A Visual Approach to Spot Statistically-Significant Differences in Event Logs Based on Process Metrics

Alfredo Bolt, Massimiliano de Leoni, and Wil M. P. van der Aalst

Eindhoven University of Technology, Eindhoven, The Netherlands
{a.bolt,m.d.leoni,w.m.p.v.d.aalst}@tue.nl

Abstract. This paper addresses the problem of comparing different variants of the same process. We aim to detect relevant differences between processes based on what was recorded in event logs. We use transition systems to model behavior and to highlight differences. Transition systems are annotated with measurements, used to compare the behavior in the variants. The results are visualized as transitions systems, which are colored to pinpoint the significant differences. The approach has been implemented in ProM, and the implementation is publicly available. We validated our approach by performing experiments using real-life event data. The results show how our technique is able to detect relevant differences undetected by previous approaches while it avoids detecting insignificant differences.

Keywords: Process Variants Comparison, Annotated Transition System, Statistical Significance, Process Mining

1 Introduction

Process mining is a relatively young research discipline that aims at discovering, monitoring and improving real processes by extracting knowledge from the behavior as recorded in the event logs readily available in today's systems [1]. The field of process mining puts forward techniques for discovering process models from event logs, for checking the conformance of normative models against the behavior observed in the event logs and analyzing bottlenecks and other *Key Performance Indicators* (KPIs).

Traditional process-mining techniques typically rely on the assumption that, within any organization, all executions of a certain process are characterized by an homogenous behavior, which can be easily compared. This assumption is often not met in reality: several variants of the same process may exist even within the same organization. As an example, consider an organization, such as a bank, that is composed by dozens of geographically spread branches. The same process, e.g. the loan's management, can be executed differently in these branches. Even within a branch, the observed behavior can vary according to different criteria; for example, the behavior may change over time or depend on the amount involved.

The comparative analysis of different process variants is obviously relevant and through the availability of event data also possible. This paper presents a generic technique to compare process variants by identifying statistically significant differences.

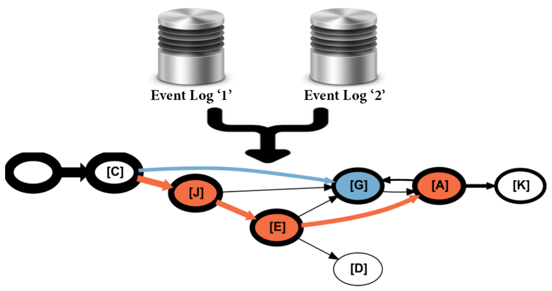


Fig. 1: Overview of the approach: two event logs are compared, producing a single annotated transition system, where the colors of nodes and edges represent the relevance of the differences found.

Figure 1 sketches the idea: two event logs are compared for differences that are projected onto a transition system where states and transitions are colored to highlight differences. The thickness of the node's borders and arcs indicates the frequencies with which states and transitions are respectively visited or occur. The portions of behavior that are rarely observed are filtered out. Also, differences are not highlighted if they are not statistically significant. The visual properties of these transition systems, and their meaning, are discussed in Sec. 3.

The two event logs that are used for comparison can have actually been extracted from different information systems, e.g. of two branches of the same company or of different companies. Alternatively, they can be extracted from a process cube [2,3] using the typical operations of, e.g., dicing, slicing and filtering. In the case that more than two event logs need to be compared, they can be grouped and merged into two event logs.

As detailed in Section 6, existing work mainly focuses on reporting differences for what concerns the control flow, meaning the frequency with which activities occur and the causal relations between activities (i.e., which activities are typically observed to follow given activities). However, differences can be regarded from other viewpoints based on other process metrics, such as the time between activities and the overall process performance. Our approach allows end users to use several process metrics for detecting such differences. Figure 1 shows an overview of the approach: two event logs are taken as input and an annotated transition system showing the differences is produced as output.

In order to assess the practical relevance of the differences highlighted by our technique, we used real-life event data extracted from the information system of an Italian local police, which records the executions of the process of handling road-traffic fines. In particular, we show how the management of high fines varies from that of low fines, including differences in the behaviors of offenders in paying the fines.

The remainder of this paper is structured as follows. Section 2 introduces the basic concepts that are used throughout the papers, whereas Section 3 details our technique for comparing the behaviors observed in two event logs. Section 4 describes the software tool that implements this approach, whereas Section 5 presents the evaluation discussed above. Section 6 discusses related work; in particular, using the same dataset of an Italian local police, we illustrate how existing approaches highlight insignificant differences instead of highlighting many of the relevant differences, which conversely, our approach can. Finally, Section 7 summarizes our contributions and discusses future work.

2 Transition Systems as a Process Representation

The behavior observed in an event log can be summarized as a *transition system* [4]. Section 2.1 introduces the formalisms used to represent event logs. Section 2.2 describes how transition systems are created. Sections 2.3 and 2.4 illustrate how measurements can be annotated into the states and transitions of a transition system.

2.1 Event Log

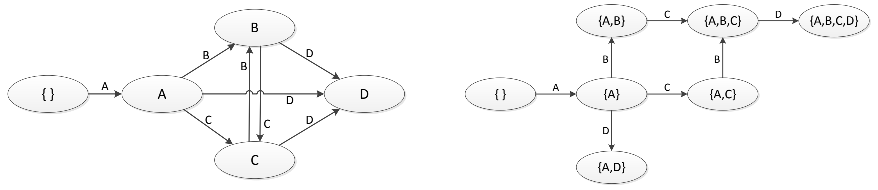
trace id	activity	timestamp	...
1	A	28-12-2015:06.30	...
1	B	28-12-2015:06.45	...
1	C	28-12-2015:07.20	...
1	D	28-12-2015:08.05	...
2	A	29-12-2015:10.10	...
2	C	29-12-2015:10.30	...
2	B	29-12-2015:11.15	...
2	D	29-12-2015:12.10	...
3	A	30-12-2015:09.30	...
3	D	30-12-2015:09.40	...

Table 1: A fragment of an event log represented as a table: each row corresponds to an event and each column corresponds to an event attribute. Events with the same *trace id* correspond to the same trace (i.e. process instance).

Let $\mathcal{E}$ be the universe of events. Events may have *attributes* (e.g., the person who executed it, associated cost, timestamp). Attribute values are related to events through the function $att_a \in \mathcal{E} \to \mathcal{V}$, where a is an attribute name and $\mathcal{V}$ is the set of possible attribute values. In this paper we do not impose a specific set of attributes. However, given the focus of this paper, we assume that each event has at least the following attributes: *activity name* and *timestamp* (denoted as $att_n(e)$ and $att_t(e)$ respectively).

Let $\sigma \in \mathcal{E}^*$ be a trace. A trace records the execution of an *instance* of a process and is a finite sequence of events. The k^{th} event of a trace is denoted as $\sigma(k)$. The length of a trace is denoted as $|\sigma|$. We assume that events in traces are ordered by timestamp i.e., $\forall \sigma \in \mathcal{E}^*, 1 \leq i < j \leq |\sigma| : att_t(\sigma(i)) \leq att_t(\sigma(j))$. The prefix of a trace containing its first k events is defined by the function $pref^k \subseteq \mathcal{E}^* \to \mathcal{E}^*$, with the special case $pref^0(\sigma) = \langle\rangle$. The set of all the prefixes of a trace σ is defined as $pref^\diamond(\sigma) = \bigcup_{k=0}^{|\sigma|}\{pref^k(\sigma)\}$. The postfix of a trace containing its last k events is defined by the function $postf^k \subseteq \mathcal{E}^* \to \mathcal{E}^*$.

Let $L \in \mathbb{B}(\mathcal{E}^*)$ be an event log. An event log is a multiset of traces. The set of all the prefixes of traces of an event log L is defined as $P_L = \bigcup_{\sigma \in L} pref^\diamond(\sigma)$. The set of all the events in an event log L is defined as $E_L = \bigcup_{\sigma \in L}\{e \in \sigma\}$. Table 1 shows an example of an event log represented as a table. This event log will be used as a running example through the remainder of this section.



(a) $r^s(\sigma) = att_n(\sigma(|\sigma|)$ (b) $r^s(\sigma) = \{att_n(e)|e \in \sigma\}$

Fig. 2: Examples of transition systems obtained from the event log L presented in Table 1 using different *state representation* functions $r^s(\sigma), \sigma \in P_L$. In both cases, the *activity representation* function used is $r^a(e) = att_n(e), e \in E_L$.

2.2 Transition Systems

Transition systems are composed of *states* and of *transitions* between them. A transition is defined by an activity being executed, triggering the current state to move from a *source* to a *target* state. Figure 2 shows two possible transition system representations of the event log presented in Table 1. The nodes indicate the *states* and the arcs indicate the *transitions* between them. Prefixes of traces can be mapped to states and transitions using representation functions that define how these prefixes are interpreted.

The *state representation* function is defined as $r^s \in \mathcal{E}^* \rightarrow \mathcal{R}^s$ where $\mathcal{E}^*$ is the universe of possible traces and $\mathcal{R}^s$ is the set of possible representations of states. This function relates (prefixes of) traces to states in a transition system.

The *activity representation* function is defined as $r^a \in \mathcal{E} \rightarrow \mathcal{R}^a$ where $\mathcal{E}$ is the set of possible events and $\mathcal{R}^a$ is the set of possible representations of activities (e.g. *activity name* or *event id*).

When using a state representation function r^s and an activity representation function r^a together, (prefixes of) traces can be related to transitions in a transition system, as the activity and the source and target states of the transition can be identified using r^s and r^a. The set of all possible representations of traces is defined as $\mathcal{R}^t \subseteq \mathcal{R}^s \times \mathcal{R}^a \times \mathcal{R}^s$. A transition $t \in \mathcal{R}^t$ is a triplet (s_1, a, s_2) where $s_1, s_2 \in \mathcal{R}^s$ are the source and target states and $a \in \mathcal{R}^a$ is the activity executed.

Figure 2.a shows the transition system that represents the event log L shown in Table 1 using the state representation function $r^s(\sigma) = att_n(\sigma(|\sigma|), \forall \sigma \in P_L$ and the activity representation function $r^a(e) = att_n(e), \forall e \in E_L$. In this transition system, (prefixes of) traces are mapped into states and transitions as the activity name of their last event.

Figure 2.b, shows a different representation of the same event log L. For this transition system the state representation function used is $r^s(\sigma) = \{att_n(e)|e \in \sigma\}, \sigma \in P_L$ and the activity representation function used is $r^a(e) = att_n(e), e \in E_L$. In this transition system, (prefixes of) traces are mapped into states as the set of activity names of all their events, and into transitions as the activity name of their last event.

Definition 1 (Transition System). *Let r^s be a state representation function, r^a an activity representation function and L an event log. A* transition system $TS^{(r^s,r^a,L)}$ *is defined as a triplet (S, A, T) where $S = \{s \in R^s | \exists_{\sigma \in P_L} s = r^s(\sigma)\}$ is the set of states, $A = \{a \in R^a | \exists_{e \in E_L} a = r^a(e)\}$ is the set of activities and $T = \{(s_1, a, s_2) \in S \times A \times S | \exists_{\sigma \in P_L, \sigma \neq \langle\rangle} s_1 = r^s(pref^{|\sigma|-1}(\sigma)) \land a = r^a(\sigma(|\sigma|)) \land s_2 = r^s(\sigma)\}$ is the set of valid transitions between states.*

Note that the structure of a transition system is affected by the state and activity representation functions used to create it. A thorough discussion on state and event representations in transition systems is presented in [4].

2.3 Measurements

In order to compare event logs, we need to introduce the measurements used for comparison. Measurement functions are computed as functions of event attributes contained in the events of a trace.

Given a state representation function r^s a *state measurement* function $sm_{r^s} \in \mathcal{E}^* \times \mathcal{R}^s \to \mathbb{B}(\mathbb{R})$, is a function that relates traces $\sigma \in \mathcal{E}^*$ and states $s \in \mathcal{R}^s$ to multisets of numerical measurements. For example, it is possible to measure whether or not a certain state s in a state representation r^s is reached during the process' execution recorded in a trace σ:

$$sm_{r^s}^{occur}(\sigma, s) = \begin{cases} [1] \text{ if } \exists \sigma' \in pref^{\diamond}(\sigma) : r^s(\sigma') = s \\ [0] \text{ otherwise} \end{cases} \tag{1}$$

It is also possible to measure the *elapsed time* between the beginning of a trace σ and the visit of a state s using a state representation r^s:

$$sm_{r^s}^{elapsed}(\sigma, s) = \biguplus_{\substack{\sigma' \in pref^{\diamond}(\sigma), \sigma' \neq \langle\rangle \\ r^s(\sigma') = s}} [att_t(\sigma'(|\sigma'|)) - att_t(\sigma'(1)] \tag{2}$$

Given a state representation function r^s and an activity representation r^a, a *transition measurement* function $tm_{(r^s,r^a)} \in \mathcal{E}^* \times \mathcal{R}^t \to \mathbb{B}(\mathbb{R})$, is a function that relates traces $\sigma \in \mathcal{E}^*$ and transitions $t \in \mathcal{R}^t$ to multisets of numerical measurements. For example, it is possible to measure whether a certain transition t is executed in a given trace σ:

$$tm_{(r^s,r^a)}^{occur}(\sigma, t) = \begin{cases} [1] \text{ if } \exists_{\sigma' \in pref^{\diamond}(\sigma), \sigma' \neq \langle\rangle} \left(r^s(pref^{|\sigma'|-1}(\sigma')), r^a(\sigma'(|\sigma'|)), r^s(\sigma')\right) = t \\ [0] \text{ otherwise} \end{cases} \tag{3}$$

It is also possible to measure the *elapsed time* of a trace until a transition is triggered within the trace:

$$tm_{(r^s,r^a)}^{elapsed}(\sigma, t) = \biguplus_{\substack{\sigma' \in pref^{\diamond}(\sigma), \sigma' \neq \langle\rangle \\ \left(r^s(pref^{|\sigma'|-1}(\sigma')), r^a(\sigma'(|\sigma'|)), r^s(\sigma')\right) = t}} [att_t(\sigma'(|\sigma'|)) - att_t(\sigma'(1)] \tag{4}$$

2.4 Annotations

As mentioned before, states and transitions can be *annotated* with the measurements obtained from an event log. Given a state measurement function sm, a transition measurement function tm and an event log L, an *annotation* function $an^{(sm,tm,L)} \in (\mathcal{R}^s \cup \mathcal{R}^t) \to \mathbb{B}(\mathbb{R})$, is a function that, given a state $s \in \mathcal{R}^s$ or transition $t \in \mathcal{R}^t$, produces a multiset of numerical measurements. The annotation function is defined as:

$$an^{(sm,tm,L)}(x) = \begin{cases} \biguplus_{\sigma \in L} sm(\sigma, x) \text{ if } x \in \mathcal{R}^s \\ \biguplus_{\sigma \in L} tm(\sigma, x) \text{ if } x \in \mathcal{R}^t \end{cases}$$

3 Comparison and Visualization of the Differences in Process Variants

Given two event logs L_1 and L_2, our approach produces comparison results (as shown in Figure 1) in three steps:

1. Create an *annotated transition system* (i.e., a transition system with multiple annotation functions) from L_1 and L_2 using the state and activity representation functions r^s and r^a and the state and transition measurement functions sm_{r^s} and $tm_{(r^s,r^a)}$.
2. *Compare* the annotations of each state or transition of the annotated transition system.
3. *Visualize* the differences in the annotated transition system.

In order to compare process variants, we need to compare the annotations that are produced for the states and transitions of a transition system. Hence, we introduce *annotated transition systems* which allows to annotate a transition system with multiple annotation functions.

Definition 2 (Annotated Transition System). *Given two event logs L_1 and L_2, state and activity representation functions r^s and r^a, state and transition measurement functions sm and tm, we define an* annotated transition system $ATS^{(r^s,r^a,L_1,L_2,sm,tm)}$ *as the triplet* $(TS^{(r^s,r^a,L_1 \uplus L_2)}, an^{(sm_{r^s},tm_{(r^s,r^a)},L_1)}, an^{(sm_{r^s},tm_{(r^s,r^a)},L_2)})$*, where* $TS^{(r^s,r^a,L_1 \uplus L_2)} = (S, A, T)$ *is a transition system and* $an^{(sm_{r^s},tm_{(r^s,r^a)},L_1)}$, $an^{(sm_{r^s},tm_{(r^s,r^a)},L_2)}$ *are annotation functions denoted as* an_1 *and* an_2 *respectively.*

Note that the transition system uses all the traces contained in the union of the event logs L_1 and L_2. Also, note that an_1 and an_2 use only the traces contained in one event log (L_1 and L_2 respectively).

Figure 3 shows an example of annotated transition system created using the event log L_1 and L_2 are created from the event log presented in Table 1 (the first two traces belong to L_1 and the third trace belongs to L_2), the state representation function $r^s(\sigma) = \{att_n(e) | e \in \sigma\}, \forall \sigma \in P_L$, the activity representation function $r^a(e) = att_n(e), \forall e \in E_L$, the state measurement function sm_{r^s} defined in Eq. 1 and the transition representation function $tm_{(r^s,r^a)}$ defined in Eq. 3. Only annotations of the function an_1 are represented (i.e., as text below the node and arc labels).

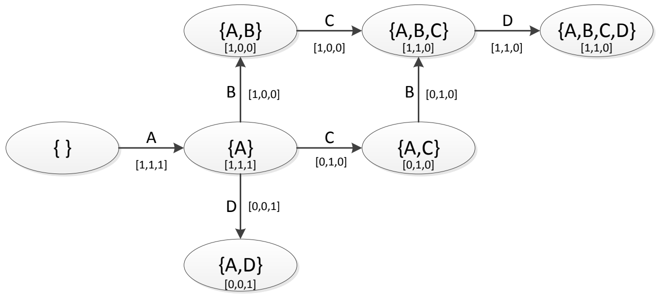


Fig. 3: Transition system annotated with the *occurrence* state and transition measurement functions defined in Equations 1 and 3. Annotations are represented as text under the node and edge labels.

State and Transition Comparison using Annotations. The comparison of annotations can be abstracted as a *comparison oracle* that is defined as the function $\mathit{diff} \in \mathbb{B}(\mathbb{R}) \times \mathbb{B}(\mathbb{R}) \rightarrow \mathit{Bool}$, which given two multi-set of numerical measurements (i.e., annotations) decides whether there are differences between them (i.e., *true*) or not (i.e., *false*).

Given an $ATS = \big((S, A, T), an_1, an_2\big)$, for each element $x \in S \cup T$ we want to detect differences by evaluating $\mathit{diff}(an_1(x), an_2(x))$.

In order to avoid detecting irrelevant differences between the means of the annotations, *statistical significance tests* are used as the comparison oracle. We have opted for the two-tailed "Welch's T-test", also known as the "two-tailed T-test with different variances" [5] because it is suited when the two sets of measurements come from independent populations, such as when they are extracted from two event logs from different branches of a company.

Visualizing Differences in Annotated Transition Systems. Annotations and comparison results of states and transitions can be represented using visual properties (i.e., *thickness* and *color*) of nodes and arcs.

Given an $ATS = \big((S, A, T), an_1, an_2\big)$, for each element $x \in S \cup T$, the *thickness* of the corresponding node (if $x \in S$) or arc (if $x \in T$) is proportional to the mean value of $an_1(x) \uplus an_2(x)$ i.e., the average value of the annotations associated with x and computed on the merged log. The thickness property provides insights about the overall behavior of both variants.

Figure 4 illustrates an example of this visualization using the ATS presented in Figure 3. In this case, the annotations obtained from an_1 and an_2 are represented as thickness instead of text.

Given an $ATS = \big((S, A, T), an_1, an_2\big)$, for each element $x \in S \cup T$, the corresponding node (if $x \in S$) or arc (if $x \in T$) will be colored black or white (depending whether it is an arc or a node) if $\mathit{diff}(an_1(x), an_2(x)) = \mathit{false}$, or it will be colored

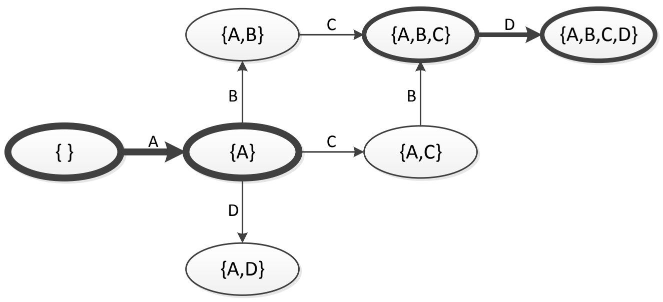


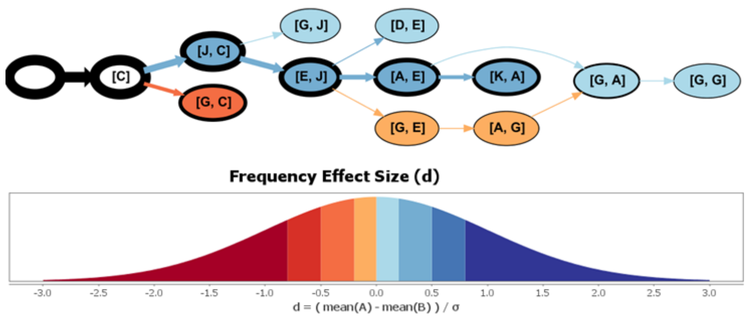
Fig. 4: An example of how the annotations are translated to the thickness of the transition's arcs and state's node borders using the annotated transition system shown in Figure 3.

using other colors if $\mathit{diff}(an_1(x), an_2(x)) = \mathit{true}$. In the latter case, the color used will depend on the measurement function used and on the *effect size* of the difference.

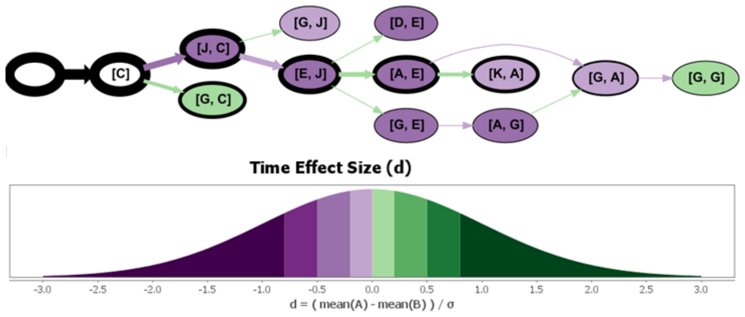
The effect size oracle is defined as the function $\mathit{eff} \in \mathbb{B}(\mathbb{R}) \times \mathbb{B}(\mathbb{R}) \rightarrow \mathbb{R}$, which given two multisets of measurements, returns the size of the effect (i.e., how small or large is the difference) and the sign of the difference (+/-) in a certain scale. In this paper, we used Cohen's d [6] to measure effect size, which measures the difference of sample means in terms of pooled standard deviation units. Cohen relates ranges of d values to effect size categories: $d = \pm 0.2$ is considered as a *small effect*, $d = \pm 0.5$ is considered as a *medium effect* and $d = \pm 0.8$ is considered as a *large effect*. However, other effect size measurements could be used instead.

Currently, we support two measurement functions and, hence, two color intervals are used, as shown in Figure 5[1]. In Figure 5.a, *occurrence* measurement functions (Eqs. 1 and 3) were used. Blue-based colors mean that the occurrence of a state or transition in a first event log is higher than in a second event log and red-based colors mean the opposite. In Figure 5.b, *elapsed time* (performance) annotation functions (Eqs. 2 and 4) were used. Green-based colors mean that the elapsed time of reaching a state or executing a transition in a first event log is higher than in a second event log and purple-based colors mean the opposite. Note that within the color intervals, different colors are used according to Cohen's d ranges of effect size values. Colors with higher intensity (i.e., darker) represent larger effect sizes (i.e., more relevant differences), whereas colors with low intensity (i.e., lighter) represent smaller effect sizes (i.e., less relevant differences).

[1] Note that the example transition system used in this figure is different than previous examples, and it is used for illustration purposes only



(a) sm and tm measure occurrence (Eq. 1 and 3)



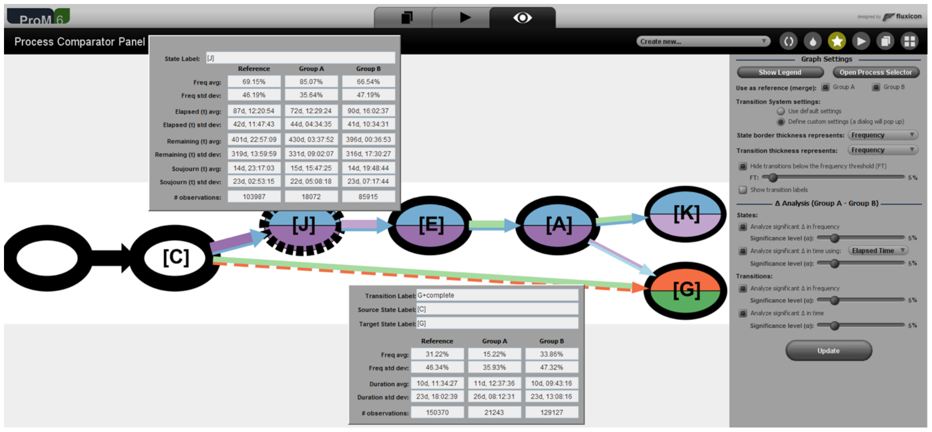
(b) sm and tm measure elapsed time (Eq. 2 and 4)

Fig. 5: Example of an annotated transition system colored with the results of statistical significance tests and effect size oracle using different state and transition measurement functions.

4 Implementation

Our approach has been implemented as the *Process Comparator* plugin in the ProM [7] framework. ProM allows researchers to implement process mining techniques in a standardized environment, providing several functionalities that can be used by the implementations, and also providing a distribution platform for other researchers to use these developments. The ProM framework is considered as the *de-facto* standard for process mining, and it can be freely downloaded form `http://promtools.org`.

The tool takes two event logs as input. However, more than two event logs can be compared. This is handled by requesting the user to group these event logs into two groups. Each of these groups is then merged into a single event log and then compared against each other. The tool also provides a "hint" functionality for the users that do not have context knowledge or do not know which processes to compare. This functionality suggests to compare a single process against all the others by calculating *similarity scores* between each process and the union of the $n-1$ remaining processes. Similarity score is calculated based on the percentage of elements that present statistically signifi-



State Label: [J]

	Reference	Group A	Group B
Freq avg:	69.15%	85.07%	66.54%
Freq std dev:	46.19%	35.64%	47.19%
Elapsed (t) avg:	87d, 12:20:54	72d, 12:29:24	90d, 16:02:37
Elapsed (t) std dev:	42d, 11:47:43	44d, 04:34:35	41d, 10:34:31
Remaining (t) avg:	401d, 22:57:09	430d, 03:37:52	396d, 00:36:53
Remaining (t) std dev:	319d, 13:59:59	331d, 09:02:07	316d, 17:30:27
Soujourn (t) avg:	14d, 23:17:03	15d, 15:47:25	14d, 19:48:44
Soujourn (t) std dev:	23d, 02:53:15	22d, 05:08:18	23d, 07:17:44
# observations:	103987	18072	85915

Transition Label: G+complete
Source State Label: [C]
Target State Label: [G]

	Reference	Group A	Group B
Freq avg:	31.22%	15.22%	33.86%
Freq std dev:	46.34%	35.93%	47.32%
Duration avg:	10d, 11:34:27	11d, 12:37:36	10d, 09:43:16
Duration std dev:	23d, 18:02:39	26d, 08:12:31	23d, 13:08:16
# observations:	150370	21243	129127

Graph Settings
Show Legend | Open Process Selector
Use as reference (merge): Group A Group B
Transition System settings:
Use default settings
[✓ Define custom settings (a dialog will pop up)]
State border thickness represents: Frequency
Transition thickness represents: Frequency
Hide transitions below the frequency threshold (FT)
FT: 5%
Show transition labels
Δ Analysis (Group A - Group B)
States:
Analyze significant Δ in frequency
Significance level (α): 5%
Analyze significant Δ in time using: Elapsed Time
Significance level (α): 5%
Transitions:
Analyze significant Δ in frequency
Significance level (α): 5%
Analyze significant Δ in time
Significance level (α): 5%
Update

Fig. 6: Screenshot of the *Process Comparator* plugin in the ProM framework. Detailed data tables pop-up when the user clicks on states or transitions.

cant differences. Finally, the process that has most differences with the rest is suggested to the user as a starting point for comparative analysis.

Our tool allows the user to change state and event representation functions, state and transition measurement functions and several useful parameters (e.g., the significance level of the statistical significance tests) in order to provide flexible representations for the event logs, as shown in Figure 6. Our tool also provides frequency filtering capabilities where all the nodes and arcs with lower frequency than a defined threshold will be hidden from the visualization. This allows to filter out rare behavior and to produce clearer visualizations. Also, the elements of the annotated transition system presented as result are *interactive*. The user can click on any state or transition, and a data table will pop-up showing the values of the annotations of such state or transition for both event logs (e.g., frequency of occurrence, elapsed time, remaining time, number of traces).

5 Evaluation

In order to show the usefulness of our approach in practice, we performed experiments using multiple real-life event logs. Here we report on a log extracted from an Italian Municipality's information system that handled the “road fines management” process [8]. For showing the comparison capabilities of our approach, we split the event logs into two sub logs (i.e., *variants*): the first one contains all the cases where the fine amount was lower than 50 euros (i.e., *low fines*) and the second contains all the cases where the amount of the fine was equal or higher than 50 euros (i.e., *high fines*). The two event logs were then compared against each other using our tool, and the differences were projected into an annotated transition system. We performed two sets of experiments:

– The first was based on an abstraction where the last event of the trace is considered. We used the following state and transition abstraction: given an event log L, a trace

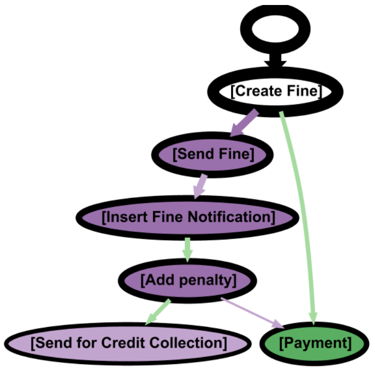


Fig. 7: Performance (*elapsed time*) comparison. Colored states (i.e., nodes) and transitions (i.e., edges) contain statistically significant differences between the two event logs. Purple shades represent earlier executions of activities or reaching of states in *high* fines. Green shades represent the other way around. White indicates that no significant differences can be observed. The shades become darker and darker with increasingly statistically significant differences.

$\sigma \in P_L$ and an event $e \in E_L$, $r^s(\sigma) = att_n(\sigma(|\sigma|))$ and $r^a(e) = att_n(e)$. As measurement for comparison, elapsed time was used as defined in Eqs. 2 and 4, thus comparing the time differences when activities were executed.

– The second was based on an abstraction where the last two events were considered: $r^s(\sigma) = \langle att_n(\sigma(|\sigma|)), att_n(\sigma(|\sigma| - 1))\rangle$ and $r^a(e) = att_n(e)$. The occurrence measurements for comparison were used as defined in Eqs. 1 and 3.

In both of experiments, we used a confidence level $\alpha = 0.05$ for the Welch's T tests.

Fig. 7 shows the results of the first experiment, where many relevant performance differences were detected. As previously shown in Fig. 5.b, green colors are assigned to states and transitions that are reached or executed statistically significantly earlier in low fines, whereas purple colors are assigned when the opposite occurs. The green color assigned to state *Payment* indicates that payments were received significantly earlier for low fines (99 days versus 151 days)[2]. Conversely, the purple-colored transition *(Create Fine, Send Fine)* indicates that high fines are sent to offenders significantly earlier (72 days versus 90 days)[2]. The thickness of this arc also indicates that, overall, sending the fine after creating is a more frequent behavior. The fact that the *Create Fine* state is white indicates that there is no statistically significant difference in how early *Create Fine* is executed.

Figure 8 illustrates the output of the second experiment. Orange shade ovals and arcs represent states reached or transitions executed significantly more often in low fines compared with high fines. Blue shades refer to the opposite. The first observation is that low fines are usually immediately paid without requiring the local police to send

[2] This is not observable in the picture but, in the implementation, by clicking on a state/transition, one can read this information in a popup equivalent to the two shown in Figure 6

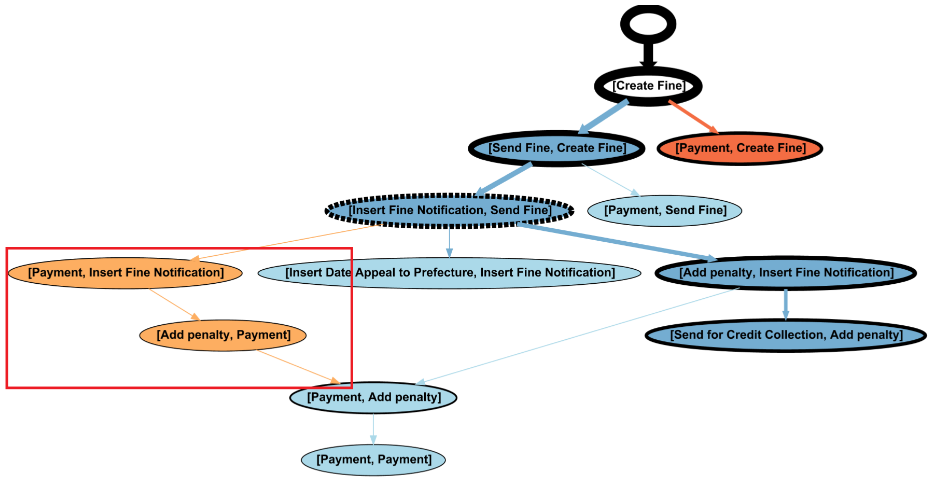

[Create Fine]
[Send Fine, Create Fine]
[Payment, Create Fine]
[Insert Fine Notification, Send Fine]
[Payment, Send Fine]
[Payment, Insert Fine Notification]
[Insert Date Appeal to Prefecture, Insert Fine Notification]
[Add penalty, Insert Fine Notification]
[Add penalty, Payment]
[Send for Credit Collection, Add penalty]
[Payment, Add penalty]
[Payment, Payment]

Fig. 8: *Occurrence* frequency comparison. Colored states (i.e., nodes) and transitions (i.e., edges) contain statistically significant differences between the two event logs. Blue colors represent a higher occurrence in *high* fines . Orange colors represent a higher occurrence in *low* fines.

a copy of fine to the offender. This can be seen through the orange-colored state *[Payment,Create Fine]* and the transition from *[Create Fine]* to this state. Conversely, high fines are more often sent to the offender than low fines, as one can observe through the blue-colored state *[Send Fine, Create Fine]*. Similar observations can be derived by looking at the other states and transitions. Figure 8 highlights part of the transition system (red rectangle). It indicates that, for low fines, it happens significantly more often that offenders perform incomplete payments, which cause a penalty to be added[3], which are subsequently followed by a second payment to probably complete the fine payment. Conversely, for high fines, it is significantly more frequent that payments only occurs after adding the penalty. This can be seen from the blue color associated with the transition between states *[Add Penalty, Insert Fine Notification]* and *[Payment,Add Penalty]*. Please observe that the latter finding could not be observed if we used an abstraction solely based on the last occurred event.

6 Related Work

Earlier work has been done on comparing process variants. The corresponding papers can be grouped in two category: model-based and log-based comparison. The main difference between these two categories is that model-based approaches require process models as inputs and log-based approaches require event logs as inputs. Indirectly,

[3] According to the Italian laws, if a fine is not paid in full within 90 days, a penalty is added so that the due amount doubles

model-based approaches can also be used starting from event logs. Models can be discovered from logs and then used as inputs for the approach. However, the obtained insights should be validated since the structure of the models (hence, the detected differences) can be drastically affected by the choice of the discovery technique or its parameters.

Model-based Comparison. Model-based comparison techniques have been developed in recent years [9,10,11,12]. La Rosa et al. [11] provide a complete overview of the different ways to compare and merge models. Most of them are based on control-flow comparison, where the structural properties of the models (represented as graphs) are compared (e.g., nodes and edges present in one of the models, but not in the other one).

A drawback of model-based approaches is that they are unable to detect differences in terms of frequency or any other process metrics (e.g., elapsed time). For example, in Section 5, we detected a frequency difference on the payment of a fine directly after being created (34% of the *low* fines versus 15% of the *high* fines). This difference is not detected by model-based approaches, since in both variants the activity "Create Fine" is followed by "Payment" in at least 15% of the cases, so this behavior would be present in the models of both variants. A severe drawback of employing model-based comparison is related to the fact that the variants are compared in terms of their model structure whereas we aim to compare the behavior. This motivates why, in this paper, we have opted for a low-level behavioral representation, i.e., transition systems, instead of high-level process modelling languages, such as BPMN or Petri nets. For instance, they are unable to detect that low-fine offenders perform incomplete payments that need to be integrated after receiving a penalization.

Log-based Comparison. The most recent approach for log-based behavior comparison is by van Beest et al. [13]. This technique is able to identify differences between two event logs by computing *frequency-enhanced prime event structures* (FPES) from the corresponding event logs, comparing the obtained FPES and report the results using two sets of textual statements: *control-flow* differences and *branching frequency* differences.

This approach has some advantages, such as the handling of concurrency in process behavior. However, it presents three main limitations described as follows. First, the technique looks at the relative frequency, only. As such, when looking at branching frequency, it possibly returns a difference (if any), even though the branching point is actually reached very rarely. Also, no statistical significant tests are employed. Second, to determine branching points, they only look at the last activity independently of what activities were previously executed. As such - as we have verified by testing the reference implementation - it is unable to detect differences that refer to the activities preceding the last, such as, in the road-traffic case study, a number of low-fine offenders perform incomplete payments that need to be integrated after receiving a penalization. Third, the approach considers event logs as sequences of event labels, thus ignoring all other event attributes (e.g., timestamp, event payload). This limits the approach to detect only frequency differences. Differences in performance or other process metrics cannot be obtained.

Other approaches based on *sequence mining* such as [14,15,16,17] obtain rules that are overcomplicated and not valuable from a business perspective (as indicated in [14] and [13]).

7 Conclusion

The problem of comparing process variants is highly relevant. Many companies are observing that the executions of their processes are not always optimal and subject to variations. Processes may change because of the influence of several factors, such as the year period, the geographical location of the process' execution or the resource unit in charge. Some recent approaches aim to compare the execution of the different process variants. Most existing approaches tend to focus on the control-flow perspective or to detect differences that are statistically insignificant.

To our knowledge, no current approach is able to detect the relevant behavioral differences between process variants in terms of any process metric (e.g., performance) based on their recorded event logs. To address this issue, we developed a new technique based on annotated transition systems that detects statistically significant differences between process variants in terms of any process metric, using event logs as input. We used annotated transition systems to avoid being mining algorithm specific.

Our implementation is provided with two concrete metrics, which are related to the control-flow frequency (in the paper, named *occurrence*) and to the time perspective (the *elapsed time* metric). However, the framework allows one to easily add new measurement functions.

The evaluation and the related-work analysis has clearly shown that the approach is relevant and allows one to pinpoint differences that previous approaches fail to provide. Also, our approach excludes all differences that are in fact statistically insignificant, which are conversely returned by other approaches.

As future work, we aim to evaluate to what extent this visual approach scales when processes get larger and more complex. In this way, we can obtain direct feedback about whether business stakeholders can understand and benefit from our visual approach. Also, we aim to integrate it with *process cubes*, thus providing a complete suite to slice, dice, drill down, roll up and compare process variants.

References

1. van der Aalst, W.M.P.: Process Mining: Discovery, Conformance and Enhancement of Business Processes. 1st edn. Springer-Verlag Berlin Heidelberg (2011)
2. Bolt, A., van der Aalst, W.M.P.: Multidimensional process mining using process cubes. In: Enterprise, Business-Process and Information Systems Modeling. Volume 214 of Lecture Notes in Business Information Processing. Springer International Publishing (2015) 102–116
3. van der Aalst, W.M.P.: Process cubes: slicing, dicing, rolling up and drilling down event data for process mining. In: Proceedings of the First Asia Pacific Conference on Business Process Management. Volume 159 of Lecture Notes in Business Information Processing., Springer International Publishing (2013) 1–22

4. van der Aalst, W.M.P., Schonenberg, M.H., Song, M.: Time prediction based on process mining. Information Systems **36**(2) (2011) 450 – 475 Special Issue: Semantic Integration of Data, Multimedia, and Services.
5. Welch, B.L.: The generalization of 'student's' problem when several different population variances are involved. Biometrika **34**(1-2) (1947) 28–35
6. Cohen, J.: Statistical Power Analysis for the Behavioral Sciences. Lawrence Erlbaum Associates (1988)
7. van Dongen, B.F., de Medeiros, A.K.A., Verbeek, H.M.W., Weijters, A.J.M.M., van der Aalst, W.M.P.: The ProM framework: A new era in process mining tool support. In: Applications and Theory of Petri Nets. Volume 3536 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (2005) 444–454
8. de Leoni, M., Mannhardt, F.: Road traffic fine management process. `10.4121/uuid:270fd440-1057-4fb9-89a9-b699b47990f5` (2015)
9. Kriglstein, S., Wallner, G., Rinderle-Ma, S.: A visualization approach for difference analysis of process models and instance traffic. In: Business Process Management. Volume 8094 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (2013) 219–226
10. Cordes, C., Vogelgesang, T., Appelrath, H.J.: A generic approach for calculating and visualizing differences between process models in multidimensional process mining. In: Business Process Management Workshops. Volume 202 of Lecture Notes in Business Information Processing. Springer International Publishing (2015) 383–394
11. La Rosa, M., Dumas, M., Uba, R., Dijkman, R.: Business process model merging: An approach to business process consolidation. ACM Trans. Softw. Eng. Methodol. **22**(2) (March 2013) 11:1–11:42
12. Ivanov, S., Kalenkova, A., van der Aalst, W.M.P.: BPMNDiffViz: A tool for BPMN models comparison. In: Proceedings of the BPM Demo Session 2015 Co-located with the 13th International Conference on Business Process Management (BPM 2015), Innsbruck, Austria, September 2, 2015. (2015) 35–39
13. van Beest, N., Dumas, M., García-Bañuelos, L., La Rosa, M.: Log delta analysis: Interpretable differencing of business process event logs. In: Proceedings of the 13th International Conference on Business Process Management (BPM'15). (2015) 386–405
14. Nguyen, H., Dumas, M., La Rosa, M., Maggi, F.M., Suriadi, S.: Mining business process deviance: A quest for accuracy. In: On the Move to Meaningful Internet Systems (OTM 2014). Volume 8841 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (2014) 436–445
15. Lakshmanan, G., Rozsnyai, S., Wang, F.: Investigating clinical care pathways correlated with outcomes. In: Business Process Management (BPM 2013). Volume 8094 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (2013) 323–338
16. Jagadeesh Chandra Bose, R., van der Aalst, W.M.P.: Abstractions in process mining: A taxonomy of patterns. In: Business Process Management (BPM 2009). Volume 5701 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (2009) 159–175
17. Swinnen, J., Depaire, B., Jans, M., Vanhoof, K.: A process deviation analysis – a case study. In: Business Process Management Workshops. Volume 99 of Lecture Notes in Business Information Processing. Springer Berlin Heidelberg (2012) 87–98