Process and Deviation Exploration with Inductive visual Miner

Sander J.J. Leemans, Dirk Fahland, and Wil M.P. van der Aalst

Eindhoven University of Technology, the Netherlands
{s.j.j.leemans, d.fahland, w.m.p.v.d.aalst}@tue.nl

Abstract Process mining aims to extract information from recorded process data, which can be used to gain insights into the process. This requires applying a discovery algorithm and settings its parameters, after which the discovered process model should be evaluated. Both steps may need to be repeated several times until a satisfying model is found; we refer to this as *process exploration*. Existing commercial tools usually do not provide models having executable semantics, thereby disallowing for accurate map evaluation, while most academic tools lack features and by the repetitive nature of process exploration, their use is tedious. In this paper, we describe a novel process exploration tool: the *Inductive visual Miner*. It aims to bridge this gap between commercial and academic tools, by combining the executable semantics of academic tools with the exploration support of commercial tools. It also adds animation and deviation visualisation capabilities.

Keywords: Process mining, process exploration, deviation analysis

1 Process Exploration

To gain insights in business processes based on factual knowledge, recorded event data can be analysed using process mining. Process mining aims to extract information from recorded process data, stored in an event log, and starts with discovering a process model from the event log. However, many process discovery algorithms exist, their parameters have to be set, and the question at hand might require to focus on specific parts of the event log. The implications of these choices are, although well-studied for academic approaches, unclear for the average user, which makes it difficult to obtain a model that is suitable to answer the question at hand. In this paper we focus on *process exploration*, which is the process of repeatedly trying settings until a satisfactory model is discovered [4].

The first step to take in process exploration is to select a process discovery algorithm and to set its parameters. Moreover, the scope of the exploration needs to be set by applying all kinds of filters and choosing a perspective, e.g. one can focus on the control flow or resource perspective.

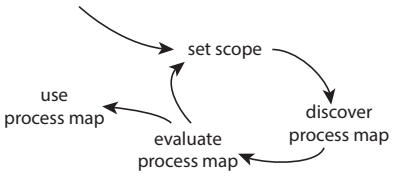


Figure 1: Exploration cycle.

In the second step of the exploration cycle, one needs to apply the algorithm in the selected scope to the event log to obtain a process model. Before conclusions can be drawn and insights can be gained, the model should be evaluated. For instance, compliance related questions, such as whether the four-eyes principle was adhered to, can only be answered if the model represents a large part of the behaviour in the event log, and future related questions should only be answered using models that are likely able to represent future behaviour. Evaluation of a model with respect to an event log can only be done accurately if the behaviour that the model allows is well-defined, i.e. if it has executable semantics, and different parts of the model might have different problems.

Often, general questions, such as what a process looks like, lead to more specific questions such as where in the process delays or deviations occur, or to questions that need to be answered using other perspectives on the event log. Or, the evaluation shows that the question cannot be answered with the discovered process model. Then, the parameters need to be set again and a new model must be discovered; process exploration is a highly iterative process.

After a user has found a suitable model, that model can be used in for instance automatic enactment of models in systems [5], in automatic prediction [7] and in compliance checking [6]. The full process exploration cycle is shown in Figure 1. All of these uses for process models require that the model can be processed automatically, for which it needs to have executable semantics.

Current commercially available process exploration tools offer plenty of options to set the scope of the exploration, but usually do not produce models having executable semantics, which thus cannot be used for automated evaluation or further use. There is plethora of academic tools available to set the scope of the exploration, to discover a process model and to evaluate it, but given the nature of process exploration, using them iteratively is tedious. In this paper, we introduce a tool, *Inductive visual Miner* (IvM), that aims to bridge this gap between commercial and academic tools. It supports the steps of process exploration by chaining existing academic tools and streamlining their use. Moreover, it improves on evaluation by a new notation and the addition of animation and quick node selection filtering. Thus far, such capabilities only existed for tools having no or just weak semantics or without formal guarantees (Fuzzy Miner, Disco, BPM|One, Celonis, Perceptive, etc.).

IvM has been implemented as a plug-in of the ProM framework, which can be obtained by installing ProM 6.4 from `http://promtools.org` and, using the ProM package manager, installing the plug-in Inductive visual Miner. Example event logs can be obtained from `http://www.processmining.org/logs/start`; a screencast is available at `http://vimeo.com/user29103154/inductivevisualminer`.

In the remainder of this paper, we explain the implementation of IvM, highlight the deviation visualisation and give an example. For a detailed comparison with existing exploration approaches, please refer to [4].

2 Inductive visual Miner: Implementation

The architecture of IvM resembles a chain of analysis and visualisation tasks, shown in Figure 2. To encourage exploration, a user can change any parameter at any time. IvM will ensure that the current computation is discarded and the chain is restarted from the first task that is influenced by the parameter change. For instance, if the user selects or deselects a node, only the tasks 'filter node selection' and 'animate' are redone. As especially the align task can take some time, intermediate visual results are shown to the user until the next task is finished.

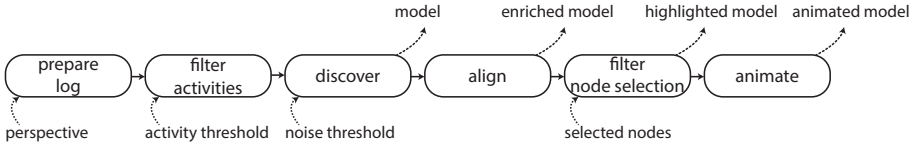


Figure 2: Chain of tasks, their parameters (bottom) and their visual results (top). If a user changes a parameter, the necessary tasks restart immediately.

In the *prepare log* task, the events in the log are classified using the provided perspective classifier. Next, in the *filter activities* task, given a threshold value, the most-frequent activities are kept, the events of other activities are filtered out. The Inductive Miner - infrequent (IMi) [3] discovery algorithm is applied in the *discover* task. IMi takes as an input parameter the amount of noise filtering to be applied to paths and produces a process tree. In the *align* task, the traces of the event log are aligned to find the best matching runs through the model (needed in case of deviations between model and log) [1]. This provides the information needed to enrich the model with information how often model elements were executed in the event log. The *filter node selection* task filters the aligned traces to keep only those that go through a selected node. The final task, *animate*, computes when traces passed model elements; this information is used to show a quick animated preview of traces in the log onto the model[1]. If the log contains no timestamps, random timestamps are inserted for demonstration purposes.

Once the model is available, it can be exported to ProM for further analysis, both as a Petri net and as a process tree; a user can perform its own evaluation without waiting for the evaluation of IvM to finish. At any point during the exploration, the model can be saved as bitmap (png) and vector (pdf, svg) image formats. The full animation of the complete log can be exported to bitmap (avi) and vector (svg) based movie formats once it is computed.

[1] At time of writing, we limited the quick preview to 50 traces for performance reasons.

Deviations. Deviations are a crucial part of the evaluation: they show precisely what parts of the model deviate with respect to the log. Deviations are visualised to show shich parts of the model fit well and which parts do not. This is important for drawing reliable conclusions. Two types of deviations have been identified [1]: if a trace contains an event that is not allowed by the model, it is a *log move*; if the model requires an event that is not present in the trace, it is a *model move*. Log and model moves are identified by the align task, that chooses a run through the process model such that the number of such deviating moves is minimal. As shown in Figure 3, IvM visualises both of them using dashed red edges; such an edge that circumvents an activity represents a model move, while a self-edge represents a log move.

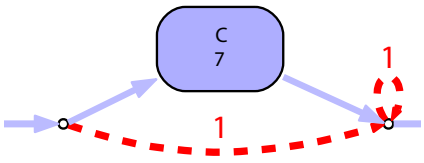
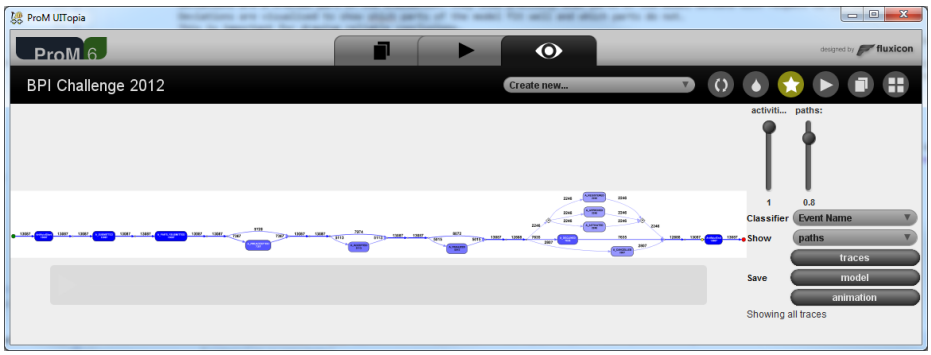
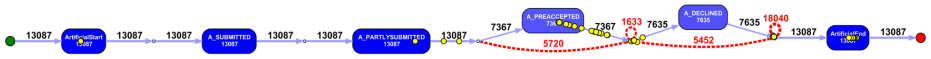


Figure 3: Model with the result of the align task. The edge circumventing C denotes a model move; the self-edge on the right a log move.

Example. Figure 4 shows the initial model with default values for all parameters. Looking at this model, the question rose what the happy flow of the process was, i.e. the most frequently taken path. After a few iterations, parameters were settled: using only the 50% most frequent activities and applying noise filtering of 20%, a happy flow of 6 activities was uncovered. Before exporting this model for further analysis, the deviation visualisation was turned on, resulting in the model shown in Figure 4b. This shows that the fourth and fifth activity are often skipped.



(a) Default parameters.



(b) After a few iterations; with deviations and animation.

Figure 4: Screenshot of IvM applied to 'A' activies of [2]; default parameters.

3 Conclusion

In this paper, we discussed the cycle of process exploration, consisting of repeatedly setting parameters, discovering a process model and evaluating it. We identified a gap between existing commercial and academic process exploration tools: commercial tools usually do not provide models having executable semantics, thereby disallowing for accurate map evaluation, while most academic tools lack features such as seamless zooming and animation, thus do not support the repetitive nature of process exploration well.

We introduced a process exploration tool, Inductive visual Miner (IvM), that aims to bridge this gap. When started, IvM immediately applies a chain of analysis and visualisation tasks to show the user not only a model, but also the traces of the event log animated on it, and where the log and model deviate from one another. IvM encourages the user to interact by enabling setting parameters at anytime: computations will be restarted as necessary in the background. IvM is not as feature-rich as some of the commercial tools, but shows that it is possible to use powerful techniques with formal guarantees in a user-friendly package. We hope that IvM will inspire commercial vendors to consider models with executable semantics and support deviation analysis. Extensions to IvM can be made in all tasks, for instance other process tree discovery algorithms can be plugged in instead of IMi.

In the future, we'd like to include approximation algorithms to compute the alignments in order to speed it up. To allow for even better evaluation, several extensions are possible, such as global quality measures (fitness, precision and generalisation) and identification of traces in the animation. Furthermore, several other filters such as filters on specific activity, timestamp, resource and on data could be included to give a user more freedom in setting the scope.

References

1. Adriansyah, A.: Aligning Observed and Modeled Behavior. Ph.D. thesis, Eindhoven University of Technology (2014)
2. van Dongen, B.: BPI Challenge 2012 Dataset (2012), `http://dx.doi.org/10.4121/uuid:3926db30-f712-4394-aebc-75976070e91f`
3. Leemans, S., Fahland, D., van der Aalst, W.: Discovering block-structured process models from event logs containing infrequent behaviour. In: Business Process Management Workshops. pp. 66–78 (2013)
4. Leemans, S., Fahland, D., van der Aalst, W.: Exploring processes and deviations. In: Business Process Management Workshops (2014), to appear
5. Meyer, A., Pufahl, L., Fahland, D., Weske, M.: Modeling and enacting complex data dependencies in business processes. In: BPM. Lecture Notes in Computer Science, vol. 8094, pp. 171–186. Springer (2013)
6. Ramezani, E., Fahland, D., van der Aalst, W.: Where did I misbehave? Diagnostic information in compliance checking. In: BPM. Lecture Notes in Computer Science, vol. 7481, pp. 262–278. Springer (2012)
7. Wynn, M., Rozinat, A., van der Aalst, W., ter Hofstede, A., Fidge, C.: Process mining and simulation. In: Modern Business Process Automation, pp. 437–457. Springer (2010)