A Unified Approach for Measuring Precision and Generalization Based on Anti-Alignments

B.F. van Dongen[1], J. Carmona[2], and T. Chatain[3]

[1] Eindhoven University of Technology, The Netherlands
b.f.v.dongen@tue.nl
[2] Universitat Politècnica de Catalunya, Barcelona (Spain)
jcarmona@cs.upc.edu
[3] LSV, ENS Cachan, CNRS, INRIA, Universit Paris-Saclay, Cachan (France)
chatain@lsv.ens-cachan.fr

Abstract. The holy grail in process mining is an algorithm that, given an event log, produces fitting, precise, properly generalizing and simple process models. While there is consensus on the existence of solid metrics for fitness and simplicity, current metrics for precision and generalization have important flaws, which hamper their applicability in a general setting. In this paper, a novel approach to measure precision and generalization is presented, which relies on the notion of *anti-alignments*. An anti-alignment describes highly deviating model traces with respect to observed behavior. We propose metrics for precision and generalization that resemble the *leave-one-out cross-validation* techniques, where individual traces of the log are removed and the computed anti-alignment assess the model's capability to describe precisely or generalize the observed behavior. The metrics have been implemented in ProM and tested on several examples.

1 Introduction

The goal of *process mining* is to gain insights into the behavior of operational information systems by analyzing event logs. Often, process mining is considered synonymous to *process discovery*, which aims at describing observed behavior of a business process in the form of an (executable) process model. The behavior used as input is considered to be given in the form of an *event log* [1].

Traditionally, event logs are considered to be accurate representations of the behavior of a system in such as way that each event refers to an *activity* that was executed in the context of a *case*. By deriving a process model from such an event log, process discovery algorithms give insights into the underlying system.

There has been always a discussion on how to interpret process discovery results, i.e. how does the produced model relate to the actual, but unknown, system in four quality dimensions [2]:

Fitness quantifies how much of the observed behavior is captured by the model,
Generalization quantifies how well the model explains unobserved system behavior,
Precision quantifies how much behavior exists in the model that was not observed, and
Simplicity quantifies the complexity of the model.

In recent years, many metrics have been developed to measure fitness, precision and generalization by comparing the event log with the generated model. For fitness, the state-of-the-art is in alignments, a technique that given a trace and a model produces the most likely explanation for that trace [3]. As the focus of this paper is not on fitness, we assume our models to be perfectly fitting. If a trace in an event log does not fit the model, we use the alignment-based explanation of that trace instead.

Measuring precision is typically done by projecting the observed traces onto the model and then count the number of ways to "escape" from the observed behavior [4]. The more "escaping edges" there are, the lower the precision. The downside of such an approach is that precision only considers the behavior of the model that is very close to the event log.

For generalization, only few metrics exist [5,6]. Some of them are again based on the projection of the log onto the model. For instance, the approach in [6] considers "frequency of use", where models are assumed to generalize if all parts of the model are used equally frequently when reproducing the event log.

In this paper, we take a fresh look at precision and generalization by using the concept of an *anti-alignment* [7]. An anti-alignment of a model with respect to a log is an execution of a model which is as different as possible from the observed log. We instruct and adapt *cross-validation*-based techniques in combination with anti-alignments to derive solid metrics that show a better estimation with respect to the state-of-the-art metrics. The following example illustrates this.

1.1 Motivating Example

Table 1. An example event log

Trace	Frequency
$\langle A, B, D, E, I\rangle$	1207
$\langle A, C, D, G, H, F, I\rangle$	145
$\langle A, C, G, D, H, F, I\rangle$	56
$\langle A, C, H, D, F, I\rangle$	23
$\langle A, C, D, H, F, I\rangle$	28

Throughout the paper, we use an example of a log and several models. The example we use is taken from page 64 of [8] and consists of the simple event log shown in Table 1. The log consists of only five different traces, with various frequencies. The models in Figures 1 through 4 are four examples of models often used to show the differences between fitness, precision and generalization. The models in Figure 5 to Figure 8 are models over the same set of activities with varying loop and/or parallel constructs.

The model in Figure 1 shows the "ideal" process discovery result, i.e. the model that is fitting, fairly precise and properly generalizing. The other models are chosen such that they score poorly on at least one of the dimensions fitness, precision or generalization.

Table 2 compares some conformance metrics for the models in Figure 1 to Figure 8 with the metrics proposed in this paper: P (computed as the average of two metrics P_t and P_l) and G (average of G_t and G_l)[4]. The fitness value F is measured using the alignment based technique of [9] and from the same author are the values of P_a and G_a which are defined in [3]. The precision values in P_{ET} and P_{ETC} are defined in [10]. Finally, the values P_{ne} and G_{ne} denote the precision and generalization metrics

[4] Throughout the paper, we will use P and G letters to denote precision and generalization metrics, respectively.

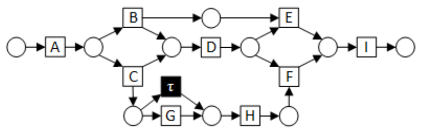

Fig. 1. The ideal model. Fitting, fairly precise and properly generalizing.

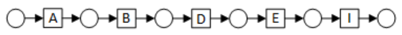

Fig. 2. Most frequent trace. Precise, but not fitting or generalizing.

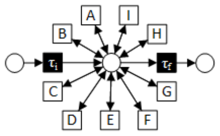

Fig. 3. The flower model. Fitting and generalizing, but very imprecise.

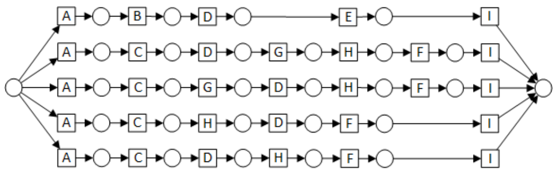

Fig. 4. All traces separate. Fitting, precise, but not generalizing.

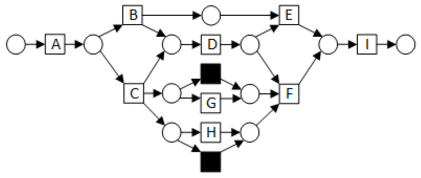

Fig. 5. A model with G and H in parallel.

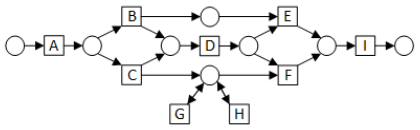

Fig. 6. A model with G and H in self-loops

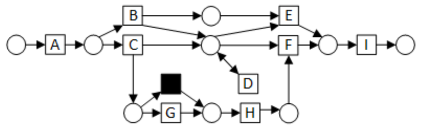

Fig. 7. A model with D in a self-loop

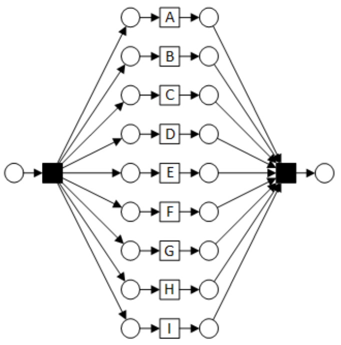

Fig. 8. A model with all transitions in parallel.

Table 2. Precision and Generalization for all models

Model		P_{ET}	P_{ETC}	P_a	G_a	P_{ne}	G_{ne}	F	P_t	P_l	P	G_t	G_l	G
Figure 1	Generating model	0.992	0.994	0.982	0.585	0.995	0.594	1.000	0.886	0.857	0.871	0.270	0.143	0.206
Figure 2	Single trace	1.000	1.000	1.000	0.900	0.893	0.000	0.915	1.000	1.000	1.000	0.000	0.000	0.000
Figure 3	Flower model	0.136	0.119	0.142	0.903	0.117	1.000	1.000	0.000	0.000	0.000	1.000	1.000	1.000
Figure 4	Separate traces	1.000	0.359	1.000	0.145	0.985	0.114	1.000	1.000	1.000	1.000	0.000	0.000	0.000
Figure 5	G,H in parallel	0.894	0.936	0.947	0.511	0.950	0.615	1.000	0.800	0.800	0.800	0.268	0.183	0.225
Figure 6	G,H as self-loops	0.884	0.889	0.947	0.722	0,874	0.615	1.000	0.819	0.357	0.588	0.290	0.643	0.466
Figure 7	D as self-loop	0.763	0.760	0.797	0.728	0.720	0.619	1.000	0.688	0.357	0.523	0.485	0.643	0.564
Figure 8	All parallel	0.273	0.170	0.336	0.178	0.158	0.972	0.739	0.067	0.000	0.033	0.417	0.500	0.459
Figure 11	C,F equal loop	0.820	0.589	0.839	0.585	0.600	0.594	1.000	0.490	0.429	0.459	0.259	0.341	0.300
Figure 12	Round-robin	0.579	0.185	0.889	0.400	0.194	0.118	0.616	0.000	0.000	0.000	0.000	0.000	0.000

from [5], respectively. Clearly, the existing metrics do not agree on all models and do not always agree with the intuition behind precision and generalization. For example, the very precise model of Figure 4 is considered to have a precision of 0.359 by the P_{ETC} metric. Furthermore, the model in Figure 2 is considered to be very generalizing by the G_a metric, while this model clearly does not generalize the observed behavior. Also, the model of Figure 6 scores very high in P_{ET}-P_{ETC}-P_a, although a trace with a thousand G's is possible in the model. One can see that the metrics presented in this paper are free from the aforementioned problems.

The paper is structured as follows: in the next section a brief description of related work is provided. Preliminaries are presented in Section 3. The core of the paper is provided in Sections 4 and 5, where techniques for precision and generalization are presented, respectively. Evaluation with further examples and tool support is reported in Section 6, and Section 7 concludes the paper.

2 Related Work

The seminal work in [2] was the first one in relating observed behavior (in form of a set of traces), and a process model. In order to asses how far can the model deviate from the log, the *follows* and *precedes* relations for both model and log are computed, storing for each relation whereas it *always* holds or only *sometimes*. In case of the former, it means that there is more variability. Then, log and model follows/precedes matrices are compared, and in those matrix cells where the model has a *sometimes* relation whilst the log has an *always* relation indicate that the model allows for more behavior, i.e., a lack of precision. This technique has important drawbacks: first, it is not general since in the presence of loops in the model the characterization of the relations is not accurate [2]. Second, the method requires a full state-space exploration of the model in order to compute the relations, a stringent limitation for models with large or even infinite state spaces.

In order to overcome the limitations of the aforementioned technique, a different approach was proposed in [4]. The idea is to find *escaping arcs*, denoting those situations where the model starts to deviate from the log behavior, i.e., events allowed by the model not observed in the corresponding trace in the log. The exploration of escaping arcs is restricted by the log behavior, and hence the complexity of the method is always bounded. By counting how many escaping arcs a pair (model, log) has, one can estimate the precision of a model. Although being a practical and fast estimation for precision, it may underestimate precision when escaping arcs lead to highly deviating behavior.

In [5] the notion of *weighted artificial negative events* from a log is proposed. Given a log L, an artificial negative event is a trace $\sigma' = \sigma \cdot a$ where $\sigma \in L$, but $\sigma' \notin L$. Algorithms are proposed to weight the confidence of an artificial negative event, and they can be used to estimate the precision and generalization of a process model by computing four sets of events: i) positive events which could be replayed without error (TP), ii) negative events which could be replayed and thus erroneously permitted by the process model (FP), iii) generalized events (negative events with low confidence) which could be replayed without error and confirm model's ability to generalize (AG), and iv) generalized events which could not be replayed by the process model (DG). The formula $\frac{TP}{TP+FP}$ provides a metric for precision, whilst $\frac{AG}{AG+DG}$ provides a metric for generalization. Like in [4], by only considering one step ahead of log/model's behavior, these metrics may underestimate precision/generalization considerably. For instance, the very high generalization provided by this metric to the model of Figure 8 (0.972, i.e., almost perfect generalization) contrast with the value provided by our metric (0.459), the latter being more in line with the real generalization of this model with respect to the log of Table 1. Furthermore, the model used to generate the log is considered more precise (0.995) than a model that only allows for a single trace (0.893), while a model with only one possible trace is as precise as it can be.

3 Preliminaries

In this paper we choose Petri nets as process modeling notation, although the theory presented is valid for any other formalism that has replay semantics.

3.1 Petri nets and Process Mining

Definition 1 ((Labeled) Petri net). *A (labeled) Petri Net [11] is a tuple $N = \langle P, T, \mathcal{F}, m_0, m_f, \Sigma, \lambda \rangle$, where P is the set of places, T is the set of transitions (with $P \cap T = \emptyset$), $\mathcal{F} : (P \times T) \cup (T \times P) \rightarrow \{0, 1\}$ is the flow relation, m_0 is the initial marking, m_f is the final marking,*

Σ is an alphabet of actions and $\lambda : T \rightarrow \Sigma$ labels every transition by an action.

A marking is an assignment of a non-negative integer to each place. If k is assigned to place p by marking m (denoted $m(p) = k$), we say that p is marked with k tokens. Given a node $x \in P \cup T$, its pre-set and post-set are denoted by ${}^\bullet x$ and $x^\bullet$ respectively.

A transition t is *enabled* in a marking m when all places in ${}^\bullet t$ are marked. When a transition t is enabled, it can *fire* by removing a token from each place in ${}^\bullet t$ and putting a token to each place in $t^\bullet$. A marking m' is *reachable* from m if there is a sequence of firings $t_1 t_2 \dots t_n$ that transforms m into m', denoted by $m[t_1 t_2 \dots t_n\rangle m'$. A sequence of actions $a_1 a_2 \dots a_n$ is a *feasible sequence* (or *run*) if there exists a sequence of transitions $t_1 t_2 \dots t_n$ firable from m_0 and such that for $i = 1 \dots n$, $a_i = \lambda(t_i)$. Let $\mathcal{L}(N)$ be the set of feasible sequences of Petri net N. The set of reachable markings from m_0 is denoted by $[m_0\rangle$, and form a graph called *reachability graph*. Let

$\mathcal{L}^n(N) \subseteq \mathcal{L}(N)$ be the set of complete traces of N with length n or shorter, i.e. $\mathcal{L}^n(N) = \{\sigma \in \mathcal{L}(N) \mid m_0[\sigma\rangle m_f \wedge |\sigma| \leq n\}$.

An event log is a collection of traces, where a trace may appear more than once. Formally:

Definition 2 (Event Log). *An* event log (L, ϕ) *is a set of traces* $L \subseteq \Sigma^*$ *and function denoting the occurrence frequency of each trace denoted by* $\phi : L \to \mathbb{N}$*, i.e.* $\phi(t) = 1$ *implies that trace* t *was observed once in the log. If for all* $t \in L$ *holds* $\phi(t) = 1$*, we omit* ϕ *from the notation. The number of traces in a log is denoted by* $|L|$*.*

Quality Dimensions. Process mining techniques aim at extracting from a log L a process model N (e.g., a Petri net) with the goal to elicit the process underlying in $\mathcal{S}$. By relating the behaviors of L, $\mathcal{L}(N)$ and $\mathcal{S}$, particular concepts can be defined [6]. A model N *fits* log L if $L \subseteq \mathcal{L}(N)$. A model is *precise* in describing a log L if $\mathcal{L}(N)\backslash L$ is small. A model N represents a *generalization* of log L with respect to system $\mathcal{S}$ if some behavior in $\mathcal{S}\backslash L$ exists in $\mathcal{L}(N)$. Finally, a model N is *simple* when it has the minimal complexity in representing $\mathcal{L}(N)$, i.e., the well-known *Occam's razor principle*.

3.2 Anti-Alignments

Anti-alignments were introduced in [7]. An anti-alignment is a run of a model which differs sufficiently from all the observed traces in a log. In order to measure how much a run differs from an observed trace, one needs a notion of *distance*; actually, a mapping $d : \Sigma^* \times \Sigma^* \to [0..1]$ is sufficient to define anti-alignments: the other axioms of distance functions (symmetry, triangle inequality, ...) are not required for the definition of anti-alignments. For a log L, we write $d(\sigma, L) = \min_{t \in L} d(\sigma, t)$. If $L = \emptyset$, then $d(\sigma, L) = 1$.

Definition 3 (Anti-alignment). *A* (n, δ)*-anti-alignment of a model* N *w.r.t. a log* L *and a distance function* d *is a run* $\sigma \in \mathcal{L}(N)$ *such that* $|\sigma| = n$ *and* $d(\sigma, L) \geq \delta$*.*

Choice of the distance function. A simple choice of a distance function, used in [7], can be constructed using the Hamming distance after truncating or padding γ to the length of σ, it simply counts the number of mismatches between the actions in the two words, i.e. the number of indices i such that $\sigma_i \neq \gamma_i$ divided by the length of σ_i. But concerning the application to process mining, Hamming distance is usually too rigid: indeed, every symbol σ_i is compared only to the exact corresponding symbol γ_i. This puts for instance the word *ababababab* at distance 1 from *bababababa*. In process mining techniques, other distances are usually preferred (see for instance [3]), typically Levenshtein's distance (or edit distance) which counts how many replacements, deletions and insertions of symbols are needed to obtain γ projected to labeled transitions starting from σ, divided by the length of the longest trace. Unless explicitly stated otherwise, all examples in this paper use the edit distance function with equal costs for remove, replace and insert operations.

Example 1. Consider the Petri net shown in Figure 1, and the log of Table 1. The trace $\langle A, C, G, H, D, F, I\rangle$ is a $(7, \frac{1}{7})$ anti-alignment when considering edit-distance as a distance metric: it can be obtained by inserting G in the observed trace $\langle A, C, H, D, F, I\rangle$;

and the length of the longest trace is 7. Notice that for $\delta > \frac{1}{7}$ there are no anti-alignments for this example. When considering Hamming distance, the same trace is a $(7, \frac{2}{7})$ anti-alignment.

Example 2. Consider the Petri net shown in Figure 2, and the log of Table 1. The trace $\langle A, B\rangle$ is a $(2, \frac{3}{5})$ anti-alignment for this model when considering either edit-distance or Hamming distance as a distance metric: in both case, the closest observed trace is $\langle A, B, D, E, I\rangle$.

Example 3. Consider the Petri net shown in Figure 3, and the log of Table 1. The trace $\langle A, B, D, E, I, A, A, A, A\rangle$ is a $(9, \frac{4}{9})$ anti-alignment for either edit or Hamming distance. Given $n = 9$, the trace $\langle \tau_i, B, A, A, A, A, A, A, A, A, \tau_f\rangle$ is a $(9, 1)$ anti-alignment when considering either edit-distance or Hamming distance as a distance metric. Notice that for any $0 \leq n$ and $0 \leq \delta \leq 1$ an (n, δ) anti-alignment exists.

Example 4. Consider the Petri net shown in Figure 4, and the log of Table 1. The trace $\langle A, C, D, G, H, F, I\rangle$ is a $(7, 0)$ anti-alignment when considering any distance metric.

In the context of process mining, discovered models typically consist of a model and an initial and final marking (where the latter is often implicit), i.e. each execution of the underlying system is assumed to be a sequence in the model from the initial to the final marking. Therefore, we define the concept of a maximal, complete anti-alignment as follows:

Definition 4 (Maximal, Complete Anti-alignments, $\boldsymbol{\Gamma_n^{d,mx}(N, L)}$). *Let N be a model. We define $\Gamma_n^{d,mx}(N, L) \subseteq \mathcal{L}^n(N)$ as the set of maximal, complete anti-alignments, such that for all $\sigma \in \Gamma_n^{d,mx}(N, L)$ holds that $\nexists \sigma' \in \mathcal{L}^n(N) \setminus \Gamma_n^{d,mx}(N, L)$ with $d(\sigma', L) > d(\sigma, L)$.*

In the remainder of this paper, we write $\gamma_n^{d,mx}(N, L)$ whenever we need an arbitrary element from the set $\Gamma_n^{d,mx}(N, L)$.

Note that the set of maximal complete anti-alignments can be empty in case there is no trace in the model with length less than n. Furthermore, in this paper, we use a representative $\gamma_n^{d,mx}(N, L) \in \Gamma_n^{d,mx}(N, L)$ in case there are more maximal complete anti-alignments. One could argue that an average over the entire (by definition finite) set could be used as well. However, this is computationally expensive and for the examples covered in this paper does not add to the qualitative results.

4 Measuring Precision

As stated earlier, a model N is *precise* in describing a log L if $\mathcal{L}(N) \setminus L$ is small, i.e. if the language of the discovered model is not much larger than the observed behavior. As the behavior of model N is often infinite (when loops are present in the model) and the log L is by definition finite, directly comparing $\mathcal{L}(N)$ with L is meaningless. Therefore, classical precision metrics [4] *estimate* precision by analyzing so-called "escaping edges", i.e. the points where the model allows to deviate from observed behavior. The more deviation points there are, the lower the precision. Existing metrics however rely on an abstraction mechanism to decide how to count the deviation points and in [4] a number of abstraction mechanisms is presented, each with their own pro's and cons. Each of the abstraction mechanisms works well in one example, but not in the other and vice versa.

In this paper, we suggest a fresh view on precision, using anti-alignments. The intuition behind our metric is as follows. A very precise process model allows for exactly the observed traces to be executed and not more. Hence, if one trace is removed from the log, this trace becomes the anti-alignment for the remaining log as it is the only execution of the model that is not in the log. We use this property to estimate precision.

Definition 5 (Trace-based Precision). *Let (L, ϕ) be an event log and N a model. We define trace-based precision as follows:*

$$P_t(N, L) = 1 - \frac{1}{|L|} \cdot \sum_{\sigma \in L} d(\sigma, \gamma_{|\sigma|}^{d,mx}(N, L \setminus \{\sigma\})).$$

We assume a perfectly fitting log, i.e. $\sigma \in \mathcal{L}^{|\sigma|}(N)$ and hence $\gamma_{|\sigma|}^{d,mx}(N, L \setminus \{\sigma\})$ exists.

For each trace σ in the log, we compute a maximal anti-alignment γ for the model N and the log without that trace $L \setminus \{\sigma\}$. This anti-alignment is guaranteed to reach the final marking m_f and hence represents an element of $\mathcal{L}(N)$. Then, we compute the distance between σ and γ which we average over the log, *not* taking into account the relative frequencies of the traces in the log. If the language of the model equals the log, then the anti-alignments γ will be equal to σ for every σ, hence the precision is 1. If for every trace σ, an anti-alignment can be produced which has maximal distance from σ, the precision is 0.

Frequencies of traces are not considered as the comparison is between the language of the model and the observed traces. Observing one trace more frequently than another should not influence the precision of the model as the amount of unobserved behavior does not change. This contrasts with current metrics for precision (e.g., [4]).

In trace-based precision, the length of the anti-alignment considered is bounded by the length of the removed trace σ. This guarantees that an anti-alignment exists in the log without trace σ, but also limits the possibility to see imprecise executions of the model that are much longer than the lengths of the observed traces. Therefore, we also define a log-based precision metric, which uses an anti-alignment of the model with respect to the entire log of a much greater length than the longest trace observed in the log.

Definition 6 (Log-based Precision). *Let (L, ϕ) be an event log and N a model. We define Log-based precision as follows:*

$$P_l^n(N, L) = 1 - d(\gamma_n^{d,mx}(N, L), L).$$

where n represents the maximal length of the anti-alignment, typically in the order of several times the length of the longest trace in the log.

The log-based precision metric uses a single anti-alignment of considerable maximum length to determine the amount of behavior allowed by the model, but not observed in the event log. Our final precision metric is a weighted sum of log- and trace-based precision.

Definition 7 (Precision). *Let (L, ϕ) be an event log and N a model. We define anti-alignment based precision as follows:*

$$P(N, L) = \alpha P_t(N, L) + (1 - \alpha) P_l^n(N, L)$$

This definition is parameterized by α and n. In the remainder of the paper, we choose $\alpha = 0.5$ and $n = 2 \cdot \max\limits_{\sigma \in L} |\sigma|$.

Our precision metric has two parameters, α, indicating the relative importance of the trace-based vs. the log-based part and n indicating the maximum length of the log-based anti-alignment. In this paper, we use $\alpha = 0.5$ and n equal to twice the length of the longest observed trace. Allowing for longer anti-alignments could lower the log-based precision if there are loops in the model (in the limit, log-based precision in a model with loops goes to 0). Striking the right balance between α and n in the context of real-life process discovery is beyond the scope of this paper. Instead, we focus on the qualitative aspects of our metrics more than the quantitative ones.

Example 5. Let's consider the Petri net shown in Figure 1 again, with the log of Table 1. Earlier, we identified the trace $\langle A, C, G, H, D, F, I \rangle$ as a $(7, \frac{1}{7})$ anti-alignment. Furthermore, when leaving one trace out, we get the following anti-alignments[5]:

σ	$\gamma_{\lvert\sigma\rvert}^{d,mx}(N, L \setminus \{\sigma\})$	γ projected	$d(\gamma, \sigma)$
$\langle A, B, D, E, I \rangle$	$\langle A, B, D, E, I \rangle$	$\langle A, B, D, E, I \rangle$	0
$\langle A, C, D, G, H, F, I \rangle$	$\langle A, C, G, H, D, F, I \rangle$	$\langle A, C, G, H, D, F, I \rangle$	$\frac{2}{7}$
$\langle A, C, G, D, H, F, I \rangle$	$\langle A, C, G, H, D, F, I \rangle$	$\langle A, C, G, H, D, F, I \rangle$	$\frac{2}{7}$
$\langle A, C, H, D, F, I \rangle$	$\langle A, C, \tau, H, D, F, I \rangle$	$\langle A, C, H, D, F, I \rangle$	0
$\langle A, C, D, H, F, I \rangle$	$\langle A, C, \tau, D, H, F, I \rangle$	$\langle A, C, D, H, F, I \rangle$	0

The trace-based precision $P_t(N, L) = \dfrac{1 + \frac{5}{7} + \frac{5}{7} + 1 + 1}{5} = \frac{31}{35} = 0.886$ and the log-based precision is $P_l(N, L) = 1 - \frac{1}{7} = \frac{6}{7} = 0.857$, hence overall precision with $\alpha = 0.5$ for this model and log is $P(N, L) = 0.5 \cdot \frac{31}{35} + 0.5 \cdot \frac{6}{7} = 0.871$.

Besides precision, we can also use anti-alignments for measuring generalization.

5 Measuring Generalization

In contrast to precision, which relates the log and the model, generalization relates the system to the log and the model. Generalization aims to estimate the extent to which unobserved, but likely possible behavior, is explained by the model. In terms of process modeling, generalization is often obtained by introducing parallel structures or loops into a model when the log suggests this to be the case. Unfortunately, we do not have any knowledge of the system other than that the log forms a representation of the most common behavior in it.

In order to quantify generalizations, we consider not only the sequential behavior that is actually allowed by the model, but we also quantify how different this behavior is

[5] Note that for the edit distance between the anti-alignment and the removed trace, the trace is first projected onto labeled elements, i.e. the τ transition is removed first.

when considering the state space of the model. (Structured) loops and parallel structures which are most commonly used to achieve generalization when modeling a system have the tendency to allow for many different sequential traces while introducing fewer states as for structured loops, the number of states does not increase with the number of executions of the loop, while for parallel transitions, the number of states 2^n grows slower than the number of sequences ($n!$). Therefore, in our generalization metric, we consider the notion of a recovery distance for an anti-alignment.

Definition 8 (Recovery distance). *Let (L, ϕ) be an event log and N a model. Let $\gamma = \gamma_n^{max}(N, L)$ be an anti-alignment of length n. Let $M_\gamma = \langle m_0, \ldots, m_n \rangle$ be the sequence of states visited by γ, i.e. m_0 is the initial marking of the model, m_n is the final marking of the model and for all $0 \leq i < n$ holds $m_i[\gamma_i\rangle m_{i+1}$. Let $S \subseteq [m_0\rangle = \{m \mid \exists\, \sigma \cdot \sigma' \in L \text{ s.t. } m_0[\sigma\rangle m\}$ be the set of states reached by L. The recovery distance is defined as:*

$$d_{rec}(\gamma) = \frac{1}{|\gamma| - 1} \cdot max_{m \in M_\gamma}\, min_{\sigma \in \Sigma^*, m[\sigma\rangle s \in S} |\sigma|$$

i.e. the recovery distance is the maximum distance between any of the states reached in the anti-alignment and the states visited by the log.

Note that in a process mining setting, we assume that there is a single reachable final marking and that the anti-alignment guarantees to reach this final marking. Hence the length of the firing sequence to reach a previously visited marking is bounded by the length of the anti-alignment minus 1. Using the recovery distance, we define a generalization metric in a similar fashion as we did for precision, i.e. we remove one trace from the log and compute an anti-alignment for which we obtain the minimum distance to the log and the maximum recovery distance.

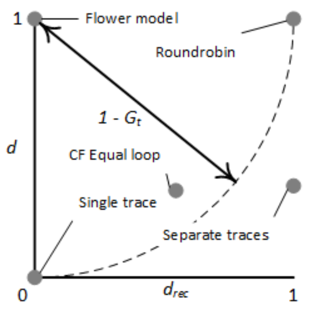

Fig. 9. Positioning of examples for trace-based generalization.

Figures 9 and 10 show the positioning of the models discussed earlier with respect to the anti-alignment distance and the recovery distance, both for the trace-based and log-based metric. Our generalization score is defined such that it favors only models that have a high anti-alignment distance and low recovery distance, i.e. models that introduce new traces without introducing new states. Recall that generalization typically occurs in structures that add fewer states than traces. If a model is properly generalizing, it is likely that the behavior observed in the log covers a significant part of the state space introduced by the generalizing structure, hence a previously unobserved trace will not introduce new states, but rather new paths between existing states, even if the introduced trace is completely different from anything observed in the log.

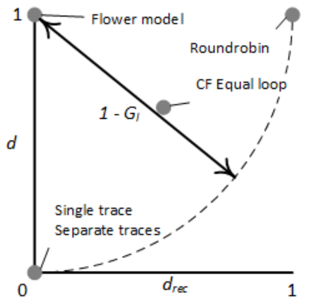

Fig. 10. Positioning of examples for log-based generalization.

Like for precision, we first consider trace-based generalization following the same leave-one-out procedure. This way, the model is guaranteed to contain an anti-alignment of some distance (i.e. the removed trace). Not using trace-based generalization would lead us to consider all models non-generalizing if the log equals the language of the model.

Definition 9 (Trace-based Generalization). *Let (L, ϕ) be an event log and N a model. We define the trace-based generalization metric for each trace. First, for every trace $\sigma \in L$, we define:*

$$G_t^{\sigma}(N, (L, \phi), \sigma) = 1 - ||1 - d(\gamma_\sigma, L \setminus \{\sigma\}), d_{rec}(\gamma_\sigma)||,$$

where $\gamma_\sigma = \gamma_{|\sigma|}^{d,mx}(N, L \setminus \{\sigma\})$ and $||a, b|| = min(1, \sqrt{a^2 + b^2})$, i.e. the Euclidean distance from $(0, 0)$, bound by 1.

Second, we define trace-based generalization as the weighted average:

$$G_t(N, (L, \phi)) = \frac{1}{\sum\limits_{\sigma \in L} \phi(\sigma)} \cdot \sum_{\sigma \in L} \phi(\sigma) \cdot G_t^{\sigma}(N, (L, \phi), \sigma).$$

Definition 9 uses the Euclidean distance from the perfectly generalizing model to compute a generalization score, where the perfectly generalizing model has maximally different anti-alignments without introducing new states, such as the model in Figure 3.

Similar to precision, we also define a log-based generalization metric which identifies an anti-alignment much longer than the longest trace in the log in order to detect if there is a part of the state space which can only be reached through longer traces.

Definition 10 (Log-based Generalization). *Let (L, ϕ) be an event log and N a model. Referring to Figure 10, we define log-based generalization as follows:*

$$G_l^n(N, (L, \phi)) = 1 - ||1 - d(\gamma, L), d_{rec}(\gamma)||,$$

where $\gamma = \gamma_n^{d,mx}(N, L)$ and n represents the maximal length of the anti-alignment, typically in the order of several times the length of the longest trace in the log. Again, we assume $||a, b|| = min(1, \sqrt{a^2 + b^2})$

Notice that both in the two previous definitions, the frequency of traces in the log is considered. Finally, combining the trace-based and the log-based generalization metric yields our final generalization metric:

Definition 11 (Generalization). *Let (L, ϕ) be an event log and N a model. We define anti-alignment based generalization as follows:*

$$G(N, (L, \phi)) = \alpha G_t(N, (L, \phi)) + (1 - \alpha) G_l^n(N, (L, \phi)).$$

This definition is parameterized by α and n. In the remainder of the paper, we choose $\alpha = 0.5$ and $n = 2 \cdot \max\limits_{\sigma \in L} |\sigma|$.

Example 6. Let's once again consider the Petri net shown in Figure 1, with the log of Table 1. Earlier, we identified the trace $\langle A, C, G, H, D, F, I\rangle$ as a $(7, \frac{1}{7})$ anti-alignment for the whole log and we measured precision to be $P(N, L) = 0.871$. The recovery distance for the trace $\langle A, C, G, H, D, F, I\rangle$ is 0 as it does not visit new states in the state space as this anti-alignment visits exactly the same set of states as the trace $\langle A, C, \tau, H, D, F, I\rangle$ which is in the log when correctly aligning the log to the model. When leaving one trace out, we got the following anti-alignments:

σ	freq.	$\gamma^{d,mx}_{\|\sigma\|}(N, L \setminus \{\sigma\})$	$d(\gamma, L \setminus \{\sigma\})$	$d_{rec}(\gamma)$
$\langle A, B, D, E, I\rangle$	1207	$\langle A, B, D, E, I\rangle$	$\frac{3}{6}$	$\frac{2}{4}$
$\langle A, C, D, G, H, F, I\rangle$	145	$\langle A, C, G, H, D, F, I\rangle$	$\frac{1}{7}$	0
$\langle A, C, G, D, H, F, I\rangle$	56	$\langle A, C, G, H, D, F, I\rangle$	$\frac{1}{7}$	0
$\langle A, C, H, D, F, I\rangle$	23	$\langle A, C, \tau, H, D, F, I\rangle$	$\frac{2}{6}$	$\frac{1}{6}$
$\langle A, C, D, H, F, I\rangle$	28	$\langle A, C, \tau, D, H, F, I\rangle$	$\frac{1}{6}$	0

The trace-based generalization $G_t(N, (L, \phi)) = (1207 \cdot (1 - \sqrt{\frac{9}{36} + \frac{4}{16}}) + 145 \cdot (1 - \sqrt{\frac{36}{49}}) + 56 \cdot (1 - \sqrt{\frac{36}{49}}) + 23 \cdot (1 - \sqrt{\frac{16}{36} + \frac{1}{36}}) + 28 \cdot (1 - \sqrt{\frac{25}{36}}))/1459 = 0.270$. The log-based precision is $G_l(N, (L, \phi)) = 1 - \sqrt{\frac{36}{49}} = 0.143$, hence overall generalization with $\alpha = 0.5$ for this model and log is $G(n, (L, \phi)) = 0.5 \cdot 0.270 + 0.5 \cdot 0.143 = 0.206$.

Consider again our example. The model presented in Figure 3 (the Flower model) clearly generalizes as it allows for very different traces (high anti-alignment distance), but all within the same state space (low recovery distance).

A model like Figure 4 (separate traces) does not generalize. If we consider the log as a whole, each anti-alignment will have distance 0 from the log and will have recovery distance 0. If we remove one trace from the log, the maximal anti-alignment found will be the removed trace, with some distance from the rest of the log, but with maximal recovery distance.

Now consider the model in Figure 11 (CF Equal loop). This model requires transitions C and F to fire equally often in order to reach the intended final marking of one token in the sink place. This model is similar to the original, but will show a high recovery distance as the number of executions of C and F determine the part of the state space which is visited by the anti-alignment, but likely not by the rest of the log.

The models in Figure 2 (Single trace) and Figure 12 (Round-robin) show examples of non-fitting models which also do not generalize. After making the log fit using alignment techniques [9], Figure 2 will have both minimal anti-alignment distance and minimal recovery distance (both 0), while the model in Figure 12 will have maximal anti-alignment distance and maximal recovery distance. Both models however are not generalizing.

6 Evaluation and Implementation

In this section, we first consider our example log of Table 1 and the models presented in Figures 1 through 8. Furthermore, we introduce two new models for our example log of Table 1, depicted in Figure 11 and Figure 12.

Table 2 shows the fitness, precision and generalization values for all models. For our precision and generalization metrics, we present both the trace-based values as well as the log-based values. The trace-based values are computed using the leave-one-out procedure presented earlier. The log-based values are computed by taking a maximal anti-alignment given the model and the entire log with maximum length equal to three times the length of the longest trace in the log.

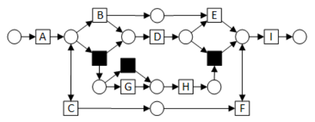

Fig. 11. A model where C and F are in a loop, but need to be executed equally often to reach the final marking.

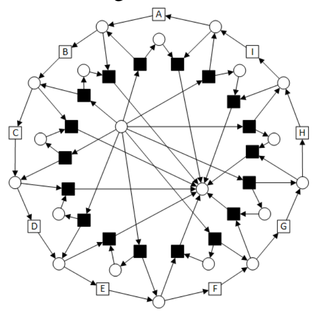

Fig. 12. Round-robin model. The outer loop can be started at any point and then exited one transition before completing the loop.

For the models that are not fitting (Figure 2, Figure 8 and Figure 12) the log is aligned to the model and then the aligned event log is used for computing precision and generalization. In case of Figure 2 this implies that all traces in the log are equal as the model only allows for one trace and therefore, the precision is always 1 and the generalization is always 0. Figure 12 is more interesting, as this model has both poor precision and poor generalization. No matter which trace is removed from the log, there is always an anti-alignment that does not look anything like the removed trace, hence precision is 0. Furthermore, as each of the starting points of the loops generates a completely distinct subgraph in the state space, the recovery distance for an anti-alignment is always very high and hence generalization is poor, despite the fact that the model allows for many different traces.

The model of Figure 1 has results as expected. The fitness is 1, and precision is fairly high. Furthermore, generalization is not so high as this model does not actually allow for much more behavior than observed. In fact, only the trace $\langle A, C, G, H, D, F, I \rangle$ is possible in the model, but not observed in the log. Figures 2, 3 and 4 indeed show extreme values for precision and/or generalization. As expected the self-loop model of Figure 3 has precision 0 as it allows for many different traces, but since the recovery distance is always 0 the generalization is maximal. Figure 4 is the opposite as it does not allow for any trace not in the log, and has a maximal recovery distance.

Now consider the models in Figure 11 and Figure 12. For Figure 11 we consider the *relaxed-sound semantics* of this model as it was translated from a causal net as introduced in [12]. The model is constructed in such a way that transitions C and F can be executed multiple times, but equally often. This model should be considered fairly imprecise as there is a lot of behavior in the model that is not in the log. However, the automaton-based metrics for precision are unable to capture this long-term dependency and they will penalize for the fact that C *can* be executed multiple times, but not for the fact that F may *have to* be executed multiple times.

Figure 12 is a model that allows for a loop over transitions A through I to be started at any point. However, when starting the loop at a given point, the model needs to terminate after executing $8 + n \times 9$ transitions. This model again should be considered imprecise as the language of the model is very different from the language of the (aligned) log. Only the P_{ET} captures this, the others consider this model very precise.

Some differences stand out between existing metrics and our anti-alignment based metrics. Consider for example the model in Figure 12. This model has minimal precision as it allows for much more behavior than observed in the log. However, both the P_{ET} and the P_a metric are unable to capture this since these metrics only consider behavior *directly* adjacent to the observed behavior with respect to a specific abstraction. Interestingly, the P_{ETC} metric considers the model of Figure 4 to be imprecise, while this model allows for exactly the observed behavior and nothing more. Again, the chosen abstraction causes this effect.

Due to the nice monotonicity property of anti-alignments shown in [7], our precision metric is the only one that consistently ranks models in such a way that a model with more possible traces (of a given maximal length) is always considered less precise.

When comparing our generalization metric with the existing ones, we see a big difference in the model of Figure 2. The behavior of this model consists of a single trace and is considered generalizing by the metric G_a since the aligned event log (the event log where non-fitting traces have been adapted to fit the model) shows great evidence of this model being the correct one for that log. In our metric however, a model that allows for only one trace will always be considered to have minimal generalization.

Interestingly, the model in Figure 8 is considered more generalizing by our metric than by most existing ones. This is due to the fact that we consider the recovery distance as important. This model allows for more behavior than observed, but does not introduce too many new states, i.e. the recovery distance is low while the distance of the anti-alignment to the log is large. This is what we consider to be generalization. The G_{ne} metric finds this model to be almost perfectly generalizing since any label is allowed to appear at almost any position, but this metric fails to recognize that labels can only appear once in each trace.

Again, consider Figure 11 and Figure 12. In both cases, the various parts of the language of the models are represented by completely separated parts of the state space. In Figure 11, the number of tokens in the place between C and F determines which part of the state space the middle part is executed, and in Figure 12, the initial decision where to start the loop does. In both cases, once a particular part of the state space is reached which is not covered by traces observed in the log, the recovery distance is maximal, i.e. only after emptying the place between C and F in Figure 11 or terminating the model in case of Figure 12, a state is reached which is covered by the observed log. Therefore, these models should not be considered generalizing.

6.1 Models found in literature

Rather than only considering our example models, we used models found in [6] for further comparison with our approach. In [6], several process mining results are presented to illustrate the importance of fitness, precision, generalization and simplicity in process mining. The paper introduces a precision and a generalization metric which are specific for process trees, or block-structured process models. The precision metric is comparable to the P_a metric used earlier. The generalization metric however focuses on the frequency with which each transition is executed in relation to the number of transitions in the model. Generalization is considered low if some parts of the model are infrequent in the log.

We compared our generalization and precision metrics with these models and there are some interesting observations. One of the models, depicted in Figure 13 contains an inclusive OR block of three activities B, C and D, implying that the model allows for 15 different traces. The log used contains 10 different traces in which B, C and D are executed in parallel, but D can be skipped and in [6] the model with the OR block is considered to be the fairly precise (precision 0.830). Our precision metric however identifies anti-alignments that have maximum distance of 0.5 from the removed trace or the log and therefore, our precision metric yields 0.477, which is what you would expect from a model that contains a large OR block to explain (almost) parallel behavior.

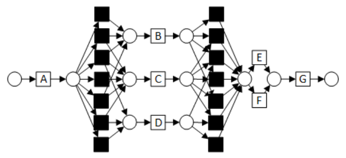

Fig. 13. Figure 9 from [6]

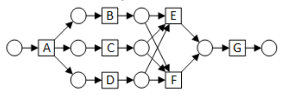

Fig. 14. Figure 5 from [6]

Another interesting model is model in Figure 14 which removed the option to skip D. This model has only one trace that is not observed in the original log and this trace, when executed, does not visit any new states compared to the rest of the log. Therefore, both trace- and log-based generalization are considered low and hence our generalization metric is 0.172 while the metric used in the paper reports a generalization of 0.889. A low generalization value is in line with the intuition behind generalization, i.e. the ability of the model to predict possible but unobserved behavior. As almost all behavior of this model has been observed before, generalization should not be high.

6.2 Implementation

The authors of [7] have shown that the problem of finding a (n, δ)-anti-alignment w.r.t. Hamming-distance is NP-complete. They have presented a way to convert this problem into a SAT problem and implemented an efficient tool in OCaml, available at `http://www.lsv.ens-cachan.fr/~chatain/darksider/`).

But, as discussed in Section 3.2, concerning the application to process mining, Hamming distance is usually too rigid. This is why, for the examples in this paper, we have chosen Levenshtein's edit distance, in spite of the higher complexity of finding anti-alignments for this distance. We have used a brute-force, depth-first search algorithm to find the anti-alignments. Since the maximum length of the anti-alignment is bounded, the size of the search space is finite which allows us to use a brute-force approach. This approach is implemented in the ProM package "anti-alignments" which can be installed through the ProM package manager available on `http://www.promtools.org/`. The package is included in the nightly build and in ProM 6.6.

In future work, we plan to improve the efficiency of the presented approach using heuristic implementations. Furthermore, we plan to integrate more distance metrics.

7 Conclusions

In this paper, we presented new metrics for measuring precision and generalization of a process model with respect to an event log. Both metrics rely on the concept of an anti-alignment, which is a trace of the model which is as different as possible from the event log given a certain distance function. The anti-alignments are applied using a cross-validation strategy to obtain measurements fro precision. Furthermore, we introduce the notion of recovery distance which is included in the generalization metric, basically expressing the ability of the model to recover from any deviation.

We have compared both metrics with the state-of-the-art metrics for precision and generalization on well-known examples, and the results clearly position the proposal of this paper as a significant improvement in terms of the quality of the estimations provided, albeit at the expense of a higher computational complexity.

Acknowledgments. This work has been partially supported by funds from the Spanish Ministry for Economy and Competitiveness (MINECO), the European Union (FEDER funds) under grant COMMAS (ref. TIN2013-46181-C2-1-R).

References

1. van der Aalst, W.M.P.: Process Mining - Discovery, Conformance and Enhancement of Business Processes. Springer (2011)
2. Rozinat, A., van der Aalst, W.M.P.: Conformance checking of processes based on monitoring real behavior. Inf. Syst. **33**(1) (2008) 64–95
3. Adriansyah, A.: Aligning observed and modeled behavior. PhD thesis, Eindhoven (2014)
4. Munoz-Gama, J.: Conformance Checking and Diagnosis in Process Mining. PhD thesis, Universitat Politecnica de Catalunya (2014)
5. vanden Broucke, S.K.L.M., Weerdt, J.D., Vanthienen, J., Baesens, B.: Determining process model precision and generalization with weighted artificial negative events. IEEE Trans. Knowl. Data Eng. **26**(8) (2014) 1877–1889
6. Buijs, J.C.A.M., van Dongen, B.F., van der Aalst, W.M.P.: Quality dimensions in process discovery: The importance of fitness, precision, generalization and simplicity. Int. J. Cooperative Inf. Syst. **23**(1) (2014)
7. Chatain, T., Carmona, J.: Anti-Alignments in Conformance Checking – The Dark Side of Process Models. In: Proceedings of ICATPN'16. Volume 9698 of Lecture Notes in Computer Science., Springer (2016)
8. Rozinat, A.: Process mining : conformance and extension. PhD thesis (2010)
9. van der Aalst, W.M.P., Adriansyah, A., van Dongen, B.F.: Replaying history on process models for conformance checking and performance analysis. Wiley Interdisc. Rew.: Data Mining and Knowledge Discovery **2**(2) (2012) 182–192
10. Adriansyah, A., Munoz-Gama, J., Carmona, J., van Dongen, B.F., van der Aalst, W.M.P.: Measuring precision of modeled behavior. Inf. Syst. E-Business Management **13**(1) (2015) 37–67
11. Murata, T.: Petri nets: Properties, analysis and applications. Proceedings of the IEEE **77**(4) (April 1989) 541–574
12. van der Aalst, W.M.P., Adriansyah, A., van Dongen, B.F.: Causal nets: A modeling language tailored towards process discovery. In Katoen, J., König, B., eds.: CONCUR 2011. Proceedings. Volume 6901 of Lecture Notes in Computer Science., Springer (2011) 28–42