A Framework for Interactive Multidimensional Process Mining

Thomas Vogelgesang[1], Stefanie Rinderle-Ma[2], and H.-Jürgen Appelrath[1]

[1] Department of Computer Science, University of Oldenburg, Germany
{thomas.vogelgesang | appelrath}@uni-oldenburg.de
[2] Faculty of Computer Science, University of Vienna, Austria
stefanie.rinderle-ma@univie.ac.at

Abstract. The emerging concept of multidimensional process mining adopts the ideas of data cubes and OLAP to analyze processes from multiple views. Analysts can split the event log into a set of homogenous sublogs according to its case and event attributes. Process mining techniques are used to create an individual process model for each sublog representing variants of the process. These models can be compared to identify the differences between the variants. Due to the explorative character of the analysis, interactivity is crucial to successfully apply multidimensional process mining. However, current approaches lack interactivity, e.g., they require the analyst to re-perform the analysis steps after changing the view on the data cube. In this paper, we introduce a novel framework to improve the interactivity of multidimensional process mining. As our main contribution, we provide a generic concept for interactive process mining based on a stack of operations.

Keywords: Interactive Process Mining, Multidimensional Process Mining, Process Cubes

1 Motivation

Process mining comprises a set of techniques for the automated analysis of processes [2]. They are based on collections of events (event logs) which are recorded during the execution of the process. Each event represents the execution of an activity and refers to a case representing a particular instance of the process. Typically, the events of an event log are grouped and chronologically ordered by their case representing the trace of a process instance. Additionally, events and cases may have arbitrary attributes to describe their specific properties.

Most process mining techniques focus on process discovery which aims to generate a descriptive and representative process model from a given event log. However, there are also other kinds of process mining techniques. Process enhancement maps the additional attributes of the event log onto a given process model to add further perspectives (e.g. waiting times). Conformance checking techniques compare process models to event logs, e.g. to measure its quality.

The notion of multidimensional process mining (MPM) [13,15,6] is an emerging concept to analyze variants of the same process. It adopts the idea of data

cubes and Online Analytical Processing (OLAP) [7] from the data warehouse domain to the field of process mining and organizes the event data by its attributes which are considered as dimensions. While traditional process mining techniques typically only consider a single event log resulting in one overall process model, MPM extracts an individual subset of an event log (sublog) for each variant. Analyzing these sublogs with process discovery returns a set of process models which can be compared to each other in order to identify differences between the process variants. Providing OLAP operators like drill-down and slice on the event data, MPM enables the analyst to define arbitrary views on the event log and to iteratively explore the processes.

In contrast to static reporting, OLAP aims to interactively explore the data. The analysis typically consists of an individual sequence of queries where each query results from the preceding ones. Therefore, OLAP requires flexible and effective user interfaces which are easy-to-use [7]. Due to this explorative character, interactivity is also vital for MPM [13]: To get the desired insights, the analysts need to refine the OLAP query for many times (e.g., filter or aggregate events, change dimension granularity) and apply different process mining techniques like process discovery, process enhancement and conformance checking. Even though they provide interactive user interfaces, current tools for MPM lack interactivity with regard to four aspects:

1. *Interaction with process models:* Process models as the major result of process mining are typically visualized in a static way. Though they are complex objects composed of nodes and edges, they are typically just drawn like an image. Except for scrolling and zooming, there is often no way to interact with the process models. Direct interaction like clicking on a node has the potential, to improve the exploration of processes by making the analysis faster and more intuitive.
2. *Dynamic analysis workflow:* Especially in MPM, analysts have to follow a more or less restrictive workflow. For example, they first have to define an OLAP query (selecting some dimensions to drill down the event data, adding optional filters), select and configure a process discovery algorithm, enhance the resulting process models with time information, and finally apply conformance checking to measure the quality of the models. If a previous analysis step has to be changed (e.g., adding another filter to the OLAP query), all subsequent analysis steps have to be done again. Consequently, even minimal changes may require a lot of effort for the analysts.
3. *Undo/redo of analysis steps:* It is desirable to provide support to undo and redo particular analysis steps. This enables analysts to try arbitrary analysis steps without any risks. The chance to return to the previous view on the process without any additional effort supports the explorative character of MPM because it avoids to end up in an "analysis dead end".
4. *Performance:* Even though MPM is not a time-critical application, performance is crucial for interactivity. Long processing times may disrupt the workflow preventing analysts from defining views which they expect to be less promising.

In this paper, we will present a novel framework which addresses the first three aspects as discussed above regarding the interactivity of MPM. We will identify different concepts and use cases to improve the interactivity of MPM. As the main contribution of our paper, we will provide a generic concept for interactive process mining based on a stack of operations relevant for MPM. We have to point out that performance (4) is only considered with respect to the workflow of MPM whereas the performance of particular analysis steps (e.g., executing OLAP queries or process discovery) is out of the scope of this paper.

The structure of the paper is as follows. Section 2 briefly discusses related work. In Sec. 3, we first introduce the assumptions of our work, before we elaborate on the general ideas and concepts of our approach. The proof-of-concept implementation of our approach is presented in Sec. 4. Finally, we conclude our work in Sec. 5 and give an overview of future work.

2 Related Work

The process mining manifesto [4] gives an overview of the field of process mining. For a comprehensive introduction to this topic, we refer to van der Aalst [2]. Event Cubes [12] are an initial but very specific approach for MPM based on a multidimensional adoption of the Flexible Heuristics Miner algorithm [16]. In contrast, Process Cubes [3,6] and PMCube [15] are generic frameworks that are not limited to a particular mining algorithm or process model representation. Moreover, they are able to incorporate process enhancement and conformance checking. PMCube also introduces advanced concepts to MPM like process model comparison using difference models and the process model consolidation which provides the automatic pre-selection of possibly more relevant process models. Even though they enable the analysts to explore the processes from a multidimensional perspective, Process Cubes as well as PMCube are limited in their interactivity. For example, they do not provide interactive process representation. The lack of interactivity was also identified during the case study presented in [15] and the systematic literature review provided in [13].

In the general context of process mining, interactivity has been initially addressed by the Fuzzy Miner algorithm [9] which adopts the basic idea of interactive maps to create so-called process maps. This representation enables the analyst to dynamically abstract from less relevant parts of the process model by clustering activities in order to focus on the major relationships of the process. Similar interactivity is also implemented in commercial tools like Fluxicon Disco [8] which considers process enhancement, too. However, we aim for a more comprehensive and generic approach which considers aspects of MPM (e.g., OLAP, difference calculation for process models, process model consolidation) and supports arbitrary process discovery algorithms and process model notations. In [1], van der Aalst extends the idea of process maps by the metaphor of navigation systems to provide interactive recommendation and prediction, e.g. to estimate the remaining execution time of a process instance. In contrast to that, we only consider historical data as we aim to improve the interactivity of the overall

MPM workflow. The application of MPM to real-time event data has not been investigated in research so far.

Interactivity is also considered in the intersection of process mining and visual analytics. As stated in the process mining manifesto [4], the combination of both fields has the potential "to extract more insights from event data". An example for such a visual process mining is the EventExplorer [5] which allows to browse an event log in order to visually explore and assess it. Visual analytics techniques for process mining algorithms realized in ProM[3] along the control flow, organizational, case, and time perspective are evaluated in [10]. However, the multidimensional perspective has not been considered yet.

RapidProM [11] allows analysts to interactively model scientific workflows for process mining by connecting operators in order to define complex analysis scenarios combining several techniques which should be repeated using different parameter settings or data sets. In contrast, we focus on supporting explorative ad-hoc analysis by interactivity instead of predefining an analysis workflow.

3 Approach

In this section, we will present the basic ideas and the concept for an interactive process mining. Section 3.1 introduces the underlying assumptions for this work. In Sec. 3.2, we discuss direct interaction with process models and present an operator framework for supporting interactivity in Sec. 3.3.

3.1 Assumptions

For this work, we assume that events are stored in a data cube which is accessible by OLAP queries that are expressed by a query language or a graphical user interface. This query is expected to return a set of sublogs in the usual event log structure (e.g., as defined in [2]). We do not assume a limitation to particular algorithms for process discovery, conformance checking, process enhancement, and consolidation. For the process models, we assume an arbitrary graph-based representation consisting of nodes and edges. However, there are different kinds and styles of nodes possible. Furthermore, we only focus on postmortem analysis. Process mining techniques combining historical data with real-time events like recommendation and prediction are not in the scope of this work.

3.2 Direct Interaction with Process Models

To make the process analysis as intuitive as possible, it is desirable to allow for a direct interaction with the process models (cf. first aspect *Interaction with process models* introduced in Sec. 1). While modeling tools, for example, enable the user to directly interact with the presented model and its elements, process mining tools (especially tools for MPM) usually only provide a static

[3] www.promtools.org

representation. As the models form the central subject of the analysis, MPM and process mining in general can also benefit from direct interactions with the process models. Therefore, we aim for an approach that allows the analysts to directly interact with the models. An example for such an interaction is to click on a node to highlight all model elements that are part of a trace containing the selected node. Which action to perform for a particular interaction can be selected from a toolbox. Alternative actions may be the presentation of additional information in a dialog or a sidebar. Furthermore, it is possible to enrich the process models' nodes and edges with additional user controls like buttons or context menus. This enables the model to provide different interactions for the same object at the same time.

Nonetheless, the direct process model interaction can be complemented by an interaction with external user controls, e.g. sliders, buttons etc. to perform actions on the model. E.g., the Fuzzy Miner [9] provides sliders to adjust filter thresholds and update the presented process model. The interaction with the process model should also incorporate the selection of different perspectives to dynamically add additional information to the model. To provide a generic solution, we define variation points of the process model elements that can be used for the visualization of additional data. Examples are the edge labels, the border and background colors of nodes, or the line thickness of arcs. Which information (e.g. frequencies, waiting times) a variation point will visualize is determined by the user selecting a particular perspective of the process model.

3.3 Operator Framework

In order to address the second and the third aspect (*Dynamic analysis workflow* and *Und/redo of analysis steps*, cf. Sec. 1), our approach introduces an operator framework. It maps the interactions with the software – especially each analysis step – onto operators which are organized at different levels according to their position in the analysis workflow and their data dependencies. Therefore, the operators of a particular level only consume the results of lower levels as input while their results are only available for the levels above. Fig. 1 shows the stack of operators defined by the framework and the data items forming their input and output. In the following, we will explain the operator levels in more detail.

OLAP An OLAP operation of the framework represents a query that extracts data from the data cube and returns a set of sublogs as a result. Parameters of the operation are the granularity of the considered dimensions, filter predicates etc. Note that each of this operations comprises multiple low-level OLAP operations on the data cube like roll-up, drill-down, slice, and dice. As the extraction of event data from the cube is necessary to conduct a process analysis, this operation level is mandatory.

Event Log Processing The optional event log processing enables the analysts to manipulate the extracted sublogs, e.g., by filtering or aggregating events. It is also possible to derive new attributes from other attributes (e.g., calculating event durations from the events' start and end time-stamps). The

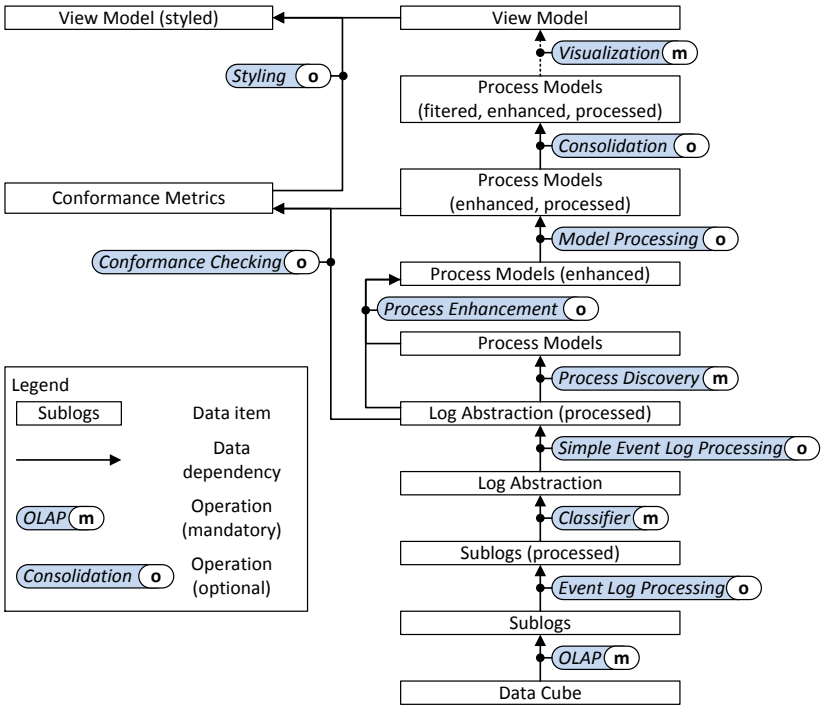

Fig. 1. Stack of operator levels with data items and their dependencies

event log processing operations are functionally overlapping with the OLAP operators. However, providing filters directly on the sublogs may significantly reduce the waiting times as loading data from the data cube is the most expensive operation in MPM in matters of performance. Therefore, applying these operations can be more efficient than changing OLAP queries.

Classifier The classifier operations select a classifier which is a function that defines how to map the event data onto a node label necessary for process discovery. Using this classifier, these operations create a log abstraction which consists of a number of classifier-dependent log metrics (e.g., relations between events and event counts). The log abstraction also comprises a more abstract representation of the sublog (similar to simple event logs, cf. [2]) which is derived using the selected classifier. It consists of a set of (unique) traces and the frequency of their occurrence. Additionally, they also maintain references to the original cases and events to be able to use them for process enhancement. Classifier operations are mandatory, because all process discovery algorithms require a classifier to select the node labels. Additionally, many process discovery algorithms like the Fuzzy Miner [9] or the Flexible Heuristics Miner [16] rely on these log metrics. Furthermore, other techniques like conformance checking (e.g., token replay) can benefit from the simple event log with respect to performance.

Simple Event Log Processing Similar to the sublogs, the simple event logs contained by the log abstraction can be manipulated, e.g., traces can be filtered by the number of occurrence. As the simple event logs provide a more condensed representation of the data giving the frequency of each trace, they can be processed more efficiently than normal sublogs (which may have duplicate traces), especially if they only contain few trace variations. However, manipulations of simple event logs may effect the log metrics, so they have to be recalculated after all simple event log operations were executed.

Process Discovery Based on the log abstractions, the process discovery operations create a process model for each cell using an arbitrary algorithm. Parameters of the operations are the selected algorithm and its individual settings (e.g., algorithm-specific thresholds). Note that changing the settings creates a new process discovery operation even if the selected algorithm remains the same. This operation level is mandatory, because the resulting process models are required for subsequent operation levels.

Process Enhancement These operations apply arbitrary process enhancement algorithms to the previously discovered process models. As different algorithms may add different perspectives (e.g., organizational or time perspective), this operation level is chainable.

Model Processing These optional operations manipulate process models, e.g., filtering nodes and edges or converting process models from one representation (e.g., petri nets) to another (e.g., BPMN). As multiple manipulations of the models may be desired, these operations are chainable.

Conformance Checking Based on the log abstractions and the process models, these operations measure the quality of the discovered process models. Examples for this operation are a token replay or a trace alignment in order to measure the fitness of the process models. As the quality metric may incorporate additional perspectives as well, the enhanced and possibly manipulated process models are used.

Consolidation The optional consolidation operations select a subset of potentially interesting process models while hiding irrelevant models in order to reduce the result complexity of MPM. An example for such an operation is the clustering consolidation (cf. [15]) which creates clusters of similar process models and selects a representative for each cluster.

Visualization The visualization operations translate the process model into a corresponding view model to be displayed to the analysts. In contrast to the previous operations, the visualization is not automatically executed for all process models but only for the models currently presented to the user.

Styling These optional operations manipulate the view model by changing its style (e.g., highlighting nodes by color) and linking the variation points of the visual elements to additional information (e.g., mapping the frequency of an edge onto its label). Like the visualization operations, they are only executed for the currently shown models. Furthermore, they are chainable, because multiple perspectives may be mapped to the view model.

Some levels are required to have at least one operation to be executed to be able to execute the level above. We call these levels mandatory (marked with

m in Fig. 1). However, there are also optional levels (marked with **o**) that do not need to have an operation to be executed. For example, process discovery operations are mandatory because the resulting process model is required for the execution of other levels like process enhancement or conformance checking. In contrast, enhancing the process model is optional.

For each level, the framework manages all executed operators in a separated list to keep track of the analysis history. If the analyst performs an interaction, an operation representing this interaction is added to the list of its corresponding level. Then, the operations of this level and the levels above are consecutively executed to propagate the changes to the final analysis results. The operations of the levels below are not executed again as their results remain unchanged. This avoids unnecessary data processing and results in shorter waiting times and a faster system response. By managing the operations at different levels, it is possible to undo or redo operations separately. This enables the analyst to, for example, go back to the previous OLAP query without discarding the other performed analysis steps like process enhancement or conformance checking. The change propagation ensures that the perspective on the process remains the same, while the underlying data is updated.

The framework keeps track of all performed interactions. Besides providing an advanced undo/redo functionality, this also prevents the analysts from repeating analysis steps when they change the underlying OLAP query. The operators also form an intermediate layer decoupling user interaction and data processing which makes it easier to link direct interactions (cf. Sec. 3.2) to particular analysis step.

The operations only store the parameters required for their execution, e.g., the selected process discovery algorithm and its settings. The data items that should be processed are centrally managed by the framework and only passed for execution, so the operators are always considering the current data. For each level, the framework manages a reference to an operation marking the latest operation to execute. We call this reference the *latest active operation* of the level. There are two different kinds of operator levels:

Chainable levels All operations up to the latest active operation of this level are consecutively executed in the respective order of their position in the operation list. All other operations of that list are considered as inactive and consequently not executed. Operation levels are only chainable if their input and the output structure are similar so the results of an operation can be passed to its successive operation as input. This can be useful, e.g., to execute multiple filter operations on the sublogs, where each operation filters by a different attribute.

Non-chainable levels For this kind of operation levels, only the latest active operation is executed. This only applies to operator levels where the operators transform the input to a differently structured output. An example for this kind of operator levels is process discovery which takes an event log as input in order to create a process model as output.

The separation of styling and visualization operations makes it possible to differentiate between the calculation and the visualization of enhanced perspec-

tives. This way, the analyst can show or hide specific information without any expensive recalculations. The styling operations link the operator framework to the direct interaction (cf. Sec. 3.2) by binding event handlers to particular visual elements. This way, it is possible to provide different reactions, e.g., when the user clicks on a node, depending on the selected styling operation.

As a view model should provide an integrated view onto to the process which incorporates multiple perspectives, it is necessary that the styling operations are chainable. However, it may happen that the user selects two operations that affect the same visual element, e.g., that map different values to the same label. To avoid confusion and misinterpretations by overriding the result of previous styling operations, such conflicts need to be resolved. Therefore, each operation provides a list of all properties of the visual elements it affects. Before adding a new styling operation, all existing operations are checked if their affected properties overlap with the affected properties of the new operation. Each operation that is in conflict with the new operation, will be discarded.

The described concepts contribute to the four aspects introduced in Sec. 1. The first aspect (*Interaction with process models*) is directly addressed by the concept for direct interaction introduced in Sec. 3.2. The operator framework contributes to the second aspect (*dynamic analysis workflow*). It allows the user to dynamically add and execute operations on an arbitrary operation level. Consequently, the users are less restriced during the workflow, because they do not have to follow a step-by-step configuration of the analysis. The change propagation also ensures that the users do not have to repeat previously performed analysis steps (like process enhancement) when applying changes to the underlying analysis steps (e.g., by adding filters to the OLAP query).

The operator framework also directly contributes to the third aspect (*Undo/ redo of analysis steps*; cf. Sec. 1): Changing the reference to the latest active operation to the previous operation will revert it, because all operations following the reference will be ignored. However, as the undone operation is kept in the list, a redo can be easily achieved by setting the latest active operation reference to the subsequent operation. Finally, the operator framework also contributes to aspect four (*performance*). It ensures that changes to the analysis by adding, undoing or redoing an operation will only affect the subsequent operations. Operations on lower operation levels do not have to be repeated. This significantly reduces the waiting times for the user because the extraction of data from the cube is typically the most expensive task in MPM in matters of processing time.

4 Implementation

We implemented our approach in a prototype called Interactive PMCube Explorer[4] using C# and the .NET framework. It is based on the PMCube Explorer tool [14] and replaces its orginal user interface. The operator framework (cf. Sec. 3.3) is managed by a central operation manager component while the operators

[4] Screencast and tool download available at `http://www.uol.de/pmcubeexplorer`

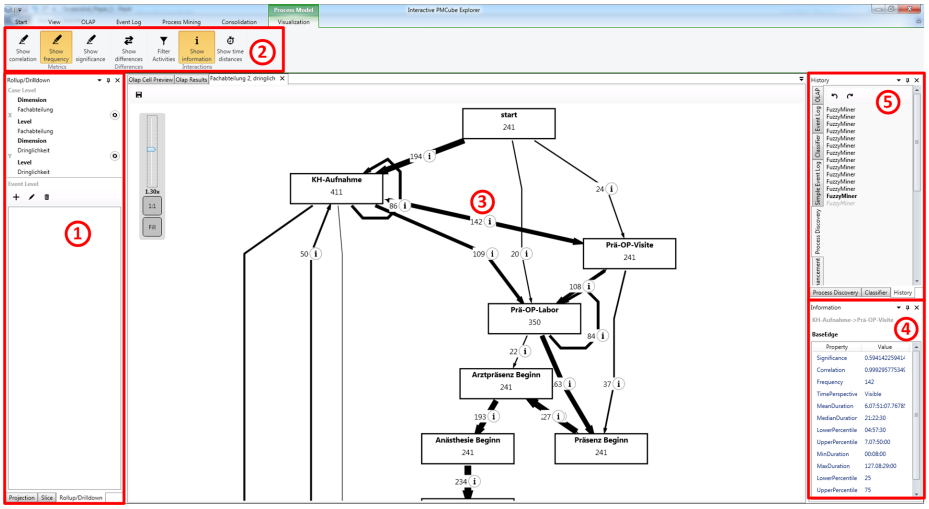

Fig. 2. Screenshot of the Interactive PMCube Explorer tool

are (mainly) integrated as plug-ins. Only the visualization and styling operations are managed by a separated visualization manager component due to their asynchronous execution (triggered by events) and specific challenges like the conflict resolution.

Figure 2 shows a screenshot of the Interactive PMCube Explorer tool. As they form the main subject of the analysis, the process mining results are presented at the center of the screen using tabs. Tool boxes, options, and additional settings are flexibly arranged around them to keep the focus on the analysis results. In this example, one can see the options for defining the OLAP operation (1) and the available styling operations for the presented process model (2). The styling operations for annotating the process model with the frequencies of nodes and edges (left highlighted button) and adding direct interaction to show additional information (right highlighted button) are activated. The effect of these style operations on the visualization can be seen in the process model (3), where additional labels are attached to nodes and edges indicating their frequencies. The event handlers for direct interaction are linked to the activity nodes and to additional buttons of the edges, because clicking on thin edges might be difficult and hindering. Clicking on a node or the edge's button opens the additional information in a separated view aside of the process model (4). Above that, the history view separately shows all performed operations for each level (5). In this example, one can see the history of process discovery operations highlighting the latest active operation while the discarded operation is toned down.

The history view allows the user to undo or redo any operation separately for each level. This enables the user, e.g., to safely try a different parameter setting of the process discovery algorithm and to return to the previous settings without repeating the configuration of other analysis steps like process enhancement.

The only exception are the styling operations, which can be directly activated or deactivated without following the order that they were performed.

To show the feasibility of the approach, the prototype provides several operators for most levels. A plug-in system allows for the easy integration of additional operators. The direct interaction is demonstrated by a number of styling operations, e.g., attaching different information like frequencies and metrics to the model, color-coding of difference models, and adding event handlers to show time differences and the additional information view. Besides, it is possible to bind event handlers to visual elements in order to trigger operations on other levels. For example, one interaction filters the simple event log to all traces having a certain activity just by clicking the respective activity node in the process model.

The implementation shows that the operator framework (cf. Sec. 3.3) can be directly applied to the PMCube approach. However, the operator framework is a generic concept. Except for the assumptions from Sec. 3.1, it is not restricted to a particular approach for MPM. In principle, it can be also applied to other MPM approaches like Process Cubes [3,6] which meets our assumptions for MPM. For instance, Process Cubes uses OLAP queries to extract sublogs from the cube and is able to apply different process mining techniques like process discovery, process enhancement and conformance checking. However, some special analysis steps (e.g., consolidation of process models) are not considered in Process Cubes. However, our concepts can still be applied to it if the correspondent operation levels will be removed. On the contrary, the operator framework cannot be applied to the Event Cube because it defines MPM in a very different way and does not meet our assumptions, e.g., it does not create sublogs and defines the OLAP operations like roll-up and drill-down as manipulations of the process model.

The concept for direct interaction is generally applicable to other MPM approaches (like Process Cubes) as well as to non-multidimensional process mining. The basic idea of the operator framework can also be adopted for process mining tools in general because MPM can be considered as a generalization of traditional process mining. The differences of non-multidimensional process mining approaches are mainly related to the number of data items (typically one log and process model is processed at a time) and some operation levels have to be skipped (e.g., OLAP) as they are only meaningful in MPM context. Therefore, also the process mining community in general may benefit from this concept.

5 Conclusions and Future Work

In this paper, we presented a novel research towards interactive process mining. Its key points are the direct interaction with the process models and the operator framework which aims to avoid the unnecessary repetition of analysis steps. We implemented our approach based on the PMCube Explorer tool [14] as a proof-of-concept prototype which showed the feasibility of the approach. We plan to evaluate our approach by a user study to investigate the contribution of our approach to the interactivity of process mining.

References

1. van der Aalst, W.M.P.: Using process mining to generate accurate and interactive business process maps. In: Abramowicz, W., Flejter, D. (eds.) Business Information Systems Workshops, BIS 2009 International Workshops. Lecture Notes in Business Information Processing, vol. 37, pp. 1–14. Springer (2009)
2. van der Aalst, W.M.P.: Process Mining - Discovery, Conformance and Enhancement of Business Processes. Springer (2011)
3. van der Aalst, W.M.P.: Process Cubes: Slicing, Dicing, Rolling Up and Drilling Down Event Data for Process Mining. In: Song, M., et al. (eds.) Asia Pacific Business Process Management. LNBIP, vol. 159, pp. 1–22. Springer (2013)
4. van der Aalst et al., W.M.P.: Process mining manifesto. In: Daniel, F., et al. (eds.) BPM Workshops. LNBIP, vol. 99, pp. 169–194. Springer (2011)
5. Bodesinsky, P., Alsallakh, B., Gschwandtner, T., Miksch, S.: Exploration and Assessment of Event Data. In: Bertini, E., Roberts, J.C. (eds.) EuroVis Workshop on Visual Analytics (EuroVA). The Eurographics Association (2015)
6. Bolt, A., van der Aalst, W.M.P.: Multidimensional Process Mining Using Process Cubes. In: Gaaloul, K., et al. (eds.) Enterprise, Business-Process and Information Systems Modeling. LNBIP, vol. 214, pp. 102–116. Springer (2015)
7. Golfarelli, M., Rizzi, S.: Data Warehouse Design: Modern Principles and Methodologies. McGraw-Hill, Inc., New York, NY, USA, 1 edn. (2009)
8. Günther, C.W., Rozinat, A.: Disco: Discover your processes. In: Lohmann, N., Moser, S. (eds.) Proceedings of the BPM 2012 Demonstration Track. CEUR Workshop Proceedings, vol. 940, pp. 40–44. CEUR-WS.org (2012)
9. Günther, C.W., van der Aalst, W.M.P.: Fuzzy mining: adaptive process simplification based on multi-perspective metrics. In: Proceedings of the 5th international conference on Business process management. pp. 328–343. BPM'07, Springer-Verlag, Berlin, Heidelberg (2007)
10. Kriglstein, S., Pohl, M., Rinderle-Ma, S., Stallinger, M.: Visual analytics in process mining: Classification of process mining techniques. In: 7th International Eurovis Workshop on Visual Analytics. Groningen (2016), (accepted for publication)
11. Mans, R., van der Aalst, W.M.P., Verbeek, H.M.W.E.: Supporting process mining workflows with rapidprom. In: Limonad, L., Weber, B. (eds.) Proceedings of the BPM Demo Sessions 2014. CEUR Workshop Proceedings, vol. 1295, p. 56. CEUR-WS.org (2014)
12. Ribeiro, J.T.S., Weijters, A.J.M.M.: Event Cube: Another Perspective on Business Processes. In: Robert Meersman et al. (ed.) On the Move to Meaningful Internet Systems: OTM 2011. LNCS, vol. 7044, pp. 274–283. Springer (2011)
13. Vogelgesang, T., Kaes, G., Rinderle-Ma, S., Appelrath, H.: Multidimensional process mining: Questions, requirements, and limitations. In: Proceedings of the CAiSE'16 Forum. pp. 169–176 (2016)
14. Vogelgesang, T., et al.: Multidimensional process mining with pmcube explorer. In: Daniel, F., Zugal, S. (eds.) Proceedings of the BPM Demo Session 2015. CEUR Workshop Proceedings, vol. 1418, pp. 90–94. CEUR-WS.org (2015)
15. Vogelgesang, T., et al.: PMCube: A Data-Warehouse-based Approach for Multidimensional Process Mining. In: Business Process Management Workshops (2015)
16. Weijters, A.J.M.M., Ribeiro, J.T.S.: Flexible heuristics miner (FHM). Tech. rep., Technische Universiteit Eindhoven (2011)