Multi-Perspective Comparison of Business Process Variants Based on Event Logs

Hoang Nguyen[1], Marlon Dumas[2], Marcello La Rosa[3], and Arthur H.M. ter Hofstede[1]

[1] Queensland University of Technology
huanghuy.nguyen@hdr.qut.edu.au, a.terhofstede@qut.edu.au
[2] University of Tartu
marlon.dumas@ut.ee
[3] University of Melbourne
marcello.larosa@unimelb.edu.au

Abstract. A process variant represents a collection of cases with certain shared characteristics, e.g. cases that exhibit certain levels of performance. The comparison of business process variants based on event logs is a recurrent operation in the field of process mining. Existing approaches focus on comparing variants based on directly-follows relations such as "a task directly follows another one" or a "resource directly hands-off to another resource". This paper presents a more general approach to log-based process variant comparison based on so-called *perspective graphs*. A perspective graph is a graph-based abstraction of an event log where a node represents any entity referred to in the log (e.g. task, resource, location) and an arc represents a relation between these entities within or across cases (e.g. directly-follows, co-occurs, hands-off to, works-together with). Statistically significant differences between two perspective graphs are captured in a so-called *differential perspective graph*, which allows us to compare two logs from any perspective. The paper illustrates the approach and compares it to an existing baseline using real-life event logs.

Keywords: Process mining, variant analysis, comparison, multi-perspective.

1 Introduction

The performance of a business process may vary over time, geographically, or across business units, products, or customer types. And even within a given time period, place, business unit, product, and customer type, there are usually performance variations between cases of a process. Some cases lead to a positive outcome (e.g. on-time completion), while others lead to negative outcomes. A typical question that arises in this setting is: *"What differentiates the positive and the negative cases?"*, or more broadly: *"What (statistically) significant differences exist between two variants of a process?"*

Recently, several approaches for comparing variants of a process based on their event logs have been proposed [1–4]. Given two event logs L1 and L2 corresponding to two variants of a business process, these techniques allow us to identify characteristics that are commonly found in the cases in L1, but are rare or non-existent in L2 and vice-versa. These approaches are restricted to identifying differences in the directly-follows relations, such as "task A always directly follows task B in one variant but never in the other" or "resource X often hands-off work to resource Y in one variant but rarely in the other". However, events in a log may carry a richer set of attributes besides tasks and resources – e.g. customer attributes, location attributes, product-related attributes, etc. Differences between two event logs may be found along any of these attributes.

This paper presents an approach for comparing process variants from multiple perspectives corresponding to arbitrary sets of attributes. Specifically, the paper introduces a graph-based abstraction of an event log, namely a *perspective graph*, where a node represents any entity referenced in an attribute of the event log (task, resource, location, etc.) and an arc represents an arbitrary relation between entities (e.g. directly-follows,

co-occurs, hands-off to, works-together with, etc.) within or across cases. Statistically significant differences between two perspective graphs are captured in a so-called *differential perspective graph*, which allows a user to visually compare two event logs from any given perspective using either a graphical or a matrix representation.

The proposed approach has been implemented as a proof-of-concept prototype in the ProM open-source process mining toolset. The paper illustrates the capabilities provided by differential perspective graphs using two real-life event logs, and compares them against an existing state-of-the-art approach for process variant analysis.

The paper is structured as follows. Section 2 discusses existing process variant analysis approaches. Section 3 presents the proposed approach, while Section 4 discusses its evaluation. Section 5 summarizes the contributions and outlines future work directions.

2 Related Work

Existing approaches to log-based process variant comparison can be classified into indicator-based, graph-based, and model-based. Indicator-based approaches extract performance indicators from two input logs and compare these indicators using visualization techniques (e.g. bar charts), e.g. risk indicators [5], performance indicators [6], and resource behavior indicators [7]. These approaches allow one to determine how two variants perform relative to each other on an aggregate basis or at a task- or resource level. For example, these techniques allow us to determine which tasks have higher cycle time in one variant than in the other. However, they do not allow us to identify behavioral differences and their impact on performance.

Graph-based approaches rely on pairwise differencing of graphs, such as event structures [2], directly-follows graphs [8], or transition systems [4]. For example, the approach in [4] abstracts a process as a transition system where each state represents an equivalence class of trace prefixes, e.g. a state may represent all prefixes that coincide on their last n events. Each transition is labeled with an event label or event attribute value. Transition systems are contrasted and differences are visually highlighted. The approach in [4] is integrated with a Process Cube approach for log slicing and dicing to generate sublogs for comparison [9]. Our technique falls into this category. In particular, the technique in [4] is used as a baseline in our evaluation.

Other related techniques take as input process models and enrich them with performance measures extracted from event logs, such as occurrence frequency or cycle time [3]. These techniques assume that a process model is available, which captures the dominant behaviors of the process variants. In contrast, in this paper we assume that no models are given a priori. Instead, the comparison of variants is conducted on an exploratory basis, from multiple perspectives, and purely based on event logs.

3 Approach

Given two event logs as input, each representing a variant of the same business process, our approach mines two perspective graphs, one from each log. Next, it compares the two graphs and visualizes their statistically significant differences using a differential graph. In the remainder, we define all ingredients of our approach: event logs, process abstraction, perspective graphs, and differential graphs.

3.1 Event Logs and Process Abstraction

An event log consists of cases where each case has a number of associated events. Cases and events can have various attributes. An example log is shown in Table 1. Rows are events and columns are event attributes. The format of event logs has been standardized in the eXtensible Event Stream by the IEEE CIS Task Force on Process Mining [10]. Table 1 provides an example of a *log schema*. Our technique can work with any log schema.

Let Ω be a universe of values, $\mathcal{E}$ be a universe of events and $\mathcal{A}$ be a universe of attribute names, where for each $A \in \mathcal{A}, A: \mathcal{E} \rightarrow \Omega$.

Definition 1 (Event Log) *An event log L over a schema $S = \{A_1, A_2, \ldots, A_n\} \subset \mathcal{A}$ is a set of events, i.e. a subset of $\mathcal{E}$.*

Assume that events in a case occur sequentially as shown in Table 1. Based on the event sequence in each case, Fig. 1 shows an example of event clusterings in each case along the time line according to the *Department* and *Location* attributes. For example, in Fig. 1a, events e_1 and e_2 share the same department attribute d_1, and the next occurring events e_3 and e_4 share the same department d_2. This is followed by event e_5 which has occurred with department attribute d_1 again. Fig. 1a and Fig. 1b each is seen as an *abstraction* of the process in Table 1. This method of process abstraction is formalized as follows.

CaseID	EventID	Timestamp	Activity	Resource	Department	Location
c_1	e_1	01.10 10:00:00	a_1	r_1	d_1	l_1
c_1	e_2	02.10 10:00:00	a_2	r_2	d_1	l_1
c_1	e_3	03.10 10:00:00	a_3	r_3	d_2	l_1
c_1	e_4	04.10 10:00:00	a_1	r_3	d_2	l_2
c_1	e_5	05.10 10:00:00	a_2	r_1	d_1	l_2
c_2	e_6	02.10 10:00:00	a_3	r_1	d_1	l_1
c_2	e_7	04.10 10:00:00	a_1	r_2	d_1	l_2
c_2	e_8	06.10 10:00:00	a_3	r_4	d_2	l_2
c_2	e_9	08.10 10:00:00	a_2	r_2	d_1	l_1

Table 1. Event log example

Let L be a log over schema $S, CaseID$ the CaseID attribute of events, *timestamp* the timestamp attribute of events, CID_L the set of CaseIDs in L, and $T \subseteq S$ a schema with $T \neq \emptyset$ and $timestamp \notin T$. We assume that all timestamps are different.

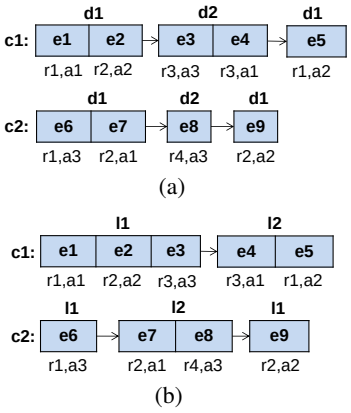

Fig. 1. Abstraction by (a) *Department*, (b) *Location*

First, we define a number of relations between events as the basis for our later formalisation. Events can be related in terms of *timestamp*, *CaseID*, and other attributes. Given two events, first they can be *ordered* based on their timestamps, i.e. $e_1 < e_2$ iff $timestamp(e_1) < timestamp(e_2)$. Second, they are *case-related* if they occur in the same case, i.e. $e_1 \doteq e_2$ iff $CaseID(e_1) = CaseID(e_2)$. Finally, in terms of a schema $T \subseteq S$, they are *T-equal* if they share values for all attributes in T, i.e. $e_1 =_T e_2$ iff $\forall t \in T\ [t(e_1) = t(e_2)]$; otherwise, they are *T-unequal*, i.e. $e_1 \neq_T e_2$.

Based on the above relations, two events are *case-ordered* iff they are *ordered* and *case-related*, i.e. $e_1 \lessdot e_2$ iff $e_1 < e_2 \wedge e_1 \doteq e_2$. Further, two events are *T-case-ordered* iff they are *case-ordered* and *T-unequal* or *case-ordered*, *T-equal* but separated in time from each other by another case-related but T-unequal event, i.e. $e_1 \lessdot_T e_2$ iff $(e_1 \lessdot e_2 \wedge e_1 \neq_T e_2) \vee (e_1 \lessdot e_2 \wedge e_1 =_T e_2 \wedge \exists e_3 \in L\ [e_1 \lessdot e_3 \wedge e_3 \lessdot e_2 \wedge e_3 \neq_T e_1])$.

The T-case-ordered relation $\lessdot_T$ can be observed in Table 1. In case c_1, in terms of schema T={Department}, events e_1 and e_3 are *T-case-ordered* because they are ordered and T-unequal; events e_1 and e_5 are also *T-case-ordered* because they are ordered, T-equal and there is an event, e.g. e_3, that occurs between them in the same case but is T-unequal to them. The T-case-ordered relation $\lessdot_T$ forms a strict partial order over $\mathcal{E}$ for each CaseID. This can be sketched briefly. T-case-ordered is an *irreflexive* relation because $e \lessdot_T e$ implies $e \lessdot e$ hence $timestamp(e) < timestamp(e)$ which is not possible. From the definition of T-case-ordered, it can be proved that T-case-ordered is a *transitive* relation by case distinction with four cases.

Given two case-related events and a schema T, if there exists no *T-case-ordered* relation between the two events, it means that they are T-equal and non-separable in time from each other by another case-related T-unequal event. In this situation, we say that they are *T-equivalent*, i.e. $e_1 \sim_T e_2$ iff $e_1 \doteq e_2 \wedge \neg(e_1 \lessdot_T e_2) \wedge \neg(e_2 \lessdot_T e_1)$.

The T-equivalent relation $\sim_T$ on the event set $E_c = \{e \in L \mid CaseID(e) = c\}$ of a caseid c forms an equivalence relation, where the corresponding quotient set of E_c is $E_c\backslash_{\sim_T}$. Two T-equivalent events are in the same equivalence class of $E_c\backslash_{\sim_T}$. We refer to an equivalence class in $E_c\backslash_{\sim_T}$ as a *fragment*. Visually, a fragment is a row of events as shown in Fig. 1, e.g. in Fig. 1a, $\{e_1, e_2\}$ is a fragment in case c_1.

Based on the notion of fragments, we now define process abstraction.

Definition 2 (Process Abstraction) *Let L be a log, $T \subset S$ a schema, and CID_L the set of CaseIDs in L. An abstraction over T from L is defined as $\mathcal{A}_T^L = \bigcup_{c \in CID_L} E_c\backslash_{\sim_T}$.*

Let $\mathcal{A}_T^L$ be an abstraction over schema T from log L. From the definition, $\mathcal{A}_T^L$ is a set of fragments where each fragment is a set of T-equivalent events. There is a *follows* relation between fragments $F_1, F_2 \in \mathcal{A}_T^L$, i.e. $F_1 \twoheadrightarrow F_2$ iff $\exists e_1 \in F_1\ \exists e_2 \in F_2\ [e_1 \lessdot_T e_2]$, and a *directly-follows* relation between fragments $F_1, F_2 \in \mathcal{A}_T^L$, i.e. $F_1 \rightarrow F_2$ iff $F_1 \twoheadrightarrow F_2 \wedge \nexists F_3 \in \mathcal{A}_T^L\ [F_1 \twoheadrightarrow F_3 \wedge F_3 \twoheadrightarrow F_2]$.

3.2 Perspective Graphs

From a process abstraction as shown in Fig. 1, one can look at different relations between event attributes. For example, one can look at the co-occurrence of two attributes in the same fragment, e.g. two resources working in the same department or location. Alternatively, one can look at an inter-fragment relation where an attribute occurs in one fragment and the other attribute occurs in a directly following fragment, e.g. the flow from an activity performed in one department to another activity performed in the next department. More generally, instead of focusing on one attribute, one may focus on a number of attributes depending on the type of analysis, e.g. it may be a pair (resource, activity) representing a task assignment.

In order to represent different types of relation between event attributes, we propose two types of graphs, *intra-fragment* and *inter-fragment*, defined as follows.

Let $\mathcal{A}_T^L$ be an abstraction over schema T from log L. Let $V \subseteq S$, then $\pi_V(e)$ denotes a projection on event e of attributes in V which is defined as $\{(v, v(e)) \mid v \in V\}$[1].

Definition 3 (Intra-Fragment Graph) *Let $\mathcal{A}_T^L$ be an abstraction over schema T from log L and let $U, V \subseteq S$. An Intra-Fragment Graph $IAG_T^{U,V}(L)$ is a node and arc weighted undirected graph $G = (N, E, W_N, W_E)$, defined by:*

- $N = \{\pi_U(e) \mid e \in L\} \cup \{\pi_V(e) \mid e \in L\}$,
- $E = \{\{\pi_U(e), \pi_V(e')\} \mid e \in L \wedge e' \in L \wedge e \sim_T e' \wedge \pi_U(e) \neq \pi_V(e')\}$,
- $W_N(n) = |\{e \in L \mid \pi_U(e) = n \vee \pi_V(e) = n\}|$ *for all* $n \in N$,
- $W_E(\{n_1, n_2\}) = |\{e \in L \mid \pi_U(e) = n_1 \wedge \pi_V(e) = n_2\}| + |\{\{e_1, e_2\} \mid e_1 \in L \wedge e_2 \in L \wedge \pi_U(e_1) = n_1 \wedge \pi_V(e_2) = n_2 \wedge e_1 \sim_T e_2 \wedge e_1 \neq e_2\}|$ *for all* $\{n_1, n_2\} \in E$.

Intra-Fragment Graphs represent a *co-occurrence relation* between event attributes as they co-occur in the same fragment. For example, it can be a task assignment relation when a resource and an activity co-occur in an event, or a co-location relation when two resources co-occur in the same location.

Let $\mathcal{A}_T^L$ be an abstraction over schema T from log L. Let $\phi, \varphi: \mathcal{A}_T^L \rightarrow \mathcal{E}$ such that for all $F \in \mathcal{A}_T^L$, $\phi(F) \in \mathcal{E}$ and $\varphi(F) \in \mathcal{E}$. ϕ and φ are two choice functions on $\mathcal{A}_T^L$, i.e. choice functions can be used to extract events with certain properties from fragments. We will not specify their semantics. Just as an example, ϕ could be chosen such that it returns the latest event in a fragment, and φ could be chosen such that it returns the earliest event in a fragment.

Given two choice functions ϕ and φ, two events are in an *inter-fragment directly-follows* relation, denoted $e_1 \rightarrow_T e_2$, iff $\exists F_1, F_2 \in \mathcal{A}_T^L \mid F_1 \rightarrow F_2 \wedge e_1 = \phi(F_1) \wedge e_2 = \varphi(F_2)$. The set of all inter-fragment directly-follows event pairs in log L is $\mathcal{E}_{\rightarrow_T} = \{(e_1, e_2) \in \mathcal{E} \times \mathcal{E} \mid e_1 \rightarrow_T e_2\}$.

[1] $(v, v(e))$ is abbreviated to $v(e)$ when it is clear from the context for the purpose of readability

Definition 4 (Inter-Fragment Graph) *Let $\mathcal{A}_T^L$ be an abstraction over schema T from log L and let $U, V \subseteq S$. An Inter-Fragment Graph $IEG_T^{U,V}(L)$ is a node and arc weighted directed graph $G = (N, E, W_N, W_E)$, defined by:*

- $N = \{\pi_U(e) \mid e' \in \mathcal{E} \wedge (e, e') \in \mathcal{E}_{\rightarrow_T}\} \cup \{\pi_V(e') \mid e \in \mathcal{E} \wedge (e, e') \in \mathcal{E}_{\rightarrow_T}\}$,
- $E = \{(\pi_U(e), \pi_V(e')) \mid (e, e') \in \mathcal{E}_{\rightarrow_T}\}$,
- $W_N(n) = |\{(e, e') \in \mathcal{E}_{\rightarrow_T} \mid \pi_U(e) = n\}| + |\{(e, e') \in \mathcal{E}_{\rightarrow_T} \mid \pi_V(e') = n\}|$, *for all* $n \in N$,
- *Let* $(n_1, n_2) \in E$ *and* $\mathcal{E}_{\rightarrow_T}^{n_1,n_2}$ *be the set of all inter-fragment directly-follows event pairs corresponding to* (n_1, n_2), *i.e.* $\mathcal{E}_{\rightarrow_T}^{n_1,n_2} = \{(e_1, e_2) \in \mathcal{E}_{\rightarrow_T} \mid \pi_U(e_1) = n_1 \wedge \pi_V(e_2) = n_2\}$.

$$W_E((n_1, n_2)) = \begin{cases} |\mathcal{E}_{\rightarrow_T}^{n_1,n_2}| & \textit{if frequency} - \textit{based} \\ \frac{\sum_{(e_1,e_2)\in\mathcal{E}_{\rightarrow_T}^{n_1,n_2}} \left(\textit{timestamp}(e_2) - \textit{timestamp}(e_1)\right)}{|\mathcal{E}_{\rightarrow_T}^{n_1,n_2}|} & \textit{if time} - \textit{based} \end{cases}.$$

Inter-Fragment Graphs represent a *flow relation* between event attributes. For example, it can be a hand-over from a resource in one department to another resource in a directly following department, or a flow from an activity executed in one location to another activity executed in a directly following location.

3.3 Comparing Perspective Graphs and Visualizing Differences

In comparing two perspective graphs, common nodes and edges on the two graphs are compared in terms of their weights. Note that the weights defined in Section 3.2 are computed for the whole log. Instead of comparing graphs based on these weights, this paper looks for statistically significant differences by comparing sample populations of weights obtained from log observations.

Different techniques can be used to make observations of logs. *Case-wise observations* are made on cases in the log, i.e. weights of nodes and edges are computed from events in each case. Differences determined by the tests can be understood as differences between the two variants synthesized from all cases. *Time-wise observation* allows one to see differences between logs over time. This technique uses a sliding time window starting from the earliest event in each log. Observations are made on each window, i.e. weights of nodes and edges are computed from events occurring within each window.

The result of graph comparison is a *differential graph* containing common nodes and edges and also uncommon nodes and edges that appear in one graph only. If nodes and edges are common with a statistically significant difference, their weight is the *effect size* of the difference. This paper chooses *common language effect size* [11] due to its interpret-ability. For example, an effect size of 80% indicates that given any random observations of the two variants, variant A has 80% chance of having a higher mean weight than variant B. If nodes or edges are common without a statistically significant difference, their weight is simply zero. Lastly, if they are uncommon, their weight is the relative weight among all uncommon nodes or edges in the graph. Differential graphs are visualised in the form of matrices (nodes are row and column headers while edges are cells). Matrices can be symmetric (for undirected graphs) as shown in Fig. 5 or asymmetric (for directed graphs) as shown in Fig. 4. Nodes and edges are color coded based on their weight and the *color scheme* shown in Fig. 2.

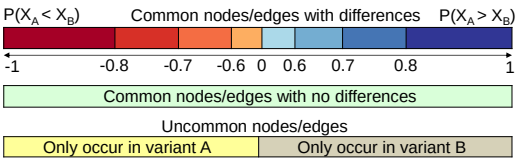

Fig. 2. Color scheme

4 Evaluation

We implemented our approach as a ProM plugin named *Multi-Perspective Process Comparator* (MPC).[2] The plugin allows one to import two event logs in MXML or

[2] Executable and source code are available from `http://apromore.org/platform/tools`

XES format as input, mine different perspective graphs and compare them to identify statistically significant differences. Using this implementation, we evaluated our approach on two real-life datasets and compared the results with the *ProcessComparator* (or PC) plugin in ProM [4].

We looked at the public real-life event logs available in the *4TU Data Center*[3] and selected two representative datasets, namely BPIC13 and BPIC15. These two datasets come with business questions that entail variants comparison, which have been posed by the process stakeholders of these datasets, as part of public contests on process mining. Due to space limits, we only report the result of our technique on the BPIC13 log focusing on aspects our technique can improve over the baseline. Detailed evaluation is documented in a technical report [12].

BPIC13[4] records cases of an IT incident handling process at Volvo Belgium. An IT ticket is raised for each incident to be investigated by various IT support teams. Teams are organized into technology-wide functions (*org:role* attribute), organization lines (*organization involved* attribute), and countries (*resource country* attribute). For our evaluation, we selected the following question from the description accompanying this dataset: "*Where do the two IT organisations (A2 and C) differ?*" where *A2* and *C* are the main organization lines responsible for most of the IT tickets.

For each dataset and business question above, we compared process variants using three sub-questions. With reference to Fig. 1, these questions are focused on two levels of granularity, event and fragment, and time-wise differences.

Q1. What are the differences at the event level?

At the event level, we can look into either inter-event or intra-event relations where each event is a fragment. Regarding the former, both PC and MPC can provide the same insight. Specifically in the case of MPC, we can use the *event ID* attribute to create a process abstraction, then create an inter-fragment graph using the pair of *event name* and *status* attributes as nodes.

However, regarding the intra-event relations, PC cannot provide a solution while MPC can investigate these relations through intra-fragment graphs. For example, on the event-based abstraction, we can use the *country* attribute as a node and the *activity status* attribute as another node. The matrix in Fig. 3 reveals that the teams in Brazil, India and the USA in the organization C choose the "Wait User" status for IT tickets more frequently than in the organization A2. This is an operational concern since IT staff can choose this status as an excuse to delay incident investigation.

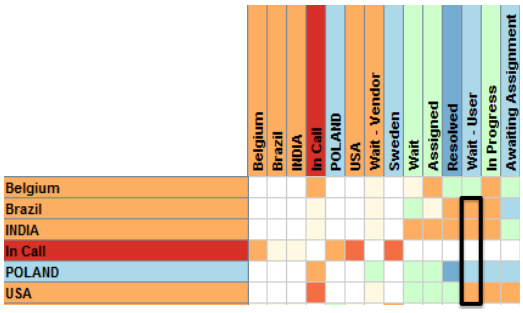

Fig. 3. Resource Country and Activity Status

Q2. What are the differences at the fragment level?

In the BPIC13 dataset, we can create fragments using the *country* attribute to look into how process activities are related between IT teams from different countries. In this aspect, PC aggregates the activity flow between fragments. For example, PC shows a flow from [Sweden] to [Poland] through the Accepted activity meaning that this activity is performed by Sweden and then work is transferred to Poland. It may however consist of two possible flows: either Accepted by Sweden followed by activity Queued performed by Poland, or Accepted by Sweden followed by Completed performed by Poland.

[3] https://data.4tu.nl/repository/collection:event_logs_real
[4] doi:10.4121/uuid:500573e6-accc-4b0c-9576-aa5468b10cee

Similarly to PC, MPC can look into the same flow by first using the *country* attribute to create the process abstraction, then choosing the pair of *country* and *event name* attributes as node and the *country* attribute as another node to create an inter-fragment graph. However, beyond that, MPC can elaborate the activity flow between countries by using other event attributes. Fig. 4 shows an example where we chose *impact*, *country* and *event name* to represent a node. From this figure, we can see that the process activity flow from Sweden to Poland through the "Accepted" activity is actually the control flow from "Medium_Sweden_Accepted" to "Medium_Poland_Accepted" (i.e. from "Accepted" to "Accepted" activities for medium impact cases only).

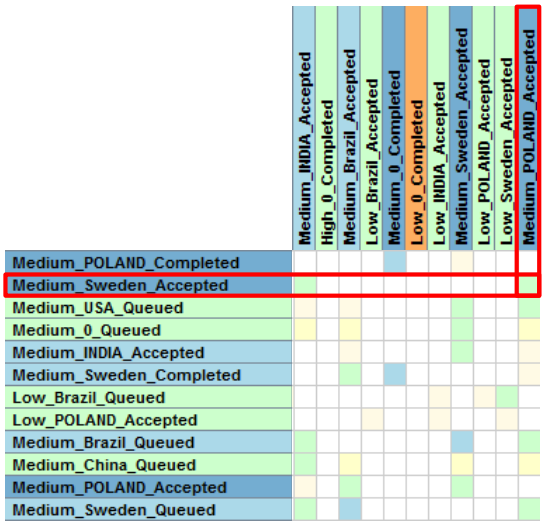

Fig. 4. Impact, Country, and Event Name

Further, MPC can look into the differences between A2 and C within each fragment through intra-fragment graphs, while this is not possible with PC. For example, we use the *impact* attribute to create a process abstraction, and the pair of *impact* and *activity status* attributes as the node. The result is shown in Fig. 5 for medium impact incidents. Remarkably, we can see that for medium-impact incidents, most of activity statuses in A2 have approximately 60-70% chance of occurring more frequently than in C. There are no significant differences between the two organization lines in high-impact incidents.

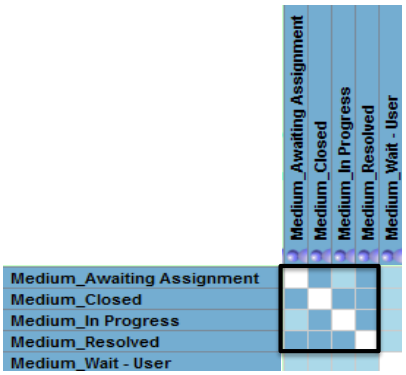

Fig. 5. Medium Impact and Activity Status

Q3. What are the time-wise differences as compared to case-wise differences?

So far, the evaluation only finds case-wise differences, i.e. differences synthesized from cases in A2 and C. Time-wise observations, however, are not available in PC.

For MPC, we use time-wise observations with a sliding window set to three days as most of events in a case occur within a day. We use the *event ID* to create a process abstraction, and the pair of *event name* and *activity status* as node. In this case, the node (edge) weight captures their relative occurrence frequency in each window. The result is shown in Fig. 6. We can see that there are two remarkable differences between A2 and C over time in the node "Accepted_Wait-User" and the edge "Accepted_In Progress" → "Accepted_Wait-User". The difference magnitude is approximately 56%, i.e. there is a 56% probability that "Accepted_Wait-User" in A2 has lower frequency than in C. In MPC, clicking on the edge "Accepted_In Progress"

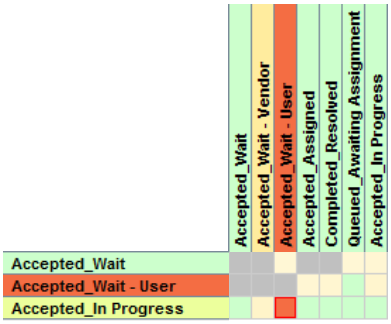

Fig. 6. Event Name and Activity Status

→ "Accepted_Wait-User" views detailed time series which shows that this difference mostly occurred between 21 Jan and 10 Mar 2012.

5 Conclusion

This paper contributes the notions of perspective graph and differential graph. A perspective graph is an abstraction of an event log in which nodes represent entities referenced by an event attribute or combination of attributes, and links refer to co-occurrence or directly-follows relations. Perspective graphs generalize directly-follows graphs and hand-off graphs, commonly supported by process mining tools. Differential perspective graphs allow us to compare two event logs (abstracted via perspective graphs) and to identify their statistically significant differences.

The example-based evaluation of differential perspective graphs on real-life logs shows that we can identify differences that are beyond the scope of the existing ProcessComparator approach, and that the matrix-based representation of differential perspective graphs provides a more compact representation for displaying such differences, compared to node-link (graphical) representations used in process mining tools.

While the examples highlighted the possible advantages of the proposed approach, these need to be confirmed via a usability evaluation with end users, which is left as future work. Another future work avenue is to extend the approach in order to identify differences between variants that can be causally related to performance, e.g. structural or behavioral differences that can explain differences in cycle time between variants.

Acknowledgements. This research is partly funded by the Australian Research Council (DP150103356) and the Estonian Research Council (grant IUT20-55).

References

1. H. Nguyen, M. Dumas, M. La Rosa, F. Maggi, and S. Suriadi. Mining business process deviance: a quest for accuracy. In *Proc. of CoopIS*. Springer, 2014.
2. N.R.T.P. Van Beest, M. Dumas, L. García-Bañuelos, and M. La Rosa. Log delta analysis: interpretable differencing of business process event logs. In *Proc. of BPM*. Springer, 2015.
3. M. Wynn, E. Poppe, J. Xu, A.H.M. ter Hofstede, R. Brown, A. Pini, and W.M.P. van der Aalst. ProcessProfiler3D: A visualisation framework for log-based process performance comparison. *DSS*, 100:93–108, 2017.
4. A. Bolt, M. de Leoni, and W.M.P. van der Aalst. Process variant comparison: using event logs to detect differences in behavior and business rules. *Inf. Syst.*, 2018.
5. A. Pika, W.M.P. van der Aalst, C. Fidge, A.H.M. ter Hofstede, and M. Wynn. Profiling event logs to configure risk indicators for process delays. In *Proc. of CAiSE*. Springer, 2013.
6. J. Gulden. Visually comparing process dynamics with Rhythm-Eye views. In *Proc. of BPM Workshops*. Springer, 2016.
7. A. Pika, M. Wynn, C. Fidge, A.H.M. ter Hofstede, M. Leyer, and W.M.P. van der Aalst. An extensible framework for analysing resource behaviour using event logs. In *CAiSE*. Springer, 2014.
8. N. Ballambettu, M. Suresh, and R. Bose. Analyzing process variants to understand differences in key performance indices. In *Proc. of CAiSE*. Springer, 2017.
9. A. Bolt and W.M.P. van der Aalst. Multidimensional process mining using Process Cubes. In *Proc. of BMMDS/EMMSAD*. Springer, 2015.
10. IEEE standard for eXtensible Event Stream (XES) for achieving interoperability in event Logs and event streams. *IEEE Std 1849-2016*, pages 1–50, 2016.
11. K.O. McGraw and S.P. Wong. A common language effect size statistic. *Psychological Bulletin*, 111(2):361–365, 1992.
12. H. Nguyen, M. Dumas, M. La Rosa, and A.H.M. ter Hofstede. Multi-perspective comparison of business process variants based on event Logs (extended paper). QUT ePrints Technical Report, Queensland University of Technology, 2018 (http://eprints.qut.edu.au/117962).