Analysis of Business Process Variants in Apromore

Raffaele Conforti[1], Marlon Dumas[2,1], Marcello La Rosa[1,3], Abderrahmane Maaradji[3,1], Hoang Huy Nguyen[1], Alireza Ostovar[1], and Simon Raboczi[1]

[1] Queensland University of Technology, Australia
{raffaele.conforti, m.larosa, alireza.ostovar, simon.raboczi}@qut.edu.au
huanghuy.nguyen@hdr.qut.edu.au
[2] University of Tartu, Estonia
marlon.dumas@ut.ee
[3] NICTA Queensland Lab, Australia
abderrahmane.maaradji@nicta.com.au

Abstract. In this paper we illustrate a set of features of the Apromore process model repository for analyzing business process variants. Two types of analysis are provided: one is static and based on differences on the process control flow, the other is dynamic and based on differences in the process behavior between the variants. These features combine techniques for the management of large process model collections with those for mining process knowledge from process execution logs. The tool demonstration will be useful for researchers and practitioners working on large process model collections and process execution logs, and specifically for those with an interest in understanding, managing and consolidating business process variants both within and across organizational boundaries.

1 Overview

Understanding the differences between variants of the same business process, a.k.a. *business process variants analysis*, has manifold applications. It is the starting point of any business process consolidation and standardization initiative, where similar variants can be replaced with a consolidated or standardized process to reduce running and infrastructure costs, e.g. the costs of a supporting IT infrastructure. It is also useful for identifying root causes of performance issues, e.g. those affecting certain variants that negatively *deviate* from a normative process specification. Finally, variants analysis can provide input to process conformance and compliance checking.

In this paper we describe a set of plugins of the Apromore platform which can be combined in a toolchain to support users in analyzing business process variants. Apromore [4] is an open and extensible repository which provides a range of dedicated services for the management of large process model collections.

To illustrate the features of these plugins, let us consider the case of a financial firm offering loan services through multiple branches, distributed over different states of a country. Due to local regulations, the loan application process will manifest itself in different variants, one per state.

One key characteristic of our variants analysis approach is its evidence-based orientation, i.e. we take the process execution logs (event logs for short) of the various process variants as the starting point for our analysis. Coming back to our loan application example, after collecting the event logs for each state, we first need to extract the sublog representing how the process is currently executed in each state. In fact, an event

log typically records the evolution of a process over time so restricting the analysis to the current process behavior (i.e. the last "stable" process behavior) is required to obtain meaningful results when comparing process variants. This means extracting the sublog since the last *process drift*, i.e. the last significant change in the process behavior. This can be achieved by the ProDrift plugin of Apromore [5]. This plugin implements a fully automated and scalable method for detecting process drifts by monitoring the process behavior recorded in an event log. The core idea is to perform statistical hypothesis testing over the distributions of runs observed in two consecutive time windows. The underlying assumption is that if a change occurs at or around a given time, the distribution of runs before and after this time point will be statistically different.

Alternatively, one could use process model change logs to identify the last stable process behavior. However, here we do not assume the existence of any manually-built process model, as often in reality process models, even when available, are hardly up-to-date [6].

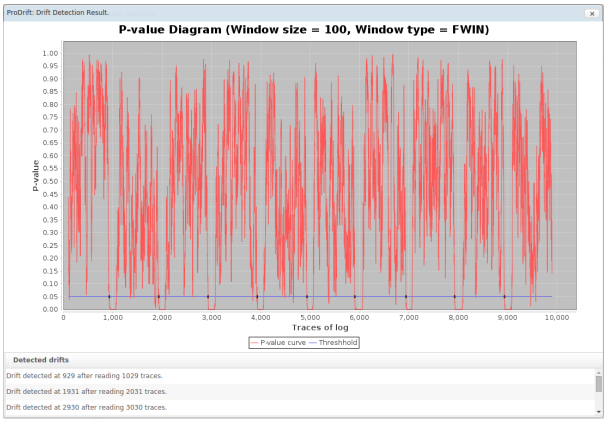

Fig. 1. ProDrift plugin showing the drifts identified in an event log

The plugin accepts a log in XES or MXML format, performs the statistical test and detects with high accuracy the drifts in the log (see Fig. 1). The user is then offered to extract the sublogs between any two drifts. For our purposes, we are only interested in the sublog since the last drift as this represents the last stable process behavior.

Next, we need to discover a process model from the sublog of each variant. For this we use the BPMN Miner [1] plugin. This plugin is able to discover a hierarchically structured BPMN process models from an event log. It relies on approximate functional and inclusion dependency discovery techniques to infer a process-subprocess hierarchy from an event log and on a set of heuristics to identify boundary events and activity markers. The plugin takes as input a process log in XES or MXML format, a baseline automated process model discovery algorithm (Heuristics, Fodina, Inductive, ILP and Alpha supported) and a set of thresholds for the identification of boundary events and activity markers (see Fig. 2.a). After selecting the set of attributes to use for the identification of primary keys, the plugin assigns the most likely attribute to each activity. This assignment can then be modified by the user, who can remove any false positive identified (see Fig. 2.b). The final result is a BPMN model which can be saved in the repository.

Afterwards, we can merge all discovered BPMN models using the Process Merge [3] plugin to create a consolidated model. This model uses an algorithm based on maximum

common regions to guarantee that the consolidated model is the union of the behaviors of all variants. The result is a *configurable* BPMN (C-BPMN) model, where commonalities are distinguished from variant-specific fragments via a notion of *variation point*, i.e. a gateway represented with a thicker border that indicates where a common fragment branches out into, or joins back from, variant-specific fragments.

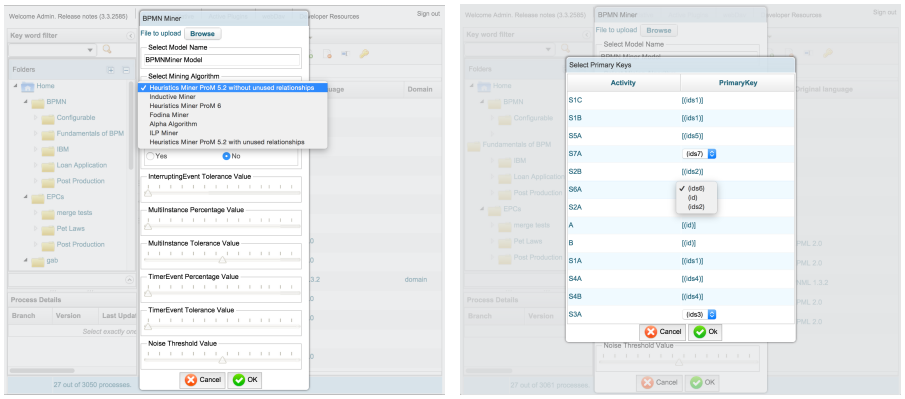

Fig. 2. BPMNMiner: (a) configuration window; (b) selection of candidate keys

Apromore offers two analysis capabilities on top of this consolidated model: a *static* one, concerning the process structure, and a *dynamic* one, concerning the process behavior. First, by exploiting the structure of the configurable model, the user can highlight variant specific fragments as well as common fragments by coloring the involved nodes, e.g. one color per variant and one color for the common parts (see see Fig. 3). Further, the user can create a *digest*, i.e. a projection of the model control flow to a specific variant or set thereof, in which case the parts not relevant will be greyed out. Highlighting can also be governed by frequency, e.g. highlighting all elements of the model participating in at least 2 but not more than 4 variants. Highlighting is obtained by modifying the underlying SVG of the process model shown in the Apromore Editor.

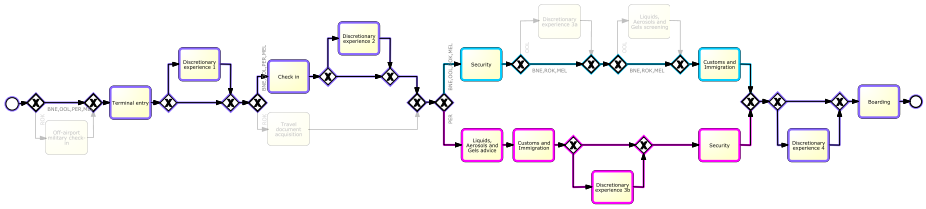

Fig. 3. Model projection and coloring on a C-BPMN model

The dynamic analysis capability is offered by the LogReplay plugin. This plugin can replay one or multiple logs simultaneously on top of a (C-)BPMN model (see Fig. 4), where the progress of each log is indicated by a set of colored tokens (one color per log, one token per case). By animating multiple logs at the same time, it is possible to gain insights on the behavioral differences between the process variants under analysis, e.g. paths or activities that are frequent in one variant and not in the other.

The LogReplay plugin relies on a Java Servlet running on the Apromore server, which builds an animation file, and on a Web client, which plays the animation on top of a BPMN process model open in the Apromore Editor. The communication between

client and server is via AJAX with content transferred in JSON format. The input to the tool is one or more event logs in XES or MXML format. The output is an SVG animation file. The plugin relies on a range of heuristics to efficiently compute the animation file, including: costs of various moves on the log and moves on the model, depth of model traversal, percentage of model-log fitness, number of nodes explored on the model, number of consecutive mismatches, vicious cycle checking option, approximate or exact fitness calculation option, and time limit for replaying. The user can configure the speed of animation and move the animation backward or forward in time, including step-wise movements at given time intervals. A timeline and pie chart show the progress of each log replay.

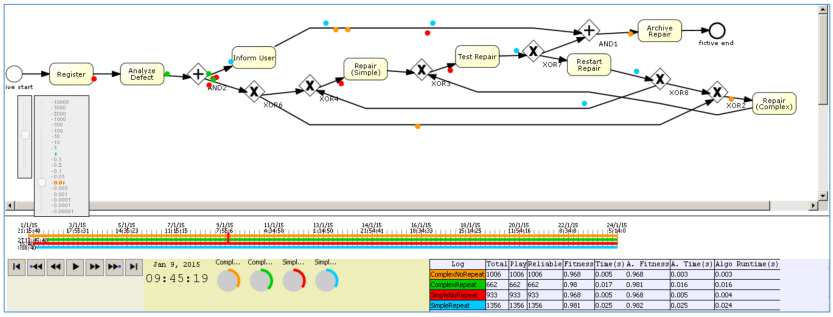

Fig. 4. LogReplay plugin replaying four logs on top of a C-BPMN model

2 Significance and Maturity

Our toolchain embodies a novel approach for identifying and analyzing variants of a business process over a logical dimension (e.g. different variants of a process per product, brand or location) or over time (e.g. different variants reflecting the evolution of a given process). The same approach can be applied in intra-organizational settings, as well as in cross-organizational settings, e.g. in the public sectors or within domain-specific consortia of organizations, i.e. where there is a benefit in sharing business practices. The approach is innovative as it combines for the first time features for managing process model collections with those from process mining, to offer two types of complementary analyses: a static one based on differences in the process control flow, and a dynamic one and based on differences in the process behavior.

The accuracy and scalability of the Apromore features used in this approach, except from those of the LogReplay plugin, have been extensively evaluated using both artificial and real-life process model collections and event logs. The results of these evaluations are reported in [5] (ProDrift), [1] (BPMN Miner), [2] (Process Similarity) and [3] (Process Merger and model projection).

These features are part of a larger ecosystem of advanced capabilities for managing process model collections provided by the Apromore repository. These include *design* capabilities such as automated discovery, merging and questionnaire-driven configuration; *filtering* capabilities such as structure-based similarity search and clone detection, and behavior-based model comparison and querying; *evaluation* capabilities such as simulation and log replay; and *presentation* capabilities such as model restructuring. In addition, Apromore provides basic repository functionality such as import/export and SVN-like version control.

The repository, in its version 3.3 at the time of writing, is the result of more than five years of development and improvements. It is implemented via a three-level service-oriented architecture where the logic layer, blended between the data and the presentation layers, hosts all the plugins and features of the platform; the presentation layer provides connections with an internal visual editor based on Signavio CoreComponents, and links to various external environments (ProM, WoPeD, BAB and YAWL); and the data layer controls access to the files stored in the repository. These files are mainly process models, which are stored and indexed in a relational database using Apromore's internal canonical format for efficient querying, as well as in their native format (all major formats are supported, including BPMN, XPDL, EPML, PNML and YAWL). Apromore can also store files of any format, e.g. event logs, via a dedicated WebDAV.

The repository is publicly accessible over the Web as a Software as a Service (SaS). It uses the latest J2EE technology, including Spring and Maven for development, OSGi for its plugin architecture, Eclipse Virgo as its OSGi-based application server and ZK for its AJAX-based Web interface. The public release of Apromore counts over 150 active users, including both practitioners and academics (e.g. several PhD students around the world are using Apromore for their research). The development has been supported by various grants, including funding from Suncorp, the largest insurance company in Australia, and from NICTA, Australia's national ICT research center.

3 Screencasts and links

Using the financial firm scenario, in the demonstration we will showcase two use cases for variants analysis: one where variants of the loan application process are determined by the states in which the firm operates, the other where variants are identified by the drifts of this process over time, in the context of a specific branch. The screencasts for these two use cases are available at `http://youtu.be/3sxBTtHqBLI` and `https://youtu.be/wrbelRtlOGs`.

The public release of Apromore is available at `http://apromore.qut.edu.au` while its source code can be downloaded under the GNU LGPL license version 3.0 at `https://github.com/apromore/ApromoreCode`.

References

1. R. Conforti, M. Dumas, L. García-Bañuelos, and M. La Rosa. Beyond tasks and gateways: Discovering bpmn models with subprocesses, boundary events and activity markers. In *Business Process Management*, volume 8659 of *LNCS*, pages 101–117. Springer, 2014.
2. R.M. Dijkman, M. Dumas, B.F. van Dongen, R. Käärik, and J. Mendling. Similarity of business process models: Metrics and evaluation. *Inf. Syst.*, 36(2):498–516, 2011.
3. M. La Rosa, M. Dumas, R. Uba, and R. M. Dijkman. Business process model merging: An approach to business process consolidation. *ACM Trans. Softw. Eng. Methodol.*, 22(2):11, 2013.
4. M. La Rosa, H. A. Reijers, W. M. P. van der Aalst, R. M. Dijkman, J. Mendling, M. Dumas, and L. García-Bañuelos. APROMORE: an advanced process model repository. *Expert Syst. Appl.*, 38(6):7029–7040, 2011.
5. A. Maaradji, M. Dumas, M. La Rosa, and A. Ostovar. Fast and accurate business process drift detection. In *Business Process Management*, LNCS. Springer, 2015.
6. W.M.P. van der Aalst, M. La Rosa, A.H.M. ter Hofstede, and M.T. Wynn. Liquid business process model collections. In *Modeling and Simulation-Based Systems Engineering Handbook*, pages 401–424. CRC Press, 2014.