BPMN Miner 2.0: Discovering Hierarchical and Block-Structured BPMN Process Models

Raffaele Conforti[1], Adriano Augusto[1], Marcello La Rosa[1], Marlon Dumas[2], and Luciano García-Bañuelos[2]

[1] Queensland University of Technology, Australia
{a.augusto, raffaele.conforti, m.larosa}@qut.edu.au
[2] University of Tartu, Estonia
{marlon.dumas, luciano.garcia}@ut.ee

Abstract. We present *BPMN Miner 2.0*: a tool that extracts hierarchical and block-structured BPMN process models from event logs. Given an event log in XES format, the tool partitions it into sub-logs (one per subprocess) and discovers a BPMN process model from each sub-log using existing techniques for discovering BPMN process models via heuristics nets or Petri nets. A drawback of these techniques is that they often produce spaghetti-like models and in some cases unsound models. Accordingly, BPMN Miner 2.0 applies post-processing steps to remove unsound constructions as well as a technique to block-structrure the resulting process models in a behavior-preserving manner. The tool is available as a standalone Java tool as well as a ProM and an Apromore plugin. The target audience of this demonstration includes process mining researchers as well as practitioners interested in exploring the potential of process mining using BPMN.

Keywords: Process Discovery, BPMN, Structured Process Models, Hierarchical Process Models, ProM, Apromore

Process mining is a discipline within Business Process Management that aims to extract actionable insights from process execution logs [9]. Such execution logs, called *event logs* for short, can be extracted from common or special-purpose IT systems available in today's organizations, such as an ERP system or a Claims Management System.

One particular category of process mining methods is that of *automated process model discovery*. These methods extract a process model from an event log. Real-life logs, however, are typically affected by noise (e.g. in the form of infrequent behavior) and are incomplete (i.e. they do not contain all possible behavior of the business process under analysis). Consequently, process model discovery is not a trivial procedure and leads to process models of varying quality, which often approximate the actual business process, depending on the discovery algorithm that is employed.

A large number of automated process discovery algorithms have been proposed. These algorithms strike various tradeoffs between accuracy (measured in terms of fitness, precision and generalization [9]) and *complexity* (measured in terms of model size and other complexity metrics [7]). Some algorithms tend to discover process models with high accuracy at the cost of high model complexity. The importance of model

complexity should not be underestimated as this underpins the understandability of the extracted process model, and so ultimately its value to users. In this respect, empirical studies [5] have shown that besides model size, an important proxy for process model understandability is the *structuredness* of the model. This latter observation has led to the design of discovery algorithms such as the *Inductive Miner* [6] and the *Evolutionary Tree Miner* [2], which discover structured process models by design. Models discovered using such algorithms may be further away from reality depending on the degree of unstructuredness of the actual business process the log refers to. For example, the Inductive Miner tends to over-generalize the behavior in the log, leading to low precision while maximizing fitness [1]. On the other hand, discovery algorithms such as the *Heuristics Miner* [11] and the *Fodina Miner* [10] tend to strike better results in terms of accuracy but produce more complex and sometimes syntactically incorrect and unsound process models [1]. Furthermore, the majority of discovery algorithms produce models that are represented in languages that are either not widely accepted among practitioners (e.g. Petri nets), too technical (e.g. Heuristics nets) or too abstract and over-generalizing (e.g. fuzzy nets or process maps).

The *BPMN Miner* algorithm [3, 4] is designed to address the above limitations. This algorithm discovers models in the BPMN 2.0 language [8], a *de jure* standard widely supported by vendors and practitioners. Moreover, by exploiting implicit functional and foreign-key dependencies between attributes in the event log, the algorithm can generate hierarchical BPMN models, i.e. models organized over two or more abstraction levels via the use of subprocess models. In order to further reduce the complexity of the discovered models, the algorithm exploits an extensive set of notational elements provided by the BPMN language, such as boundary events (to model interrupting exceptions), activity markers (loops and multi-instance) and event subprocesses.

BPMN Miner relies on existing (*baseline*) automated process discovery algorithms to generate an initial model of each subprocess. The baseline algorithms supported are Inductive Miner, Heuristics Miner, Fodina, ILP Miner and Alpha Miner. Version 2.0 of BPMN Miner additionally embeds an algorithm that discovers sound and maximally-structured BPMN models, namely the *Structured Miner* [1]. This latter algorithm removes unsound and unstructured constructs in the discovered model, thus further increasing its potential usability.

The minimum input required by the tool is the log from which to discover a BPMN model. By default, the Heuristics Miner is chosen as the baseline discovery algorithm. It is however possible to customize a number of input parameters in order to tune the results (see Figure 1).

Besides the baseline discovery algorithm, one can choose whether the partitioning of the log into sublogs is achieved via a noise tolerant dependency discovery algorithm, or not, in which case the log is assumed to be noise-free. Additionally, it is possible to sort the input log based on the timestamp of its events, and to switch on the structuring of the discovered process model. The latter function is only effective when the baseline discovery algorithm does not already produce a structured model by design.

Additional parameters can be set to fine-tune the discovery of BPMN-specific notational elements, on the basis of a number of heuristics. For example, one can set tolerance levels for the identification of boundary timer and message events, and of multi-instance activity markers.

Once these parameters are set, the algorithm retrieves event attributes which may be used as primary keys for the partitioning of the log into sublogs. In this context, a primary key is an attribute which is recurrent in all events that are related to the activities of a particular subprocess. For example, an attribute "invoice" would be recurrent across all events related to the handling of the invoice, as part of an overarching order-to-cash process. All such events that are related to the handling of the invoice would be isolated in a separate sublog, from which the corresponding subprocess for handling invoices will then be discovered. The user has the possibility of steering the use of particular event attributes by selecting/deselecting them from a list that is automatically populated by the tool.

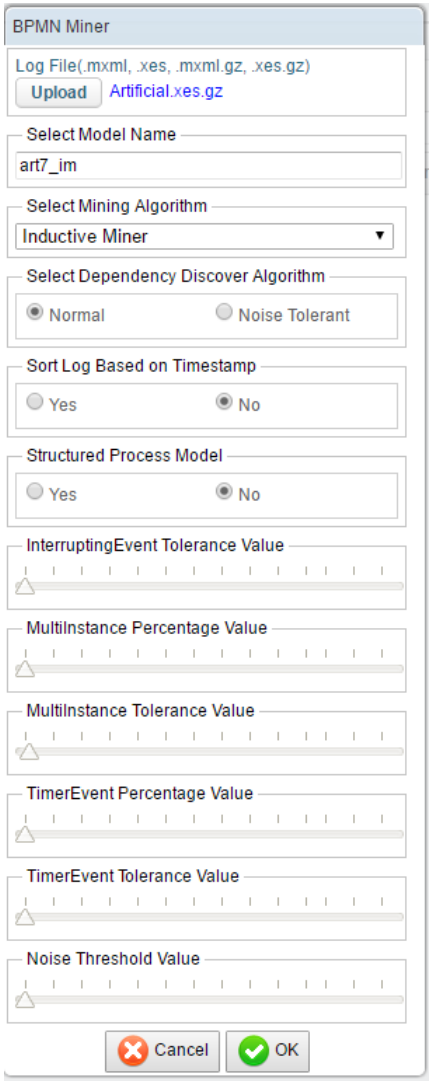

Fig. 1. Parameters.

Next, the algorithm assigns a specific primary key to each sublog. If more than one primary key can be assigned to the same sublog, the user is asked to choose from a droplist, where the first key is the most fitting one. At this point the event log is partitioned into sublogs using the chosen primary keys, and each sublog is passed as input to the baseline discovery algorithm for model discovery. When the discovery has completed, each subprocess model is structured separately, if this option is enabled, and assembled together as part of a single hierarchical BPMN model.

Figure 2 shows an example of hierarchical process model discovered by BPMN Miner, which was fully structured after the discovery. This model exhibits two expanded subprocesses (one with loop marker, the other with multi-instance marker) and one event subprocess. Figure 3 shows an example of hierarchical process model discovered with BPMN Miner in ProM. This latter model is maximally structured and exhibits two expanded subprocesses (again, with loop and multi-instance markers), two event subprocesses (one nested within a subprocess), and boundary timer and message events with exception flows. Both models were discovered from synthetic log of an order-to-cash process.

The accuracy and scalability of BPMN Miner have been extensively evaluated using over 600 event logs, including both artificial and real-life event logs. The majority of the artificial logs were generated from the SAP R/3 and IBM BIT process model collections, which group models from a variety of domains, including finance, sales,

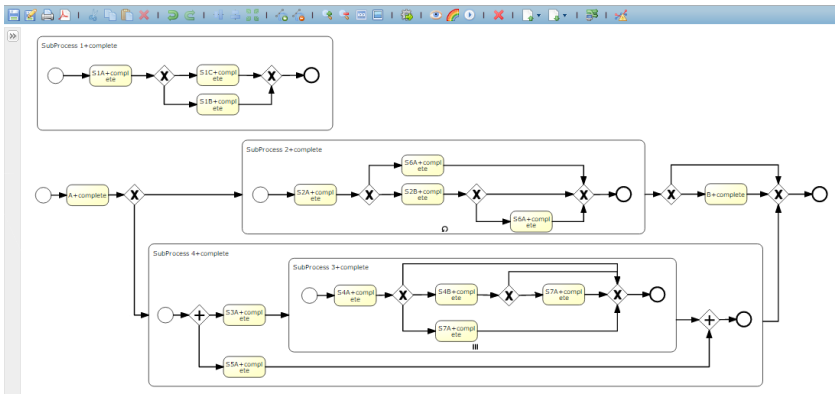

Fig. 2. Example of fully-structured process model discovered by BPMN Miner in Apromore.

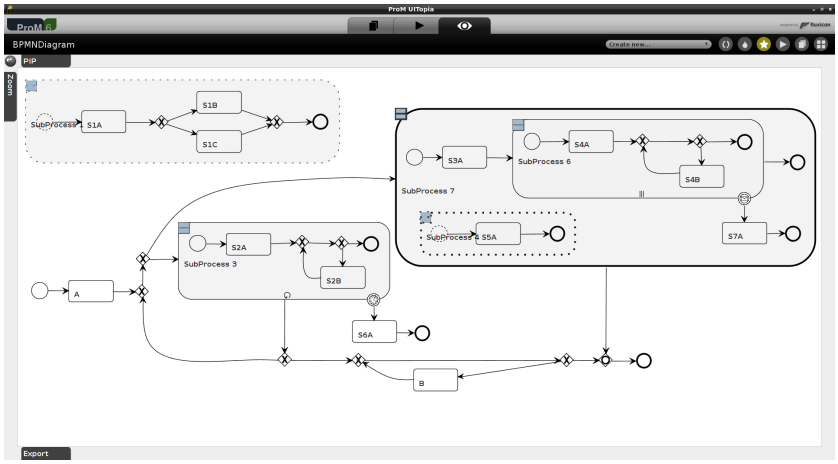

Fig. 3. Example of maximally-structured process model discovered by BPMN Miner in ProM.

accounting, logistics, communication and human resources. The results of these evaluations are reported in [3, 4, 1].

The results of the experiments show that the tool scales well to large and noisy real-life logs, performing within reasonable time bounds in the order of minutes. Moreover, the results indicate a statistically significant improvement of discovery accuracy and model complexity over all the baseline discovery algorithms supported by the tool.

BPMN Miner 2.0 is available as an OSGi plugin of the Apromore process model repository, as a plugin of the ProM Framework, as well as a standalone commandlineJava tool. Apromore is an online open-source ecosystem of advanced capabilities for managing large process model collections, including process modeling, simulation, filtering, querying, similarity search, behavioral comparison and model merging. ProM is the largest on-source process mining framework, offering over 300 plugins (in its latest incarnation) offering process model discovery, conformance checking, variants and deviance mining, and log analysis capabilities.

A screencast is available at https://youtu.be/eb0k2RO2PQ8. This video illustrates different examples and provides a brief explanation of the tool settings along with the possible outputs that can be obtained by varying the input parameters. BPMN Miner 2.0 is embedded as an OSGi plugin in the online platform Apromore, which has been used for the screencast (http://apromore.qut.edu.au). The artificial log used in the screencast is available at https://goo.gl/AdvnEd.

BPMN Miner is also available as a ProM plugin (http://promtools.org) and as a standalone Java tool (http://apromore.org/platform/tools).

References

1. A. Augusto, R. Conforti, M. Dumas, M. La Rosa, and G. Bruno. Automated discovery of structured process models: Discover structured vs. discover and structure. In *Proc. of ER*. Springer, 2016. Preprint available at http://eprints.qut.edu.au/95189/.
2. J. Buijs, B.F. van Dongen, and W.M.P. van der Aalst. On the role of fitness, precision, generalization and simplicity in process discovery. In *Proc. of CoopIS*, volume 7565 of *LNCS*, pages 305–322. Springer, 2012.
3. 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 *Proc. of BPM*, LNCS, 2014.
4. Raffaele Conforti, Marlon Dumas, Luciano García-Bañuelos, and Marcello La Rosa. Bpmn miner: Automated discovery of bpmn process models with hierarchical structure. *Information Systems*, 56:284–303, 2016.
5. M. Dumas, M. La Rosa, J. Mendling, R. Mäesalu, H.A. Reijers, and N. Semenenko. Understanding business process models: the costs and benefits of structuredness. In *Proc. of CAiSE*, volume 7328 of *LNCS*, pages 31–46. Springer, 2012.
6. S.J.J. Leemans, D. Fahland, and W.M.P. van der Aalst. Discovering block-structured process models from event logs - a constructive approach. In *Proc. of PETRI NETS*, volume 7927 of *LNCS*. Springer, 2013.
7. J. Mendling. *Metrics for Process Models: Empirical Foundations of Verification, Error Prediction, and Guidelines for Correctness*. Springer, 2008.
8. Object Management Group (OMG). *Business Process Model and Notation (BPMN) ver. 2.0*. Object Management Group (OMG), January 2011.
9. W.M.P. van der Aalst. *Process Mining - Discovery, Conformance and Enhancement of Business Processes*. Springer, 2011.
10. S.K.L.M. vanden Broucke, J. De Weerdt, J. Vanthienen, and B. Baesens. Fodina: a robust and flexible heuristic process discovery technique. http://www.processmining.be/fodina/. Last accessed: 03/27/2014.
11. A.J.M.M. Weijters and J.T.S. Ribeiro. Flexible Heuristics Miner (FHM). In *Proc. of CIDM*. IEEE, 2011.