows for duplicate and silent transitions in the process model.

**Advanced behavioral appropriateness ( $prec_J$ ).** In the same paper, Rozinat and van der Aalst [27] define advanced behavioral appropriateness. This approach abstracts from the process model by describing the relation between activities of both the log and model with respect to whether these activities *follow* and/or *precede* each other. Hereby

they differentiate between *never*, *sometimes* and *always* precede/follow relations. To calculate precision the set of sometimes followed relations of the log  $S_F^l$  and the model  $S_P^m$  are considered, as well as their sometimes precedes relations  $S_P^l$  and  $S_P^m$ . The fraction of sometimes follows/precedes relations of the model which are also observed by the event log defines precision. Note, that the approach assumes a sound workflow net. However, it allows for duplicate and silent transitions in the process model.

**ETC-one/ETC-rep ( $prec_K$ ) and ETC-all ( $prec_L$ ).** Munoz-Gama and Carmona [25] introduced a precision measure which constructs an automaton that reflects the states of the model which are visited by the event log. For each state, it is evaluated whether there are activities which were allowed by the process model but not observed by the event log. These activities are added to the automaton as so-called escaping edges. Since this approach is not able to handle unfitting behavior, [6] and [3] extended the approach with a preprocessing step that aligned the log to the model before the construction of the automaton. Since it is possible that traces result in multiple optimal alignments, there are three variations of the precision measure. One can randomly pick one alignment and construct the alignment automaton based on it (ETC-one), select a representative set of multiple alignments (ETC-rep) or use all optimal alignments (ETC-all). For each variation, [3] defines an approach that assigns appropriate weights to the edges of the automaton. Precision is then computed by comparing for each state of the automaton, the weighted number of non-escaping edges to the total number of edges.

**Behavioral specificity ( $prec_M$ ) and Behavioral precision ( $prec_N$ ).** Goedertier et al. [16] introduced a precision measure based on the concept of negative events that is defined based on the concept of a confusion matrix as used in the data mining field. In this confusion matrix, the induced negative events are considered to be the ground truth and the process model is considered to be a prediction machine that predicts whether an event can or cannot occur. A negative event expresses that at a certain position in a trace, a particular event cannot occur. To induce the negative events into an event log, the traces are split in subsequences of length  $k$ . For each event  $e$  in the trace, it is checked whether another event  $e_n$  could be a negative event. Therefore the approach searches whether the set of subsequences contains a similar sequence to the one preceding  $e$ . If no matching sequence is found that contains  $e_n$  at the current position of  $e$ ,  $e_n$  is recorded as a negative event of  $e$ . To check conformance the log, that was induced with negative events, is replayed on the model.

For both measures, the log that was induced with negative events is replayed on the model. Specificity and precision are measured according to their data mining definition using true positive (TP), false positive (FP) and true negative (TN) counts.

Goedertier et al. [16] ( $prec_M$ ) defined behavioral specificity precision as  $prec_M(l, m) = \frac{TN(l, m)}{TN(l, m) + FP(l, m)}$ , i.e., the ratio of the induced negative events that were also disallowed by  $m$ . More recently, De Weerd et al. [32] gave an inverse definition, called behavioral precision ( $prec_N$ ), as the ratio of behavior that is allowed by  $m$  that does not conflict an induced negative event, i.e.  $prec_N(l, m) = \frac{TP(l, m)}{TP(l, m) + FP(l, m)}$ .

**Weighted negative event precision ( $prec_O$ ).** Van den Broucke et al. [30] proposed an improvement to the approach of Goedertier et al. [16], which assigns weights to negative events. These weights indicate the confidence of the negative events actually being negative. To compute the weight, the approach takes the sequence preceding event  $e$  and searches for the matching subsequences in the event log. All events that have never followed such a subsequence are identified as negative events for  $e$  and their weight is computed based on the length of the matching subsequence. To calculate precision the enhanced log is replayed on the model, similar to the approach introduced in [32]. However, instead of increasing the counters by 1 they are increased by the weight of the negative event. Furthermore, van den Broucke et al. [30] also introduced a modified trace replay procedure which finds the best fitting firing sequence of transitions, taking force firing of transitions as well as paths enabled by silent transitions into account.

**Projected precision ( $prec_P$ ).** Along with projected recall ( $rec_E$ ) Leemans et al. [23] introduce projected precision. To compute precision, the approach creates a DFA which describes the conjunction of the behavior of the model and the event log. The number of outgoing edges of  $DFA(m|_A)$  and the conjunctive automaton  $DFAc(l, m, A)$  are compared. Precision is calculated for each subset of size  $k$  and averaged over the number of subsets.

**Anti-alignment precision ( $prec_Q$ ).** Van Dongen et al. [13] propose a conformance checking approach based on anti-alignments. An anti-alignment is a run of a model which differs from all the traces in a log. The principle of the approach assumes that a very precise model only allows for the observed traces and nothing more. If one trace is removed from the log, it becomes the anti-alignment for the remaining log.

Therefore, trace-based precision computes an anti-alignment for each trace in the log. Then the distance  $d$  between the anti-alignment and the trace  $\sigma$  is computed. This is summed up for each trace and averaged over the number of traces in the log. The more precise a model, the lower the distance. However, the anti-alignment used for trace-based precision is limited by the length of the removed trace  $|\sigma|$ . Therefore, log-based precision uses an anti-alignment between the model and the complete log which has a length which is much greater than the traces observed in the log. Anti-alignment precision is the weighted combination of trace-based and log-based anti-alignment precision. Note, that the approach allows for duplicate and silent transitions in the process model.

**Eigenvalue precision ( $prec_R$ ).** Polyvyanyy et al. [26] also define a precision measure along with the Eigenvalue recall ( $rec_G$ ). For precision, they rely on the relation between the language of a deterministic finite automaton (DFA) that describes the behavior of the model and the language of the log. To overcome the problems arising with infinite languages of the model or log, they compute their corresponding eigenvalues and set them in relation. To compute these eigenvalues, the languages have to be irreducible. Since this is not the case for the language of event logs, Polyvyanyy et al. [26] introduce a short-circuit measure over languages and prove that it is a deterministic measure over any arbitrary regular language.

Table 1: Overview of the recall propositions that hold for the existing measures (under the assumption that  $l \neq []$ ,  $\tau(m) \neq \emptyset$  and  $\langle \rangle \notin \tau(m)$ ):  $\checkmark$  means that the proposition holds for any log and model and  $\times$  means that the proposition does not always hold.

Proposition	Name	$rec_A$	$rec_B$	$rec_C$	$rec_D$	$rec_E$	$rec_F$	$rec_G$
1	<b>DetPro</b> <sup>+</sup>	$\checkmark$	$\times$	$\checkmark$	$\times$	$\checkmark$	$\times$	$\checkmark$
2	<b>BehPro</b> <sup>+</sup>	$\checkmark$	$\times$	$\checkmark$	$\times$	$\checkmark$	$\checkmark$	$\checkmark$
3	<b>RecPro1</b> <sup>+</sup>	$\times$	$\times$	$\checkmark$	$\times$	$\checkmark$	$\checkmark$	$\checkmark$
4	<b>RecPro2</b> <sup>+</sup>	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$
5	<b>RecPro3</b> <sup>0</sup>	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\checkmark$
6	<b>RecPro4</b> <sup>0</sup>	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$
7	<b>RecPro5</b> <sup>+</sup>	$\times$	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$	$\times$	$\checkmark$

#### 4.6 Evaluation of Existing Recall and Precision Measures

Several of the existing precision measures are not able to handle non-fitting behavior and remove it by aligning the log to the model. We use a baseline approach for the alignment, which results in a deterministic event log:  $l$  is the original event log, which is aligned in a deterministic manner. The resulting event log  $l'$  corresponds to unique paths through the model. We use  $l'$  to evaluate the propositions.

**Evaluation of Existing Recall Measures** The previously presented recall measures are evaluated using the corresponding propositions. The results of the evaluation are displayed in Table 1. To ensure the readability of this paper, only the most interesting findings of the evaluation are addressed in the following section. For full details refer to Appendix A.2.

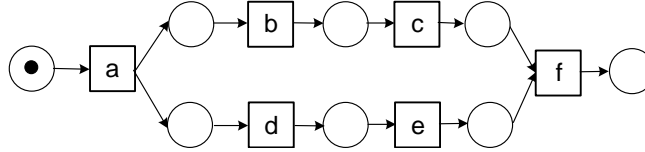


Fig. 3: A process model  $m_4$ .

The evaluation of the *causal footprint recall measure* ( $rec_A$ ) showed that it is deterministic and solely relies on the behavior of the process model. However, the measure violates several propositions such as **RecPro1**<sup>+</sup>, **RecPro3**<sup>0</sup>, and **RecPro5**<sup>+</sup>. These violations are caused by the fact that recall records every difference between the footprint of the log and the model. Behavior that is described by the model but is not observed in the event log has an impact on recall, although Definition 5 states otherwise. To illustrate this, consider  $m_4$  in Figure 3, event log  $l_4 = [\langle a, b, c, d, e, f \rangle, \langle a, b, d, c, e, f \rangle]$  and **RecPro5**<sup>+</sup>. The traces in  $l_4$  perfectly fit process model  $m_4$ . The footprint of  $l_4$  is shown



Table 2: The causal footprints of  $m_4$  (a),  $l_4$  (b). Mismatching relations are marked in red.

(a)						(b)					
a	b	c	d	e	f	a	b	c	d	e	f
a	#	→	#	→	# #	a	#	→	#	#	# #
b	←	#	→		#	b	←	#	→	→	# #
c	#	←	#		→	c	#	←	#		→ #
d	←			#	→ #	d	#	←		#	→ #
e	#			←	# →	e	#	#	←	←	# →
f	#	#	←	#	← #	f	#	#	#	#	← #

in Table 2 (b). Comparing it to the footprint of  $m_4$  in Table 2 (a) shows mismatches although  $l_4$  is perfectly fitting. These mismatches are caused by the fact that the log does not show all possible behavior of the model and, therefore, the footprint cannot completely detect the parallelism of the model. Consequently 10 of 36 relations of the footprint represent mismatches:  $rec_A(l_4, m_4) = 1 - \frac{10}{36} = 0.72 \neq 1$ . Van der Aalst mentions in [1] that checking conformance using causal footprints is only meaningful if the log is complete in term of directly followed relations. Moreover, the measure also includes precision and generalization aspects, next to recall.

In comparison, recall based on *token replay* ( $rec_B$ ) depends on the path taken through the model. Due to duplicate activities and silent transitions, multiple paths through a model can be taken when replaying a single trace. Different paths can lead to different numbers of produced, consumed, missing and remaining tokens. Therefore, the approach is neither deterministic nor independent from the structure of the process model and, consequently, violates **RecPro1**<sup>+</sup>. The *continued parsing measure* ( $rec_F$ ) builds on a similar replay principle as token-based replay and also violates **DetPro**<sup>+</sup>. However, the approach translates the process model into a causal matrix and is therefore independent of its structure.

Table 1 also shows that most measures violate **RecPro3**<sup>0</sup>. This is caused by the fact, that we define non-fitting behavior in this paper on a trace level: traces either fit the model or they do not. However, the evaluated approaches measure non-fitting behavior on an event level. A trace consists of fitting and non-fitting events. In cases where the log contains traces with a large number of deviating events, recall can be improved by adding non-fitting traces which contain several fitting and only a few deviating events. To illustrate this, consider *token replay* ( $rec_B$ ), process model  $m_5$  in Figure 4,  $l_5 = \langle a, b, f, g \rangle$  and  $l_6 = l_5 \uplus \langle a, d, e, f, g \rangle$ . The log  $l_5$  is not perfectly fitting and replaying it on the model results in 6 produced and 6 consumed tokens, as well as 1 missing and 1 remaining token.  $rec_B(l_5, m_5) = \frac{1}{2}(1 - \frac{1}{6}) + \frac{1}{2}(1 - \frac{1}{6}) = 0.833$ . Event log  $l_6$  was created by adding non-fitting behavior to  $l_5$ . Replaying  $l_6$  on  $m_5$  results in  $p = c = 13$ ,  $r = m = 2$  and  $rec_B(l_6, m_6) = \frac{1}{2}(1 - \frac{2}{13}) + \frac{1}{2}(1 - \frac{2}{13}) = 0.846$ . Hence, the additional unfitting trace results in proportionally more fitting events than deviating ones which improves recall:  $rec_B(l_6, m_6) > rec_B(l_5, m_5)$ .

To overcome the problems arising with the differences between trace-based and event-based fitness, one could alter the definition of **RecPro3**<sup>0</sup> by requiring, that the

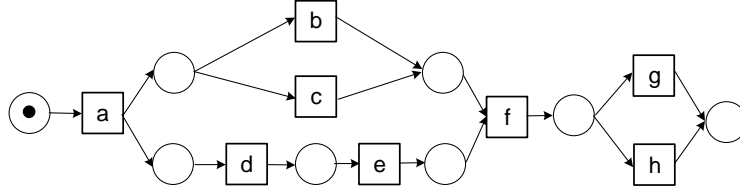


Fig. 4: Petri net  $m_5$

Table 3: Overview of the precision propositions that hold for the existing measures (under the assumption that  $l \neq []$ ,  $\tau(m) \neq \emptyset$  and  $\langle \rangle \notin \tau(m)$ ):  $\checkmark$  means that the proposition holds for any log and model and  $\times$  means that the proposition does not always hold.

Prop.	Name	$prec_H$	$prec_I$	$prec_J$	$prec_K$	$prec_L$	$prec_M$	$prec_N$	$prec_O$	$prec_P$	$prec_Q$	$prec_R$
1	<b>DetPro</b> <sup>+</sup>	$\times$	$\times$	$\times$	$\times$	$\checkmark$	$\times$	$\times$	$\times$	$\checkmark$	$\checkmark$	$\checkmark$
2	<b>BehPro</b> <sup>+</sup>	$\checkmark$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\checkmark$	$\checkmark$	$\checkmark$
8	<b>PrecPro1</b> <sup>+</sup>	$\checkmark$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\checkmark$	$\checkmark$
9	<b>PrecPro2</b> <sup>+</sup>	$\checkmark$	$\times$	$\checkmark$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\checkmark$
10	<b>PrecPro3</b> <sup>0</sup>	$\checkmark$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\checkmark$
11	<b>PrecPro4</b> <sup>0</sup>	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$
12	<b>PrecPro5</b> <sup>+</sup>	$\checkmark$	$\times$	$\checkmark$	$\times$	$\times$	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$
13	<b>PrecPro6</b> <sup>0</sup>	$\checkmark$	$\times$	$\checkmark$	$\times$	$\times$	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$

initial log  $l_1$  only contains fitting behavior ( $\tau(l_1) \subseteq \tau(m)$ ). However, to stay within the scope of this paper, we decide to use the propositions as defined in [2] and keep this suggestion for future work.

**Evaluation of Existing Precision Measures** The previously presented precision measures are evaluated using the corresponding propositions. The results of the evaluation are displayed in Table 3. To ensure the readability of this paper, only the most interesting findings of the evaluation are addressed in the following section. For full details, we refer to Appendix A.3.

The evaluation showed that several measures violate the determinism **DetPro**<sup>+</sup> proposition. For example, the soundness measure ( $prec_H$ ) solely relies on the number of unique paths of the model  $|\tau(m)|$  and unique traces in the log that comply with the process model  $|\tau(l) \cap \tau(m)|$ . Hence, precision is not defined if the model has infinite possible paths. Additionally to **DetPro**<sup>+</sup>, behavioral specificity ( $rec_M$ ) and behavioral precision ( $rec_N$ ) also violate **BehPro**<sup>+</sup>. If during the replay of the trace duplicate or silent transitions are encountered, the approach explored which of the available transitions enables the next event in the trace. If no solution is found, one of the transitions is randomly fired, which can lead to different recall values for traces with the same behavior.

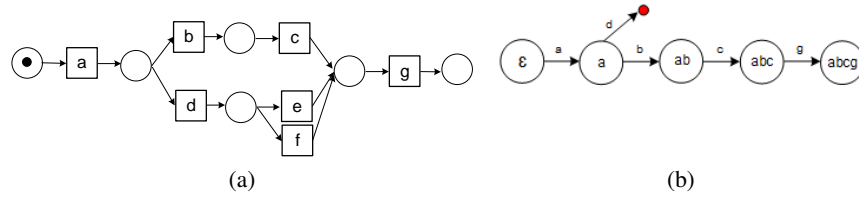


Fig. 5: Petri net  $m_6$  (a) and the alignment automaton describing the state space of  $\sigma = \langle a, b, c, g \rangle$  (b)

Table 3 shows that simple behavioral appropriateness ( $prec_I$ ) violates all but one of the propositions. One of the reason is that it relies on the average number of enabled transitions during replay. Even when the model allows for all exactly observed behavior (and nothing more), precision is not maximal when the model is not strictly sequential. Advanced behavioral appropriateness ( $prec_J$ ) overcomes these problems by relying on follow relations. However, it is not deterministic and depends on the structure of the process model.

The results presented in [29] show that ETC precision ( $prec_K$  and  $prec_L$ ), weighted negative event precision ( $prec_O$ ) and projected precision ( $prec_P$ ) violate **PrecPro1**<sup>+</sup>. Additionally, all remaining measures aside from anti-alignment precision ( $prec_Q$ ) and eigenvalue precision ( $prec_R$ ) violate the proposition. The proposition states that removing behavior from a model that does not happen in the event log cannot lower precision. Consider, projected precision ( $prec_P$ ) and a model with a length-one-loop. We remove behavior from the model by restricting the model to only execute the looping activity twice. This changes the DFA of the model since future behavior now depends on how often the looping activity was executed: the DFA contains different states for each execution. If these states show a low local precision, overall precision decreases. Furthermore, [29] showed that ETC precision ( $prec_K$  and  $prec_L$ ), projected precision ( $prec_P$ ) and anti-alignment precision ( $prec_Q$ ) also violate **PrecPro2**<sup>+</sup>.

In general, looking at Table 3 shows that all precision measures, except for soundness ( $prec_H$ ) and eigenvalue precision ( $prec_R$ ) violate **PrecPro3**<sup>0</sup>, which states that adding unfitting behavior to the event log should not change precision. However, for example, all variations of the ETC-measure ( $prec_K$ ,  $prec_L$ ) align the log before constructing the alignment automaton. Unfitting behavior can be aligned to a trace that was not seen in the log before and introduce new states to the automaton. Consider process model  $m_6$ , together with trace  $\sigma = \langle a, b, c, g \rangle$  and its alignment automaton displayed in Figure 5. Adding the unfitting trace  $\langle a, d, g \rangle$  could result in the aligned trace  $\langle a, d, e, g \rangle$  or  $\langle a, d, f, g \rangle$ . Both aligned traces introduce new states into the alignment automaton, alter the weights assigned to each state and, consequently, change precision. Weighted negative precision ( $prec_O$ ) also violates this proposition. The measure accounts for the number of negative events that actually could fire during trace replay (FP). These false positives are caused by behavior that is shown in the model but not observed in the log. As explained in the context of **RecPro3**<sup>0</sup>, although the trace is not

fitting when considered as a whole, certain parts of the trace can fit the model. These parts can possibly represent the previously missing behavior in the event log that leads to the wrong classification of negative events. Adding these traces will, therefore, lead to a decrease in false positives and changes precision.  $FP(l_1, m) > FP(l_2, m)$  and  $\frac{TP(l_1, m)}{(TP(l_1, m) + FP(l_1, m))} < \frac{TP(l_2, m)}{(TP(l_2, m) + FP(l_2, m))}$ .

Table 3 shows that  $prec_I$ ,  $prec_K$  and  $prec_L$  violate proposition **PrecPro6**<sup>0</sup>, which states that if all modeled behavior was observed, precision should be maximal and unfitting behavior cannot effect precision.  $prec_I$  only reports maximal precision if the model is strictly sequential and both ETC measures ( $prec_K$  and  $prec_L$ ) can run into problems with models containing silent or duplicate transitions.

The ETC ( $prec_K$ ,  $prec_L$ ) and anti-alignment measures ( $prec_Q$ ) form a special group of measures as they are unable to handle unfitting behavior without pre-processing unfitting traces and aligning them to the process model. Accordingly, we evaluate the conformance measure based on this aligned log. The evaluation of **PrecPro3**<sup>0</sup> and the ETC measure ( $prec_K$ ,  $prec_L$ ) is an example of the alignment of the log resulting in a violation. However, there are also cases where the proposition only holds because of this alignment. Consider, for example, anti-alignment precision ( $prec_Q$ ) and proposition **PrecPro6**<sup>0</sup>. By definition, an anti-alignment will always fit the model. Consequently, when computing the distance between the unfitting trace and the anti-alignment it will never be minimal. However, after aligning the log, it exactly contains the modeled behavior, precision is maximal and the proposition holds.

## 5 Generalization

Generalization is a challenging concept to define, in contrast to recall and precision. As a result, there are different viewpoints within the process mining community on what generalization precisely means. The main reason for this is, that generalization needs to reason about behavior that was not observed in the event log and establish its relation to the model.

The need for a generalization dimension stems from the fact that, given a log, a model can be fitting and precise, but be overfitting. The algorithm that simply creates a model  $m$  such that  $\tau(m) = \{t \in l\}$  is useless because it is simply enumerating the event log. Consider an unknown process. Assume we observe the first four traces  $l_1 = [\langle a, b, c \rangle, \langle b, a, c \rangle, \langle a, b, d \rangle, \langle b, a, d \rangle]$ . Based on this we may construct the model  $m_3$  in Figure 2 with  $\tau(m_3) = \{\langle a, b, c \rangle, \langle b, a, c \rangle, \langle a, b, d \rangle, \langle b, a, d \rangle\}$ . This model allows for all the traces in the event log and nothing more. However, because the real underlying process is unknown, this model may be overfitting event log  $l_1$ . Based on just four example traces we cannot be confident that the model  $m_3$  in Figure 2 will be able to explain future behavior of the process. The next trace may as well be  $\langle a, c \rangle$  or  $\langle a, b, b, c \rangle$ . Now assume that we observe the same process for a longer time and consider the first 100 traces (including the initial four):  $l_2 = [\langle a, b, c \rangle^{25}, \langle b, a, c \rangle^{25}, \langle a, b, d \rangle^{25}, \langle b, a, d \rangle^{25}]$ . After observing 100 traces, we are more confident that model  $m_3$  in Figure 2 is the right model. Intuitively, the probability that the next case will have a trace not allowed by  $m_3$  gets smaller. Now assume that we observe the same process for an even longer time and obtain the event log  $l_2 = [\langle a, b, c \rangle^{53789}, \langle b, a, c \rangle^{48976}, \langle a, b, d \rangle^{64543}]$ ,

$\langle b, a, d \rangle^{53789}$ ]. Although we do not know the underlying process, intuitively, the probability that the next case will have a trace not allowed by  $m_3$  is close to 0. This simple example shows that recall and precision are not enough for conformance checking. We need a generalization notion to address the risk of overfitting example data.

It is difficult to reason about generalization because this refers to unseen cases. Van der Aalst et al. [3] was the first to quantify generalization. In [3], each event is seen as an observation of an activity  $a$  in some state  $s$ . Suppose that state  $s$  is visited  $n$  times and that  $w$  is the number of different activities observed in this state. Suppose that  $n$  is very large and  $w$  is very small, then it is unlikely that a new event visiting this state will correspond to an activity not seen before in this state. However, if  $n$  and  $w$  are of the same order of magnitude, then it is more likely that a new event visiting state  $s$  will correspond to an activity not seen before in this state. This reasoning is used to provide a generalization metric. This estimate can be derived under the Bayesian assumption that there is an unknown number of possible activities in state  $s$  and that probability distribution over these activities follows a multinomial distribution.

It is not easy to develop an approach that accurately measures generalization. Therefore, some authors define generalization using the notion of a “system” (i.e., a model of the real underlying process). The system refers to the real behavior of the underlying process that the model tries to capture. This can also include the context of the process such as the organization or rules. For example, employees of a company might exceptionally be allowed to deviate from the defined process model in certain situations [20]. In this view, *system fitness* measures the fraction of the behavior of the system that is captured by the model and *system precision* measures how much of the behavior of the model is part of the system. Buijs et al. [11] link this view to the traditional understanding of generalization. They state that both system fitness and system precision are difficult to obtain under the assumption that the system is unknown. Therefore, state-of-the-art discovery algorithms assume that the process model discovered from an event log does not contain behavior outside of the system. In other words, they assume system precision to be 1. Given this assumption, system fitness can be seen as generalization [11]. Janssenswillen et al. [20] agree that in this comparison between the system and the model, especially the system fitness, in fact is what defines generalization. Furthermore, Janssenswillen and Depaire [18] demonstrated the differences between the traditional and the system-based view on conformance checking by showing that state-of-the-art conformance measures cannot reliably assess the similarity between a process model and the underlying system.

Although capturing the unobserved behavior by assuming a model of the system is a theoretically elegant solution, practical applicability of this solution is hindered by the fact that it is often impossible to retrieve full knowledge about the system itself. Furthermore, [2] showed the importance of trace probabilities in process models. To accurately represent reality, the system would also need to include probabilities for each of its traces. However, to date, there is only one conformance measure that can actually support probabilistic process models [22]. This approach uses the Earth Movers’ distance which measures the effort to transform the distributions of traces of the event log into the distribution of traces of the model.

Some people would argue that one should use cross-validation (e.g.,  $k$ -fold checking). However, this is a very different setting. Cross validation aims to estimate the quality of a discovery approach and not the quality of a given model given an event log. Of course, one could produce multiple process models using fragments of the event log and compare them. However, such forms of cross-validation evaluate the quality of the discovery technique and are unrelated to generalization.

For these reasons, we define generalization in the traditional sense.

**Definition 9 (Generalization).** A generalization measure  $gen \in \mathcal{L} \times \mathcal{M} \rightarrow [0, 1]$  aims to quantify the probability that new unseen cases will fit the model.<sup>5</sup>

This definition assumes that a process generates a stream of newly executed cases. The more traces that are fitting and the more redundancy there is in the event, the more certain one can be that the next case will have a trace that fits the model. Note that we deliberately do not formalize the notion of probability, since in real-life we cannot know the real process. Also phenomena like concept drift and contextual factors make it unrealistic to reason about probabilities in a formal sense.

Based on this definition, we present a set of propositions. Note that we do not claim our set of propositions to be complete and invite other researchers who represent a different viewpoint on generalization to contribute to the discussion.

## 5.1 Generalization Propositions

Generalization ( $gen \in \mathcal{L} \times \mathcal{M} \rightarrow [0, 1]$ ) aims to quantify the probability that new unseen cases will fit the model. This conformance dimension is a bit different than the two previously discussed conformance dimensions because it reasons about future *unseen* cases (i.e., not yet in the event log). If the recall is good and the log is complete with lots of repeating behavior, then future cases will most likely fit the model. Analogous to recall, model extensions cannot lower generalization (**GenPro1**<sup>+</sup>), extending the log with fitting behavior cannot lower generalization (**GenPro2**<sup>+</sup>), and extending the log with non-fitting behavior cannot improve generalization (**GenPro3**<sup>0</sup>).

**Proposition 14 (GenPro1<sup>+</sup>).** For any  $l \in \mathcal{L}$  and  $m_1, m_2 \in \mathcal{M}$  such that  $\tau(m_1) \subseteq \tau(m_2)$ :  $gen(l, m_1) \leq gen(l, m_2)$ .

Similar to recall, this proposition implies **BehPro**<sup>+</sup>. Generalization measures violating **BehPro**<sup>+</sup> also violate **GenPro1**<sup>+</sup>.

**Proposition 15 (GenPro2<sup>+</sup>).** For any  $l_1, l_2, l_3 \in \mathcal{L}$  and  $m \in \mathcal{M}$  such that  $l_2 = l_1 \uplus l_3$  and  $\tau(l_3) \subseteq \tau(m)$ :  $gen(l_1, m) \leq gen(l_2, m)$ .

**Proposition 16 (GenPro3<sup>0</sup>).** For any  $l_1, l_2, l_3 \in \mathcal{L}$  and  $m \in \mathcal{M}$  such that  $l_2 = l_1 \uplus l_3$  and  $\tau(l_3) \subseteq \bar{\tau}(m)$ :  $gen(l_1, m) \geq gen(l_2, m)$ .

<sup>5</sup> Note that the term “probability” is used here in an informal manner. Since we only have example observations and no knowledge of the underlying (possibly changing) process, we cannot compute such a probability. Of course, unseen cases can have traces that have been observed before.

Duplicating the event log does not necessarily influence recall and precision. According to propositions **RecPro4<sup>0</sup>** and **PrecPro4<sup>0</sup>** this should have no effect on recall and precision. However, making the event log more redundant, should have an effect on generalization. For fitting logs, adding redundancy without changing the distribution can only improve generalization (**GenPro4<sup>+</sup>**). For non-fitting logs, adding redundancy without changing the distribution can only lower generalization (**GenPro5<sup>+</sup>**). Note that **GenPro4<sup>+</sup>** and **GenPro5<sup>+</sup>** are special cases of **GenPro6<sup>0</sup>** and **GenPro7<sup>0</sup>**. **GenPro6<sup>0</sup>** and **GenPro7<sup>0</sup>** consider logs where some traces are fitting and others are not. For a log where more than half of the traces is fitting, duplication can only improve generalization (**GenPro6<sup>0</sup>**). For a log where more than half of the traces is non-fitting, duplication can only lower generalization (**GenPro7<sup>0</sup>**).

**Proposition 17 (GenPro4<sup>+</sup>).** For any  $l \in \mathcal{L}$ ,  $m \in \mathcal{M}$ , and  $k \geq 1$  such that  $\tau(l) \subseteq \tau(m)$ :  $gen(l^k, m) \geq gen(l, m)$ .

**Proposition 18 (GenPro5<sup>+</sup>).** For any  $l \in \mathcal{L}$ ,  $m \in \mathcal{M}$ , and  $k \geq 1$  such that  $\tau(l) \subseteq \bar{\tau}(m)$ :  $gen(l^k, m) \leq gen(l, m)$ .

**Proposition 19 (GenPro6<sup>0</sup>).** For any  $l \in \mathcal{L}$ ,  $m \in \mathcal{M}$ , and  $k \geq 1$  such that most traces are fitting ( $|\{t \in l \mid t \in \tau(m)\}| \geq |\{t \in l \mid t \notin \tau(m)\}|$ ):  $gen(l^k, m) \geq gen(l, m)$ .

**Proposition 20 (GenPro7<sup>0</sup>).** For any  $l \in \mathcal{L}$ ,  $m \in \mathcal{M}$ , and  $k \geq 1$  such that most traces are non-fitting ( $|\{t \in l \mid t \in \tau(m)\}| \leq |\{t \in l \mid t \notin \tau(m)\}|$ ):  $gen(l^k, m) \leq gen(l, m)$ .

When the model allows for any behavior, clearly the next case will also be fitting (**GenPro8<sup>0</sup>**). Nevertheless, it is marked as controversial because the proposition would also need to hold for an empty event log.

**Proposition 21 (GenPro8<sup>0</sup>).** For any  $l \in \mathcal{L}$  and  $m \in \mathcal{M}$  such that  $\tau(m) = \mathcal{T}$ :  $gen(l, m) = 1$ .

## 5.2 Existing Generalization Measures

The following sections introduce several state-of-the-art generalization measures, before they will be evaluated using the corresponding propositions.

**Alignment generalization ( $gen_S$ ).** Van der Aalst et al. [3] also introduce a measure for generalization. This approach considers each occurrence of a given event  $e$  as observation of an activity in some state  $s$ . The approach is parameterized by a  $state_M$  function that maps events onto states in which they occurred. For each event  $e$  that occurred in state  $s$  the approach counts how many different activities  $w$  were observed in that state. Furthermore, it counts the number of visits  $n$  to this state. Generalization is high if  $n$  is very large and  $w$  is small, since in that case, it is unlikely that a new trace will correspond to unseen behavior in that state.

Table 4: An overview of the generalization propositions that hold for the measures: (assuming  $l \neq []$ ,  $\tau(m) \neq \emptyset$  and  $\langle \rangle \notin \tau(m)$ ):  $\checkmark$  means that the proposition holds for any log and model and  $\times$  means that the proposition does not always hold.

Proposition	Name	$gen_S$	$gen_T$	$gen_U$
1	<b>DetPro</b> <sup>+</sup>	$\checkmark$	$\times$	$\checkmark$
2	<b>BehPro</b> <sup>+</sup>	$\checkmark$	$\times$	$\times$
14	<b>GenPro1</b> <sup>+</sup>	$\times$	$\times$	$\times$
15	<b>GenPro2</b> <sup>+</sup>	$\times$	$\times$	$\times$
16	<b>GenPro3</b> <sup>0</sup>	$\times$	$\times$	$\times$
17	<b>GenPro4</b> <sup>+</sup>	$\checkmark$	$\checkmark$	$\checkmark$
18	<b>GenPro5</b> <sup>+</sup>	$\times$	$\checkmark$	$\checkmark$
19	<b>GenPro6</b> <sup>0</sup>	$\checkmark$	$\checkmark$	$\checkmark$
20	<b>GenPro7</b> <sup>0</sup>	$\times$	$\checkmark$	$\checkmark$
21	<b>GenPro8</b> <sup>0</sup>	$\times$	$\checkmark$	$\times$

**Weighted negative event generalization ( $gen_T$ ).** Aside from improving the approach of Goedertier et al. [16] van den Broucke et al. [30] also developed a generalization measure based on weighted negative events. It defines allowed generalizations  $AG$  which represent events, that could be replayed without errors and confirm that the model is general and disallowed generalizations  $DG$  which are generalization events, that could not be replayed correctly. If during replay a negative event  $e$  is encountered that actually was enabled the  $AG$  value is increased by  $1 - weight(e)$ . Similarly, if a negative event is not enabled the  $DG$  value is increased by  $1 - weight(e)$ . The more disallowed generalizations are encountered during log replay the lower generalization.

**Anti-alignment generalization ( $gen_U$ ).** Van Dongen et al. [13] also introduce an anti-alignment generalization and build on the principle that with a generalizing model, newly seen behavior will introduce new paths between the states of the model, however no new states themselves. Therefore, they define a recovery distance  $d_{rec}$  which measures the maximum distance between the states visited by the log and the states visited by the anti-alignment  $\gamma$ . A perfectly generalizing model according to van Dongen et al. [13] has the maximum distance to the anti-alignment with minimal recovery distance. Similar to recall they define trace-based and log-based generalization. Finally, anti-alignment generalization is the weighted combination of trace-based and log-based anti-alignment generalization.

### 5.3 Evaluation of Existing Generalization Measures

The previously presented generalization measures are evaluated using the corresponding propositions. The results of the evaluation are displayed in Table 4. To improve the readability of this paper, only the most interesting findings of the evaluation are addressed in the following section. For full details refer to Appendix A.4.



Table 4 displays that alignment based generalization ( $gen_S$ ) violates several propositions. Generalization is not defined if there are unfitting traces since they cannot be mapped to states of the process model. Therefore, unfitting event logs should be aligned to fit to the model before calculating generalization. Aligning a non-fitting log and duplicating it will result in more visits to each state visited by the log. Therefore, adding non-fitting behavior increases generalization and violates the propositions **GenPro3<sup>0</sup>**, **GenPro5<sup>+</sup>** and **GenPro7<sup>0</sup>**.

In comparison, weighted negative event generalization ( $gen_T$ ) is robust against the duplication of the event log, even if it contains non-fitting behavior. However, this measure violates **DetPro<sup>+</sup>**, **BehPro<sup>+</sup>**, **GenPro1<sup>+</sup>**, **GenPro2<sup>+</sup>** and **GenPro3<sup>0</sup>**, which states that extending the log with non-fitting behavior cannot improve generalization. However, in this approach, negative events are assigned a weight which indicates how certain the log is about these events being negative ones. Even though the added behavior is non-fitting it might still provide evidence for certain negative events and therefore increase their weight. If these events are then not enabled during log replay the value for disallowed generalizations (DG) decreases  $DG(l_1, m) < DG(l_2, m)$  and generalization improves:  $\frac{AG(l_1, m)}{AG(l_1, m) + DG(l_1, m)} < \frac{AG(l_2, m)}{AG(l_2, m) + DG(l_2, m)}$ .

Table 4 shows that anti-alignment generalization ( $gen_U$ ) violates several propositions. The approach considers markings of the process models as the basis for the generalization computation which violates the behavioral proposition. Furthermore, the measure cannot handle if the model displays behavior that has not been observed in the event log. If the unobserved model behavior and therefore also the anti-alignment introduced a lot of new states which were not visited by the event log, the value of the recovery distance increases and generalization is lowered. This clashes with propositions **GenPro1<sup>+</sup>** and **GenPro8<sup>+</sup>**. Finally, the approach also excludes unfitting behavior from its scope. Only after aligning the event log, generalization can be computed. As a result, the measure fulfills **GenPro5<sup>+</sup>**, **GenPro6<sup>0</sup>** and **GenPro7<sup>0</sup>**, but violates **GenPro3<sup>0</sup>**.

## 6 Conclusion

With the process mining field maturing and more commercial tools becoming available [21], there is an urgent need to have a set of agreed-upon measures to determine the quality of discovered processes models. We have revisited the 21 conformance propositions introduced in [2] and illustrated their relevance by applying them to baseline measures. Furthermore, we used the propositions to evaluate currently existing conformance measures. This evaluation uncovers large differences between existing conformance measures and the properties that they possess in relation to the propositions. It is surprising that seemingly obvious requirements are not met by today's conformance measures. However, there are also measures that do meet all the propositions.

It is important to note that we do not consider the set of propositions to be complete. Instead, we consider them to be an initial step to start the discussion on what properties are to be desired from conformance measures, and we encourage others to contribute to this discussion. Moreover, we motivate researchers to use the conformance propositions as design criteria for the development of novel conformance measures.

One relevant direction of future work is in the area of conformance propositions that have a more fine-grained focus than the trace-level, i.e., that distinguish between *almost fitting* and *completely non-fitting* behavior. Another relevant area of future work is in the direction of probabilistic conformance measures, which take into account branching probabilities in models, and their desired properties.

**Acknowledgements** We thank the Alexander von Humboldt (AvH) Stiftung for supporting our research.

## References

1. W.M.P. van der Aalst. *Process Mining: Data Science in Action*. Springer-Verlag, Berlin, 2016.
2. W.M.P. van der Aalst. Relating Process Models and Event Logs: 21 Conformance Propositions. In W.M.P. van der Aalst, R. Bergenthum, and J. Carmona, editors, *Workshop on Algorithms & Theories for the Analysis of Event Data (ATAED 2018)*, pages 56–74. CEUR Workshop Proceedings, 2018.
3. W.M.P. van der Aalst, A. Adriansyah, and B. van Dongen. Replaying History on Process Models for Conformance Checking and Performance Analysis. *WIREs Data Mining and Knowledge Discovery*, 2(2):182–192, 2012.
4. W.M.P. van der Aalst, A.J.M.M. Weijters, and L. Maruster. Workflow Mining: Discovering Process Models from Event Logs. *IEEE Transactions on Knowledge and Data Engineering*, 16(9):1128–1142, 2004.
5. A. Adriansyah, B. van Dongen, and W.M.P. van der Aalst. Conformance Checking using Cost-Based Fitness Analysis. In C.H. Chi and P. Johnson, editors, *IEEE International Enterprise Computing Conference (EDOC 2011)*, pages 55–64. IEEE Computer Society, 2011.
6. A. Adriansyah, J. Munoz-Gama, J. Carmona, B. F. van Dongen, and W. M. P. van der Aalst. Alignment based precision checking. In M. La Rosa and P. Soffer, editors, *Business Process Management Workshops*, pages 137–149. Springer, 2013.
7. A. Augusto, A. Armas-Cervantes, R. Conforti, M. Dumas, M. La Rosa, and D. Reissner. Abstract-and-compare: A family of scalable precision measures for automated process discovery. In M. Weske, M. Montali, I. Weber, and J. vom Brocke, editors, *Proceedings of the International Conference on Business Process Management*, pages 158–175, Cham, 2018. Springer International Publishing.
8. S.K.L.M. Vanden Broucke, J. De Weerd, J. Vanthienen, and B. Baesens. On replaying process execution traces containing positive and negative events. Technical report, KU Leuven-Faculty of Economics and Business, 2013.
9. J.C.A.M. Buijs. *Flexible evolutionary algorithms for mining structured process models*. PhD thesis, Department of Mathematics and Computer Science, 2014.
10. J.C.A.M. Buijs, B.F. van Dongen, and W.M.P. van der Aalst. On the Role of Fitness, Precision, Generalization and Simplicity in Process Discovery. In R. Meersman, S. Rinderle, P. Dadam, and X. Zhou, editors, *OTM Federated Conferences, 20th International Conference on Cooperative Information Systems (CoopIS 2012)*, volume 7565 of *Lecture Notes in Computer Science*, pages 305–322. Springer-Verlag, Berlin, 2012.
11. J.C.A.M. Buijs, B.F. van Dongen, and W.M.P. van der Aalst. Quality Dimensions in Process Discovery: The Importance of Fitness, Precision, Generalization and Simplicity. *International Journal of Cooperative Information Systems*, 23(1):1–39, 2014.

12. J. Carmona, B. van Dongen, A. Solti, and M. Weidlich. *Conformance Checking: Relating Processes and Models*. Springer-Verlag, Berlin, 2018.
13. B.F. van Dongen, J. Carmona, and T. Chatain. A Unified Approach for Measuring Precision and Generalization Based on Anti-alignments. In M. La Rosa, P. Loos, and O. Pastor, editors, *International Conference on Business Process Management (BPM 2016)*, volume 9850 of *Lecture Notes in Computer Science*, pages 39–56. Springer-Verlag, Berlin, 2016.
14. B.F. van Dongen, J. Carmona, T. Chatain, and F. Taymouri. Aligning Modeled and Observed Behavior: A Compromise Between Computation Complexity and Quality. In E. Dubois and K. Pohl, editors, *International Conference on Advanced Information Systems Engineering (Caise 2017)*, volume 10253 of *Lecture Notes in Computer Science*, pages 94–109. Springer-Verlag, Berlin, 2017.
15. L. Garcia-Banuelos, N. van Beest, M. Dumas, M. La Rosa, and W. Mertens. Complete and Interpretable Conformance Checking of Business Processes. *IEEE Transactions on Software Engineering*, 44(3):262–290, 2018.
16. S. Goedertier, D. Martens, J. Vanthienen, and B. Baesens. Robust Process Discovery with Artificial Negative Events. *Journal of Machine Learning Research*, 10:1305–1340, 2009.
17. G. Greco, A. Guzzo, L. Pontieri, and D. Saccà. Discovering Expressive Process Models by Clustering Log Traces. *IEEE Transaction on Knowledge and Data Engineering*, 18(8):1010–1027, 2006.
18. G. Janssenswillen and B. Depaire. Towards confirmatory process discovery: Making assertions about the underlying system. *Business & Information Systems Engineering*, Dec 2018.
19. G. Janssenswillen, N. Donders, T. Jouck, and B. Depaire. A comparative study of existing quality measures for process discovery. *Information Systems*, 50(1):2:1–2:45, 2017.
20. G. Janssenswillen, T. Jouck, M. Creemers, and B. Depaire. Measuring the quality of models with respect to the underlying system: An empirical study. In M. La Rosa, P. Loos, and O. Pastor, editors, *Business Process Management*, pages 73–89. Cham, 2016. Springer International Publishing.
21. M. Kerremans. Gartner Market Guide for Process Mining, Research Note G00353970. [www.gartner.com](http://www.gartner.com), 2018.
22. S. Leemans, A. Syring, and W.M.P. van der Aalst. Earth Movers’ Stochastic Conformance Checking. In T. Hildebrandt, B.F. van Dongen, M. Röglinger, and J. Mendling, editors, *Business Process Management Forum (BPM Forum 2019)*, volume 360 of *Lecture Notes in Business Information Processing*, pages 1–16. Springer-Verlag, Berlin, 2019.
23. S.J.J. Leemans, D. Fahland, and W.M.P. van der Aalst. Scalable Process Discovery and Conformance Checking. *Software and Systems Modeling*, 17(2):599–631, 2018.
24. F. Mannhardt, M. de Leoni, H.A. Reijers, and W.M.P. van der Aalst. Balanced Multi-Perspective Checking of Process Conformance. *Computing*, 98(4):407–437, 2016.
25. J. Munoz-Gama and J. Carmona. A Fresh Look at Precision in Process Conformance. In R. Hull, J. Mendling, and S. Tai, editors, *Business Process Management (BPM 2010)*, volume 6336 of *Lecture Notes in Computer Science*, pages 211–226. Springer-Verlag, Berlin, 2010.
26. A. Polyvyanyy, A. Solti, M. Weidlich, C. Di Ciccio, and J. Mendling. Behavioural Quotients for Precision and Recall in Process Mining. Technical report, University of Melbourne, 2018.
27. A. Rozinat and W.M.P. van der Aalst. Conformance Checking of Processes Based on Monitoring Real Behavior. *Information Systems*, 33(1):64–95, 2008.
28. A. Rozinat, A.K. Alves de Medeiros, C.W. Günther, A.J.M.M. Weijters, and W.M.P. van der Aalst. The Need for a Process Mining Evaluation Framework in Research and Practice. In M. Castellanos, J. Mendling, and B. Weber, editors, *Informal Proceedings of the International Workshop on Business Process Intelligence (BPI 2007)*, pages 73–78. QUT, Brisbane, Australia, 2007.
29. N. Tax, X. Lu, N. Sidorova, D. Fahland, and W.M.P. van der Aalst. The Imprecisions of Precision Measures in Process Mining. *Information Processing Letters*, 135:1–8, 2018.

30. S. K. L. M. vanden Broucke, J. De Weerd, J. Vanthienen, and B. Baesens. Determining process model precision and generalization with weighted artificial negative events. *IEEE Transactions on Knowledge and Data Engineering*, 26(8):1877–1889, Aug 2014.
31. J. De Weerd, M. De Backer, J. Vanthienen, and B. Baesens. A Multi-Dimensional Quality Assessment of State-of-the-Art Process Discovery Algorithms Using Real-Life Event Logs. *Information Systems*, 37(7):654–676, 2012.
32. J. De Weerd, M. De Backer, J. Vanthienen, and B. Baesens. A Robust F-measure for Evaluating Discovered Process Models. In N. Chawla, I. King, and A. Sperduti, editors, *IEEE Symposium on Computational Intelligence and Data Mining (CIDM 2011)*, pages 148–155, Paris, France, April 2011. IEEE.
33. A.J.M.M. Weijters, W.M.P. van der Aalst, and A.K. Alves de Medeiros. Process Mining with the Heuristics Miner-algorithm. BETA Working Paper Series, WP 166, Eindhoven University of Technology, Eindhoven, 2006.

## A Appendix

### A.1 Evaluation results of the baseline conformance measures

Table 5: Overview of the conformance propositions that hold for the three baseline measures (under the assumption that  $l \neq []$ ,  $\tau(m) \neq \emptyset$  and  $\langle \rangle \notin \tau(m)$ ):  $\checkmark$  means that the proposition holds for any log and model and  $\times$  means that the proposition does not always hold.

Proposition	Name	$rec_{TB}$	$rec_{FB}$	$prec_{FB}$
1	<b>DetPro</b> <sup>+</sup>	$\checkmark$	$\checkmark$	$\times$
2	<b>BehPro</b> <sup>+</sup>	$\checkmark$	$\checkmark$	$\checkmark$
3	<b>RecPro1</b> <sup>+</sup>	$\checkmark$	$\checkmark$	
4	<b>RecPro2</b> <sup>+</sup>	$\checkmark$	$\checkmark$	
5	<b>RecPro3</b> <sup>0</sup>	$\checkmark$	$\checkmark$	
6	<b>RecPro4</b> <sup>0</sup>	$\checkmark$	$\checkmark$	
7	<b>RecPro5</b> <sup>+</sup>	$\checkmark$	$\checkmark$	
8	<b>PrecPro1</b> <sup>+</sup>			$\checkmark$
9	<b>PrecPro2</b> <sup>+</sup>			$\checkmark$
10	<b>PrecPro3</b> <sup>0</sup>			$\checkmark$
11	<b>PrecPro4</b> <sup>0</sup>			$\checkmark$
12	<b>PrecPro5</b> <sup>+</sup>			$\checkmark$
13	<b>PrecPro6</b> <sup>0</sup>			$\checkmark$

## A.2 Detailed Results of the Recall Measure Evaluation

### Proposition DetPro<sup>+</sup>

**Causal footprint recall** ( $rec_A$ ). *Proposition holds.*

**Reasoning.** The causal footprint fully describes the log as well as the model in terms of directly followed relations. By comparing the footprints of the model and the log recall can be determined.

**Token replay recall** ( $rec_B$ ). *Proposition does not hold.*

**Reasoning.** This technique depends on the path taken through the model. Due to duplicate activities and silent transitions, multiple paths through a model can be taken when replaying a single trace. Different paths can lead to different numbers of produced, consumed, missing and remaining tokens and recall is not deterministic.

**Alignment recall** ( $rec_C$ ). *Proposition holds.*

**Reasoning.** When computing alignments the algorithm searches per trace for the optimal alignment: the alignment with the least cost associated to it. There may be multiple alignments, but these all have the same cost. Recall is computed based on this cost. Therefore, given a log, a model and a cost function the recall computation is deterministic.

**Behavioral recall** ( $rec_D$ ). *Proposition does not hold.*

**Reasoning.** If duplicate or silent transitions are encountered during replay of the traces, which were enhanced with negative events, it is explored which of the available transitions enables the next event in the trace. If no solution is found one of the transitions is randomly fired, which can lead to different recall values for traces with the same behavior.

**Projected recall** ( $rec_E$ ). *Proposition holds.*

**Reasoning.** This technique splits log and model into subsets and calculates how many traces in the sub-log can be replayed on the corresponding sub-model, which is represented as a deterministic finite automaton. The sub-logs and models are created by projection on a subset of activities which is a deterministic process. Therefore, the computation of the average recall value over all subsets is also deterministic.

**Continued parsing measure** ( $rec_F$ ). *Proposition does not hold.*

**Reasoning.** The continued parsing measure translates the behavior of the process model into a causal matrix. This translation is not defined if the model contains duplicate or silent transitions. Consequently, the continued parsing measure is not defined for these models, which violates this proposition.

**Eigenvalue recall** ( $rec_G$ ). *Proposition holds.*

**Reasoning.** The measure compares the languages of the model and the language of the process model. These have to be irreducible, to compute their eigenvalue. Since the language of an event log is not irreducible, Polyvyanyy et al. [26] introduce a short-circuit measure over languages and proof that it is a deterministic measure over any arbitrary regular language.

## Proposition 2 BehPro<sup>+</sup>

**Causal footprint recall** ( $rec_A$ ). *Proposition holds.*

**Reasoning.** The causal footprint completely describes the log as well as the model in terms of directly followed relations and therefore does not depend on the representation of the model.

**Token replay recall** ( $rec_B$ ). *Proposition does not hold.*

**Reasoning.** Due to duplicate activities and silent transitions, one can think of models with the same behavior but a different structure. It is also possible to have implicit places that do not change the behavior but do influence the number of produced, consumed, missing and remaining tokens. For example, if a place often has missing tokens, then duplicating this place will lead to even more missing tokens (also relatively). Moreover, nondeterminism during replay can lead to a difference in the replay path and therefore in different numbers of produced, consumed, missing and remaining tokens and shows that token replay-based recall depends on the representation of the model.

**Alignment recall** ( $rec_C$ ). *Proposition holds.*

**Reasoning.** Silent transitions are used for routing behavior of the Petri net and are not linked to the actual behavior of the process. During alignment computation, there is no cost associated with silent transitions. Also, the places do not play a role (e.g., implicit places have no effect). Therefore, the structure of the model itself has no influence on the alignment computation and two different models expressing the same behavior will result in the same recall measures.

**Behavioral recall** ( $rec_D$ ). *Proposition does not hold.*

**Reasoning.** If duplicate or silent transitions are encountered during the replay of the traces, which were enhanced with negative events on the model, it is explored which of the available transitions enables the next event in the trace. If no solution is found, one of the transitions is randomly fired, which can lead to different recall values for a trace on two behaviorally equivalent but structurally different models.

**Projected recall** ( $rec_E$ ). *Proposition holds.*

**Reasoning.** This technique translates the event log as well as the process model into deterministic finite automata before computing recall. Therefore, it is independent of the representation of the model itself.

**Continued parsing measure** ( $rec_F$ ). *Proposition holds.*

**Reasoning.** The continued parsing measure translates the possible behavior into so-called input expressions and output expressions, which describe possible behavior before and after the execution of each activity. Therefore, it abstracts from the structure of the process model.

**Eigenvalue recall** ( $rec_G$ ). *Proposition holds.*

**Reasoning.** The approach computes recall based on the languages of the event log and the language of the process model. This abstracts from the representation of the process model and, therefore, the proposition holds.

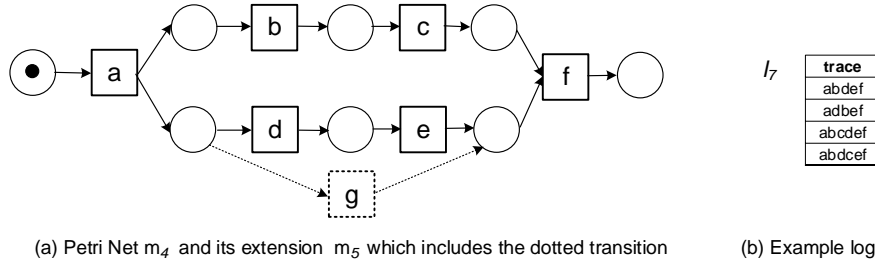


Fig. 6: Petri net  $m_4$  and its extension  $m_5$  which includes the dotted transition (b), as well as example log  $l_7$ .

Table 6: The Causal footprints of  $m_4$  (a),  $l_7$  (b) and  $m_5$  (c). Mismatching relations are marked in red.

(a)						(b)						(c)									
a	b	c	d	e	f	a	b	c	d	e	f	a	b	c	d	e	f	g			
a	#	→	#	→	#	#	a	#	→	#	→	#	a	#	→	#	→	#	#	→	
b	←	#	→			#	b	←	#	→		→	#	b	←	#	→			#	
c	#	←	#			→	c	#	←	#		→	#	c	#	←	#			→	
d	←			#	→	#	d	←			#	→	#	d	←			#	→	#	#
e	#			←	#	→	e	#	←	←	←	#	→	e	#			←	#	→	#
f	#	#	←	#	←	#	f	#	#	#	#	←	#	f	#	#	←	#	←	#	←
														g	←			#	#	←	#

**Proposition 3 RecPro1<sup>+</sup>**

*Causal footprint recall (rec<sub>A</sub>). Proposition does not hold.*

**Reasoning.** Recall is calculated by dividing the number of relations where log and model differ by the total number of relations. When adding behavior to the model while keeping the log as is, the causal footprint of the model changes while the causal footprint of the log stays the same. This may introduce more differences between both footprints and therefore lowers recall. Process model  $m_4$ , its extension  $m_5$  in Figure 6 (a) and event log  $l_7 = [\langle a, d, b, e, f \rangle, \langle a, b, d, e, f \rangle, \langle a, b, c, d, e, f \rangle, \langle a, b, d, c, e, f \rangle]$  illustrate this. The corresponding causal footprints are displayed in Table 6. Since the log  $l_7$  does not contain activity  $g$ , when computing recall between  $l_7$  and  $m_5$ , we assume that all activities show a #-relation with  $g$ . Computing recall based on the footprints results in a  $rec_A(l_7, m_4) = 1 - \frac{6}{36} = 0.83$  and  $rec_A(l_7, m_5) = 1 - \frac{14}{49} = 0.71$ . The proposition is violated since  $rec_A(l_7, m_4) < rec_A(l_7, m_5)$ . Van der Aalst mentions in [1] that checking conformance using causal footprints is only meaningful if the log is complete in term of directly followed relations. Furthermore, the approach intended to cover precision, generalization and recall in one conformance value.

*Token replay recall (rec<sub>B</sub>). Proposition does not hold.*

**Reasoning.** BehPro<sup>+</sup> does not hold, which implies that RecPro1<sup>+</sup> does not hold.

**Alignment recall** ( $rec_C$ ). *Proposition holds.*

**Reasoning.** Note that the model extension only adds behavior to the model and does not restrict it further. During alignment computation, this means that either the initial alignments are computed or that the additional behavior resulted in an optimal alignment with even lower cost (i.e. alignments with less log/model moves). Therefore, recall of the extended model cannot be lower than the value calculated for the initial model.

**Behavioral recall** ( $rec_D$ ). *Proposition does not hold.*

**Reasoning.** **BehPro**<sup>+</sup> does not hold, which implies that **RecPro1**<sup>+</sup> does not hold.

**Projected recall** ( $rec_E$ ). *Proposition holds.*

**Reasoning.** Based on the definition of projected recall it can only be lowered if fewer traces of the log can be replayed on the model. This is only possible if the model extension also restricts parts of its behavior. Hence, by purely extending the model the number of fitting traces can only be increased:  $|\{t \in l|_A | t \in DFA(m_1|_A)\}| \leq |\{t \in l|_A | t \in DFA(m_2|_A)\}|$  if  $\tau(m_1) \subseteq \tau(m_2)$ . As a result, recall cannot be lowered and the proposition holds.

**Continued parsing measure** ( $rec_F$ ). *Proposition holds.*

**Reasoning.** Note that the model extension only adds behavior to the model and does not restrict it further. When replaying the log on the extended causal matrix the number of missing and remaining activated expressions stays the same or decreases which consequently cannot lower recall.

**Eigenvalue recall** ( $rec_G$ ). *Proposition holds.*

**Reasoning.** Trivially, adding behavior to the model can only increase the intersection between the language of the log and the model.:  $\mathcal{L}(l) \cap \mathcal{L}(m_1) \leq \mathcal{L}(l) \cap \mathcal{L}(m_2)$  if  $\tau(m_1) \subseteq \tau(m_2)$ . Polyvyany et al. [26] proved in lemma 5.6 that the short-circuit measure based on eigenvalue is increasing, i.e. that  $\frac{eig(\mathcal{L}(l) \cap \mathcal{L}(m_1))}{eig(\mathcal{L}(l))} \leq \frac{eig(\mathcal{L}(l) \cap \mathcal{L}(m_2))}{eig(\mathcal{L}(l))}$ .

#### **Proposition 4 RecPro2**<sup>+</sup>

**Causal footprint recall** ( $rec_A$ ). *Proposition holds.*

**Reasoning.** Adding fitting behavior to the event log either does not change the footprint of the log because no new relations were observed or it changes the corresponding causal footprint in a way that more of its relations match the causal footprint of the model. The only three options are, that a #-relation changes to  $\rightarrow$ ,  $\rightarrow$  becomes  $\parallel$  or  $\leftarrow$  becomes  $\parallel$ . Hence the differences between the two footprints are minimized and recall improved.

**Token replay recall** ( $rec_B$ ). *Proposition holds.*

**Reasoning.** Adding fitting behavior to the event log means that these traces can be replayed on the model without any problems. Here we assume that if a trace is perfectly replayable it will also be replayed perfectly. In case of duplicate and silent transitions this does not need to be the case. Consider two very long branches in the process model



allowing for the same behavior. Only at the end, they have differences. This may lead to the situation where initially the wrong branch was chosen. In this paper, we make the assumption that fitting behavior is replayed correctly. Hence, adding fitting traces results in more produced and consumed tokens without additional missing or remaining tokens ( $c_1 < c_2$  and  $p_1 < p_2$ ). Therefore, recall can only be improved by adding fitting behavior.  $\frac{1}{2}(1 - \frac{m}{c_1}) + \frac{1}{2}(1 - \frac{r}{p_1}) \leq \frac{1}{2}(1 - \frac{m}{c_2}) + \frac{1}{2}(1 - \frac{r}{p_2})$ .

**Alignment recall** ( $rec_C$ ). *Proposition holds.*

**Reasoning.** Fitting behavior results in a perfect alignment which only consists of synchronous moves. Consequently, this alignment has no costs assigned and adding it to the existing log cannot lower recall.

**Behavioral recall** ( $rec_D$ ). *Proposition holds.*

**Reasoning.** For fitting log  $l_3$ ,  $FN(l_3, m) = 0$  and  $TP(l_3, m)$  is proportional to the size of  $l_3$ . For  $l_2 = l_1 \uplus l_3$ , we have  $TP(l_2, m) = TP(l_1, m) + TP(l_3, m)$ , and  $FN(l_2, m) = FN(l_1, m) + FN(l_3, m) = FN(l_1, m)$ . Therefore,  $rec_D(l_2, m) = \frac{TP(l_1, m) + TP(l_3, m)}{FN(l_1, m)}$  and since  $rec_D(l_1, m) = \frac{TP(l_1, m)}{FN(l_1, m)}$ , we have  $rec_D(l_2, m) \geq rec_D(l_1, m)$ .

**Projected recall** ( $rec_E$ ). *Proposition holds.*

**Reasoning.** Based on the definition of projected recall it can only be lowered if fewer traces of the log can be replayed on the model. Fitting traces can be replayed and recall cannot be lowered by adding them to the log.  $||[t \in l_1|_A | t \in DFA(m|_A)]|| \leq ||[t \in l_2|_A | t \in DFA(m|_A)]||$  if  $l_2 = l_1 \uplus l_3$  and  $\tau(l_3) \subseteq \tau(m)$ . Averaging recall over all subsets of a given length also does not influence this.

**Continued parsing measure** ( $rec_F$ ). *Proposition holds.*

**Reasoning.** Adding fitting behavior to the event log means that these traces can be replayed on the causal matrix without any problems. Here we assume, similar to  $rec_B$ , that if a trace is perfectly replayable it will also be replayed perfectly. Hence fitting behavior does not yield additional missing or remaining activated expressions. Therefore recall can only be improved.

**Eigenvalue recall** ( $rec_G$ ). *Proposition holds.*

**Reasoning.** Trivially, adding fitting behavior to the event log can only increase the intersection between the language of the log and the model, i.e.,  $\mathcal{L}(l_1) \cap \mathcal{L}(m) \leq \mathcal{L}(l_2) \cap \mathcal{L}(m)$  if  $l_2 = l_1 \uplus l_3$  and  $\tau(l_3) \subseteq \tau(m)$ . Polyvyany et al. [26] proved in lemma 5.6 that the short-circuit measure based on eigenvalue is increasing, and therefore it also holds that  $\frac{eig(\mathcal{L}(l_1) \cap \mathcal{L}(m))}{eig(\mathcal{L}(l_1))} \leq \frac{eig(\mathcal{L}(l_2) \cap \mathcal{L}(m))}{eig(\mathcal{L}(l_2))}$ .

### Proposition 5 RecPro3<sup>0</sup>

**Causal footprint recall** ( $rec_A$ ). *Proposition does not hold.*

**Reasoning.** The causal footprint technique defines fitting and non-fitting behavior on an event level, while the proposition states that a trace is either fitting or non-fitting. The added non-fitting behavior could consist of multiple fitting and one non-fitting event.

Table 7: The causal footprints of  $l_8$ . The differences to the footprint of  $m_4$  in Figure 6 are marked in red

	a	b	c	d	e	f
a	#	→	#	→	→	#
b	←	#	→			→
c	#	←	#			#
d	←			#	→	#
e	←			←	#	→
f	#	←	#	#	←	#

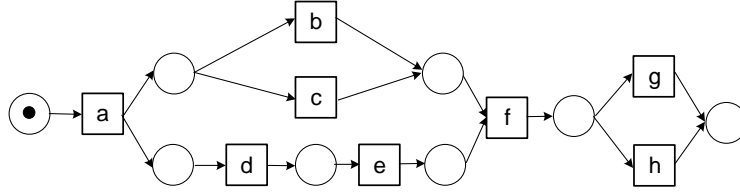


Fig. 7: Petri net  $m_6$

The fitting events could actually decrease the differences between the log and the model while the single non-fitting event introduces an additional difference. The added non-fitting trace in total introduce more similarities than differences and therefore improve recall. Beyond that, it is possible that the other traces in the log already resulted in the appropriate causal relations and the non-fitting event does not change anything.

To illustrate that, consider  $m_4$  and  $l_7$  in Figure 6. We extend  $l_4$  with four clearly unfitting traces:  $l_8 = l_7 \uplus [\langle a, d, b, f \rangle, \langle a, e, b, f \rangle, \langle a, e, c \rangle, \langle a, b, c, e \rangle]$ . The footprint of  $l_8$  is displayed in Table 7. It shows that adding the unfitting traces reveals the parallelism between the activities  $d, e$  and  $b$ , and decreases the differences to the footprint of  $m_4$  in Figure 6 (a). Consequently,  $rec_A(l_8, m_4) = 1 - \frac{4}{36} = 0.88$  and  $rec_A(l_7, m_4) < rec_A(l_8, m_4)$ .

**Token replay recall ( $rec_B$ ). Proposition does not hold.**

**Reasoning.** In the recall formula used during token replay, the number of produced and consumed tokens is in the denominator, while the number of missing and remaining token is in the nominator. If we add a very long non-fitting trace to the event log, which yields a lot of produced and consumed token but only a few missing and remaining token, recall is improved. For long cases with only a few deviations, the approach gives too much weight to the fitting part of the trace.

To illustrate this consider process model  $m_6$  of Figure 7 and log  $l_9 = [\langle a, b, f, g \rangle]$ . This log is not perfectly fitting and therefore results in 6 produced and 6 consumed tokens, as well as 1 missing and 1 remaining token.  $Rec_B(l_9, m_6) = \frac{1}{2}(1 - \frac{1}{6}) + \frac{1}{2}(1 - \frac{1}{6}) = 0.833$ . We extend the log  $l_9$  with non-fitting behavior:  $l_{10} = l_9 \uplus [\langle a, d, e, f, g \rangle]$ . Replaying  $l_{10}$  on  $m_6$  results in  $p = c = 13$ ,  $r = m = 2$  and  $rec_B(l_{10}, m_6) = \frac{1}{2}(1 - \frac{2}{13}) + \frac{1}{2}(1 - \frac{2}{13}) = 0.846$ . Hence,  $rec_B(l_9, m_6) < rec_B(l_{10}, m_6)$ .

**Alignment recall** ( $rec_C$ ). *Proposition does not hold.*

**Reasoning.** Consider model  $m_4$  in Figure 6 and event log  $l_{11} = [\langle f, a \rangle]$ , which result in costs  $fcost(L, M) = 6$  and worst-case cost  $move_L(L) = 2$  and  $move_M(M) = 6$ . Consequently, recall is  $rec_C(l_{11}, m_4) = 1 - \frac{6}{2+6} = 0.25$ . We add a non-fitting trace to the log  $l_{12} = l_{11} \uplus \langle a, b, c, d, e \rangle$ , which shows less deviations than  $l_{11}$ . This results in an additional cost of 1 and additional worst-case costs of  $5 + 6$ . This leads to  $rec_C(l_{12}, m_4) = 1 - \frac{7}{(2+5)+2 \times 6} = 0.64$ . Hence, adding a non-fitting trace with fewer deviations improves recall  $rec_C(l_{11}, m_4) < rec_C(l_{12}, m_4)$  and violates the proposition.

**Behavioral recall** ( $rec_D$ ). *Proposition does not hold.*

**Reasoning.** In the recall formula used during token replay, the number correctly replayed events (TP) is in the denominator as well as in the nominator, while the number of transitions that were forced to fire although they were not enabled (FN) is only in the denominator. If we now add a very long non-fitting trace to the event log, which consists of a large number of correctly replayed events but only a few force firings, recall is improved. Furthermore, remaining tokens during replay are not considered in the formula and cannot lower recall, although these tokens are clear indications for unfitting traces.

Consider  $m_6$  of Figure 7 and log  $l_9 = [\langle a, b, f, g \rangle]$ . After replaying the log on the model there are 3 recorded true positive events and 1 recorded false negative events. This results in  $rec_D(l_9, m_6) = \frac{3}{3+1} = 0.75$ . We extend the log  $l_9$  with non-fitting behavior:  $l_{10} = l_9 \uplus [\langle a, d, e, f, g \rangle]$ . Replaying  $l_{10}$  on  $m_6$  results in 7 recorded true positive events and 2 recorded false negative events. Consequently  $rec_D(l_{10}, m_6) = \frac{7}{7+2} = 0.77$  and  $rec_D(l_9, m_6) < rec_D(l_{10}, m_6)$ .

**Projected recall** ( $rec_E$ ). *Proposition does not hold.*

**Reasoning.** According to this approach, there can be traces that are fitting most of the automata and traces that are fitting only a few automata. The counter-example of  $rec_C$  illustrates this and shows that the proposition does not hold. Projecting the model and both logs on  $\{\langle a, b \rangle\}$  results in recall values for both logs of  $rec_E(l_9|_{\langle a, b \rangle}, m_6|_{\langle a, b \rangle}) = \frac{0}{1}$  and  $rec_E(l_{10}|_{\langle a, b \rangle}, m_6|_{\langle a, b \rangle}) = \frac{1}{2}$ . The additional non-fitting trace in  $l_{10}$  similarly fits the other projected DFAs of size 2, except for the ones containing  $f$ . However, event log  $l_9$  fits none of the projected DFAs of size 2. Therefore adding non-fitting traces that fit most projected automata can increase the aggregated recall.

**Continued parsing measure** ( $rec_F$ ). *Proposition does not hold.*

**Reasoning.** In the recall formula, the number of events  $e$  in the log is present in the denominator as well as in the nominator while the number of missing  $m$  and remaining activated expressions  $r$  is subtracted from the nominator. Similar to token replay recall (B), adding a very long non-fitting trace to the event log which introduces a large number of events but only a few missing and remaining activated expressions, improves recall.

**Eigenvalue recall** ( $rec_G$ ). *Proposition holds.*

**Reasoning.** Trivially, unfitting behavior to the event log cannot change the intersection

between the language of the log and the model and consequently also not their eigenvalue. However, the language of the log might increase which lowers recall.  $\mathcal{L}(l_1) \cap \mathcal{L}(m) = \mathcal{L}(l_2) \cap \mathcal{L}(m)$  if  $l_2 = l_1 \uplus l_3$  and  $\tau(l_3) \subseteq \bar{\tau}(m)$ . Polyvyany et al. [26] proved in lemma 5.6 that the short-circuit measure based on eigenvalue is increasing, and therefore it also holds that  $\frac{eig(\mathcal{L}(l_1) \cap \mathcal{L}(m))}{eig(\mathcal{L}(l_1))} \geq \frac{eig(\mathcal{L}(l_2) \cap \mathcal{L}(m))}{eig(\mathcal{L}(l_2))}$ .

**Proposition 6 RecPro4<sup>0</sup>**

**Causal footprint recall** ( $rec_A$ ). *Proposition holds.*

**Reasoning.** The causal footprint does not account for frequencies of traces and events. Therefore, multiplying the log has no influence on the causal footprint and therefore recall does not change.

**Token replay recall** ( $rec_B$ ). *Proposition holds.*

**Reasoning.** Multiplying the log  $k$  times will equally increase the number of produced, consumed, missing and remaining token. Their ratios stay the same and recall does not change.  $\frac{1}{2}(1 - \frac{k \cdot m}{k \cdot c}) + \frac{1}{2}(1 - \frac{k \cdot r}{k \cdot p}) = \frac{1}{2}(1 - \frac{m}{c}) + \frac{1}{2}(1 - \frac{r}{p})$ ,  $rec_B(l^k, m) = rec_B(l, m)$ .

**Alignment recall** ( $rec_C$ ). *Proposition holds.*

**Reasoning.** Multiplying the event log  $k$  times has no influence on recall since the formula accounts for trace frequency in denominator and nominator. The ratio of replay cost and cost of the worst case scenario stays the same and recall does not change.  $1 - \frac{k \times fcost(L, M)}{k \times move_L(L) + k \times |L| \times move_M(M)} = 1 - \frac{fcost(L, M)}{move_L(L) + |L| \times move_M(M)}$ ,  $rec_C(l^k, m) = rec_C(l, m)$ .

**Behavioral recall** ( $rec_D$ ). *Proposition holds.*

**Reasoning.** Multiplying the log  $k$  times will equally increase the number of correctly replayed events and force fired transitions.

For any  $l \in \mathcal{L}$ ,  $rec_D(l^k, m) = \frac{k \times TP(l, m)}{k \times TP(l, m) + k \times FN(l, m)} = \frac{k \times TP(l, m)}{k \times (TP(l, m) + FN(l, m))} = \frac{TP(l, m)}{(TP(l, m) + FN(l, m))} = rec_D(l, m)$ .

**Projected recall** ( $rec_E$ ). *Proposition holds.*

**Reasoning.** Multiplying the log will equally increase the total number of traces and fitting traces. Their ratio stays the same and recall does not change.  $\frac{k \cdot |[t \in l|_A|t \in DFA(m|_A)]|}{(k \cdot |l|_A)} = \frac{|[t \in l|_A|t \in DFA(m|_A)]|}{|l|_A}$ ,  $rec_E(l^k, m) = rec_E(l, m)$ . Averaging over all subsets of a given length also does not influence this.

**Continued parsing measure** ( $rec_F$ ). *Proposition holds.*

**Reasoning.** Multiplying the log will equally increase the number of parsed events, missing and remaining activated expressions. Their ratio stays the same and recall does not change.  $\frac{1}{2} \frac{k \cdot (e-m)}{k \cdot e} + \frac{1}{2} \frac{k \cdot (e-r)}{k \cdot e} = \frac{1}{2} \frac{(e-m)}{e} + \frac{1}{2} \frac{(e-r)}{e}$ ,  $rec_F(l^k, m) = rec_F(l, m)$ .

**Eigenvalue recall** ( $G$ ). *Proposition holds.*

**Reasoning.** Eigenvalue recall is defined purely on the language of the log and the model and it does not take into account trace frequencies in the log, therefore, this proposition holds.

Table 8: The causal footprints of  $l_{13}$ . Mismatches with the footprint of  $m_4$  are marked in red.

	a	b	c	d	e	f
<b>a</b>	#	→	#	#	#	#
<b>b</b>	←	#	→	→	#	#
<b>c</b>	#	←	#		→	#
<b>d</b>	#	←		#	→	#
<b>e</b>	#	#	←	←	#	→
<b>f</b>	#	#	#	#	←	#

**Proposition 7 RecPro5<sup>+</sup>**

*Causal footprint recall ( $rec_A$ ). Proposition does not hold.*

**Reasoning.** The recall measure based on causal footprints compares the behavior in both directions. If the model has additional behavior that is not present in the log, even in the case where all traces in the log fit the model, the footprint comparison will show the difference and recall will not be maximal.

To illustrate this, consider  $m_4$  in Figure 6. We compute recall for  $m_4$  and  $l_{13} = [\langle a, b, c, d, e, f \rangle, \langle a, b, d, c, e, f \rangle]$ . The traces in  $l_{13}$  perfectly fit process model  $m_4$ . The footprint of  $l_{13}$  is shown in Table 8. Comparing it to the footprint of  $m_4$  in Table 6 (a) shows mismatches although  $l_{13}$  is perfectly fitting. These mismatches are caused by the fact that the log does not show all possible paths of the model and therefore the footprint cannot completely detect the parallelism of the model. Consequently,  $rec_A(l_{13}, m_4) = 1 - \frac{10}{36} = 0.72 \neq 1$  even though  $\tau(l) \subseteq \tau(m)$ .

Van der Aalst mentions in [1] that checking conformance using causal footprints is only meaningful if the log is complete in term of directly followed relations.

*Token replay recall ( $rec_B$ ). Proposition holds.*

**Reasoning.** There will be no missing and remaining tokens if all traces in the log fit the model. Hence, recall is maximal, if  $\tau(l) \subseteq \tau(m)$ .  $\frac{1}{2}(1 - \frac{0}{p}) + \frac{1}{2}(1 - \frac{0}{c}) = 1$ . Note, that again we make the assumption that perfectly fitting behavior is replayed perfectly. Due to the nondeterministic nature of replay in the presence of silent and duplicate transition, this is not guaranteed.

*Alignment recall ( $rec_C$ ). Proposition holds.*

**Reasoning.** The alignments only consist of synchronous moves if all traces in the log fit the model. Consequently, the alignment costs  $fcost(L, M)$  are 0 and recall is maximal.

$$rec_C = 1 - \frac{fcost(L, M)}{(move_L(L) + |L| \times move_M(M))} = 1 - \frac{0}{(move_L(L) + |L| \times move_M(M))} = 1, \text{ if } \tau(l) \subseteq \tau(m).$$

*Behavioral recall ( $rec_D$ ). Proposition holds.*

**Reasoning.** If all traces in log  $l$  fit model  $m$ , then  $FN(l, m) = 0$ . As a result,  $rec_D(l, m) = \frac{TP(l, m)}{TP(l, m) + FN(l, m)} = \frac{TP(l, m)}{TP(l, m)} = 1$ .

**Projected recall** ( $rec_E$ ). *Proposition holds.*

**Reasoning.** If all traces in the log fit the model, the number of correctly replayed traces equals the number of traces in the log  $|\{t \in l_A | t \in DFA(m|_A)\}| = |l_A|$ , if  $\tau(l) \subseteq \tau(m)$  and recall is maximal. The approach also defines that recall is maximal if the log is empty. [23].

**Continued parsing measure** ( $rec_F$ ). *Proposition does not hold.*

**Reasoning.** Flower models consisting of one place that connects to all transitions do not have a final place. Translating this model into a causal matrix will cause that there is no activity with an empty output expression. Hence, after replaying the fitting log, there will always be remaining activated output expressions and recall is not maximal.

**Eigenvalue recall** ( $G$ ). *Proposition holds.*

**Reasoning.** Proven in Corollary 5.15 of [26].

### A.3 Detailed Results of the Precision Measure Evaluation

#### Proposition 1 DetPro<sup>+</sup>

**Soundness** ( $prec_H$ ). *Proposition does not hold.*

**Reasoning.** The formula divides the unique traces observed in the log by the unique paths through the model. If the model contains loops there are infinitely many unique paths and precision is not defined.

**Simple behavioral appropriateness** ( $prec_I$ ). *Proposition does not hold.*

**Reasoning.** Shown to be non-deterministic in [29] and was already stressed in the original paper [27] that introduced the measure.

**Advanced behavioral appropriateness** ( $prec_J$ ). *Proposition does not hold.*

**Reasoning.** Shown to be undefined for some combinations of logs and models in [29]. Note, that implementation of the approach in the process mining tool ProM<sup>6</sup> defines precision for these combinations and is, therefore, deterministic. However, in this paper, we only consider the approach as formally defined in the paper [27].

**ETC-one/ETC-rep** ( $prec_K$ ). *Proposition does not hold.*

**Reasoning.** For the construction of the state space and its escaping edges, the aligned log is used. In the case of multiple optimal alignments, one (ETC-one) or a set of representative alignments (ETC-rep) is used to construct the state space. During regular conformance checking based on alignments, all optimal alignments are equal. However different alignments can lead to different escaping edges and therefore to different precision measures.

Consider process  $m_7$  in Figure 8 along with event log  $l_{14} = [\langle a, g \rangle]$ . It is clear that the log does not fit the process model and after aligning log and model there are three possible aligned traces:  $\sigma_1 = \langle a, b, c, g \rangle$ ,  $\sigma_2 = \langle a, d, e, g \rangle$  and  $\sigma_3 = \langle a, d, f, g \rangle$ . The

<sup>6</sup> <http://www.promtools.org>

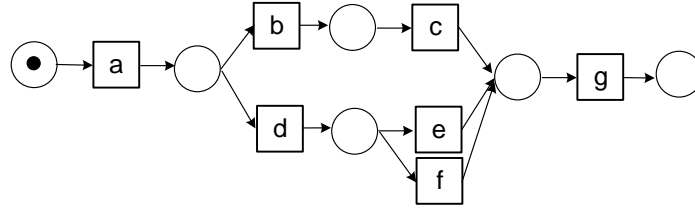


Fig. 8: Petri net  $m_7$

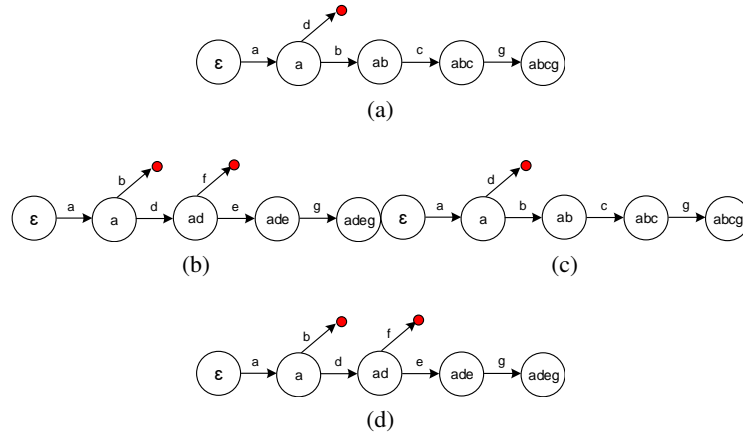


Fig. 9: Two alignment automata describing the state space of  $\sigma_1 = \langle a, b, c, g \rangle$  (a) and  $\sigma_2 = \langle a, d, e, g \rangle$  (b).

ETC-one approach randomly picks one of the traces and construct the corresponding alignment automaton. The automata of  $\sigma_1$  and  $\sigma_2$  in Figure 9 show the different escaping edges that result from both traces. As a result, precision is different for these two aligned traces:  $prec_K(\sigma_1, m_7) = \frac{4}{5} = 0.8$  and  $prec_K(\sigma_2, m_7) = \frac{4}{6} = 0.67$ .

**ETC-all** ( $prec_L$ ). *Proposition holds.*

**Reasoning.** For ETC-all all optimal alignments are used. Which leads to a complete state space and a deterministic precision measure.

**Behavioral specificity** ( $prec_M$ ). *Proposition does not hold.*

**Reasoning.** If during the replay of the trace duplicate or silent transitions are encountered, the approach explored which of the available transitions enables the next event in the trace. If no solution is found, one of the transitions is randomly fired, which can lead to different recall values for traces with the same behavior.

Furthermore, to balance the proportion of negative and positive events in the log, the algorithm induces the log with the calculated negative events based on a probabilistic parameter. Only if this parameter is set to 1 all negative events are added to the log. Hence the recall measure is non-deterministic for parameter settings smaller than 1.

Finally, it is possible that the negative event induction algorithm does not induce any negative events. This, for example, happens when the algorithm assesses that all activity types in the log are in parallel. When there are no negative events found, it follows from the definition that precision is  $\frac{0}{0}$  and thus undefined.

**Behavioral precision** ( $prec_N$ ). *Proposition does not hold.*

**Reasoning.**  $prec_N$  uses the same non-deterministic replay procedure and the same negative event induction approach (possibly also non-deterministic, depending on parameter settings) as  $prec_M$ .

$prec_N$  does not have the same problem as  $prec_M$  with regards to being undefined when there are no negative events, as this measure additionally has the number of true positives in the formula.

**Weighted negative event precision** ( $prec_O$ ). *Proposition does not hold.*

**Reasoning.**  $prec_O$  uses a non-deterministic replay procedure, which is detailed in [8]. Therefore, the precision calculation is non-deterministic.

**Projected precision** ( $prec_P$ ). *Proposition holds.*

**Reasoning.** This technique projects the behaviors of the log and the model onto subsets of activities and compares their deterministic finite automata to calculate precision. The sub-logs and models are created by projection on a subset of activities which is a deterministic process. Moreover, the process of creating a deterministic automaton is also deterministic. There is a unique DFA which has the minimum number of states, called the minimal automaton. Therefore, the computation of the average precision value over all subsets is also deterministic.

**Anti-alignment precision** ( $prec_Q$ ). *Proposition holds.*

**Reasoning.** Precision is computed based on the maximal anti-alignment. Even if there are multiple maximal anti-alignments, the distance will always be maximal and, therefore, precision is deterministic. Note, that we assume in case of non-fitting behavior that the log is first aligned before evaluating this proposition.

**Eigenvalue precision** ( $prec_R$ ). *Proposition holds.*

**Reasoning.** The measure compares the languages of the model and the language of the process model. These have to be irreducible, to compute their eigenvalue. Since the language of an event log is not irreducible, Polyvyanyy et al. [26] introduce a short-circuit measure over languages and proof that it is a deterministic measure over any arbitrary regular language.

## Proposition 2 BehPro<sup>+</sup>

**Soundness** ( $prec_H$ ). *Proposition holds.*

**Reasoning.** The behavior of the model is defined as sets of traces  $\tau(m)$ , which abstracts from the representation of the process model itself.



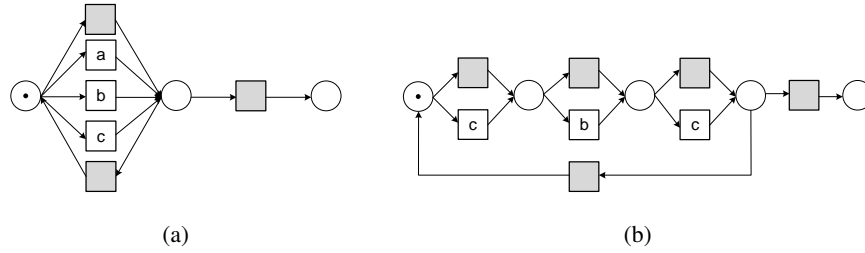


Fig. 10: Two process models  $m_8$  (a) and  $m_9$  (b) that show the same behavior but different representations.

**Simple behavioral appropriateness** ( $prec_I$ ). *Proposition does not hold.*

**Reasoning.** A counter-example to Axiom 4 (as introduced in [29]), which is equivalent to  $\mathbf{BehPro}^+$ , was shown in [29].

**Advanced behavioral appropriateness** ( $prec_J$ ). *Proposition does not hold.*

**Reasoning.** A counter-example to Axiom 4 (as introduced in [29]), which is equivalent to  $\mathbf{BehPro}^+$ , was shown to hold in [29].

**ETC-one/ETC-rep** ( $prec_K$ ). *Proposition does not hold.*

**Reasoning.** A counter-example to Axiom 4 (as introduced in [29]), which is equivalent to  $\mathbf{BehPro}^+$ , was shown in [29].

**ETC-all** ( $prec_L$ ). *Proposition does not hold.*

**Reasoning.** This technique depends on the path taken through the model to examine the visited states and its escaping edges. One can think of Petri nets with the same behavior but described by different paths through the net which then also results in different escaping edges and hence different precision. The two models shown in Figure 10 prove this case.

**Behavioral specificity** ( $prec_M$ ). *Proposition does not hold.*

**Reasoning.** If duplicate or silent transitions are encountered while replaying a trace, the approach checks if one of the available transitions enables the next event in the trace. Whether this is the case can depend on the structure of the model.

**Behavioral precision** ( $prec_N$ ). *Proposition does not hold.*

**Reasoning.** For  $prec_M$ ,  $\mathbf{BehPro}^+$  did not hold because of its replay procedure.  $prec_N$  uses the same replay procedure as  $prec_M$ .

**Weighted negative event precision** ( $prec_O$ ). *Proposition does not hold.*

**Reasoning.** Like with  $prec_M$  and  $prec_N$  the outcome of the replay procedure can be impacted by duplicate transitions and by silent transitions. Therefore, this proposition does not hold.

**Projected precision** ( $prec_P$ ). *Proposition holds.*

**Reasoning.** This technique translates the event log as well as the process model into deterministic finite automata before computing recall (recall that the minimal deterministic automaton is unique due to the Myhill–Nerode theorem). Therefore, it is independent of the representation of the model itself.

**Anti-alignment precision** ( $prec_Q$ ). *Proposition holds.*

**Reasoning.** The authors define an anti-alignment as a run of a model which differs sufficiently from the observed traces in a log. This anti-alignment is solely constructed based on the possible behavior of the process model and the observed behavior of the log. It is independent of the structure of the net.

**Eigenvalue precision** ( $prec_R$ ). *Proposition holds.*

**Reasoning.** The approach calculates precision based on the languages of the model and the language of the process model. This abstracts from the representation of the process model and, consequently, the proposition holds.

### **Proposition 8 PrecPro1<sup>+</sup>**

**Soundness** ( $prec_H$ ). *Proposition holds.*

**Reasoning.** This proposition holds since, removing behavior from the model that does not happen in the event log decreases the set of traces allowed by the model  $|\tau(m_1)| \geq |\tau(m_2)|$ , while the set of traces of the event log complying with the model stays the same  $|\tau(l) \cap \tau(m_1)| = |\tau(l) \cap \tau(m_2)|$ .

**Simple behavioral appropriateness** ( $prec_I$ ). *Proposition does not hold.*

**Reasoning.** **BehPro<sup>+</sup>** does not hold, which implies that **PrecPro1<sup>+</sup>** does not hold.

**Advanced behavioral appropriateness** ( $prec_J$ ). *Proposition does not hold.*

**Reasoning.** **BehPro<sup>+</sup>** does not hold, which implies that **PrecPro1<sup>+</sup>** does not hold.

**ETC-one/ETC-rep** ( $prec_K$ ). *Proposition does not hold.*

**Reasoning.** A counter-example to Axiom 2 (as introduced in [29]) was presented in [29]. Since **PrecPro1<sup>+</sup>** is a generalization of Axiom 2, the same counter-example shows that **PrecPro1<sup>+</sup>** does not hold. Furthermore, **BehPro<sup>+</sup>** does not hold, which implies that **PrecPro1<sup>+</sup>** does not hold.

**ETC-all** ( $prec_L$ ). *Proposition does not hold.*

**Reasoning.** A counter-example to Axiom 2 (as introduced in [29]) was presented in [29]. Since **PrecPro1<sup>+</sup>** is a generalization of Axiom 2, this implies that **PrecPro1<sup>+</sup>** does not hold. Furthermore, **BehPro<sup>+</sup>** does not hold, which implies that **PrecPro1<sup>+</sup>** does not hold.

**Behavioral specificity** ( $prec_M$ ). *Proposition does not hold.*

**Reasoning.** **BehPro<sup>+</sup>** does not hold, which implies that **PrecPro1<sup>+</sup>** does not hold.

**Behavioral precision** ( $prec_N$ ). *Proposition does not hold.*

**Reasoning.** **BehPro**<sup>+</sup> does not hold, which implies that **PrecPro1**<sup>+</sup> does not hold.

**Weighted negative event precision** ( $prec_O$ ). *Proposition does not hold.*

**Reasoning.** A counter-example to Axiom 2 (as introduced in [29]) was presented in [29]. Since **PrecPro1**<sup>+</sup> is a generalization of Axiom 2, this implies that **PrecPro1**<sup>+</sup> does not hold. Furthermore, **BehPro**<sup>+</sup> does not hold, which implies that **PrecPro1**<sup>+</sup> does not hold.

**Projected precision** ( $prec_P$ ). *Proposition does not hold.*

**Reasoning.** A counter-example to Axiom 2 (as introduced in [29]) was presented in [29]. To illustrate this, the paper considers a model with a length-one-loop and its more precise corresponding model that unrolled the loop up to two executions. The DFA of the unrolled model will contain more states since the future allowed behavior depends on the number of executions of the looping activity, while the DFA of the initial model will contain only one state for this activity. This can cause that the unrolled model is considered less precise which violates the proposition.

**Anti-alignment precision** ( $prec_Q$ ). *Proposition holds.*

**Reasoning.** The behavior of the model that is not observed in the log will become the anti-alignment between the log and the model. The distance between the log and the anti-alignment is big which leads to low precision. If this behavior is removed from the model an anti-alignment closer to the log is found which leads to a higher precision.

**Eigenvalue precision** ( $prec_R$ ). *Proposition holds.*

**Reasoning.** Proven in Lemma 5.6 of [26].

### **Proposition 9 PrecPro2**<sup>+</sup>

**Soundness** ( $prec_H$ ). *Proposition holds.*

**Reasoning.** Adding fitting behavior to the event log can lead to additional unique process executions that comply with the process model:  $|\tau(l_1) \cap \tau(m)| \leq |\tau(m)| \leq |\tau(l_2)|$ . This cannot lower precision according to the definition of soundness.  $\frac{|\tau(l_1) \cap \tau(m)|}{|\tau(m)|} \leq \frac{|\tau(l_2) \cap \tau(m)|}{|\tau(m)|}$ , if  $l_2 = l_1 \uplus l_3$ .

**Simple behavioral appropriateness** ( $prec_I$ ). *Proposition does not hold.*

**Reasoning.** This approach does not consider whether the behavior of the log fits the model or not, but it focuses on the average number of enabled transitions during log replay. It is possible that the additional behavior enables a large number of transitions, this increases the average count and thereby lowers precision.

**Advanced behavioral appropriateness** ( $prec_J$ ). *Proposition holds.*

**Reasoning.** Adding fitting behavior to the log can only increase the intersection between the follow relations of the log and the model.  $|S_F^{l_1} \cap S_F^m| \leq |S_F^{l_2} \cap S_F^m|$  and  $|S_P^{l_1} \cap S_P^m| \leq |S_P^{l_2} \cap S_P^m|$ , if  $l_2 = l_1 \uplus l_3$  and  $\tau(l_3) \subseteq \tau(m)$ . Hence by adding fitting behavior to the log precision can only be increased.

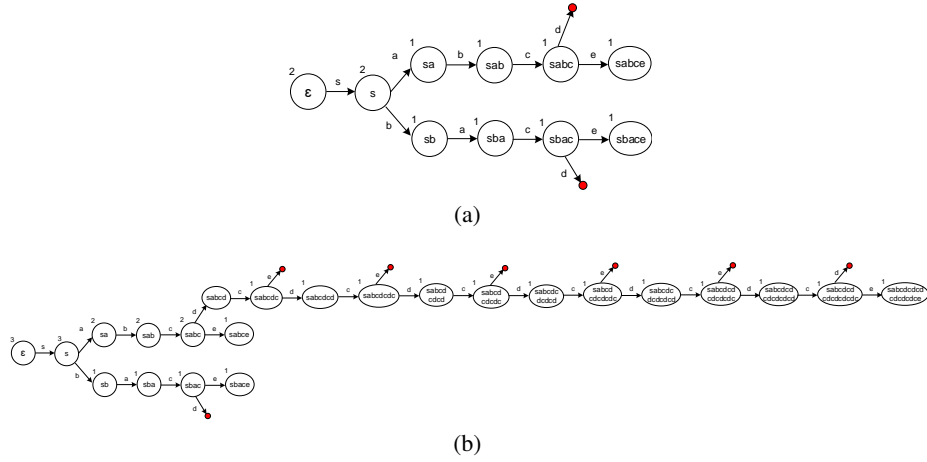


Fig. 11: Two alignment automata describing the state space of  $m_1$  and  $l_{15}$  (a) and the state space of  $m_1$  and  $l_{10}$  (b).

**ETC-one/ETC-rep** ( $prec_K$ ). *Proposition does not hold.*

**Reasoning.** A counter-example to Axiom 5 (as introduced in [29]) was presented in [29]. Since **PrecPro2<sup>+</sup>** is a generalization of Axiom 5, this implies that **PrecPro2<sup>+</sup>** does not hold.

**ETC-all** ( $prec_L$ ). *Proposition does not hold.*

**Reasoning.** The conclusions drawn in [29] for ETC-one can also be transferred to ETC-all. When adding fitting behavior it is possible that the new trace visits states that introduce a lot of new escaping edges. The increase in escaping edges is bigger than the increase in non-escaping ones which lowers precision. Consider process model  $m_1$  in Figure 1 (a) and the event log  $l_{15} = [\langle s, a, b, c, e \rangle, \langle s, b, a, c, e \rangle]$  and its extension with a fitting trace  $l_{16} = l_{15} \uplus [\langle s, a, b, c, d, c, d, c, d, c, d, c, e \rangle]$ . Note, that the “start” and “end” activities of  $m_1$  are abbreviated to “s” and “e”. The corresponding automata in Figure 11 show that the additional fitting trace adds additional states and escaping edges. This decreases precision:  $prec_L(l_{15}, m_1) = \frac{12}{14} = 0.857$  and  $prec_L(l_{16}, m_1) = \frac{31}{37} = 0.838$ .

**Behavioral specificity** ( $prec_M$ ). *Proposition does not hold.*

**Reasoning.** This proposition does not hold when the additional fitting trace introduces proportionally more negative events that could actually fire (FP) than correctly identified negative events (TN). To illustrate this, consider process model  $m_{10}$  in Figure 12 and event log  $l_{16} = [\langle a, b, b, d \rangle, \langle a, b, c, d \rangle]$ . Table 9 shows the negative events calculated for the log. We assume a window size that equals the longest trace in the event log and we generate the negative events with probability 1. After replaying the log on the process model we record  $FP(l_{16}, m_{10}) = 10$  and  $TN(l_{16}, m_{10}) = 8$ . Hence,  $\frac{TN(l_{16}, m_{10})}{TN(l_{16}, m_{10}) + FP(l_{16}, m_{10})} = \frac{8}{18} = 0.444$ . We extend the log with fitting trace  $l_{17}$ , i.e.,  $l_{17} = l_{16} \uplus [\langle a, b, c, b, b, b, d \rangle]$ . The negative events calculated for  $l_{17}$  are

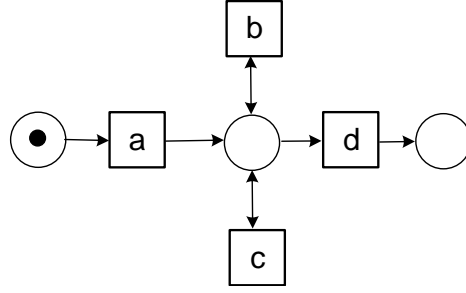


Fig. 12: Process model  $m_{10}$

Table 9: The traces of  $l_{16}$  with the corresponding negative events.

a	b	b	d	a	b	c	d
$\neg b$	$\neg a$	$\neg a$	$\neg a$	$\neg b$	$\neg a$	$\neg a$	$\neg a$
$\neg c$	$\neg c$	$\neg d$	$\neg b$	$\neg c$	$\neg c$	$\neg d$	$\neg b$
$\neg d$	$\neg d$		$\neg c$	$\neg d$	$\neg d$		$\neg c$

displayed in Table 10 and replaying it on  $m_{10}$  results in  $FP(l_{17}, m_{10}) = 31$  and  $TN(l_{17}, m_{10}) = 23$ . Consequently,  $\frac{TN(l_{17}, m_{10})}{TN(l_{17}, m_{10}) + FP(l_{17}, m_{10})} = \frac{23}{54} = 0.426$ . Although  $l_{11}$  is fitting, it introduces more negative events that are actually enabled during replay. Therefore,  $prec_M(l_{16}, m_{10}) > prec_M(l_{17}, m_{10})$ , which violates the proposition.

**Behavioral precision ( $prec_N$ ).** *Proposition does not hold.*

**Reasoning.** The counter-example of  $prec_M$  and can also be used to show that this proposition is violated. During replay of  $l_{16}$  and  $l_{17}$  we also count the positive events that can be correctly replayed (TP). This results in  $TP(l_{16}, m_{10}) = 8$  and  $TP(l_{16}, m_{10}) = 17$ . When we calculate precision, we obtain  $prec_N(l_{16}, m_{10}) = \frac{TP(l_{16}, m_{10})}{TP(l_{16}, m_{10}) + FP(l_{16}, m_{10})} = \frac{8}{8+10} = 0.44$  and  $prec_N(l_{17}, m_{10}) = \frac{17}{17+31} = 0.35$ . The additional fitting trace lowers precision:  $prec_N(l_{16}, m_{10}) > prec_N(l_{17}, m_{10})$ .

**Weighted negative event precision ( $prec_O$ ).** *Proposition does not hold.*

**Reasoning.** Consider the same counter-example as that we provided for  $prec_M$ . Negative events are weighted by the size of the longest matching window of events. For the long repetition of  $b$ -events in  $\langle a, b, c, b, b, b, b, d \rangle$ , the negative events for  $a$ ,  $c$  and  $d$  have weight 2 due to the 2 consecutive  $b$ 's in trace  $\langle a, b, b, d \rangle$ . Since the negative events in  $l_3$  that caused the precision to go up when  $l_3$  was added to  $l_1$  have above average weight, the weighting does not invalidate the counter-example.

**Projected precision ( $prec_P$ ).** *Proposition does not hold.*

**Reasoning.** A counter-example to Axiom 5 (as introduced in [29]) was presented in [29]. Since **PrecPro2<sup>+</sup>** is a generalization of Axiom 5, this implies that **PrecPro2<sup>+</sup>** does not hold.

Table 10: The traces of  $l_{17}$  with the corresponding negative events.

a	b	b	d	a	b	c	d	a	b	c	b	b	b	b	b	d
$\neg b$	$\neg a$	$\neg a$	$\neg a$	$\neg b$	$\neg a$	$\neg a$	$\neg a$	$\neg b$	$\neg a$	$\neg a$	$\neg a$	$\neg a$	$\neg a$	$\neg a$	$\neg a$	$\neg a$
$\neg c$	$\neg c$	$\neg d$	$\neg b$	$\neg c$	$\neg c$	$\neg d$	$\neg b$	$\neg c$	$\neg c$	$\neg d$	$\neg c$	$\neg c$	$\neg c$	$\neg c$	$\neg c$	$\neg b$
$\neg d$	$\neg d$		$\neg c$	$\neg d$	$\neg d$		$\neg c$	$\neg d$	$\neg d$		$\neg d$	$\neg d$	$\neg d$	$\neg d$	$\neg d$	$\neg c$

**Anti-alignment precision** ( $prec_Q$ ). *Proposition does not hold.*

**Reasoning.** A counter-example to Axiom 5 (as introduced in [29]) was presented in [29]. Since **PrecPro2<sup>+</sup>** is a generalization of Axiom 5, this implies that **PrecPro2<sup>+</sup>** does not hold.

**Eigenvalue precision** ( $prec_R$ ). *Proposition holds.*

**Reasoning.** Proven in Lemma 5.6 of [26].

### Proposition 10 PrecPro3<sup>0</sup>

**Soundness** ( $prec_H$ ). *Proposition holds.*

**Reasoning.** Adding non-fitting behavior to the event log cannot lead to additional process executions, which comply with the process model. Hence, it cannot lower precision according to the definition of soundness.  $|\tau(l_1) \cap \tau(m)| = |\tau(l_2) \cap \tau(m)|$ , if  $l_2 = l_1 \uplus l_3$  and  $\tau(l_3) \subseteq \bar{\tau}(m)$ .

**Simple behavioral appropriateness** ( $prec_I$ ). *Proposition does not hold.*

**Reasoning.** Adding non-fitting behavior to the event log might change the average number of enabled transitions during log replay and therefore precision. Note that this scenario was not considered by the approach since the authors assume a fitting log [27].

**Advanced behavioral appropriateness** ( $prec_J$ ). *Proposition does not hold.*

**Reasoning.** This approach records relations between activities and compares these relations between the model and the log. Hence, it does not consider entire traces to be fitting or non-fitting but refines it to an activity level. Therefore, it is possible that non-fitting traces contain fitting events that improve precision. For example, the non-fitting traces change a *never* follows relation of the event log to a *sometimes* follows relation that matches the process model. Consequently, precision increases and violates this proposition.

**ETC-one/ETC-rep** ( $prec_K$ ). *Proposition does not hold.*

**Reasoning.** Before the alignment automaton is constructed the log is aligned to ensure that the traces fit the model. Adding non-fitting behavior can possibly lead to new alignments that lead to new escaping edges and change precision.

Consider model  $m_7$  in Figure 8 and alignment automata in Figure 9 (a) corresponding to trace  $\langle a, b, c, g \rangle$ . Adding the unfitting trace  $\langle a, d, g \rangle$  results in the aligned trace  $\langle a, d, e, g \rangle$  or  $\langle a, d, f, g \rangle$ . Either of the two aligned traces introduces new states into the alignment automaton. Additionally, the new trace alters the weights of each state. Therefore, even though both automata contain one escaping edge, precision changes.

**ETC-all** ( $prec_L$ ). *Proposition does not hold.*

**Reasoning.** The counter-example presented for  $prec_K$  also shows that this proposition does not hold for this approach. The unfitting trace  $\langle a, d, g \rangle$  results in the aligned trace  $\langle a, d, e, g \rangle$  or  $\langle a, d, f, g \rangle$ . This variant of ETC precision uses both of the two aligned traces to construct the alignment automaton. This introduces new states, alters the weights of the states, removes the escaping edge and changes precision.

**Behavioral specificity** ( $prec_M$ ). *Proposition does not hold.*

**Reasoning.** Negative events describe behavior that was not allowed during process execution. They are constructed based on the behavior observed in the event log. From this point of view, non-fitting behavior is behavior that should have been described by negative events, but since it was observed in the event log, the algorithm does not define it as negative events anymore. Hence adding non-fitting behavior  $l_3$  to event log  $l_1$  decreases the number of correctly identified negative events (TN) in the traces of  $l_1$ .

Furthermore, this measure accounts for the number of negative events that actually could fire during trace replay (FP). These false positives are caused by the fact that behavior is shown in the model but not observed in the log. Although the trace is not fitting when considered as a whole, certain parts of the trace can fit the model, and these parts can represent the previously missing behavior in the event log that leads to the wrong classification of negative events. Adding these non-fitting traces  $l_3$  can, therefore, lead to a decrease in false positives in the traces of  $l_1$  and changes precision.

**Behavioral precision** ( $prec_N$ ). *Proposition does not hold.*

**Reasoning.** As shown in the reasoning for  $prec_M$ , adding non-fitting traces  $l_3$  to a fitting log  $l_1$  can decrease the number of false positives FP in the negatives events that were generated for the traces of  $l_1$ .

**Weighted negative event precision** ( $prec_O$ ). *Proposition does not hold.*

**Reasoning.** As shown in the reasoning for  $prec_M$ , adding non-fitting traces  $l_3$  to a fitting log  $l_1$  can decrease the number of false positives FP in the negatives events that were generated for the traces of  $l_1$ . Weighing the negative events does not change this.

**Projected precision** ( $prec_P$ ). *Proposition does not hold.*

**Reasoning.** Since projected precision calculates precision based on several projected sub-models and sub-logs, it is possible that unfitting behavior fits some of these sub-models locally. To illustrate this consider a model with the language  $\tau(m_{11}) = \{\langle a, b \rangle, \langle c, d \rangle\}$  and  $l_{18} = [\langle a, b \rangle]$ . It is clear, that the model is not perfectly precise since trace  $\langle c, d \rangle$  is not observed in the event log. Hence, if we project our model and log on  $\{c, d\}$ , precision will be 0 for this projection.

We extend the log with an unfitting trace  $l_{19} = l_{18} \uplus [\langle c, d, a \rangle]$ . Projecting  $l_{19}$  on  $\{c, d\}$  results in a precision value of 1. Since this approach aggregates the precision over several projections, it is clear, that unfitting behavior can improve precision.

**Anti-alignment precision** ( $prec_Q$ ). *Proposition does not hold.*

**Reasoning.** By definition anti-alignments always fit the model. Consequently, there will always be a distance between a non-fitting trace and an anti-alignment. Adding

non-fitting behavior to the event log will, therefore, change precision. The proposition does not require  $l_1$  to be fitting. Therefore it could be the case that  $l_1$  has a trace that has a higher distance to behavior that is allowed by  $m$  than what can be found amongst the traces of  $l_3$ . Note that this scenario is not considered by the approach since the authors assume a fitting log [13]. However, also after aligning the log, it might still change precision by resulting in alignments that were not contained in the initial log.

**Eigenvalue precision** ( $prec_R$ ). *Proposition holds.*

**Reasoning.** The measure is defined as  $\frac{eig(\tau(m) \cap \tau(l))}{eig(\tau(m))}$ . As  $\tau(m) \cap \tau(l)$  does not change when adding non-fitting traces to  $l$ , neither does the measure change.

### Proposition 11 PrecPro4<sup>0</sup>

**Soundness** ( $prec_H$ ). *Proposition holds.*

**Reasoning.** Since the approach only considers unique model executions, duplicating the log has no effect on precision.

**Simple behavioral appropriateness** ( $prec_I$ ). *Proposition holds.*

**Reasoning.** Since the number of traces  $n_i$  is present in the denominator as well as the nominator of the formula duplication has no effect on precision.  $\frac{\sum_{i=1}^k n_i (|T_V| - x_i)}{(|T_V| - 1) \cdot \sum_{i=1}^k n_i}$ . Here we assume that if a trace is perfectly replayable it will also be replayed perfectly.

**Advanced behavioral appropriateness** ( $prec_J$ ). *Proposition holds.*

**Reasoning.** The sometimes follows relations will not change by duplicating the event log. Hence, the result is unaffected.

**ETC-one/ETC-rep** ( $prec_K$ ). *Proposition holds.*

**Reasoning.** The weight of escaping and non-escaping edges is calculated based on the trace frequency. However, since the distribution of the traces does not change the weight of both edge types grows proportionally and precision does not change.

**ETC-all** ( $prec_L$ ). *Proposition holds.*

**Reasoning.** The weight of escaping and non-escaping edges is calculated based on the trace frequency. However, since the distribution of the traces does not change the weight of both edge types grows proportionally and precision does not change.

**Behavioral specificity** ( $prec_M$ ). *Proposition holds.*

**Reasoning.** Multiplying the event log  $k$  times leads to a proportional increase in true negatives and false positives. Consequently, precision does not change.

$$\frac{k \times TN(l, m)}{k \times (TN(l, m) + FP(l, m))} = \frac{TN(l, m)}{TN(l, m) + FP(l, m)}, \text{ hence } prec_M(l^k, m) = prec_M(l, m).$$

**Behavioral precision** ( $prec_N$ ). *Proposition holds.*

**Reasoning.** Multiplying the event log  $k$  times leads to a proportional increase in true positives and false positives. Consequently, precision does not change.

$$\frac{k \times TP(l, m)}{k \times (TP(l, m) + FP(l, m))} = \frac{TP(l, m)}{TP(l, m) + FP(l, m)}, \text{ hence } prec_N(l^k, m) = prec_N(l, m).$$



**Weighted negative event precision** ( $prec_O$ ). *Proposition holds.*

**Reasoning.** Multiplying the event log  $k$  times leads to a proportional increase in true negatives and false positives. Consequently precision does not change.

$$\frac{k \times TN(l,m)}{k \times (TN(l,m) + FP(l,m))} = \frac{TN(l,m)}{TN(l,m) + FP(l,m)}, \text{ hence } prec_M(l^k, m) = prec_M(l, m).$$

**Projected precision** ( $prec_P$ ). *Proposition holds.*

**Reasoning.** The precision measure does not consider trace frequency and therefore will not be changed by duplicating the event log.

**Anti-alignment precision** ( $prec_Q$ ). *Proposition holds.*

**Reasoning.** This approach sums for each trace in the log the distance between anti-alignment and trace. This sum is averaged over the number of traces in the log and consequently, duplication of the log will not change precision.

**Eigenvalue precision** ( $R$ ). *Proposition holds.*

**Reasoning.** Eigenvalue precision is defined purely on the language of the log and the model and it does not take into account trace frequencies in the log, therefore, this proposition holds.

## Proposition 12 PrecPro5<sup>+</sup>

**Soundness** ( $prec_H$ ). *Proposition holds.*

**Reasoning.** If the model allows for the behavior observed and nothing more each, unique process execution corresponds to a unique path through the model  $\tau(l) = \tau(m)$ . Therefore precision is maximal:  $|\tau(l)| / |\tau(m)| = 1$ .

**Simple behavioral appropriateness** ( $prec_I$ ). *Proposition does not hold.*

**Reasoning.** This approach only considers strictly sequential models to be perfectly precise. If the model has choices, loops or concurrency, then, multiple transitions might be enabled during replay even if the model only allows only for the observed behavior. As a result, precision is not maximal.

**Advanced behavioral appropriateness** ( $prec_J$ ). *Proposition holds.*

**Reasoning.** If the model allows for only the behavior observed and nothing more, the set of sometimes follows/precedes relations of the model are equal to the ones of the event log.  $S_F^l \cap S_F^m = S_F^m$  and  $S_P^l \cap S_P^m = S_P^m$ , if  $\tau(l) = \tau(m)$ . Consequently precision is maximal.

**ETC-one/ETC-rep** ( $prec_K$ ). *Proposition does not hold.*

**Reasoning.** Consider a model with a choice between two a-labeled transitions and a trace  $\{a\}$ . When constructing the alignment automaton, there will be an escaping edge for the other a-labeled transition. Similar problems may arise with silent transitions.

**ETC-all** ( $prec_L$ ). *Proposition does not hold.*

**Reasoning.** The counter-example for  $prec_K$  shows that also  $prec_L$  violates this proposition.

**Behavioral specificity** ( $prec_M$ ). *Proposition holds.*

**Reasoning.** When the model allows for only observed behavior (i.e., in  $l$ ), then  $FP(l, m) = 0$ , as false positives are caused by the fact that behavior is shown in the model but not observed in the log. Therefore,  $\tau(l) = \tau(m) \Rightarrow prec_M(l, m) = \frac{TN(l, m)}{TN(l, m) + FP(l, m)} = \frac{TN(l, m)}{TN(l, m) + 0} = 1$ .

**Behavioral precision** ( $prec_N$ ). *Proposition holds.*

**Reasoning.** When the model allows for only observed behavior (i.e., in  $l$ ), then  $FP(l, m) = 0$ , as false positives are caused by the fact that behavior is shown in the model but not observed in the log. Therefore,  $\tau(l) = \tau(m) \Rightarrow prec_N(l, m) = \frac{TP(l, m)}{TP(l, m) + FP(l, m)} = \frac{TP(l, m)}{TP(l, m) + 0} = 1$ .

**Weighted negative event precision** ( $prec_O$ ). *Proposition holds.*

**Reasoning.** See the reasoning for  $prec_N$  above. The weighing of negative events doesn't change the fact that false positives cannot occur when  $\tau(l) = \tau(m)$ , therefore the same reasoning applies to  $prec_O$ .

**Projected precision** ( $prec_P$ ). *Proposition holds.*

**Reasoning.** If the model allows for only the behavior observed and nothing more, the two automata describing the behavior of the log and the model are exactly the same:  $DFA(m|_A) = DFA(l|_A) = DFAc(l, m, A)$ , if  $\tau(l) = \tau(m)$ . Hence, precision is maximal.

**Anti-alignment precision** ( $prec_Q$ ). *Proposition holds.*

**Reasoning.** If the model allows for the behavior observed and nothing more, each anti-alignment will exactly match its corresponding trace. Consequently, the distance between the log and the anti-alignment is minimal and precision maximal.

**Eigenvalue precision** ( $prec_R$ ). *Proposition holds.*

**Reasoning.** Proven in Corollary 5.15 of [26].

### **Proposition 13 PrecPro6<sup>0</sup>**

**Soundness** ( $prec_H$ ). *Proposition holds.*

**Reasoning.** If the log contains non-fitting behavior and all modeled behavior was observed, the set of traces in the event log complying with the process model equals the paths through the model.  $|\tau(l) \cap \tau(m)| = |\tau(m)|$  and precision is maximal.

**Simple behavioral appropriateness** ( $prec_I$ ). *Proposition does not hold.*

**Reasoning.** This approach only considers strictly sequential models to be perfectly precise. If the model contains choices, concurrency, loops, etc., multiple transitions may be enabled, lowering precision. Moreover, the approach assumes all behavior to be fitting. Hence, behavior that is observed and not modeled is likely to lead to problems.

**Advanced behavioral appropriateness** ( $prec_J$ ). *Proposition holds.*

**Reasoning.** Non-fitting behavior cannot affect the follow relations of the process model. Furthermore, it does not influence the *sometimes* follows/precedes relations of the event log if all the modeled behavior was observed. Hence, the sets of *sometimes* follows/precedes relations of the log and the model are equal to each other.  $S_F^l \cap S_F^m = S_F^m$  and  $S_P^l \cap S_P^m = S_P^m$  and, therefore, precision is maximal.

**ETC-one/ETC-rep** ( $prec_K$ ). *Proposition does not hold.*

**Reasoning.** The counter-example from **PrecPro5<sup>+</sup>** also shows that **PrecPro6<sup>0</sup>** is violated.

**ETC-all** ( $prec_L$ ). *Proposition does not hold.*

**Reasoning.** The counter-example from **PrecPro5<sup>+</sup>** also shows that **PrecPro6<sup>0</sup>** is violated.

**Behavioral specificity** ( $prec_M$ ). *Proposition holds.*

**Reasoning.** When the model allows for only observed behavior (i.e., in  $l$ ), then  $FP(l, m) = 0$ , as false positives are caused by the fact that behavior is shown in the model but not observed in the log. Therefore,  $\tau(m) \subseteq \tau(l) \Rightarrow prec_M(l, m) = \frac{TN(l, m)}{TN(l, m) + FP(l, m)} = \frac{TN(l, m)}{TN(l, m) + 0} = 1$ .

**Behavioral precision** ( $prec_N$ ). *Proposition holds.*

**Reasoning.** When the model allows for only observed behavior (i.e., in  $l$ ), then  $FP(l, m) = 0$ , as false positives are caused by the fact that behavior is shown in the model but not observed in the log. Therefore,  $\tau(m) \subseteq \tau(l) \Rightarrow prec_M(l, m) = \frac{TP(l, m)}{TP(l, m) + FP(l, m)} = \frac{TP(l, m)}{TP(l, m) + 0} = 1$ .

**Weighted negative event precision** ( $prec_O$ ). *Proposition holds.*

**Reasoning.** See the reasoning for  $prec_N$  above. The weighing of negative events doesn't change the fact that false positives cannot occur when  $\tau(m) \subseteq \tau(l)$ , therefore the same reasoning applies to  $prec_O$ .

**Projected precision** ( $prec_P$ ). *Proposition holds.*

**Reasoning.** If the all modeled behavior is observed, the automaton describing the model and the conjunctive automaton of the model and the log are exactly the same.  $DFac(S, M, A) \setminus DFA(M|_A) = \emptyset$ , if  $\tau(m) \subseteq \tau(l)$ . Hence, precision is maximal. Furthermore, the authors define that precision is 1 if the model is empty [23].

**Anti-alignment precision** ( $prec_Q$ ). *Proposition holds.*

**Reasoning.** By definition, an anti-alignment will always fit the model. Consequently, when computing the distance between the unfitting trace and the anti-alignment, it will never be minimal. However note, that scenarios with unfitting behavior were not considered by the approach since the authors assume a fitting log. After aligning the log, it contains exactly the modeled behavior  $\tau(l) = \tau(m)$  and precision is maximal.

**Eigenvalue precision** ( $prec_R$ ). *Proposition holds.*

**Reasoning.** Corollary 5.15 of [26] proves that  $prec_R(l, m) = 1$  when  $\tau(m) = \tau(l)$ . From the definition of the precision measure (i.e.,  $prec_R(l, m) = \frac{eig(\tau(m) \cap \tau(l))}{eig(\tau(m))}$ ) it follows that the proposition holds, as the numerator  $eig(\tau(m) \cap \tau(l))$  is equal to  $eig(\tau(m))$  when  $\tau(m) \subseteq \tau(l)$ .

#### A.4 Generalization

##### Proposition 1 DetPro<sup>+</sup>

**Alignment generalization** ( $gen_S$ ). *Proposition holds.*

**Reasoning.** Generalization is calculated based on the states visited by the process. The approach counts how often each state is visited ( $n$ ) and how many different activities were observed in this state ( $w$ ). These two numbers can be obtained from the model and the log at all times. Hence generalization is deterministic. Note, that we assume in case of non-fitting behavior that the log is first aligned before evaluating this proposition.

**Weighted negative event generalization** ( $gen_T$ ). *Proposition does not hold.*

**Reasoning.** If duplicate or silent transitions are encountered during the replay of the trace, which was enhanced with negative events, the approach explores which of the available transitions enables the next event in the trace. If no solution is found, one of the transitions is randomly fired. Hence precision also depends on the representation of the model.

**Anti-alignment generalization** ( $gen_U$ ). *Proposition holds.*

**Reasoning.** Precision is computed based on the maximal anti-alignment. Even if there are multiple maximal anti-alignments the distance will always be maximal and, therefore, precision is deterministic. Note, that we assume in case of non-fitting behavior that the log is first aligned before evaluating this proposition.

##### Proposition 2 BehPro<sup>+</sup>

**Alignment generalization** ( $gen_S$ ). *Proposition holds.*

**Reasoning.** The approach abstracts from the concrete representation of the process models. A key element is the function  $state_M$  which is a parameter of the approach and maps each event onto the state in which the event occurred. This function only uses behavioral properties. Hence, the proposition holds.

**Weighted negative event generalization** ( $gen_T$ ). *Proposition does not hold.*

**Reasoning.** If duplicate or silent transitions are encountered while replaying a trace, the approach checks if one of the available transitions enables the next event in the trace. Whether this is the case can depend on the structure of the model.

**Anti-alignment generalization** ( $gen_U$ ). *Proposition does not hold.*

**Reasoning.** This approach defines a so-called recovery distance which measures the distance between the states of the anti-alignment and the states visited by the log. It defines a state as a marking of the Petri net. One can think of two Petri nets with the same behavior but different markings based on their structure. The two process models presented in Figure 10 can be used as examples. Therefore generalization depends on the representation of the process model.

#### **Proposition 14 GenPro1<sup>+</sup>**

**Alignment generalization** ( $gen_S$ ). *Proposition does not hold.*

**Reasoning.** The approach does not allow for unfitting behavior and therefore aligns the log with the process model. These aligned traces might visit states that have already been observed by the fitting behavior, which increases the number of visits  $n$  to these states and improves generalization. The extension of the model such that  $\tau(m_1) \subseteq \tau(m_2)$  might cause this previously unfitting behavior to fit  $m_2$  and aligning the log is not necessary anymore. However, these “missing” aligned traces cause a decrease in the number of visits  $n$  to each state of the previously aligned trace and generalization decreases.

**Weighted negative event generalization** ( $gen_T$ ). *Proposition does not hold.*

**Reasoning.** BehPro<sup>+</sup> does not hold, which implies that GenPro1<sup>+</sup> does not hold

**Anti-alignment generalization** ( $gen_U$ ). *Proposition does not hold.*

**Reasoning.** BehPro<sup>+</sup> does not hold, which implies that GenPro1<sup>+</sup> does not hold.

#### **Proposition 15 GenPro2<sup>+</sup>**

**Alignment generalization** ( $gen_S$ ). *Proposition does not hold.*

**Reasoning.** According to the definition of generalization, it is possible that additional fitting behavior in the event log decreases generalization if the additional traces introduce new unique events to the log and  $pnew(w, n) = 1$ . Hence, the new traces raise the number of unique activities ( $w$ ) in state  $s$  while the number of times  $s$  was visited by the event log stays low ( $n$ ).

**Weighted negative event generalization** ( $gen_T$ ). *Proposition does not hold.*

**Reasoning.** The fitting behavior can lead to the generation of additional negative events. If these negative events are correctly identified, they increase the value of disallowed generalizations (DG).  $AG(l_1, m) = AG(l_2, m)$  and  $DG(l_1, m) < DG(l_2, m)$  which decreases generalization  $\frac{AG(l_1, m)}{AG(l_1, m) + DG(l_1, m)} > \frac{AG(l_2, m)}{AG(l_2, m) + DG(l_2, m)}$ .

**Anti-alignment generalization** ( $gen_U$ ). *Proposition does not hold.*

**Reasoning.** The approach defines the perfectly generalizing model as a model with a maximal anti-alignment distance  $d$  and minimal recovery distance  $d_{rec}$ . The newly observed behavior of the general model should introduce new paths between states but

no new states [13]. However, if the model is very imprecise and with a lot of different states, it is possible that the added traces visit very different states than the anti-alignment, generalization will be low for these traces. Consequently, the average generalization over all traces decreases.

**Proposition 16 GenPro3<sup>0</sup>**

*Alignment generalization ( $gen_S$ ). Proposition does not hold.*

**Reasoning.** According to this approach, generalization is not defined if there are unfitting traces, since unfitting behavior cannot be mapped to states of the process model. Note that unfitting behavior was intentionally excluded from the approach and the authors state that unfitting event logs should be preprocessed to fit to the model. However, after the log is aligned, the added traces might improve generalization by increasing  $n$  the times how often certain states are visited while executing events that have already been observed by the fitting traces in the log.

*Weighted negative event generalization (S). Proposition does not hold.*

**Reasoning.** In this approach, negative events are assigned a weight which indicates how certain the log is about these events being negative ones. Even though the added behavior is non-fitting it might still provide evidence for certain negative events and therefore increase their weight. If these events are not enabled during log replay the value for disallowed generalizations (DG) decreases  $DG(l_1, m) > DG(l_2, m)$  and generalization improves:  $\frac{AG(l,m)}{AG(l,m)+DG(l_1,m)} < \frac{AG(l,m)}{AG(l,m)+DG(l_2,m)}$ .

*Anti-alignment generalization (R). Proposition does not hold.*

**Reasoning.** According to this approach, generalization is not defined if there are unfitting traces since unfitting behavior cannot be mapped to states of the process model. Note that the authors exclude unfitting behavior from this approach and state that unfitting event logs should be preprocessed to fit to the model. But after aligning the event log, it might be the case that the added and aligned traces report a big distance to the anti-alignment without introducing new states, which increases generalization.

**Proposition 17 GenPro4<sup>+</sup>**

*Alignment generalization ( $gen_S$ ). Proposition holds.*

**Reasoning.** Multiplying a fitting log  $k$  times will result in more visits  $n$  to each state while the number of different activities observed  $w$  stays the same and generalization increases.

*Weighted negative event generalization ( $gen_T$ ). Proposition holds.*

**Reasoning.** Multiplying the log  $k$  times will proportionally increase the number of allowed and disallowed generalizations and therefore not change generalization:  $\frac{k \times AG(l,m)}{k \times (AG(l,m) + DG(l,m))} = \frac{AG(l,m)}{AG(l,m) + DG(l,m)}$ ,  $gen_T(l^k, m) = gen_T(l, m)$ .

**Anti-alignment generalization** ( $gen_U$ ). *Proposition holds.*

**Reasoning.** This approach sums for each trace in the log the trace-based generalization. This sum is averaged over the number of traces in the log and consequently, duplication of the log will not change precision.

### Proposition 18 GenPro5<sup>+</sup>

**Alignment generalization** ( $gen_S$ ). *Proposition does not hold.*

**Reasoning.** According to this approach, generalization is not defined if there are unfitting traces, since they cannot be mapped to states of the process model. Note that unfitting behavior was intentionally excluded from the approach and the authors state that unfitting event logs should be preprocessed to fit the model. However, after the log is aligned the added traces might improve generalization by increasing  $n$  the times how often certain states are visited while being aligned to traces that have already been observed by the other traces in the log. Hence, generalization increases.

**Weighted negative event generalization** ( $gen_T$ ). *Proposition holds.*

**Reasoning.** Multiplying the log  $k$  times will proportionally increase the number of allowed and disallowed generalizations and therefore not change generalization:

$$\frac{k \times AG(l,m)}{k \times (AG(l,m) + DG(l,m))} = \frac{AG(l,m)}{AG(l,m) + DG(l,m)}, \quad gen_T(l^k, m) = gen_T(l, m).$$

**Anti-alignment generalization** ( $gen_U$ ). *Proposition holds.*

**Reasoning.** According to this approach, generalization is not defined if there are unfitting traces since unfitting behavior cannot be mapped to states of the process model. Note, that the authors exclude unfitting behavior from this approach and state that unfitting event logs should be preprocessed to fit the model. Therefore we evaluate this proposition after the event log was aligned. Duplicating the aligned log will not change generalization since the sum of trace-generalization is averaged over the number of traces in the log.

### Proposition 19 GenPro6<sup>0</sup>

**Alignment generalization** ( $gen_S$ ). *Proposition holds.*

**Reasoning.** According to this approach, generalization is not defined if there are unfitting traces since they cannot be mapped to states of the process model. Note that unfitting behavior was intentionally excluded from the approach and the authors state that unfitting event logs should be preprocessed to fit the model. Duplicating the aligned log will result in more visits to each state visited by the log. Generalization increases.

**Weighted negative event generalization** ( $gen_T$ ). *Proposition holds.*

**Reasoning.** Multiplying the log  $k$  times will proportionally increase the number of allowed and disallowed generalizations and therefore not change generalization:

$$\frac{k \times AG(l,m)}{k \times (AG(l,m) + DG(l,m))} = \frac{AG(l,m)}{AG(l,m) + DG(l,m)}, \quad gen_T(l^k, m) = gen_T(l, m).$$

**Anti-alignment generalization** ( $gen_U$ ). *Proposition holds.*

**Reasoning.** According to this approach, generalization is not defined if there are unfitting traces since unfitting behavior cannot be mapped to states of the process model. Note that the authors exclude unfitting behavior from this approach and state that unfitting event logs should be preprocessed to fit the model. Duplicating the aligned log will not change generalization since the sum of trace-generalization is averaged over the number of traces in the log.

**Proposition 20 GenPro7<sup>0</sup>**

**Alignment generalization** ( $gen_S$ ). *Proposition does not hold.*

**Reasoning.** According to this approach, generalization is not defined if there are unfitting traces since they cannot be mapped to states of the process model. Note that unfitting behavior was intentionally excluded from the approach and the authors state that unfitting event logs should be preprocessed to fit the model. Duplicating an aligned log will result in more visits to each state. Hence, generalization increases and violates the proposition.

**Weighted negative event generalization** ( $gen_T$ ). *Proposition holds.*

**Reasoning.** Multiplying the log  $k$  times will proportionally increase the number of allowed and disallowed generalizations and therefore not change generalization:  

$$\frac{k \times AG(l, m)}{k \times (AG(l, m) + DG(l, m))} = \frac{AG(l, m)}{AG(l, m) + DG(l, m)}, gen_T(l^k, m) = gen_T(l, m).$$

**Anti-alignment generalization** ( $gen_U$ ). *Proposition holds.*

**Reasoning.** According to this approach, generalization is not defined if there are unfitting traces since unfitting behavior cannot be mapped to states of the process model. Note that the authors exclude unfitting behavior from this approach and state that unfitting event logs should be preprocessed to fit the model. Duplicating the aligned log will not change generalization since the sum of trace-generalization is averaged over the number of traces in the log.

**Proposition 21 GenPro8<sup>0</sup>**

**Alignment generalization** ( $gen_S$ ). *Proposition does not hold.*

**Reasoning.** According to the definition, generalization can never become 1. It only approaches 1. Consider a model allowing for just  $\tau(m) = \langle a \rangle$  and the log  $l = [\langle ara^k \rangle]$ . The log visits the state  $k$ -times and observes one activity  $w = 1$  in this state. The function  $pnew = \frac{1(1+1)}{k(k-1)}$  will approach 0 as  $k$  increases but never actually be 0. Hence,  $gen_S(l, m) = 1 - pnew(1, k)$  approaches 1, but will never be precisely 1.

**Weighted negative event generalization** ( $gen_T$ ). *Proposition holds.*

**Reasoning.** If the model allows for any behavior, it does not contain any negative behavior which is not allowed. Hence the algorithm cannot find negative events, which are not enabled during replay (DG) and generalization will be maximal.  $gen_R = AG(l, m)/(AG(l, m) + DG(l, m)) = AG(l, m)/(AG(l, m) + 0) = 1.$



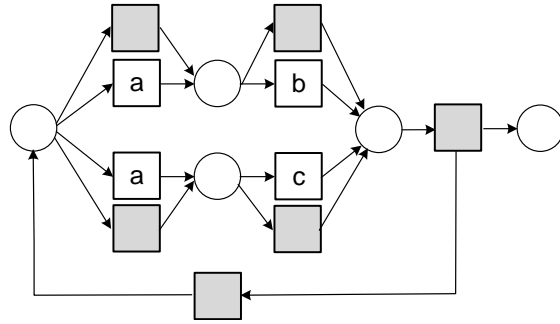


Fig. 13: A process model that allows for any behavior while displaying different states.

**Anti-alignment generalization** ( $gen_U$ ). *Proposition does not hold.*

**Reasoning.** Assume a model that allows for any behavior because of silent transition, loops and duplicate transitions. The distance between the log and the anti-alignment is maximal. However, due to the duplicate transitions which are connected to separate places the recovery distance is not minimal. Consequently, generalization would not be maximal which violates the proposition. Figure 13 is an example of such a process model.