The Imprecisions of Precision Measures in Process Mining

Niek Tax∗, Xixi Lu, Natalia Sidorova, Dirk Fahland, Wil M.P. van der Aalst

Eindhoven University of Technology, P.O. Box 513, Eindhoven, The Netherlands

Abstract

In process mining, precision measures are used to quantify how much a process model overapproximates the behavior seen in an event log. Although several measures have been proposed throughout the years, no research has been done to validate whether these measures achieve the intended aim of quantifying over-approximation in a consistent way for all models and logs. This paper fills this gap by postulating a number of axioms for quantifying precision consistently for any log and any model. Further, we show through counter-examples that none of the existing measures consistently quantifies precision.

Keywords: Process mining, Formal languages and automata, Petri nets, Design of algorithms

1. Introduction

Process mining [1] is a fast growing discipline that is focused on the analysis of events logged during the execution of a business process. Events contain information on what was done, by whom, for whom, where, when, etc. Such event data are often readily available from information systems such as ERP, CRM, or BPM systems. Process discovery, which plays a prominent role in process mining, is the task of automatically generating a process model that accurately describes a business process based on such event data. Many process discovery techniques have been developed over the last decade (e.g., [2, 3, 4, 5]), producing process models in various forms, such as Petri nets [6], process trees [7], YAWL models [8], and BPMN models [9].

The process model that is pursued by process discovery techniques ideally allows for all the behavior that was observed in the event log (called fitness), while at the same time it should not be too general by allowing for much more behavior than what was seen in the event log (called precision).

A range of measures have been proposed for quantifying precision [10, 11, 12, 13, 14]. However, to the best of our knowledge, there is currently no work on verifying whether precision measures actually quantify what they are supposed to measure in a consistent manner. Conceptually, the precision of a process model in the context of an event log should be high when the model allows for few traces not seen in the log, and it should be low when it allows for many traces not seen in the log. In this paper we propose a set of axioms that formulate desired properties of precision measures and systematically validate whether these axioms hold for existing precision measures.

In Section 2 we introduce basic notation and definitions. In Section 3 we formulate axioms for precision measures. We then continue with Section 4 where we describe existing precision measures in more detail and validate the axioms for these measures. In Section 5 we describe two contexts in which we are not able to define axioms for precision. In Section 6 we conclude this paper and state several directions for future work.

2. Preliminaries

In this section we introduce concepts used in later sections of this paper.

\[ X = \{a_1, a_2, \ldots, a_n\} \] denotes a finite set. \( \mathcal{P}(X) \) denotes the power set of \( X \), i.e., the set of all possible subsets of \( X \). \( X^* \) denotes the set of all sequences over a set \( X \) and \( \sigma = (a_1, a_2, \ldots, a_n) \) denotes a sequence of length \( n \), with \( \langle \rangle \) the empty sequence. \( X \setminus Y \) denotes the set of elements that are in set \( X \) but not in set \( Y \), e.g., \( \{a, b, c\} \setminus \{a, c\} = \{b\} \).

A multiset (or bag) over \( X \) is a function \( B : X \rightarrow \mathbb{N} \) which we write as \( [a_1^{n_1}, a_2^{n_2}, \ldots, a_n^{n_n}] \), where for \( 1 \leq i \leq n \) we have

*Corresponding author

Email addresses: n.tax@tue.nl (Niek Tax), x.lu@tue.nl (Xixi Lu), n.sidorova@tue.nl (Natalia Sidorova), d.fahland@tue.nl (Dirk Fahland), w.m.p.v.d.aalst@tue.nl (Wil M.P. van der Aalst)
A partial function \( f \in \mathcal{X} \rightarrow \mathcal{Y} \) with domain \( \text{dom}(f) \) can be lifted to sequences over \( X \) using the following recursive definition: (1) \( f(\varnothing) = \varnothing \); (2) for any \( \sigma \in \mathcal{X}^* \) and \( x \in X \):

\[
 f(\sigma \cdot \langle x \rangle) = \begin{cases} 
  f(\sigma) & \text{if } x \notin \text{dom}(f), \\
  f(\sigma) \cdot \langle f(x) \rangle & \text{if } x \in \text{dom}(f).
\end{cases}
\]

Defining an initial and final markings allows to define the language accepted by a Petri net as a set of finite sequences of activities.

Definition 3 (Accepting Petri Net). An accepting Petri net is a triplet \( \mathcal{APN} = (\mathcal{N}, \mathcal{M}_0, \mathcal{MF}) \), where \( \mathcal{N} \) is a labeled Petri net, \( \mathcal{M}_0 \in \mathcal{B}(\mathcal{P}) \) is its initial marking, and \( \mathcal{MF} \subseteq \mathcal{B}(\mathcal{P}) \) is its set of possible final markings. A sequence \( \sigma \in \mathcal{X}^* \) is a trace of an accepting Petri net \( \mathcal{APN} \) if there exists a firing sequence \( \mathcal{M}_0 \rightarrow \mathcal{M}_f \) such that \( \mathcal{M}_f \in \mathcal{MF} \), \( \gamma \in T^+ \) and \( \ell(\gamma) = \sigma \).

The language \( \mathcal{L}(\mathcal{APN}) \) is the set of all its traces, i.e., \( \mathcal{L}(\mathcal{APN}) = \{ \ell(\gamma) | \exists \mathcal{M}_0, \mathcal{MF} \mathcal{M}_0 \rightarrow^* \mathcal{M}_f \} \), which can be of infinite size when \( \mathcal{APN} \) contains loops. Even though we define language for accepting Petri nets, in theory \( \mathcal{L}(\mathcal{M}) \) can be defined for any process model \( \mathcal{M} \) with formal semantics. We denote the universe of process models as \( \mathcal{M} \). For each \( \mathcal{M} \in \mathcal{M} \), \( \mathcal{L}(\mathcal{M}) \) is defined.

For an event log \( L \), \( \hat{L} = \{ \sigma \in \mathcal{X} | L(\sigma) > 0 \} \) is the trace set of \( L \). For example, for log \( L = \{(a, b, c)^3, (b, a, c)^3\} \), \( \hat{L} = \{(a, b, c) \rightarrow (b, a, c)\} \). For an event log \( L \) and a model \( \mathcal{M} \) we say that \( L \) is fitting on model \( \mathcal{M} \) if \( \hat{L} \subseteq \mathcal{L}(\mathcal{M}) \). Precision is related to the behavior that is allowed by a model \( \mathcal{M} \) that was not observed in the event log \( L \), i.e., \( \mathcal{L}(\mathcal{M}) \backslash \hat{L} \).

3. Axioms for Precision Metrics

The properties that are desired for precision measures are not clearly defined in existing work, although they are often discussed informally. Van der Aalst et al. [15] describe the precision dimension as “Precision: measure determining whether the model prohibits behavior very different from the behavior seen in the event log. A model with low precision is underfitting.”. Vanden Broucke et al. [13] describe precision as “precision (or: appropriateness), i.e., the model’s ability to disallow unwanted behavior;”. Múñoz-Gama and Carmona [12] describe it as “Precision: refers to overly general models, preferring models with minimal behavior to represent as closely as possible to the log.”. Buijs et al. [16] describe precision as “... precision quantifies the fraction of the behavior allowed by the model which is not seen in the event log.”.
Existing precision measures normalize $\mathbb{R}$ to a $[0, 1]$-interval.

The first axiom states that precision is deterministic, i.e., given a log and model always the same result is returned.

**Axiom A1.** A precision measure is a function $\text{prec} : \mathcal{B}(\Sigma^*) \times M \rightarrow \mathbb{R}$, i.e., it is deterministic.

For models $M_1$ and $M_2$ and a log $L$, $L \subseteq \mathcal{V}(M_1) \subseteq \mathcal{V}(M_2)$ implies $\text{prec}(L, M_1) \geq \text{prec}(L, M_2)

Note that A2 does allow $L \subseteq \mathcal{V}(M_1) \subseteq \mathcal{V}(M_2)$ with $\text{prec}(L, M_1) = \text{prec}(L, M_2)$. Ideally, since $\mathcal{V}(M_1)$ is smaller than $\mathcal{V}(M_2)$ we would like to see a higher precision for $M_1$, but this requirement might be too strict. However, for a process model $M$ with $L \subseteq \mathcal{V}(M)$, we would like the precision of $M$ on $L$ to be higher than the precision of $M$ on any other model (i.e., a model that allows for all behavior over its activities) on log $L$.

**Axiom A3.** For models $M_1$ and $M_2$ and a log $L$, $\mathcal{V}(M_1) \subseteq \mathcal{V}(M_2) \land \mathcal{V}(M_2) = \mathcal{P}(\Sigma^*)$ implies $\text{prec}(L, M_1) \geq \text{prec}(L, M_2).

The precision of a log on two language equivalent models should be equal, i.e., precision should not depend on the model structure.

**Axiom A4.** For models $M_1$ and $M_2$ and a log $L$, $\mathcal{V}(M_1) = \mathcal{V}(M_2)$ implies $\text{prec}(L, M_1) = \text{prec}(L, M_2)$.

A4 was stated before in an informal manner by Rozinat and van der Aalst [11], who stated that precision should be independent of structural properties of the model.

Adding fitting traces to a fitting log can only increase the precision of a given model with respect to the log.

**Axiom A5.** For model $M$ and logs $L_1$ and $L_2$, $L_1 \subseteq L_2 \subseteq \mathcal{V}(M) \implies \text{prec}(L_2, M) \geq \text{prec}(L_1, M)$.

From A5 it follows as a corollary that precision is maximal when the log contains all the traces allowed by the model, and minimal when it contains no traces allowed by the model.

In the coming sections we will validate whether these axioms hold for several precision measures. Some articles that introduce precision measures explicitly mention that the measure is intended to be used only with a certain subclass of Petri nets. An example of such a subclass of Petri nets are bounded Petri nets, which have the restriction that all places must have a finite number of tokens in all reachable markings. When an article that introduces a precision measure states an explicit assumption on the subclass of Petri nets, then we only validate the axioms on this subclass of Petri nets. When no explicit assumption on a subclass of Petri nets is stated, we assume that the precision measure is intended for Petri nets in general.

### 4. Precision Metrics

In this section we give an overview of the precision measures that have been developed in the process mining field, and validate the axioms for precision measures introduced in Section 3 for each of those measures.

#### 4.1. Soundness

Greco et al. [10] were the first to propose a precision measure, defining it as the number of unique executions of the process that were seen in the event log divided by the number of unique paths through the process model. This measure is not usable in practice, because it is zero when the process model allows for an infinite number of paths through the model. Any process model having a loop has a precision of 0. More recent precision
measures are capable of calculating the precision of a model for an event log even when the models allows for infinite behavior.

4.2. Behavioral Appropriateness

Rozinat and Van der Aalst [11] proposed the simple behavioral appropriateness precision measure, which looks at the average number of enabled transitions during replay. The authors observed themselves that simple behavioral appropriateness is dependent on the structure of the model, and not solely dependent on the behavior that it allows, therefore A4 does not hold for this measure. Furthermore, for a process model that contains silent transitions or duplicate labels it is possible that a given trace can be replayed on this model in multiple ways, where the average number of enabled transitions can depend on the chosen replay path through the model. This replay path through the model is chosen arbitrarily from the possible ways in which the trace can be replayed. This shows that A1 does not hold for simple behavioral appropriateness, as it is not deterministic.

In the same paper, Rozinat and van der Aalst [11] propose advanced behavioral appropriateness, which is independent of the model structure. Advanced behavioral appropriateness calculates the sets $S_F \subseteq \Sigma \times \Sigma$ of pairs of activities that sometimes, but not always, follow each other. Likewise set $S_P \subseteq \Sigma \times \Sigma$ is calculated as the set of activities that sometimes, but not always, precede each other. $S_F$ and $S_P$ denote the sometimes-follows and sometimes-precedes relations on the log, and $S_F^M$ and $S_P^M$ denotes the sometimes-follows and sometimes-precedes relations according to the model. However, to calculate $S_F^M$ and $S_P^M$, exhaustive exploration of the state space of the model is required, prohibiting the application of this measure for large models or highly concurrent models, where the state-space explosion problem arises. Advanced behavioral appropriateness precision is defined as $\alpha' = \frac{\sum_{p \in F} |S_F^M|}{\sum_{p \in P} |S_P^M|}$. Because $S_F^M$ and $S_P^M$ are obtained through exhaustive exploration of the state space of the model, it is easy to see that they depend only on the behavior of the model and not on its structure, therefore A4 holds. A problem with advanced behavioral appropriateness occurs for deterministic models, where $|S_F^M|=|S_P^M|=0$, leading to undefined precision. This shows that advanced behavioral appropriateness is a partial function, which is in conflict with A1.

Rozinat and van der Aalst [11] state that simple behavioral appropriateness and advanced behavioral appropriateness assume the Petri net to be in the class of sound workflow (WF) nets [17]. A WF-net requires the Petri net to have (i) a single Start place, (ii) a single End place, and (iii) every node must be on some path from Start to End. The soundness property additionally requires that each transition can be potentially executed, and that the process can always terminate properly, i.e., finish with only one token in the End place.

Consider model $M$ of Figure 2, which belongs to the class of sound WF-nets, and any log $L$ such that $L \subseteq \Sigma(M)$. The loop in model $M$ causes $S_F^M$ and $S_P^M$ to contain all pairs of activities of $\Sigma$. Therefore, $S_F^M$ and $S_P^M$ are identical to the sometimes relations $|S_F^M|$ and $|S_P^M|$ of any model $M'$ with $\Sigma(M')=P(\Sigma')$, leading to $\text{prec}(L,M)=\text{prec}(L,M')$. As $\Sigma(M) \subseteq \Sigma(\Sigma')$, this is in conflict with A3.

4.3. Escaping Edges Precision

Escaping Edges Precision (ETC) [12] calculates precision by constructing a prefix automaton, which consists of one state per unique prefix of the event log. Figure 3b shows an example prefix automaton for an event log $L = [(a,c),(a,d)]$. For each state in the prefix automaton it is then determined which activities are allowed as next activities by the process model. Activities that are allowed as next activities for some prefix but that are never observed in the event log after this prefix are referred to as escaping edges.

In later work [18] [19], alignments [20] are used to calculate the prefix automaton on the aligned event log instead of the original event log, making the precision measure robust to non-fitting traces, i.e., traces that are not in the language of the model. For a trace $\sigma$ from a log $L$ that is fitting on an accepting Petri net APN, alignments [20] give a sequence of transition firings $\gamma \in T^*$ such that $m_0 \overset{\gamma}{\rightarrow} m_f$ with $m_0$ the initial marking and $m_f$ a final marking of APN and $\ell(\gamma)=\sigma$. Note that for a given trace $\sigma$ and model, multiple possible alignments can exist. For non-fitting traces, alignments search for a firing sequence $\gamma \in T^*$ such that $\ell(\gamma)$ is as close as possible to $\sigma$. Adriansyah et al. [18] describe two versions of the alignment-based escaping edges precision: one-align ETC, which calculates the precision based on one optimal alignment of log and model, and all-align ETC, which calculates the precision based on all optimal alignments between log and model. In practice, it is often computationally infeasible to calculate all optimal alignments. A later
precision measure, **representative-align ETC** [21], calculates the escaping edges based on a sample of optimal alignments, and can therefore be seen as a trade-off between the computational efficiency of one-align ETC and the reliability of all-align ETC. The papers on ETC precision and its variants do not state an assumption on a subclass of Petri nets. ETC, one-align ETC, all-align, and representative-align ETC precision are all implemented in the package ETConformance [14] as part of the process mining framework ProM [22].

The one optimal alignment that is used by one-align ETC is chosen arbitrarily from the set of optimal alignments of a log on a model. However, different optimal alignments result in different prefix automata, which can potentially lead to different precision values. This shows that A1 does not hold for one-align ETC.

Consider log L₁=[(a, c), (a, d)], log L₂=[(a, c), (a, d), (b, a, b, a, b, a, a, b, a, c)] and model M be the Petri net of Figure 3a. Note that L₁⊂L₂. The alignment automata generated for the calculation of prec(L₁, M) and the calculation of prec(L₂, M) are shown in Figure 3b and Figure 3c. The circles represent the states of the automaton, and the arrows the transitions. The numbers in the states represent the weights of the states for the precision calculation, i.e., the number of times that states are visited in the alignment of log L on model M [18]. In an alternative definition of one-align ETC [19] the states are weighted by the number of times that events occurred while being in this state according to the alignment of L on M, instead of the number of times that this state was reached according to this alignment. Figure 3b shows that the initial state was visited twice, activity a occurred twice at the start in log L₁, resulting in a state from which activities b, d, and c were enabled.

From this state, activities c and d were seen once, and activity b was never seen, thus it is an escaping edge. Escaping edges precision is then the (weighted) average ratio of non-escaping edges from all outgoing edges, where states are weighted by the number of times that they are visited. Counting the weighted number of non-escaping edges in the numerator and the weighted total number of edges in the denominator in our example, we find prec(L₁, M)=2×1+2×2+1×0+1×0+6×1=6/3=0.75. One-align ETC results in the following precision values for M on L₁ and L₂: prec(L₁, M)=0.75 and prec(L₂, M)=0.7143. This shows that A5 does not hold for one-align ETC. By comparing the automata of Figures 3b and 3c it becomes clear that the single trace that is in L₂ but not in L₁ brings the model to many states with three escaping edges, reducing precision. The prefix automata and the precision calculations for M on logs L₁ and L₂ were performed manually following the procedure from the paper and validation using the ETConformance plugin in ProM.

Now consider log L = [(a, b, c)], and the three Petri nets M₁, M₂, M₃ in Figures 4a, 4b, and 4c respectively. Note that M₁ and M₂ are language equivalent, as $\nu(M_1)=\nu(M_2)=[a,b,c]$. M₃ is more behaviorally constrained than M₁ and M₂, since all its traces start with activity a. The one-align precision of M₁, M₂, M₃ on L are: $\text{prec}(L, M₁)=0.3333$, $\text{prec}(L, M₂)=0.5238$, and $\text{prec}(L, M₃)=0.4444$. $\nu(M_1)\subset\nu(M_2)$, but $\text{prec}(L, M_3)>\text{prec}(L, M_1)$, implying that A2 does not hold for one-align ETC. Furthermore, $\nu(M_1)\subset\nu(M_2)$, but $\text{prec}(L, M_1)\neq \text{prec}(L, M_2)$, implying that A4 does not hold for one-align ETC.

Analyzing the ETConformance plugin in ProM we

---

1https://svn.win.tue.nl/trac/prom/browser/Packages/ETConformance
found that the prefix automaton generated for one-align precision for calculation of $\text{prec}(L, M_1)$ results in 6 states, belonging to 3 firings of observable transitions and 2 firings of $\tau$-transitions. In 3 of the 6 states, which correspond to $M_1$ being in the initial marking, there are 4 possible next activities according to the model, of which only one is observed for that prefix. Furthermore, it shows that the alignment automaton generated for $L$ and $M_1$ consists of 6 states, the automaton for $L$ and $M_2$ consists of 12 states, and the automaton for $L$ and $M_3$ consists of 5 states. This shows that the silent ($\tau$) transitions in $M_2$ generate additional states in the alignment automaton, leading to a higher precision value.

Computing the precision of $M_1$ or of $M_2$ on $L$ did not finish with all-align ETC and representative-align ETC after 8 hours of computation time. The long computation time of all-align ETC and representative-align ETC on models where many optimal alignments exist is a known issue which hinders the application of those measures in practice.

4.4. Negative Event Precision

Goedertier et al. [2] proposed a method to induce negative events, i.e., sets of events that were prevented from taking place. Negative events are induced for each position in the event log, i.e., for each event $e$ in each trace of the log a set of events is induced that could not have taken place instead of event $e$. De Weerdt et al. [21] proposed a precision measure based on negative events, behavioral precision ($p_b$), which is closely linked to how precision is defined in the area of data mining. Negative event precision regards a process model as a binary classifier that determines whether a certain event can take place given a certain prefix, and then evaluates the precision of this classifier in data mining terms taking the induced negative events as ground truth. For a given trace prefix, true positive (TP) events are defined as events that are possible according to both the process model (i.e., a transition labeled with this event is enabled) and log (i.e., this event is not a negative event). False positive events (FP) are negative events induced for a given prefix that were possible according to the model. Behavioral precision is defined as $p_b = \frac{TP}{TP + FP}$, which is in accordance to the definition of precision in the data mining field. In later work [24] induction of artificial negative events has been refined based on frequent temporal patterns which are mined from the event log. Finally, weighted artificial events, where negative events are weighted according to their confidence, are proposed in [13].

Weighted behavioral precision induces negative events for an event $e$ in the log by taking a window of events $w$ that directly precede $e$, then calculating all subsequences of events in the log that exactly match $w$, and finally negative events are identified by calculating which events have never occurred in the log directly after any subsequence matching $w$. This procedure is repeated for different windows sizes, and the resulting negative event are weighted by window size.

To induce the events that could not have happened after e.g. trace prefix $\sigma' = \{a, c, d, e, c, d, e, e\}$, the method to induce weighted negative events described in [13] searches for subsequences of events in the log that are identical to the latest $k$ events of $\sigma'$ in the event log. All the activities that have never succeeded such subsequences are considered to be negative events, furthermore, the confidence of these negative events is based on the length $k$ of those matching subsequences.

Negative event based precision measures, with the different methods for negative event induction, are implemented in the ProM package NEConformance. In this paper we evaluate the precision measure that uses weighted negative events [13], which is the most recent approach to induce negative events and the recommended approach for measuring precision [13]. No assumption on a subclass of Petri nets is stated in the papers on negative event precision.

Consider models $M_1$ and $M_2$ of Figure 5 respectively excluding and including the arcs and places indicated in dotted lines. $\Sigma(M_2) \subset \Sigma(M_1)$, since $M_2$ contains a long term dependency between activities $a$ and $f$ and between activities $b$ and $g$, which $M_1$ does not have. Consider an event log $L$ which consists of 10 traces from $M_2$, leading to $L$ being fitting on both $M_1$ and $M_2$. We found the negative event precision of $M_1$ and $M_2$ on the same $L$ to be non-deterministic, resulting in slightly different values every time that it is calculated. This shows that $A1$ does not hold for negative event precision.

Because negative event precision is non-deterministic, we calculated the precision of $M_1$

\footnote{http://processmining.be/neconformance/}

Figure 5: A process model $M_1$ and its behaviorally restricted variant $M_2$ which includes the dotted places and arcs.
and $M_2$ on $L$ both 20 times. The highest precision found in 20 repetitions for $M_1$ is 0.4876, while the lowest precision found for $M_2$ is 0.4545, showing that the non-determinism has the effect that $A_2$ does not hold for negative event precision. We found an average value of 0.4744 with a standard deviation of 0.0090 for the precision of $M_1$ on $L$ and an average value of 0.4640 with a standard deviation of 0.0072 for the precision of $M_2$ on $L$. This shows that also in terms of average precision value $A_2$ does not hold.

To test whether the difference in mean precision between $M_1$ and $M_2$ is due to chance alone we formulate a null hypothesis:

$H_0$ : The average negative event precision of $M_2$ on $L$ is higher than or equal to the average negative event precision of $M_1$ on $L$.

Testing this null hypothesis with a one-tailed Welch $t$-test we found a $p$-value of 0.0001801, indicating that we can reject the null hypothesis with significance level 0.01. This shows that, with statistical significance, the precision of $M_1$ on $L$ is higher than the precision of $M_2$ on $L$, which is in disagreement with $A_2$.

To see why $A_2$ does not hold for negative event precision, consider the negative event inducing procure being applied to trace prefix $\sigma' = \langle a, c, c, d, e, c, d, e, e \rangle$ from log $L$. Petri net $M_2$ generates many different traces because of the parallel length-one-loops on activities $c$, $d$ and $e$, which allows for any sequence of any length over these activities. Therefore, the matching subsequences of $\sigma'$ in the log generated from $M_2$ are the subsequences that by chance ended in the same behavior over $c$, $d$, and $e$. Because the sequences of $c$, $d$ and $e$ events can be long and diverse, activity $a$ and $b$ are unlikely to be present in the matching subsequences, which makes it unlikely that the procedure can induce the negative event $g$ for $\sigma'$. Because the negative events that reflect the constraint that $M_2$ introduces compared to $M_1$ cannot be induced from the log, negative event precision is not able to recognize that $M_2$ is more precise than $M_1$.

4.5. Projected Conformance Checking

Projected Conformance Checking (PCC) precision was developed by Leemans et al. \[14\] as a computationally efficient precision measure that scales to event logs with billions of events. PCC precision projects both event log and model on all subsets of activities of size $k$, and generates minimal deterministic finite automata (DFA) for the behavior over these subsets of activities in the log (i.e., log automaton) and for the behavior over these events allowed by the model (i.e., model automaton). Based on the log automaton and model automaton it then builds a conjunction automaton which allows the behavior that was allowed both in log and model automaton. It then iterates over the states of the model automaton, and calculates over the share of transitions of this state that is also possible in the corresponding state in the conjunction automaton. It defines precision as the average of this share over the model automaton states.

PCC precision is implemented in the ProM package ProjectedRecallAndPrecision\[1\]. PCC precision assumes the Petri net to be of the class of bounded Petri nets, i.e., the Petri net for which precision is calculated must have a finite number of tokens in every place for all reachable markings.

Consider log $L = \{ (a, b) \}$ and Petri nets $M_1$ and $M_2$ of Figures 6\[a\] and 6\[b\] respectively. $M_1$ starts with a length-one-loop on activity $a$, followed by activity $b$. $M_2$ unrolls the length-one-loop on activity $a$ of $M_1$ to at most two executions, thereby limiting the behavior as it only allows at most two executions of activity $a$. It is easy to see that $M_1$ and $M_2$ both belong to the class of bounded Petri nets, as in both models each place can have at most one token. For this log and these models, PCC precision results in $\text{prec}(L, M_1)=0.6$, and $\text{prec}(L, M_2)=0.5$. However, since $\forall(M_2) \subset \forall(M_1)$, $A_2$ states that the precision of $M_2$ for fitting log $L$ should be higher or equal to its precision of $M_1$. This shows that $A_2$ does not hold for PCC precision.

This drop in precision is an effect of the additional states that are created in the model DFA as an effect of unrolling the length-one-loop. The model DFA created from Petri net $M_2$ (Figure 6\[b\]) for example contains a state $s$ that is reached after firing $(a, a)$. This state however is never reached based on event log $L$, which only contains a trace $(a, b)$, which has the effect that none of the enabled transitions from state $s$ were observed in the log, bringing down the precision. In the DFA generated from Petri net $M_1$ (Figure 6\[a\]), this state $s$ is merged with the state that one reaches after observing a single $a$.

\[3\]https://svn.win.tue.nl/trac/prom/browser/ Packages/ProjectedRecallAndPrecision/
event, as future behavior allowed by the model does not depend on the number of a-events seen.

Consider Petri net $M$ of Figure 7 and event logs $L_1 = \{(b, a, c), (a, a, c)\}$, and $L_2 = \{(b, a, c), (a, a, c), (a, b, b, b, b, b, b, b, b, b, b)\}$. The single place of $M$ is bounded to one token, therefore $M$ belongs to the class of bounded Petri nets. It is easy to see that $L_1 \subseteq L_2$, since the first two traces of log $L_2$ form log $L_1$. PCC precision results in $\text{prec}(L_1, M) = 0.3125$ and $\text{prec}(L_2, M) = 0.2727$, violating A5. The two traces of $L_2$ that are not in $L_1$ are very long traces to the traces that are in $L_1$, leading to additional states in the log automaton and the conjunction automaton. The additional states of the conjunction automaton have a low precision of $\frac{1}{2}$, since for each state the model allows for four options (firing activity $a$, $b$, or $c$, or stopping), while only one is seen in the log. Therefore, if we would expand trace $(b, a, a, a, a, a, a, a, a, a, a, a, a, a)$ with more events of activity $a$, then $\text{prec}(L_2, M)$ would approach $\frac{1}{2}$.

4.6. Overview of Precision Metric Properties

We formulated five axioms that describe desirable properties for precision measures. Table 1 gives an overview of that axioms that we showed that do hold (✓) and that do not hold (✗) for each precision measure. We found that none of the existing precision measures fulfills all five axioms. Empty cells in the table are currently unknown, and no formal proof nor a counter example has been found that proves or disproves the axiom for the respective precision measure.

![Figure 7: A flower model over activities a, b and c.](image_url)

Table 1: Overview of the precision axioms and whether they hold for each precision measure.

| Metric                        | A1 | A2 | A3 | A4 | A5 |
|-------------------------------|----|----|----|----|----|
| Simple behavioral appropriateness | ✗  | ✗  | ✗  | ✓  |    |
| Advanced behavioral appropriateness | ✗  | ✗  | ✓  | ✓  |    |
| One-align ETC | ✗  | ✗  | ✓  | ✓  |    |
| Negative Event Precision      | ✗  | ✗  | ✓  | ✓  |    |
| PCC precision                 | ✗  | ✓  | ✓  | ✓  |    |

5. Contexts With Unclear Requirements for Precision Metrics

The axioms introduced in Section 4 can be regarded as necessary conditions for precision measures, but they leave precision unspecified in some contexts. Figure 8 shows a situation in which $L \subseteq \psi(M_1)$, $L \subseteq \psi(M_2)$, but $\psi(M_1) \cup \psi(M_2) \neq \emptyset$ and $\psi(M_1) \setminus \psi(M_2) = \emptyset$. In this setting, both $M_1$ and $M_2$ allow for (a possibly infinite amount of) different behavior that was not seen in $L$. Precision measures deal with this situation by quantifying the amount of behavior of $M_1$ and $M_2$. However, there are no obvious formal properties telling how the precision of $M_1$ and $M_2$ on $L$ should relate.

Furthermore, all axioms define desired properties of precision measures when the event log $L$ fits the behavior of the model $M$, i.e., $L \subseteq \psi(M)$. In practice, process discovery techniques will return process models with fitness below 1, i.e., there exists $\sigma \in L : \sigma \notin \psi(M)$. The discovery algorithm may deliberately abstract from infrequent behavior. In this paper we do not formulate axioms for precision measures in the context of event logs that do not fit the process model, since we feel that there is not enough agreement in the process mining community on how a precision measure should behave in this context. Figure 8b shows an Euler diagram of a log $L$ and two models $M_1$ and $M_2$ such that $\psi(M_1) \subseteq \psi(M_2)$ and $L \subseteq \psi(M_1)$, which is a non-fitting equivalent of A2. A2 prescribes $\text{prec}(L, M_1) \geq \text{prec}(L, M_2)$, however, when the log does not fit the models, the behavior that fits $M_2$ but not $M_1$, $(L \setminus \psi(M_1)) \cap \psi(M_2)$, makes it unclear how the precision of $M_1$ and $M_2$ should relate. Furthermore, even when $(L \setminus \psi(M_1)) \cap \psi(M_2) = \emptyset$, it can be the case that the behavior in $L$ that does not fit the models is behaviorally similar to behavior of $M_2$.

6. Conclusions & Future Work

In this paper provides a set of minimal requirements for precision measures through axioms. We validated these axioms for existing measures. Surprisingly, we
discovered that none of the existing precision measures fulfills all formulated requirements.

In future work, we would like to fill the empty cells of Table II and get a complete overview of the axioms that hold for each precision measure. Furthermore, we would like to use the insights learned from evaluating the axioms on the measures to either repair one of the existing measures or come up with a completely new measure that fulfills all five axioms.

Reproducibility. The event logs and process models that are used as part of a counterexample for a combination of an axiom and a precision measure can be found at [26].

References

[1] W. M. P. van der Aalst, Process mining: Data science in action, Springer, 2016.
[2] S. Goedertier, D. Martens, J. Vanthienen, B. Baesens, Robust process discovery with artificial negative events, Journal of Machine Learning Research 10 (2009) 1305–1340.
[3] S. J. J. Leemans, D. Fahland, W. M. P. van der Aalst, Discovering block-structured process models from event logs: a constructive approach, in: International Conference on Applications and Theory of Petri Nets and Concurrency, Springer Berlin Heidelberg, 2013, pp. 311–329.
[4] R. Conforti, M. Dumas, L. García-Bañuelos, M. La Rosa, BPMN miner: automated discovery of BPMN process models with hierarchical structure, Information Systems 56 (2016) 284–303.
[5] S. J. van Zelst, B. F. van Dongen, W. M. P. van der Aalst, Avoiding over-fitting in ILP-based process discovery, in: International Conference on Business Process Management, Springer International Publishing, 2015, pp. 163–171.
[6] T. Murata, Petri nets: Properties, analysis and applications, Proceedings of the IEEE 77 (4) (1989) 541–580.
[7] J. C. A. M. Bijls, B. F. van Dongen, W. M. P. van der Aalst, A genetic algorithm for discovering process trees, in: Proceedings of the 2012 IEEE Congress on Evolutionary Computation, IEEE, 2012, pp. 1–8.
[8] W. M. P. van der Aalst, A. H. M. ter Hofstede, YAWL: yet another workflow language, Information systems 30 (4) (2005) 245–275.
[9] Object Management Group, Notation (BPMN) version 2.0, OMG Specification.
[10] G. Greco, A. Guzzo, L. Pontieri, D. Sacca, Discovering expressive process models by clustering log traces, IEEE Transactions on Knowledge and Data Engineering 18 (8) (2006) 1010–1027.
[11] A. Rozinat, W. M. P. van der Aalst, Conformance checking of processes based on monitoring real behavior, Information Systems 33 (1) (2008) 64–95.
[12] J. Muñoz-Gama, J. Carmona, A fresh look at precision in process conformance, in: International Conference on Business Process Management, Springer, 2010, pp. 211–226.
[13] S. K. L. M. vanden Broucke, J. De Weerdt, J. Vanthienen, B. Baesens, Determining process model precision and generalization with weighted artificial negative events, IEEE Transactions on Knowledge and Data Engineering 26 (8) (2014) 1877–1889.
[14] S. J. J. Leemans, D. Fahland, W. M. P. van der Aalst, Scalable process discovery and conformance checking, Software & Systems Modeling (2016) 1–33.
[15] W. M. P. van der Aalst, A. Adriansyah, A. K. A. De Medeiros, F. Arcieri, T. Bai, T. Blickle, J. C. Bose, P. van den Brand, R. Brandtjen, J. Bijls, et al., Process mining manifesto, in: International Conference on Business Process Management, Springer, 2011, pp. 169–194.
[16] J. C. A. M. Bijls, B. F. van Dongen, 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 (01) (2014) 1440001.
[17] W. M. P. van der Aalst, Verification of workflow nets, in: International Conference on Application and Theory of Petri Nets, Springer, 1997, pp. 407–426.
[18] A. Adriansyah, J. Muñoz-Gama, J. Carmona, B. F. van Dongen, W. M. P. van der Aalst, Alignment based precision checking, in: International Conference on Business Process Management, Springer, 2012, pp. 137–149.
[19] W. M. P. Van der Aalst, A. Adriansyah, B. F. van Dongen, Replaying history on process models for conformance checking and performance analysis, Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 2 (2) (2012) 182–192.
[20] A. Adriansyah, B. F. van Dongen, W. M. P. van der Aalst, Conformance checking using cost-based fitness analysis, in: Proceedings of the 15th IEEE International Enterprise Distributed Object Computing Conference, IEEE, 2011, pp. 55–64.
[21] A. Adriansyah, J. Muñoz-Gama, J. Carmona, B. F. van Dongen, W. M. P. van der Aalst, Measuring precision of modeled behavior, Information Systems and e-Business Management 13 (1) (2015) 37–67.
[22] B. F. van Dongen, A. K. A. de Medeiros, H. M. W. Verbeek, A. J. M. M. Weijters, W. M. P. van der Aalst, The ProM framework: A new era in process mining tool support, in: International Conference on Application and Theory of Petri Nets, Springer, 2005, pp. 444–454.
[23] J. De Weerdt, M. De Backer, J. Vanthienen, B. Baesens, A robust F-measure for evaluating discovered process models, in: Proceedings of the IEEE Symposium on Computational Intelligence and Data Mining, IEEE, 2011, pp. 148–155.
[24] S. K. L. M. vanden Broucke, J. De Weerdt, B. Baesens, J. Vanthienen, Improved artificial negative event generation to enhance process event logs, in: International Conference on Advanced Information Systems Engineering, Springer, 2012, pp. 254–269.
[25] B. L. Welch, The generalization of student’s problem when different population variances are involved, Biometrika 34 (1/2) (1947) 28–35.
[26] N. Tax, Validation of precision measures - event logs and process models, Eindhoven University of Technology. Dataset, http://dx.doi.org/10.4121/uuid:991753f7-4240-4ba6-a8a8-67174a08c51b (2017).