Translucent Precision: Exploiting Enabling Information To Evaluate The Quality Of Process Models*

Harry H. Beyel[0000−0002−6541−3848] and Wil M.P. van der Aalst[0000−0002−0955−6940]

Chair of Process and Data Science, RWTH Aachen University, Aachen, Germany
{beyel, wvdaalst}@pads.rwth-aachen.de

Abstract. An event log stores information about executed activities in a process. Conformance-checking techniques are used to measure the quality of a process model using an event log. Part of the investigated quality dimensions is *precision*. Precision puts the behavior of a log and a model in relation. There are event logs that also store information about *enabled* activities besides the actual executed activities. These event logs are called *translucent event logs*. A technique for measuring precision is escaping arcs. However, this technique does not consider information on enabled activities contained in a translucent event log. This paper provides a formal definition of how to compute a precision score by considering translucent information. We discuss our method using a translucent event log and four different models. Our translucent precision score conveys the underlying concept by considering more information.

Keywords: Process Mining · Conformance Checking · Precision.

1 Introduction

In each organization, processes play a vital role. The execution of processes may leave event data in information systems. Typically, an event consists of three attributes: a *case identifier*, an *activity*, and a *timestamp*. We call a collection of these data an *event log*. Such event logs are used in *process mining* [1]. Conformance checking, an area of process mining, consists of various quality dimensions, including *precision* [12]. Precision evaluates whether a process model allows for more behavior than captured in the event log. Suppose L describes the behavior in an event log, and M captures the behavior contained in a process model. In that case, we can define the general idea of precision as follows: [1]

$$precision = \frac{|L \cap M|}{|M|}$$

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

[1] The notion of "behavior" is left vague here. There is the challenge that the event log is a finite sample, but the model may describe infinitely many traces (due to loops).

Table 1: Example translucent event log.

Event	Case	Activity	Enabled Activities	Time-stamp
e_1	1	a	{a}	13:37:37
e_2	1	b	{b, c}	13:37:38
e_3	1	c	{c}	13:37:39
e_4	1	e	{d, e}	13:37:40
e_5	2	a	{a}	13:37:41
e_6	2	c	{b, c}	13:37:42
e_7	2	b	{b}	13:37:43
e_8	2	d	{d, e}	13:37:44
e_9	2	b	{b, c}	13:37:45
e_{10}	2	c	{c}	13:37:46
e_{11}	2	e	{d, e}	13:37:47
e_{12}	3	a	{a}	13:37:48
e_{13}	3	c	{b, c}	13:37:49
e_{14}	3	b	{b}	13:37:50
e_{15}	3	e	{d, e}	13:37:51

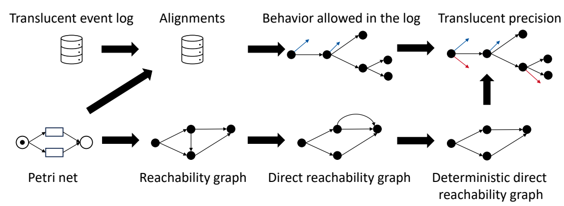


Fig. 1: Sketch of our approach to measure translucent precision.

Besides capturing only executed activities, an event can capture information on *enabled activities*. If an event log consists of such events, we call the event log a *translucent event log* [2]. A translucent event log can be, e.g., created when tasks performed in a desktop environment are captured to create training data for software bots [9]. An example of a translucent event log is shown in Table 1. When measuring precision between a translucent event log and a process model, it is vital to consider the information on enabled activities. In this paper, we present the first approach to a precision method that considers the information captured in translucent event logs and fits the intuitive meaning of precision. Our approach is based on escaping arcs [19]. An overview is depicted in Figure 1.

2 Related Work

A technique for measuring precision is based on escaping arcs [3, 7, 8, 19]. Given an event log, a prefix automaton is built. Traces are replayed on the model, and it is checked whether the model allows for more behavior than in the automaton. Another approach is based on anti-alignments [15]. The previous approach does not capture model deviation if it is not directly involved in the replay. This approach aims to solve this issue. An anti-alignment is an execution sequence in a given model that significantly differs from all traces in the log [13, 14]. Another approach relies on negative events [16]. Negative events are sets of events that were prohibited from taking place. Such events are induced for each position in the event log. [20] introduces behavioral precision. [10, 11] refine the approach. There exist stochastic-aware precision measures [18] and approaches for object-centric process mining [5]. However, none of the presented approaches uses information on enabled activities provided in the event log.

3 Preliminaries

Definition 1 (Sets, Powersets, Multisets, Sequences). *Given a set X and a function f, $f(X) = \{f(x) \mid x \in X\}$ denotes applying the function f on all elements of set X. For sets X and Y, $X \times Y = \{(x, y) \mid x \in X, y \in Y\}$ denotes the cartesian product. The powerset of a set X is denoted as $\mathcal{P}(X) = \{X' \mid X' \subseteq X\}$. $\mathcal{B}(X)$ denotes the set of all multisets over set X. E.g., if $X = \{x, y, z\}$, a possible bag is $[x, x, y] = [x^2, y]$. Given a set X, a sequence $\sigma = \langle \sigma_1, \dots, \sigma_n \rangle$, $\sigma \in X^*$, denotes a sequence over X. σ_i denotes the sequence's i-th element. The length of a sequence σ is denoted as $|\sigma|$. Given a sequence $\sigma = \langle \sigma_1, \dots, \sigma_{|\sigma|} \rangle$ and a function f, $f(\sigma) = \langle f(\sigma_1), \dots, f(\sigma_{|\sigma|}) \rangle$. $pref_i(\sigma) = \langle \sigma_1, \dots, \sigma_i \rangle$ refers to the prefix of a sequence containing the first i elements. $pref_0 = \langle \rangle$. Given a sequence σ and a set X', $\sigma\restriction_{X'}$ denotes a sequence projections, e.g., $\langle a, b, c, d \rangle\restriction_{\{a,c\}} = \langle a, c \rangle$.*

Translucent event logs capture information on enabled activities in addition to executed ones. Hence, the executed activity must also be enabled in the corresponding event. Also, we assume that all enabled activities in an event log are performed at some point. $\mathcal{U}_{case}$ is the universe of case identifiers, $\mathcal{U}_{act}$ is the universe of activity names, and $\mathcal{U}_{time}$ is the universe of timestamps.

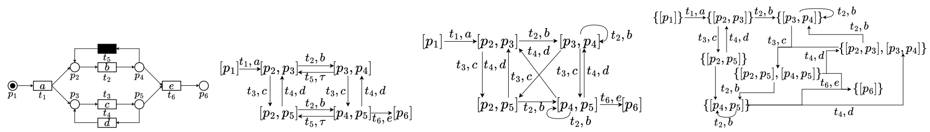
Definition 2 (Translucent Event Log, Trace). *$\mathcal{U}_{ev}$ is the universe of events. $e \in \mathcal{U}_{ev}$ is an event, $\pi_{case}(e) \in \mathcal{U}_{case}$ is the case of e, $\pi_{time}(e) \in \mathcal{U}_{time}$ is the time of e, $\pi_{en}(e) \subseteq \mathcal{U}_{act}$ are the enabled activities of e, $\pi_{act}(e) \in \pi_{en}(e)$ is the activity of e. In addition, $\bigcup_{e \in L} \pi_{en}(e) = \bigcup_{e \in L} \{\pi_{act}(e)\}$. A translucent event log L is a set of events $L \subseteq \mathcal{U}_{ev}$. For simplicity, we assume that events in L are totally ordered s.t. for $e_1, e_2 \in L$, $e_1 < e_2$ implies $\pi_{time}(e_1) \leq \pi_{time}(e_2)$. A trace is a sequence of all events of a case ordered from earliest to latest, i.e., $\sigma^{L,c} = \langle e_1, \dots, e_n \rangle$, s.t. for $c \in \pi_{case}(L)$, $\{e_1, \dots, e_n\} = \{e \in L \mid \pi_{case}(e) = c\}$ and $e_1 < ... < e_n$. The set of traces of L is denoted as $\Sigma^L = \{\sigma^{L,c} \mid c \in \pi_{case}(L)\}$.*

For the example translucent event log shown in Table 1, $\pi_{case}(e_2) = 1$, $\pi_{act}(e_2) = b$, $\pi_{en}(e_2) = \{b, c\}$, and $\pi_{time}(e_2) =$ 13:37:38, and $\Sigma^L = \{\langle e_1, e_2, e_3, e_4 \rangle, \dots\}$.

Definition 3 (Marked Labeled Petri Net). *A labeled Petri net is a tuple $N = (P, T, F, A, l)$, where P is a set of places, T is a set of transitions s.t. $P \cap T = \emptyset$, and $F \subseteq (T \times P) \cup (P \times T)$ is a set of directed arcs. $A \subseteq \mathcal{U}_{act} \cup \{\tau\}$ is a set of activity labels, and $l : T \to A$ is a labeling function where τ denotes the activity of silent transitions. A marking $M \in \mathcal{B}(P)$ is a multiset of places. We write (N, M) to refer to the Petri net N in marking M.*

We focus on sound workflow nets [4]. Petri net firing rules can be found in [1].

Definition 4 (Firing Sequence). *Let $N = (P, T, F, A, l)$ be a Petri net. The successive firing of all transitions in $\sigma \in T^*$ is denoted as $(N, M_1) \xrightarrow{\sigma} (N, M_{n+1})$.*



(a) Petri net. (b) RG. (c) DRG. (d) DDRG.

Fig. 2: Structures based on the event log shown in Table 1.

Let $M_{in} \in \mathcal{B}(P)$ be the initial marking. We define a marking $M' \in \mathcal{B}(P)$ as reachable in (N, M_{in}), if there exists $\sigma \in T^*$, s.t. $(N, M_{in}) \xrightarrow{\sigma} (N, M')$. The set of all reachable markings starting in (N, M_{in}) is denoted as $(N, M_{in}\rangle$.

In the remainder of our work, we assume the existence of alignments between a Petri net and a log [6]. For this work, we assume that our log fits perfectly. For our approach, we need to remove silent transitions from alignments.

Definition 5 (Alignment). *Let $N = (P, T, F, A, l)$ be a Petri net and $L \subseteq \mathcal{U}_{ev}$ be a translucent event log. $T_\tau = \{t \in T \mid l(t) = \tau\}$ denotes the set of silent transitions. For a trace $\sigma \in \Sigma^L$, its perfectly fitting alignment on a Petri net N is denoted as $path(\sigma, N) \in T^*$. We link non-silent transitions to the corresponding events, i.e., $\pi^N_{trans}(\sigma_i) = (path(\sigma, N)\upharpoonright_{T \setminus T_\tau})_i$, for all $i \in \{1, \cdots, |\sigma|\}$.*

To access the behavior of a Petri net, we use reachability graphs. Each node in such a graph is a marking of a Petri net. Nodes are connected if a transition exists, s.t. firing the transition leads from one marking to the other.

Definition 6 (Reachability Graph). *Let $N = (P, T', F, A, l)$ be a labeled Petri net, with the inital marking $M_{in} \in \mathcal{B}(P)$. The reachability graph (RG) of the Petri net N is defined as $RG^{N, M_{in}} = (S, E)$ with $S = (N, M_{in}\rangle$ and $E = \{(M, t, M') \in S \times T \times S \mid \exists_{t \in T}\ (N, M) \xrightarrow{t} (N, M')\}$.*

The RG of the Petri net shown in Figure 2a is displayed in Figure 2b.

4 Translucent Precision

4.1 Log Behavior

We define events' prefixes using transitions based on alignments to capture the executed and enabled behavior.

Definition 7 (Executed and Enabled Behavior). *Let $L \subseteq \mathcal{U}_{ev}$ be a translucent event log, $N = (P, T, F, A, l)$ be a Petri net, and $T_\tau = \{t \in T \mid l(t) = \tau\}$ be the set of silent transitions. For a trace $\sigma \in \Sigma^L$, we define $\pi^N_{pref}(\sigma_i) = \pi^N_{trans}(pref_{i-1}(\sigma))$. The executed behavior for $e \in L$ at some point is defined as: $\pi^N_{prefact}(e) = \{\pi_{act}(e') \mid e' \in L\ \wedge \pi^N_{pref}(e) = \pi^N_{pref}(e')\}$ Similarly, we define the enabled behavior as: $\pi^N_{prefen}(e) = \bigcup_{\substack{e' \in L \\ \pi^N_{pref}(e) = \pi^N_{pref}(e')}} \pi_{en}(e')$.*

Given our example log shown in Table 1 and the Petri net depicted in Figure 2a, $\pi^N_{prefact}(e_1) = \{a\}$, $\pi^N_{prefact}(e_2) = \{b, c\}$, $\pi^N_{prefact}(e_4) = \{e\}$, $\pi^N_{prefen}(e_4) = \{d, e\}$.

4.2 Model Behavior

When capturing model behavior, silent transitions provide a special challenge since their execution is not captured in the log. Moreover, executing them at a different point in time is often possible. As a result, we want to check if their execution enables other transitions, respectively, activities. Such activities could be captured as translucent activities. To do so, we first introduce τ-sequences. A τ-sequences is a sequence of transitions s.t. all, but the last transition is a silent transition. Given the Petri net depicted in Figure 2a, possible τ-sequences are $\langle t_5, t_2\rangle$, $\langle t_5, t_3\rangle$, and $\langle t_5, t_4\rangle$. By using τ-sequences, we can transform an RG into a *direct RG*. In this process, we remove τ-transitions from the RG and establish connections between the start and end of these sequences.

Definition 8 (Direct RG). *Let $N = (P, T, F, A, l)$ be a Petri net, with its initial marking $M_{in} \in \mathcal{B}(P)$, and let Σ_τ^N be its set of τ-sequences. The Direct RG (DRG) of N is $DRG^{N,M_{in}} = (S, E)$ with $S = (N, M_{in}\rangle$ being the set of reachable markings and $E = E' \cup E_\tau$ s.t. $E' = \{(M, t, M') \in S \times T \times S \mid \exists_{t \in T \setminus T_\tau}\ (N, M) \xrightarrow{t} (N, M')\}$ and $E_\tau = \{(M, \sigma_{|\sigma|}, M')S \times T \times S \mid \exists_{\sigma \in \Sigma_\tau^N}\ (N, M) \xrightarrow{\sigma} (N, M')\}$.*

Figure 2c shows the DRG based on the previously shown RG (see Figure 2b). Following the transition sequence $\langle t_1, t_2, t_3\rangle$ results in two markings: $[p_2, p_5]$ and $[p_4, p_5]$. Hence, following a transition sequence in the graph is not deterministic. To solve this problem, we simplify the graph using automata theory [17].

Definition 9 (Deterministic DRG). *Let N be a Petri net, with its initial marking $M_{in} \in \mathcal{B}(P)$, and $DRG^{N,M_{in}} = (S, E)$ be a DRG. The Deterministic DRG (DDRG) is a DRG, $DDRG^{N,M_{in}} = (S', E')$ s.t. $S' = \mathcal{P}(S)$ and $E' = \{(S_1, t, S_2) \in S' \times T \times S' \mid \exists_{t \in T}\ S_2 = \bigcup_{s_1 \in S_1} \{s_2 \mid (s_1, t, s_2) \in E\}\}$.*

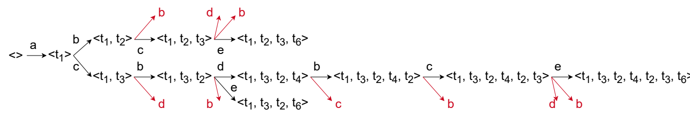
The DDRG of the DRG depicted in Figure 2c is shown in Figure 2d. After making the replay deterministic, we want to access the enabled activities in a DDRG and, therefore, in the model. We use the states of the DDRG to do so.

Definition 10 (Enabled Activities in Model). *Let $N = (P, T, F, A, l)$ be a Petri net, with its initial marking $M_{in} \in \mathcal{B}(P)$, and $DDRG^{N,M_{in}} = (S, E)$ be the corresponding DDRG. For $s, s' \in S, t \in T$, if there exists an edge $(s, t, s') \in E$, we denote this with $s \xrightarrow{t} s'$. For $\sigma \in T^*$ and states $s_1, \ldots, s_{|\sigma|+1} \in S$, we denote $s_1 \xrightarrow{\sigma} (s_{|\sigma|+1})$, if $\forall_{1 \leq i \leq |\sigma|}\ s_i \xrightarrow{\sigma_i} s_{i+1}$. Let $L \subseteq \mathcal{U}_{ev}$ be a translucent event log. For $e \in L$ and $s \in S$ we define $\pi_{state}^{DDRG^{N,M_{in}}}(e) = s$ s.t. $\{M_{in}\} \xrightarrow{\pi_{pref}^N(e)} s$. Thus, $\pi_{modelen}^{DDRG^{N,M_{in}}}(e) = \{l(t) \mid \exists_{t \in T, s \in S}\ \pi_{state}^{DDRG^{N,M_{in}}}(e) \xrightarrow{t} s\}$.*

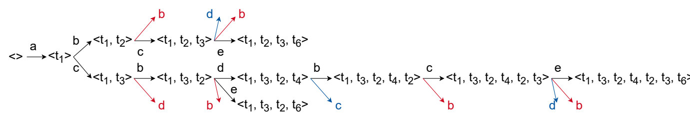
Given the example DDRG provided in Figure 2d, $\pi_{modelen}^{DDRG^{N,M_{in}}}(e_1) = \{a\}$, $\pi_{modelen}^{DDRG^{N,M_{in}}}(e_2) = \{b, c\}$, $\pi_{modelen}^{DDRG^{N,M_{in}}}(e_3) = \{b, c\}$.

4.3 Computing Precision Scores

We first define a precision score, similar to escaping arcs, which does not consider enabled activities in the provided event log. Then, a score considering them.



(a) Visualization of the computation of the precision score.



(b) Visualization of computing the translucent precision score.

Fig. 3: Visualization of the computation of the precision scores. Black arcs represent executed activities in the log, blue arcs represent enabled activities in the log, and red arcs represent activities enabled in the model but not in the log.

Definition 11 (Precision Score). *Given a translucent event log $L \subseteq \mathcal{U}_{ev}$, a Petri net N with initial marking M_{in} and $DDRG^{N,M_{in}}$, we define precision as:*

$$prec(L,N) = \frac{1}{|L|} \cdot \sum_{e \in L} \frac{|\pi^{N}_{prefact}(e)|}{|\pi^{DDRG^{N,M_{in}}}_{modelen}(e)|}$$

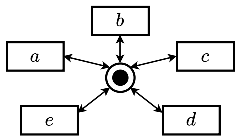
Definition 12 (Translucent Precision Score). *Given a translucent event log $L \subseteq \mathcal{U}_{ev}$, a Petri net N with its initial marking M_{in} and its $DDRG^{N,M_{in}}$, we define the translucent precision score as follows:*

$$prec_t(L,N) = \frac{1}{|L|} \cdot \sum_{e \in L} \frac{|\pi^{N}_{prefen}(e) \cap \pi^{DDRG^{N,M_{in}}}_{modelen}(e)|}{|\pi^{DDRG^{N,M_{in}}}_{modelen}(e)|}$$

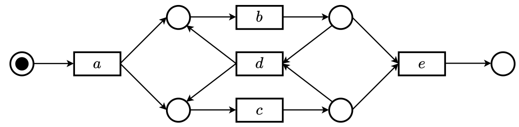
Note that we have to limit the numerator because activities that are not allowed in the model could be enabled. Illustrations for the methods are shown in Figure 3. For our running example, we denote a precision score of 0.7 and a translucent precision score of roughly 0.78.

5 Evaluation

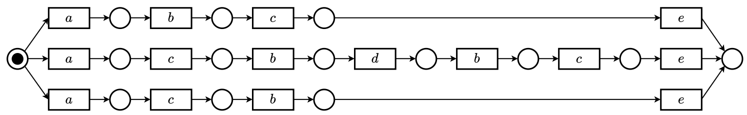
To evaluate our approach, we use the translucent event log shown in Table 1. Furthermore, we use the running example Petri net (Figure 2a) and three additional Petri nets (Figure 4) to evaluate whether the computed scores fit the meaning of precision. The results of our precision scores are displayed in Table 2. Both scores are low for the flower model. This shows that an imprecise model stays imprecise even when enabled activities are considered. The precision score from the example model suffers from b always being enabled after executing a and before executing e, thus allowing more behavior than caught in



(a) Flower model.



(b) Appropiate Petri net. Discovered by applying the discovery algorithm presented in [2].



(c) Trace-separate model.

Fig. 4: Additional Petri nets.

Table 2: Precision and translucent precision scores for different models.

Model	Precision Score	Translucent Precision Score
Flower Model (Figure 4a)	0.22	0.26
Example Model (Figure 2a)	0.70	0.78
Appropriate Model (Figure 4b)	0.90	1.00
Trace-separate Model (Figure 4c)	1.00	1.00

the log. Concerning the appropriate model, the traditional precision score suffers from the parallelism between b and c, and the choice between d and e. The translucent precision score considers enabled activities in the log, thus penalizing the afore-described behavior less. For the trace-separate model, we observe that both measurements yield a score of 1.0. In summary, the translucent precision method has the intuitive meaning of the traditional method, and considering enabled activities boosts the score for models that consider this information.

6 Conclusion

This paper presents the first notation and computational method of precision using translucent event logs. We showed that a well-established method can be extended to consider information on enabled activities. Also, we showed that our method still follows the natural understanding of precision by penalizing imprecise models. Furthermore, we showed that considering information on enabled activities is a valuable addition since process models that consider this knowledge get less penalized. Also, our method can handle duplicated transitions.

The approach we presented focuses on a fitting translucent event log. Hence, extending the approach to consider unfitting traces is valuable. Multiple methods exist for determining precision. Extending these techniques to consider information on enabled activities seems convenient. Moreover, methods for the other quality dimensions that consider translucent information should be introduced. When considering the different areas of process mining, techniques that consider the valuable information on enabled activities are needed.

References

1. van der Aalst, W.M.P.: Process Mining - Data Science in Action, Second Edition (2016)
2. van der Aalst, W.M.P.: Lucent process models and translucent event logs. Fundam. Informaticae **169**(1-2), 151–177 (2019)
3. van der Aalst, W.M.P., Adriansyah, A., van Dongen, B.F.: Replaying history on process models for conformance checking and performance analysis. WIREs Data Mining Knowl. Discov. **2**(2), 182–192 (2012)
4. van der Aalst, W.M.P., Stahl, C.: Modeling Business Processes - A Petri Net-Oriented Approach (2011)
5. Adams, J.N., van der Aalst, W.M.P.: Precision and fitness in object-centric process mining. In: ICPM (2021)
6. Adriansyah, A., van Dongen, B.F., van der Aalst, W.M.P.: Conformance checking using cost-based fitness analysis. In: EDOC. pp. 55–64 (2011)
7. Adriansyah, A., et al.: Alignment based precision checking. In: BPM Workshops (2012)
8. Adriansyah, A., et al.: Measuring precision of modeled behavior. Inf. Syst. E Bus. Manag. **13**(1), 37–67 (2015)
9. Beyel, H.H., van der Aalst, W.M.P.: Creating translucent event logs to improve process discovery. In: ICPM Workshops (2022)
10. vanden Broucke, S.K.L.M., et al.: Improved artificial negative event generation to enhance process event logs. In: CAiSE (2012)
11. vanden Broucke, S.K.L.M., et al.: Determining process model precision and generalization with weighted artificial negative events. IEEE **26**(8), 1877–1889 (2014)
12. Carmona, J., et al.: Conformance Checking - Relating Processes and Models (2018)
13. Chatain, T., Boltenhagen, M., Carmona, J.: Anti-alignments - measuring the precision of process models and event logs. Inf. Syst. **98**, 101708 (2021)
14. Chatain, T., Carmona, J.: Anti-alignments in conformance checking - the dark side of process models. In: PETRI NETS (2016)
15. van Dongen, B.F., Carmona, J., Chatain, T.: A unified approach for measuring precision and generalization based on anti-alignments. In: BPM (2016)
16. Goedertier, S., et al.: Robust process discovery with artificial negative events. J. Mach. Learn. Res. **10**, 1305–1340 (2009)
17. Hopcroft, J.E., Motwani, R., Ullman, J.D.: Introduction to automata theory, languages, and computation, 2nd Edition (2001)
18. Leemans, S.J.J., Polyvyanyy, A.: Stochastic-aware precision and recall measures for conformance checking in process mining. Inf. Syst. **115**, 102197 (2023)
19. Munoz-Gama, J., Carmona, J.: A fresh look at precision in process conformance. In: BPM (2010)
20. Weerdt, J.D., et al.: A robust f-measure for evaluating discovered process models. In: CIDM (2011)