Relating Process Models and Event Logs
21 Conformance Propositions

Wil M.P. van der Aalst

Process and Data Science (Informatik 9),
RWTH Aachen University, D-52056 Aachen, Germany
wvdaalst@rwth-aachen.de

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 can be used to confront a descriptive or normative model with the actual behavior observed. Quantifying the relationship between process models and event logs is notoriously difficult. Existing approaches seem to make ad-hoc choices, leading to results that are not easy to interpret. To address these problems, we present *21 conformance propositions*. These describe expected or desired properties of conformance measures. These cover notions such as *recall* (also called fitness), *precision*, and *generalization*. These propositions are put into a broader frame of *Log-to-Log* (L2L), *Model-to-Model* (M2M), and *Log-to-Model* (L2M) comparisons. Moreover, it is argued that conformance measures need to consider the *likelihood* of traces. Real-life event logs are incomplete showing only sample behavior. Moreover, observed traces tend to follow a Pareto distribution. Yet, mainstream process mining approaches stick to a "binary view" on process models (observed traces are fitting or not). This is hindering progress. The goal is to trigger a discussion by clearly formulating the challenges and requirements (rather than proposing new measures or techniques).

Keywords: Process mining · Conformance checking · Recall · Precision · Generalization.

1 Introduction

In process mining, there are two main types of artifacts: *event logs* and *process models* [1]. In the background, there is the "real process" that is responsible for the events in the event log. However, this process is only fully known in case one is using artificial (also called synthetic) event logs. Therefore, we assume that we only have event logs and models. However, we assume that the models are "rich enough" to generate event data. This leads to the two scenarios shown in Figure 1. Scenario 1 starts from a model that is used to generate an event log. This log can be used as input for a process discovery technique that produces another model. One can compare the discovered model with the original model using a so-called *M2M (Model-to-Model)* assessment. We can also conduct two *L2M (Log-to-Model)* assessments by comparing the log with both the original model and the discovered model. Scenario 2 starts from an event log that is used

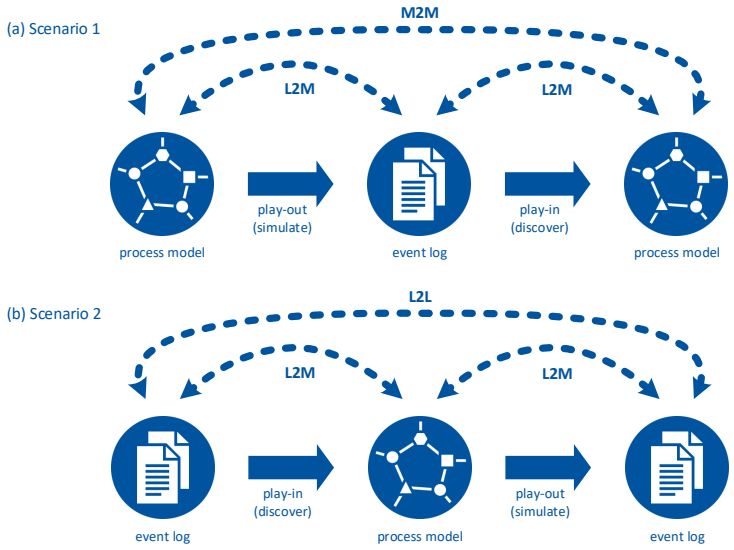


Fig. 1. Two scenarios and three types of comparisons: *L2M (Log-to-Model)*, *M2M (Model-to-Model)*, and *L2L (Log-to-Log)*.

to discover a process model that is subsequently used to produce another event log. The original and the generated event logs can be compared using an *L2L (Log-to-Log)* assessment.

Event logs only contain example behavior. This not only makes process discovery challenging; it is also difficult to assess the quality of the process model in relation to the log. Since process mining focuses on the relationship between observed behavior and modeled behavior (i.e., L2M assessments), it is important to quantify the relationship between an event log $l \in \mathcal{L}$ and a process model $m \in \mathcal{M}$. This is reflected by the different conformance measures that have been proposed [1, 3, 4, 7–10, 14, 16, 18, 21, 22]. When learning a process model from event data, there is typically a trade-off between the following four quality dimensions [1]: (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 be as simple as possible. 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.

This paper does *not* aim to provide concrete conformance measures (although some examples are given). Instead, the focus is on characteristic/desired properties of such measures. We propose 21 *conformance propositions*, i.e., properties that recall, precision, and generalizations measures could or should satisfy for any model and log combination.[1] Researchers and developers creating new measures should be clear on what conformance propositions they aim to support and ensure that the implemented measures support these. Users of existing conformance measures should be aware of seemingly obvious conformance propositions that are violated by existing approaches.

Next to defining 21 conformance propositions, this paper provides two additional contributions. First of all, we provide a more *holistic perspective* on the challenges related to comparing models and logs. Second, we show that *probabilities* play a key role in such definition. The likelihood of traces in models and the frequency of traces in logs are key elements that should not be ignored.

It is important to note that this paper focuses on *trace-based control-flow-oriented conformance measures*, i.e., we abstract from other event attributes (resources, products, time, costs, etc.) and consider full traces (not local patterns). This scope is clearly limited. However, it helps to focus the discussion.

The remainder is organized as follows. Section 2 presents some preliminaries. Next to event logs also process models which define trace likelihoods are defined. Sections 3, 4, and 5 discuss L2L (Log-to-Log), M2M (Model-to-Model), and L2M (Log-to-Model) comparisons and define example measures. Section 6 describes the 21 conformance propositions. Section 7 reflects on the conformance propositions using the simple measures defined in sections 3, 4, and 5. Section 8 discusses the viewpoint used in the paper and possible alternative viewpoints. Section 9 concludes the paper.

2 Preliminaries

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. However, unlike most other papers, we attach probabilities to traces.

2.1 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.

[1] The Merriam-Webster dictionary defines the noun *proposition* as "an expression in language or signs of something that can be believed, doubted, or denied or is either true or false" [15]. We use the term in this manner. We do not state that all 21 conformance propositions should hold universally. We merely use them as potential requirements and discuss their implications.

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 $t = \langle a, b, c, d, c \rangle \in \mathcal{T}$ 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 $l \in \mathcal{L}$. $\overline{\tau}(l) = \{t \in \mathcal{T} \mid t \notin \tau(l)\}$ is the complement of l, i.e., non-observed traces.*

An event log is a multiset of 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 $\langle a, b, d \rangle$. $l(t)$ is the number of times t appears in l, e.g., $l(\langle b, a, d \rangle) = 3$. We assume that the usual operators are defined for multisets. $l_1 \uplus l_2$ is the union of two multisets, $|l|$ is the number of elements, and $l_1 \setminus l_2$ is the difference. $l_1 \cap l_2$ is the intersection of two multisets. Hence, $(l_1 \cap l_2)(t) = l_1(t)\ max\ l_2(t)$. $[t \in l \mid b(t)]$ is the multiset of all elements in l that satisfy some condition b. $l_1 \leq l_2$ means that l_1 is contained in l_2. For example, $[a^2, b] \leq [a^2, b^2, c]$, but $[a^2, b^3] \not\leq [a^2, b^2, c^2]$ and $[a^2, b^2, c] \not\leq [a^3, b^3]$.

$\overline{\tau}(l)$ is the set traces that were not observed in the event log. Note that $\tau(l)$ and $\overline{\tau}(l)$ partition the universe of traces $\mathcal{T}$.

2.2 Process Models

Often the behavior of a process model m is simply defined by the set of traces allowed by m. However, there may be vast differences in the likelihoods of traces. For many processes, the observed traces follow a *Pareto distribution*. Typically, there are a limited number of traces that can be observed frequently, while there are often much more traces that are infrequent.

Actually, many event logs follow the "80-20 rule", i.e., more than 80% of all observed traces are explained by less than 20% of all the observed trace variants. Formally this means there is a set of traces $X \subseteq \tau(l)$ such that $|[t \in l \mid t \in X]|/|l| > 0.8$ and $|X|/|\tau(l)| < 0.2$.

Because of the distribution of traces in the log, it is often too simplistic to describe a model as a set of traces. Since "behavior is not binary", we use a *trace likelihood function* to describe the probability of each trace.

Definition 3 (Trace Likelihood Function). *$\Pi = \mathcal{T} \rightarrow [0, 1]$ is the set of all* trace likelihood functions. *$\pi \in \Pi$ assigns a likelihood $\pi(t)$ to any trace $t \in \mathcal{T}$ such that $\sum_{t \in \mathcal{T}} \pi(t) = 1$.*

When describing a trace likelihood function, we assume that the number of traces is countable. Therefore, $(\Omega, \mathcal{F}, P)$ with $\Omega = \mathcal{T}$, $\mathcal{F} = \mathbb{P}(\mathcal{T})$, and $P \in \mathcal{F} \rightarrow [0, 1]$ such that $P(X) = \sum_{t \in X} \pi(t)$, defines a proper *probability space*.[2] Note that if $\mathcal{A}$ is countable, then also $\mathcal{T}$ is countable.

[2] A probability space $(\Omega, \mathcal{F}, P)$ consists of three parts: a sample space Ω which defines the set of all possible outcomes, a set of "events" $\mathcal{F}$ where each event is a set containing zero or more outcomes, and an assignment of probabilities to the events $P \in \mathcal{F} \rightarrow [0, 1]$.

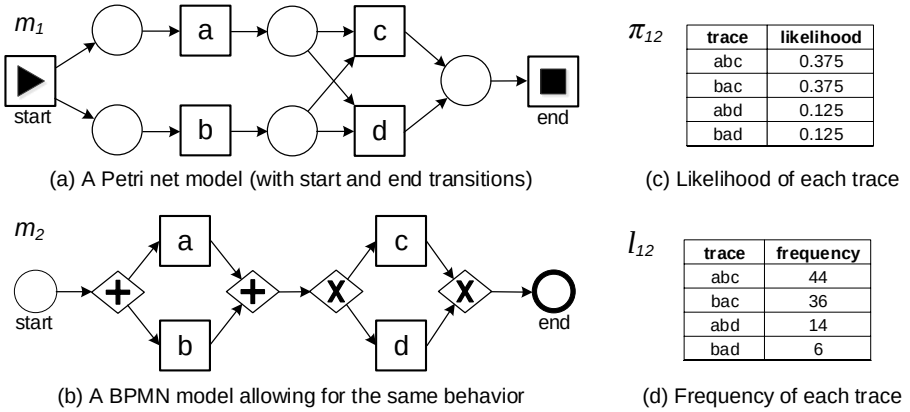


Fig. 2. Two process models m_1 and m_2 having the same behavior: $\tau(m_1) = \tau(m_2) = \{\langle a, b, c\rangle, \langle a, b, d\rangle, \langle b, a, c\rangle, \langle b, a, d\rangle\}$. m_1 is a Petri net with a special start activity (occurs once at the beginning) and a special end activity to signal the end of the trace. m_2 is a BPMN (Business Process Model and Notation) model. The likelihood of each trace $\pi_{12} = \pi(m_1) = \pi_(m_2)$ (c) and an example log l_{12} (d) are also shown.

For pragmatic reasons, we can also restrict $\mathcal{A}$ to be finite and $\mathcal{T}$ to traces of a maximal length (e.g., the maximal length seen in any log). For any set of traces $X \subseteq \mathcal{T}$, $\pi(X) = \sum_{t \in X} \pi(t)$ is the probability that a run of the process π yields a trace in X. We assume that traces are sampled independently. When considering time, this is not realistic (due to queueing), but for the control-flow (routing of a case through the model) this is a common assumption.

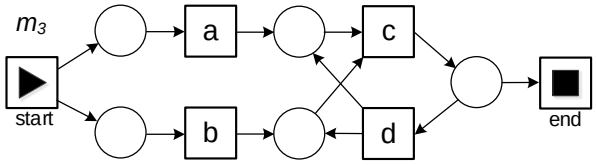
Definition 4 (Process Model). *$\mathcal{M}$ is the set of process models. A process model $m \in \mathcal{M}$ has a set of traces $\tau(m) \subseteq \mathcal{T}$ and a trace likelihood function $\pi(m) \in \Pi$.*

The relation between $\tau(m)$ and $\pi(m)$ is not very strict. Sometimes we will require $\tau(m) \subseteq \{t \in \mathcal{T} \mid \pi(t) > 0\}$ (all modeled traces are possible), $\{t \in \mathcal{T} \mid \pi(t) > 0\} \subseteq \tau(m)$ (the model allows for all traces possible), or even $\tau(m) = \{t \in \mathcal{T} \mid \pi(t) > 0\}$. Trace set $\tau(m)$ may be an abstraction of the real process and leave out unlikely behavior. Due to simplification, the trace set $\tau(m)$ may also allow for traces that cannot happen (e.g., particular interleavings or loops).

We define $\overline{\tau}(m) = \{t \in \mathcal{T} \mid t \notin \tau(m)\}$ as the complement of the set of traces allowed by model $m \in \mathcal{M}$ (like we did for event logs).

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 like $\tau(m) \subseteq \mathcal{T}$ and $\pi(m) \in \Pi$.

Figure 2 shows two process models that have the same behavior: $\tau(m_1) = \tau(m_2) = \{\langle a, b, c\rangle, \langle a, b, d\rangle, \langle b, a, c\rangle, \langle b, a, d\rangle\}$ and $\pi_{12} = \pi(m_1) = \pi(m_2)$ with $\pi_{12}(\langle a, b, c\rangle) = 0.375$, $\pi_{12}(\langle b, a, c\rangle) = 0.375$, $\pi_{12}(\langle a, b, d\rangle) = 0.125$, $\pi_{12}(\langle b, a, d\rangle) = 0.125$, and

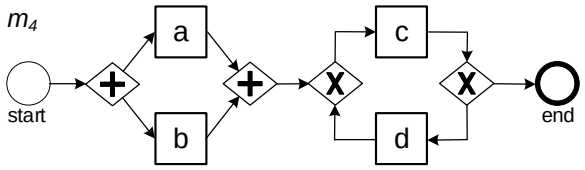


(a) A Petri net model (with start and end transitions)

π_{34}

trace	likelihood
abc	0.375
bac	0.375
abcdc	0.09375
bacdc	0.09375
abcdcdc	0.0234375
bacdcdc	0.0234375
abcdcdcdc	0.005859375
...	...

(c) Likelihood of each trace



(b) A BPMN model allowing for the same behavior

l_{34}

trace	frequency
abc	4
bac	3
abcdc	2
bacdcdcdcdc	1

(d) Frequency of each trace

Fig. 3. Two process models m_3 and m_4 allowing for the same set of traces ($\tau(m_3) = \tau(m_4)$) and having the identical trace likelihoods ($\pi_{34} = \pi(m_3) = \pi(m_4)$) and an example log l_{34} (d).

$\pi_{12}(t) = 0$ for all other traces t. The two process models m_1 and m_2 are not graphically annotated with probabilities. However, there are extensions of these notations to do so. We can use for example Stochastic Petri Nets (SPN) or BPMN-based simulation models. For the models in Figure 2, we assume that initially there is a "race" between a and b where both have a 50% probability to win. After both a and b occurred, c occurs with 75% probability and d occurs with 25% probability. This explains the likelihoods in Figure 2(c).

Figure 2(d) shows a possible event log l_{12} composed of 100 traces generated by one of these models: $l_{12}(\langle a, b, c\rangle) = 44$, $l_{12}(\langle b, a, c\rangle) = 36$, $l_{12}(\langle a, b, d\rangle) = 14$, and $l_{12}(\langle b, a, d\rangle) = 6$.

Process models may allow for an infinite number of traces. We use Figure 3 to illustrate this. Again both processes (m_3 and m_4) express the same behavior. Again there is a "race" between a and b where both have a 50% probability to win. After a and b, activity c will occur. Then there is a 75% probability that the process ends and a 25% probability that d occurs. 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. $\tau(m_3) = \tau(m_3) = \bigcup_{k\geq 0}\{t_{a,k}, t_{b,k}\}$. $\pi_{34} = \pi(m_3) = \pi(m_4)$ with $\pi_{34}(t_{a,k}) = \pi_{34}(t_{b,k}) = 0.5 \cdot (0.25)^k \cdot 0.75$. Some example values are given in Figure 3(c).

Figure 3(d) shows a possible event log l_{34} composed of 10 traces generated by one of the models (m_3 or m_4): $l_{34}(\langle a, b, c\rangle) = 4$, $l_{34}(\langle b, a, c\rangle) = 3$, $l_{34}(\langle a, b, c, d, c\rangle) = 2$, and $l_{34}(\langle b, a, c, d, c, d, c, d, c\rangle) = 1$. Since any log contains only a finite number of traces, one can never observe all traces possible in m_3 or m_4.

2.3 Process Discovery

A discovery algorithm takes an event log as input and returns a process model. For example, m_1 and m_2 in Figure 2 could have been discovered based on event log $l = [\langle a,b,c\rangle^5, \langle b,a,d\rangle^3, \langle a,b,d\rangle^2]$. Ideally, the process model captures the (dominant) behavior observed, but also generalizes without becoming too imprecise. For example, m_1 and m_2 allow for trace $t = \langle b,a,c\rangle$ although this was never observed.

Definition 5 (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]. Most of the discovery algorithms do not include probabilities. However, by replaying the event log on the discovered process model, these can be added easily. 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 $\tau(m_n) = \overline{\tau}(l)$ produces a non-fitting model that allows for all behavior *not* seen in the log. We do not define the probabilities explicitly here. Just assume some function that assigns positive values to all included traces.

3 L2L: Log-To-Log Comparison

As Figure 1 shows, we are interested in three types of conformance measurements. Although the focus will be on L2M (Log-to-Model) comparisons, we also briefly consider L2L (Log-to-Log) and M2M (Model-to-Model) comparisons. We start by providing precision and recall measures for L2L comparisons.

Definition 6 (Set-Based L2L Precision and Recall). *Let $l_1, l_2 \in \mathcal{L}$ be two logs. Set-based L2L precision and recall are defined as follows:*

$$rec_{L2L_{SB}}(l_1,l_2) = \frac{|\tau(l_1) \cap \tau(l_2)|}{|\tau(l_1)|} \qquad prec_{L2L_{SB}}(l_1,l_2) = \frac{|\tau(l_1) \cap \tau(l_2)|}{|\tau(l_2)|} \quad (1)$$

Definition 7 (Multiset-Based L2L Precision and Recall). *Let $l_1, l_2 \in \mathcal{L}$ be two logs. Multiset-based L2L precision and recall are defined as follows:*

$$rec_{L2L_{MSB}}(l_1,l_2) = \frac{|l_1 \cap l_2|}{|l_1|} \qquad prec_{L2L_{MSB}}(l_1,l_2) = \frac{|l_1 \cap l_2|}{|l_2|} \quad (2)$$

$rec_{L2L_{MSB}}$ and $prec_{L2L_{MSB}}$ take into account the frequencies of traces. In $rec_{L2L_{SB}}$ and $prec_{L2L_{SB}}$ it is not relevant how many times the same trace appears in the log. Consider l_{12} in Figure 2(d) and l_{34} in Figure 3(d). $rec_{L2L_{SB}}(l_{12}, l_{34}) = 2/4 = 0.5$, $prec_{L2L_{SB}}(l_{12}, l_{34}) = 2/4 = 0.5$, $rec_{L2L_{MSB}}(l_{12}, l_{34}) = (4+3)/100 = 0.07$, and $prec_{L2L_{MSB}}(l_{12}, l_{34}) = (4+3)/10 = 0.7$. This shows that the latter two measures only make sense if both logs have comparable sizes or things are normalized.

4 M2M: Model-To-Model Comparison

We also define two M2M (Model-to-Model) recall and precision measures.

Definition 8 (Trace-Based M2M Precision and Recall). *Let $m_1, m_2 \in \mathcal{M}$ be two process models. Trace-based M2M precision and recall are defined as follows:*

$$rec_{M2M_{TB}}(m_1, m_2) = \frac{|\tau(m_1) \cap \tau(m_2)|}{|\tau(m_1)|} \tag{3}$$

$$prec_{M2M_{TB}}(m_1, m_2) = \frac{|\tau(m_1) \cap \tau(m_2)|}{|\tau(m_2)|} \tag{4}$$

Note that these recall and precision measures do not take into account the likelihood of traces. Moreover, they are undefined if the number of traces is unbounded, i.e., $|\tau(m_1)| = \infty$ or $|\tau(m_2)| = \infty$. These two issues can be addressed by taking into account trace likelihoods, resulting in the following measures.

Definition 9 (Likelihood-Based M2M Precision and Recall). *Let $m_1, m_2 \in \mathcal{M}$ be two process models. Likelihood-based M2M precision and recall are defined as follows:*

$$rec_{M2M_{LB}}(m_1, m_2) = \frac{\pi(m_1)(\tau(m_1) \cap \tau(m_2))}{\pi(m_1)(\tau(m_1))} \tag{5}$$

$$prec_{M2M_{LB}}(m_1, m_2) = \frac{\pi(m_2)(\tau(m_1) \cap \tau(m_2))}{\pi(m_2)(\tau(m_2))} \tag{6}$$

Note that $\pi(m_1)(\tau(m_1) \cap \tau(m_2))$ quantifies the likelihood of traces in the set $\tau(m_1) \cap \tau(m_2)$ according to model m_1 (recall that $\pi(m_1) \in \Pi = \mathcal{T} \rightarrow [0, 1]$ is a trace likelihood function). To compute *recall*, we compare the traces allowed by both models ($\tau(m_1) \cap \tau(m_2)$) with the traces allowed by the first model ($\tau(m_1)$). $\pi(m_1)$ is used to quantify the fractions of traces. To compute *precision*, we compare the traces allowed by both models ($\tau(m_1) \cap \tau(m_2)$) with the traces allowed by the second model ($\tau(m_2)$). Now, $\pi(m_2)$ is used to quantify the fractions of traces. Note that it does not make any sense to use $\pi(m_1)$ when computing precision, because the behavior in $\tau(m_2) \setminus \tau(m_1)$ will be unlikely according to $\pi(m_1)$. Hence, precision based on $\pi(m_1)$ would always yield a high value even when m_2 allows for lots of additional behavior. Similarly, it does not make any sense to use $\pi(m_2)$ when computing recall.

5 L2M: Log-To-Model Comparison

The recall and precision measures in Definition 9 seem quite natural for models and there is no need for additional notions such as generalization. However, when considering event logs, recall and precision are not addressing the fact that there is a difference between (1) the behavior in the log and (2) possible future behavior. Overfitting models may have perfect recall, but unable to allow for future cases.

In the remainder, we assume the existence of three functions: $rec()$, $prec()$, $gen()$. All three take a log and model as input and return a value between 0 and 1. The higher

the value, the better. In process mining literature one can find many proposals for such functions. Although we provide a few example functions for clarity, we focus on desired properties of conformance measures rather than concrete measures.

Definition 10 (Recall). *A* recall measure $rec \in \mathcal{L} \times \mathcal{M} \to [0,1]$ *aims to quantify the fraction of observed behavior that is allowed by the model.*

Definition 11 (Precision). *A* precision measure $prec \in \mathcal{L} \times \mathcal{M} \to [0,1]$ *aims to quantify the fraction of behavior allowed by the model that was actually observed.*

Definition 12 (Generalization). *A* generalization measure $gen \in \mathcal{L} \times \mathcal{M} \to [0,1]$ *aims to quantify the likelihood that new unseen cases will fit the model.*

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

Definition 13 (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:*

$$rec_{L2M_{TB}}(l,m) = \frac{|\tau(l) \cap \tau(m)|}{|\tau(l)|} \quad prec_{L2M_{TB}}(l,m) = \frac{|\tau(l) \cap \tau(m)|}{|\tau(m)|} \tag{7}$$

Since $|\tau(l)|$ is bounded by the size of the log, $rec_{L2M_{TB}}(l,m)$ is well-defined. However, $prec_{L2M_{TB}}(l,m)$ is undefined when $\tau(m)$ is unbounded (e.g., in case of loops).

Another approach is to build a dummy process model m_l based on the event log and then apply Definition 9.

Definition 14 (Estimation-Based L2M Precision and Recall). *Let $l \in \mathcal{L}$ and $m \in \mathcal{M}$ be an event log and a process model. Let m_l be a model created based on l such that $\tau(m_l) = \tau(l)$ and $\pi(m_l)(t) = l(t)/\,|l|$. Estimation-based L2M precision and recall are defined as follows:*

$$rec_{L2M_{EB}}(l,m) = \frac{\pi(m_l)(\tau(m_l) \cap \tau(m))}{\pi(m_l)(\tau(m_l))} = \frac{|[t \in l \mid t \in \tau(m)]|}{|l|} \tag{8}$$

$$prec_{L2M_{EB}}(l,m) = \frac{\pi(m)(\tau(m_l) \cap \tau(m))}{\pi(m)(\tau(m))} = \frac{\pi(m)(\tau(l) \cap \tau(m))}{\pi(m)(\tau(m))} \tag{9}$$

The above measures do not capture generalization. Note that generalization is good when the process model is also able to explain future traces (i.e., unseen cases). Therefore, generalization takes into account the redundancy of fitting behavior. If the log contains many fitting traces that are also frequent, then generalization is likely to be good.

Definition 15 (L2M Generalization). *Let $l \in \mathcal{L}$ and $m \in \mathcal{M}$ be an event log and a process model. Two example L2M precision measures are defined as follows:*

$$gen_{L2M_q}(l,m) = \frac{|[t \in l \mid t \in \tau(m) \ \wedge \ l(t) \geq q]|}{|l|} \quad \textit{with} \quad q \geq 1 \tag{10}$$

$$gen_{L2M_{HB}}(l,m) = \frac{|[t \in l \mid t \in \tau(m)]| - |\tau(l) \cap \tau(m)|}{|l|} \tag{11}$$

$gen_{L2M_q}(l, m)$ uses a threshold q. The more fitting traces there are that have a frequency of at least q, the better. Non-fitting traces or traces that are unique lower generalization. $gen_{L2M_{HB}}(l, m)$ measures the fraction of traces that is fitting, but subtracts a penalty for each unique trace. Consider $l' = [\langle a, b, c\rangle^{60}, \langle b, a, d\rangle^{38}, \langle a, b, d\rangle^{1}, \langle c, b, a\rangle^{1}]$ and m_1 in Figure 2 and let $q = 10$. $gen_{L2M_q}(l', m_1) = (60 + 38)/100 = 0.98$ and $gen_{L2M_{HB}}(l', m_1) = ((60 + 38 + 1) - 3)/100 = 0.96$.

Assume now that $l'' = [\langle a, b, c\rangle^{7}, \langle b, a, d\rangle^{1}, \langle a, b, d\rangle^{1}, \langle c, b, a\rangle^{1}]$. $gen_{L2M_q}(l'', m_1) = 0/10 = 0.0$ and $gen_{L2M_{HB}}(l'', m_1) = ((7 + 1 + 1) - 3)/10 = 0.6$. Note that, although the set of traces in the event log did not change ($\tau(l') = \tau(l'')$), it is reasonable to assume that generalization is impacted.

6 A Collection of Conformance Propositions

Many recall measures have been proposed in literature [1, 3, 4, 7–10, 14, 16, 18, 21, 22]. In recent years, also several precision measures have been proposed [5, 20]. Only few generalization measures have been proposed [3]. The goal of this paper is not to present new measures, but to define properties for such quality measures. Through this, we hope to facilitate a better understanding of process discovery and conformance checking. Moreover, these properties may help to choose an existing measure or to define new ones.

6.1 General Propositions

In the remainder, we provide *21 conformance propositions*. The Merriam-Webster dictionary defines the noun *proposition* as "an expression in language or signs of something that can be believed, doubted, or denied or is either true or false" [15]. 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 "+". For example, the first two propositions are commonly accepted; the computation of a quality measure should be deterministic (**DetPro**$^+$) and only depend on behavioral aspects (**BehPro**$^+$). The latter is a design choice. We deliberately exclude simplicity notions. More controversial propositions are marked with a "0" (rather than a "+").

Proposition 1 (DetPro$^+$). *$rec()$, $prec()$, $gen()$ are deterministic functions, i.e., the measures $rec(l, m)$, $prec(l, m)$, $gen(l, m)$ are fully determined by $l \in \mathcal{L}$ and $m \in \mathcal{M}$.*

Proposition 2 (BehPro$^+$). *For any $l \in \mathcal{L}$ and $m_1, m_2 \in \mathcal{M}$ such that $\tau(m_1) = \tau(m_2)$ and $\pi(m_1) = \pi(m_2)$: $rec(l, m_1) = rec(l, m_2)$, $prec(l, m_1) = prec(l, m_2)$, and $gen(l, m_1) = gen(l, m_2)$, i.e., the measures are fully determined by the behavior observed and the behavior described by the model (representation does not matter).*

6.2 Recall Propositions

First, we consider a few *recall propositions*. $rec \in \mathcal{L} \times \mathcal{M} \to [0, 1]$ aims to quantify the fraction of observed behavior that is allowed by the model.

Proposition 3 (RecPro1$^+$). *For any $l \in \mathcal{L}$ and $m_1, m_2 \in \mathcal{M}$ such that $\tau(m_1) \subseteq \tau(m_2)$: $rec(l, m_1) \leq rec(l, m_2)$.*

Proposition **RecPro1$^+$** states that extending the model to allow for more behavior can never result in a lower recall. Similarly, it cannot be the case that adding fitting behavior to the event logs, lowers recall (**RecPro2$^+$**). Adding non-fitting behavior to the log, cannot improve recall (**RecPro3$^+$**).

Proposition 4 (RecPro2$^+$). *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 $\tau(l_3) \subseteq \tau(m)$: $rec(l_1, m) \leq rec(l_2, m)$.*

Proposition 5 (RecPro3$^+$). *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 $\tau(l_3) \subseteq \overline{\tau}(m)$: $rec(l_1, m) \geq rec(l_2, m)$.*

For any natural number k: $l^k(t) = k \cdot l(t)$, e.g., if $l = [\langle a, b\rangle^3, \langle c\rangle^2]$, then $l^4 = [\langle a, b\rangle^{12}, \langle c\rangle^8]$. We use this notation to enlarge event logs without changing the distribution. One could argue that this should not influence recall (**RecPro4^0**), e.g., $rec([\langle a, b\rangle^3, \langle c\rangle^2], m) = rec([\langle a, b\rangle^{12}, \langle c\rangle^8], m)$. However, unlike the previous propositions, this requirement is debatable as is indicated by the "0" tag.

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

Finally, we provide a proposition stating that recall should be 1 if all traces in the log fit the model (**RecPro5$^+$**). As a result, the empty log has recall 1 for any model.

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

Based on this proposition, $rec(l, disc_{ofit}(l)) = rec(l, disc_{ufit}(l)) = 1$ for any log l.

6.3 Precision Propositions

Precision ($prec \in \mathcal{L} \times \mathcal{M} \rightarrow [0, 1]$) aims to quantify the fraction of behavior allowed by the model that was actually observed. In [20] several precision axioms were introduced. These partly overlap with the propositions below (but more are added and some have been strengthened). **PrecPro1$^+$** states that removing behavior from a model that does not happen in the event log cannot lead to a lower precision. Adding fitting traces to the event log can also not lower precision (**PrecPro2$^+$**). However, adding non-fitting traces to the event log should not change precision (**PrecPro3^0**).

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

Proposition 9 (PrecPro2$^+$). *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 $\tau(l_3) \subseteq \tau(m)$: $prec(l_1, m) \leq prec(l_2, m)$.*

Proposition 10 (PrecPro3^0). *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 $\tau(l_3) \subseteq \overline{\tau}(m)$: $prec(l_1, m) = prec(l_2, m)$.*

One could also argue that duplicating the event log should not influence precision because the distribution remains the same (**PrecPro4**^{0}), e.g., $prec([\langle a,b\rangle^{20}, \langle c\rangle^{20}], m) = prec([\langle a,b\rangle^{40}, \langle c\rangle^{40}], m)$.

Proposition 11 (PrecPro4^0). *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**$^+$). One could also argue that if all modeled behavior was observed, precision should also be 1 (**PrecPro6**^{0}). The latter proposition is debatable, because it implies that the non-fitting behavior cannot influence perfect precision. Consider for example extreme cases where the model covers just a small fraction of all observed behavior (or even more extreme situations like $\tau(m) = \emptyset$).

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

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

According to **PrecPro5**$^+$and **PrecPro6**^{0}, $rec(l, disc_{ofit}(l)) = 1$ for any log l.

6.4 Generalization Propositions

Generalization ($gen \in \mathcal{L} \times \mathcal{M} \rightarrow [0, 1]$) aims to quantify the likelihood that new unseen cases will fit the model. This conformance dimension is a bit different than the other two 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**$^+$), extending the log with fitting behavior cannot lower generalization (**GenPro2**$^+$), and extending the log with non-fitting behavior cannot improve generalization (**GenPro3**$^+$).

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

Proposition 15 (GenPro2$^+$). *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$^+$). *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 $\tau(l_3) \subseteq \overline{\tau}(m)$: $gen(l_1, m) \geq gen(l_2, m)$.*

Duplicating the event log does not necessarily influence recall and precision. According to propositions **RecPro4**^{0} and **PrecPro4**^{0} 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**$^+$). For non-fitting logs, adding redundancy without changing the distribution can only lower generalization (**GenPro5**$^+$). Note that

GenPro4$^+$ and **GenPro5**$^+$ are special cases of **GenPro6**^{0} and **GenPro7**^{0}. **GenPro6**^{0} and **GenPro7**^{0} 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**^{0}). For a log where more than half of the traces is non-fitting, duplication can only lower generalization (**GenPro6**^{0}).

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

Proposition 19 (GenPro6^0). *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 (GenPro7^0). *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**^{0}). Nevertheless, it is marked as controversial because the proposition would also need to hold for an empty event log.

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

This concludes this first inventory of conformance propositions. As mentioned, their purpose is twofold. First of all, they help to understand the goals and challenges of process discovery. Second, they provide a checklist for existing and future conformance measures. As demonstrated in [20] for precision, most of the existing approaches violate seemingly obvious requirements. This justifies the formulation of the above 21 conformance propositions.

7 Example Evaluation

Although we do not aim to propose new conformance measures, we defined 6 L2M measures in Section 5. We use these example measures to discuss the usefulness of the 21 conformance propositions. Table 7 shows for each of the six conformance measure defined in Section 5 which conformance propositions always hold (marked with $\surd$). The cells tagged with $\times$ are measure-proposition combinations that can be violated.

Consider for example proposition **RecPro1**$^+$ and measure $rec_{L2M_{EB}}$. Obviously, $|[t \in l \mid t \in \tau(m_1)]| / |l| \leq |[t \in l \mid t \in \tau(m_2)]| / |l|$ if $\tau(m_1) \subseteq \tau(m_2)$. Next, we consider **PrecPro1**$^+$ and $prec_{L2M_{TB}}$. $|\tau(l) \cap \tau(m_1)| / |\tau(m_1)| \leq |\tau(l) \cap \tau(m_2)| / |\tau(m_2)|$ if $\tau(m_1) \subseteq \tau(m_2)$ and $\tau(l) \cap (\tau(m_2) \setminus \tau(m_1)) = \emptyset$ (because $\tau(l) \cap \tau(m_2) = \tau(l) \cap \tau(m_1)$). Similarly, we can show the correctness of all $\surd$ entries in Table 7.

More interesting are the $\times$ entries in the table. $prec_{L2M_{EB}}$ does not satisfy proposition **PrecPro1**$^+$ because $\pi(m_1)$ may be very different from $\pi(m_2)$ even when $\tau(m_1) = \tau(m_2)$. Both generalization measures do not satisfy **GenPro2**$^+$. When adding fitting

Table 1. Overview of the conformance propositions that hold for the six example measures (under the assumption that $l \neq [\,]$ and $\pi(m)(\tau(m)) > 0$): ✓ means that the proposition holds for any log and model and × means that the proposition does not always hold.

Proposition	Name	$rec_{L2M_{TB}}$	$rec_{L2M_{EB}}$	$prec_{L2M_{TB}}$	$prec_{L2M_{EB}}$	gen_{L2M_q}	$gen_{L2M_{HB}}$
1	**DetPro**$^+$	✓	✓	✓	✓	✓	✓
2	**BehPro**$^+$	✓	✓	✓	✓	✓	✓
3	**RecPro1**$^+$	✓	✓				
4	**RecPro2**$^+$	✓	✓				
5	**RecPro3**$^+$	✓	✓				
6	**RecPro4**^{0}	✓	✓				
7	**RecPro5**$^+$	✓	✓				
8	**PrecPro1**$^+$			✓	×		
9	**PrecPro2**$^+$			✓	✓		
10	**PrecPro3**^{0}			✓	✓		
11	**PrecPro4**^{0}			✓	✓		
12	**PrecPro5**$^+$			✓	✓		
13	**PrecPro6**^{0}			✓	✓		
14	**GenPro1**$^+$					✓	✓
15	**GenPro2**$^+$					×	×
16	**GenPro3**$^+$					✓	✓
17	**GenPro4**$^+$					✓	✓
18	**GenPro5**$^+$					✓	✓
19	**GenPro6**^{0}					✓	✓
20	**GenPro7**^{0}					×	×
21	**GenPro8**^{0}					×	×

traces, these traces may be low frequent, possibly lowering both generalization measures. They also do not satisfy **GenPro7**^{0}. From the two definitions we can see that replication can only lead to better generalization values even when the majority of traces is not fitting. gen_{L2M_q} and $gen_{L2M_{HB}}$ also do not satisfy **GenPro8**^{0} because the proposition does not depend on the event log as long as any trace can generated by the model. The two measures do depend on the event log. Actually, $gen_{L2M_{HB}}$ can only approach value 1.

The × entries in the table illustrate that the properties are not as straightforward as they may seem at first glance.

8 Discussion

The goal of this paper is to trigger discussion and to address common misconceptions. When it comes to relating process models and event logs different viewpoints are possible and often authors tend to impose their own assumptions on others without fully understanding the different settings. Therefore, we put the definitions and propositions presented in this paper in the context of related work. As mentioned in the introduction, a broad range of conformance notions has been proposed in literature [1, 3, 4, 7–10, 14, 16, 18, 21, 22]. Moreover, different evaluation frameworks have been proposed [11, 19,

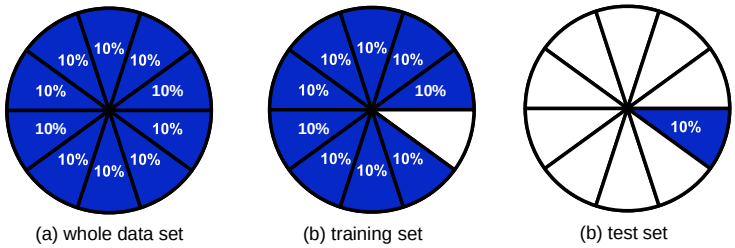


Fig. 4. The original data set is partitioned into 10 equal size subsets for 10-fold cross-validation. This allows for 10 experiments where 9 subsets are used for training and one subset is used for testing.

21]. The four quality dimensions described in [1] (recall/fitness, precision, generalization, simplicity) are generally accepted. In [6] it was shown that removing one of these dimensions may lead to degenerate models that are obviously undesirable but scoring well on the three remaining dimensions.

8.1 Evaluating Models Versus Evaluating Algorithms

In this paper, we considered the setting where two artifacts (e.g., a process model and an event log) are given and need to be compared. We are not evaluating or comparing different discovery algorithms. Hence, one *cannot* use cross-validation. Cross-validation is a technique to evaluate algorithms by partitioning the original data set into a training set to discover a model, and a test set to evaluate it. In k-fold cross-validation, the original data set is randomly partitioned into k equal size subsets (cf. Figure 4). Of the k subsets, a single subset is retained as the validation data for testing the model, and the remaining $k-1$ sets are used as training data. The cross-validation process is then repeated k times (the so-called folds), with each of the k subsets used exactly once as the validation data. The k results from the folds can then be combined to produce a single estimation. The whole data set is used for both training and validation and each data element is used for validation exactly once. Moreover, by comparing the performance on the different folds one gets insights into the robustness of the approach.

Cross-validation approaches can be used to evaluate process discovery algorithms. One can distribute the cases in the original event log over 10 smaller event logs. The discovery algorithm can be applied to the 10 folds and tested on the corresponding test logs. As usual, there is the problem that event logs do not provide negative examples. Therefore, recall is easier to measure than precision. Moreover, one can only use (k-fold) cross-validation to evaluate an algorithm and not a process model. Therefore, it is unrelated to most of the things discussed in this paper, e.g., L2M comparison and the 21 propositions.

8.2 Log Quality Versus Process Variability

In this paper, we assumed that all events are recorded correctly. In [1] a classification is given of different quality issues. Different entities (cases, activity instances, or events)

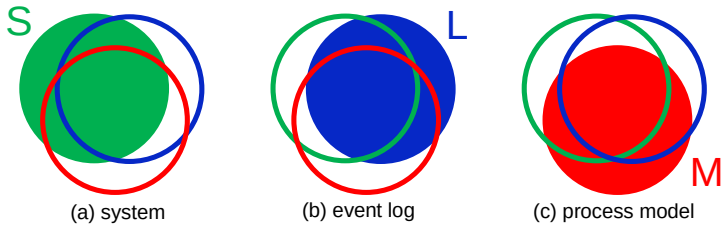


Fig. 5. Venn diagram showing the relationships between system S, log L, and model M.

can be missing in the log, missing in reality (recording of things that did not happen), or concealed in log (recorded but not easy to locate and select). Event attributes may be missing, incorrect, or imprecise. For example, the timestamp of an event may be missing, an event is related to another case, or the value of a timestamp is too coarse-grained (just a date). A particular data quality problem (e.g., a missing attribute) may persist, repeat in an unpredictable manner, or return periodically. In total 108 data quality problems were identified in [1].

Process variability is orthogonal to log quality. A process that exhibits many different behaviors, some of which are frequent and others infrequent, is difficult to analyze even when the data quality is high. See [13] for a detailed discussion of these two dimensions referred to as individual trustworthiness (quality of event data from a data perspective) and global conclusiveness (quality of event data from an analysis perspective). In [17] a generalized conformance checking framework is proposed that caters for the common case, when one does neither fully trust the log nor the model. Although data quality problems are important, the above papers are mostly unrelated to the propositions discussed in this paper.

8.3 System Model Versus Event Log

Scenario 1 shown in Figure 1 starts from a model that is used to generate an event log (e.g. using simulation) which is subsequently used as input for a process discovery technique that produces another model. One can compare the discovered model with the original model using a M2M (Model-to-Model) assessment. The idea is that there is a known "ground truth", i.e., the original process model. In [6] a similar approach is used by introducing the so-called *system* that represents the real process generating the behavior. Assume that S, L, and M are the behaviors allowed by the system (S), found in the event log (L), and allowed by the model (M) respectively. To understand the above let us first assume that S, L, and M are sets of traces. Hence, intersection and union are well-defined (e.g., $S \cap L \cap M$ and $S \cup L \cup M$ are sets of traces). This allows us to use a Venn diagram to show the relationships between system S, log L, and model M (see Figure 5).

Given the Venn diagram showing the relationships between system S, log L, and model M, we can identify the following seven types of behavior (numbers refer to Figure 6):

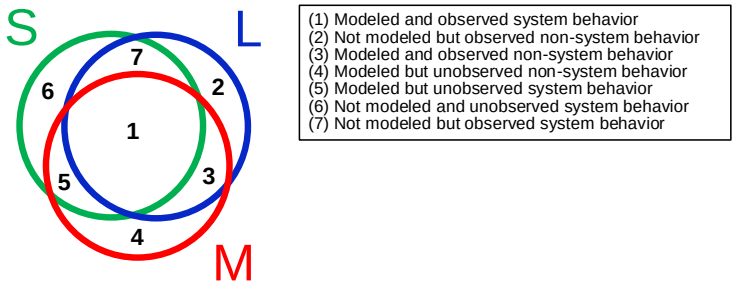


Fig. 6. Given a system S, log L, and model M we can identify seven types of behavior (ignoring non-system behavior that is not modeled and not observed) [6].

- (1) Modeled and observed system behavior ($S \cap L \cap M$).
- (2) Not modeled but observed non-system behavior ($L \setminus (M \cup S)$).
- (3) Modeled and observed non-system behavior ($(L \cap M) \setminus S$).
- (4) Modeled but unobserved non-system behavior ($M \setminus (S \cup L)$).
- (5) Modeled but unobserved system behavior ($(M \cap S) \setminus L$).
- (6) Not modeled and unobserved system behavior ($S \setminus (M \cup L)$).
- (7) Not modeled but observed system behavior ($(S \cap L) \setminus M$).

Clearly, one would like to like S and M to coincide. System-model recall can be defined as $|S \cap M| \,/\, |S|$ and can be maximized by minimizing (6) and (7). Precision can be defined as $|S \cap M| \,/\, |M|$ and can be maximized by minimizing (3) and (4). One can also define log-model recall/precision and system-log recall/precision.

Although Figure 6 is useful and consistent with the earlier definitions, it is also too simplistic and partly misleading. First of all, in reality, we do not know the system S. It is only possible to control and know S in an experimental setting (e.g. using a simulation model). Second, the notion of S suggests that behavior can be classified into system behavior and non-system behavior (i.e., a binary classification). This ignores the likelihood of behavior and the purpose of the model (e.g., striving for the 80/20 model). "Murphy's Law of Process Mining" suggest that in principle anything can happen if one is prepared to wait long enough, but it is uninteresting to return a model that allows for any behavior. This paper clearly shows the need to include probabilities. Third, system S may be changing over time. If we partition the timeline into segments $i \in \{1, 2, 3, \ldots\}$, then we will be confronted with different versions of the system S_i and new events logs L_i. How to relate model M_i to S_i and L_i while also using earlier observations (e.g., $\bigcup_{j<i} L_j$)? Fourth, the assumption that S, L, and M are sets of traces can be very misleading as shown in this paper. When the model has loops, $|M|$ will be infinite. However, $|L|$ is always finite, making both incomparable. Fortunately, can use other abstractions such as causal footprints [1]. See [2] for different abstractions. What is important is that the system, log, and model need to be able to use the same abstractions. The choice of abstraction to characterize system S, log L, and model M will highly impact the outcome of an analysis based on overlaps in Figure 6.

9 Conclusion

As process mining is maturing and more commercial tools become available [12], there is an urgent need to have a set of agreed-upon measures to determine the quality of discovered processes models.

By defining 21 conformance propositions, we discussed possible requirements for conformance measures. It is surprising that seemingly obvious requirements are not met by today's conformance measures. We used a few very simple measures to illustrate this.

The paper also stressed the need to consider probabilities. There are two main reasons for this. Likelihoods of traces in models are as natural as considering frequencies in event logs. Moreover, models with loops allow for infinitely many traces. Models with concurrency allow for a factorial number of interleavings. This shows that we cannot simply count and need a weight function to indicate the relative importance of traces.

We hope that this paper will trigger a deeper and more systematic discussion on existing and future conformance notions.

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. Process Discovery from Event Data: Relating Models and Logs Through Abstractions. *WIREs Data Mining and Knowledge Discovery*, 8(3), 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. 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.
5. A. Adriansyah, J. Munoz-Gama, J. Carmona, B.F. van Dongen, and W.M.P. van der Aalst. Measuring Precision of Modeled Behavior. *Information Systems and e-Business Management*, 13(1):37–67, 2015.
6. 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.
7. J. Carmona, B. van Dongen, A. Solti, and M. Weidlich. *Conformance Checking: Relating Processes and Models*. Springer-Verlag, Berlin, 2018.
8. 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.
9. 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.
10. 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.

11. G. Janssenswillen, N. Donders, T. Jouck, and B. Depaire. A Comparative Study of Existing Quality Measures for Process Discovery. *Information Systems*, 71:1–15, 2017.
12. M. Kerremans. Gartner Market Guide for Process Mining, Research Note G00353970. www.gartner.com, 2018.
13. X. Lu. *Using Behavioral Context in Process Mining: Exploration, Preprocessing and Analysis of Event Data*. Phd thesis, Eindhoven University of Technology, 2018.
14. 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.
15. Merriam-Webster. Online Dictionary. www.merriam-webster.com, 2018.
16. 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.
17. A. Rogge-Solti, A. Senderovich, M. Weidlich, J. Mendling, and A. Gal. In Log and Model We Trust? A Generalized Conformance Checking Framework. 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 179–196. Springer-Verlag, Berlin, 2016.
18. 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.
19. 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.
20. 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.
21. J. De Weerdt, 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.
22. J. De Weerdt, 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.