

Complex Behaviours Detection and Analysis in Process Mining

Yang Lu

School of Computer Science, The University of Sydney, Sydney, NSW 2006, Australia

Abstract

Process mining builds a bridge between traditional process modelling and data mining. Process discovery algorithms aim at constructing process models automatically from event logs. Existing process discovery algorithms perform well with simple event logs, but their performance can be affected when the input event logs contain complex behaviours, and the discovered process models may not represent the real behaviours of the business processes. In this research, we plan to develop methods to automatically detect different complex behaviours in event logs. The detection of complex behaviours can be based on the extensions of existing process discovery algorithms or stand-alone complex behaviours detection tools. The paper presents our research questions, methodologies, current research progress and potential challenges.

Keywords

Process Mining, Process Discovery, Complex Behaviours Detection

1. Introduction

Process mining is a relatively new subject which builds a bridge between traditional process modeling and data mining [1]. Process discovery is the most critical part of process mining which aims at extracting process insights of a system. The discovered process models should not only have high quality measures (i.e., fitness, precision and generalization), but also be an accurate representation of the real process behaviours.

Many process discovery algorithms have been proposed, and some can return process models with high fitness and precision values [2]. However, when input logs contain complex behaviours, the discovered models may not describe the process behaviours accurately. Firstly, some process discovery algorithms cannot guarantee the soundness of discovered models. When complex behaviours are included in event logs, they may return process models which are not sound [2]. Secondly, in order to accommodate complex process behaviours in a single process model, process discovery algorithms may return a complex and incomprehensible process models [3]. For example, when the event log contains information of several micro-processes, it is hard to use a single process model to describe the processes [4]. Thirdly, due to the representation bias [5] of process modeling languages (eg., Petri nets, BPMNs, Process

Proceedings of the Doctoral Consortium Papers Presented at the 34th International Conference on Advanced Information Systems Engineering (CAiSE 2022), June 06–10, 2022, Leuven, Belgium


✉ yalu8986@uni.sydney.edu.au (Y. Lu)

🌐 <https://www.sydney.edu.au/engineering/about/our-people/research-students/yang-lu-503.html> (Y. Lu)

🆔 0000-0002-9002-8650 (Y. Lu)



© 2022 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

 CEUR Workshop Proceedings (CEUR-WS.org)

trees), some process behaviours may not be able to be accurately presented. For example, if an activity happens in the context rather than in the control flow of the process (i.e., the activity can happen at anytime during the process), it cannot be accurately described by traditional process modeling languages [6]. Lastly, even though a comprehensive and structured process model is discovered, it assumes the process to be static and may ignore the changes of the process during the execution time [7].

To discover comprehensive process models, many event log filtering tools have been proposed [8] to simplify the event logs. Although event logs can be simplified by using log filtering tools, important knowledge about the processes can be ignored.

As a goal of process mining is to facilitate business process improvement taking the advantage of event logs, it is important for us to get deep understanding about business processes. Filtering out and ignoring complex behaviours in event logs is inappropriate.

To this end, the proposed PhD project focuses on the detection and analysis of different complex behaviours in event logs. The research questions are presented below:

- Can we extend existing process modeling languages to present different complex behaviours?
- Can we extend existing process discovery algorithms to discover different complex behaviours?
- Can we develop stand-alone tools to discover different complex behaviours directly from event logs?

On the one hand, existing process discovery algorithms can be extended to handle complex behaviours. By extending existing process discovery algorithms, their advantages can be inherited. On the other hand, stand-alone tools to discover different complex process behaviours can be developed. These tools can be applied before performing data pre-processing and process discovery algorithms to help us understand the complex behaviours ignored by the discovered model.

The rest of this paper is structured as follows: Section 2 is a literature review of related work. Section 3 presents our research approach. Section 4 presents our current research progress. Section 5 presents potential challenges of our research project and concludes the paper.

2. Related Work

2.1. Extending Existing Process Modeling Languages and Discovery Algorithms to Handle Complex Behaviours

The original alpha algorithm [9] is one of the earliest process discovery algorithms. It guarantees to rediscover the process model when the event log satisfies certain conditions. However, the process behaviours which can be discovered by the original alpha algorithm are limited. The original alpha algorithm has been extended to handle complex behaviours such as the short loops [10], invisible tasks [11, 12] and non-free-choice behaviours [11, 13]. These algorithms allow complex behaviours to be discovered while maintaining the guarantees of the original alpha algorithm.

The inductive miner [14, 15] is one of the most popular process discovery algorithms which guarantees to return sound process models. Given the direct outcome of the inductive miner is a process tree, the behaviours being represented are limited. A so-called "flow model" (i.e., process model with high fitness but very low precision) can be returned when the process behaviours are unable to be represented by process trees. To allow more behaviours to be represented and discovered, both the inductive miner and process tree are extended. For example, Leemans et al. [16] extend the inductive miner to discover cancellation behaviours, Lu et al. [17] extend the inductive miner to discover switch behaviours (i.e., change of paths on exclusive choice branches). Leemans et al. [18] also extend the inductive miner to discover recursive behaviours in the execution of software source code. In all these extensions, the discovered models are always sound.

2.2. Stand-alone Tools to Discover Complex Behaviours

Many methods have been developed in order to cluster traces in event logs [4]. Traces with similar behaviours are put into the same cluster. Instead of using one process model to represent the process in the event log, a separate process model is discovered for each cluster.

Besides trace clustering algorithms, some methods also focus on the discovery of hierarchical process models to handle complex event logs with many activities and sub-processes (eg., [19, 20]). A high-level process model can be used to represent the relationship between different sub-processes while a low-level model can be used to represent the process model of each single sub-process.

To deal with changes within business processes, various methods have been proposed to deal with concept drifts in event logs [7]. Some algorithms focus on detecting the time points of concept drifts (eg., [21, 22]). Others focus on providing comprehensive results to users (eg., visualise process changes [23]). When time points of concept drifts are detected, a separate process model can be discovered between each pair of points to help us understand the evolving of processes.

Some research also focuses on detecting other complex behaviours. For example, Lu et al. [24] propose a method to handle duplicate activities in event logs. More specifically, the same activity may execute in different stages of a process. A more comprehensive process model can be discovered if we split such an activity into multiple activities. Dees et al. [6] propose a method to visualise the behaviours of context activities on the edges of process models (i.e., activities which can happen at anytime during the execution of the process). Dubinsky et al. [25] propose a method to detect the "split-cases" behaviours in event logs (i.e., when a case is illegally split into multiple cases).

In a nutshell, stand-alone tools are independent from process discovery algorithms. They can discover more insights about the process from event logs which usually cannot be discovered directly by process discovery algorithms. These methods can be used together with process discovery algorithms to enhance our understanding of business processes described in event logs.

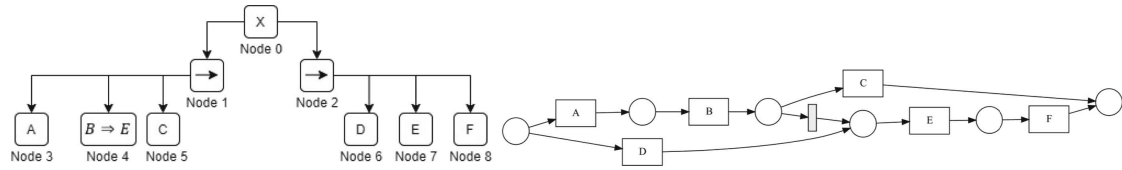


Figure 1: An example switch process tree and its corresponding workflow net

3. Research Approach

Our research approach is adapted from the Design Science Research (DSR) [26] methodology. Our research contains the following stages:

- **Identifying Problems:** At this stage, literature review is conducted. The literature review focuses on research related to detecting process behaviours from event logs (eg., process discovery algorithms, concept drift detection algorithms, trace clustering algorithms, etc.). Based on the literature review, gaps can be identified among existing literature (eg., existing concept drift detection algorithms cannot distinguish process drifts from noises).
- **Designing and Developing Solutions:** Once a gap has been identified among existing literature, a solution can then be proposed to solve the problem. Then a software can be built to implement the proposed solution. Existing open-source software packages can be modified for faster development process.
- **Demonstration and Evaluation:** When a potential solution is implemented, evaluation should be conducted to show the capabilities of the proposed solution. The evaluation usually contains two parts: firstly, the method is evaluated empirically using a big number of synthetic datasets. The performance of the proposed method can also be compared with existing methods. Secondly, the method is evaluated using real-life datasets.
- **Communication:** If a valid solution is built, the work can be presented to the research community through academic conferences, workshops or journals.

4. Current Results

In this section, we briefly introduce some of our work to discover complex process behaviours in event logs. Please refer to the corresponding references for the details of our proposed methods and evaluation results. It has to be noted that all work presented in this section has been evaluated using publicly available datasets. The implementations of these methods are also publicly available.

4.1. Discovering Switch Behaviours in Process Mining

In [17], we propose a novel method to extend the inductive miner to discover switch behaviours. Assume there is an exclusive decision choice in a process model, and the decision point is split into multiple branches. In real-life event logs, it is possible to switch between different

Table 1

Comparing the extended inductive miner with the original inductive miner

Log: <A, B, C>, <D, E, F>, <A, F>	
IM	IM augmented with switch behaviours
Fitness: 1.0, Precision: 0.38	Fitness: 1.0, Precision: 1.0
Log: <A, B, C>, <D, E, F>, <A, E, F>, <D, E, C>, <A, E, C>	
IM	IM augmented with switch behaviours
Fitness: 1.0, Precision: 0.42	Fitness: 1.0, Precision: 1.0

branches after the decision has been made. However, due to the limitation of process trees, the original inductive miner is unable to discover such behaviours. A "follower model" with very low precision can be returned when such behaviours exist in event logs.

To solve the problem, we firstly extend the process tree notation to allow the presentation of switch behaviours. The new notation is called "switch process tree". Each switch process tree can be translated into an equivalent workflow net. In addition, with some constraints of switch process trees, their corresponding workflow nets can be guaranteed to be sound. Fig. 1 shows an example process tree and its corresponding workflow net.

The extension of process trees allows us to extend the inductive miner to discover switch behaviours. As shown in Table 1, when switch behaviours exist in event logs, our proposed method can significantly improve the precision of the discovered process models. The discovered process models by the extended inductive miner can accurately present the process behaviours in event logs.

Finally, the method is also evaluated using a real-life dataset ("BPIC13-incident" event log from the "4TU Center for Research Data"). As shown in Table 2, a more accurate and simpler process model is discovered by our proposed method.

Table 2

Evaluation results with the publicly-available dataset (IMs refers to our approach, SM refers to the Split Miner [27], and IMf refers to the inductive miner infrequent [15])

	Accuracy			Complexity	
	Fitness	Precision	F-Score	Size	CFC
IMf	0.95	0.59	0.73	35	33
SM	0.98	0.71	0.82	39	48
IMs	0.97	0.80	0.88	33	46

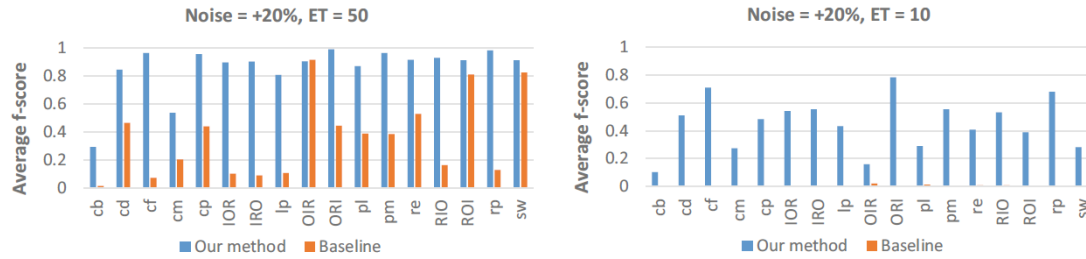


Figure 2: Comparing our drift detection algorithm with the baseline [22] under different settings when noises are inserted in event logs. Please refer to [22] for details of the evaluation results

4.2. Discovering Concept Drifts in Process Mining

In [21], we propose a method to accurately detect time points of process drifts in event logs. A process drift point is defined as the time point when there is a statistically significant difference among the observed process behaviours before and after the change point. Most existing process drift detection algorithms assume the input event logs to be clean and cannot differentiate process drifts from noises. Our proposed method can not only accurately detect process drift points, but can also be robust to noises. Evaluation results show that our proposed method can consistently perform better than existing methods. Fig. 2 shows the evaluation results comparing to the baseline [22] when 20% of noises are inserted into the event logs.

In [28], we propose a method to detect branching frequency changes in process models. Branching frequency changes refer to changes in frequencies between different options when there is an exclusive choice. Our method is evaluated using a publicly available event log (“Italian Help Desk” event log from the “4TU Center for Research Data”). The process model is presented in Fig. 3 and the results are presented in Fig. 4. Two frequency changes are detected in the event log, and the frequencies between choosing activity “Wait”, “Require upgrade” and “Resolve ticket” after “Take in charge ticket” change significantly after each drift point.

5. Conclusions and Future Work

In this paper, we explain the need for developing new techniques to discover complex behaviours in event logs for better understanding of business processes. On the one hand, existing process

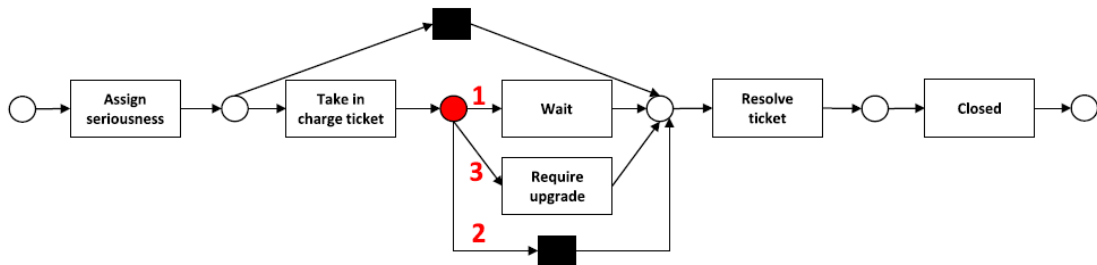


Figure 3: Process model of the “Italian help desk” log

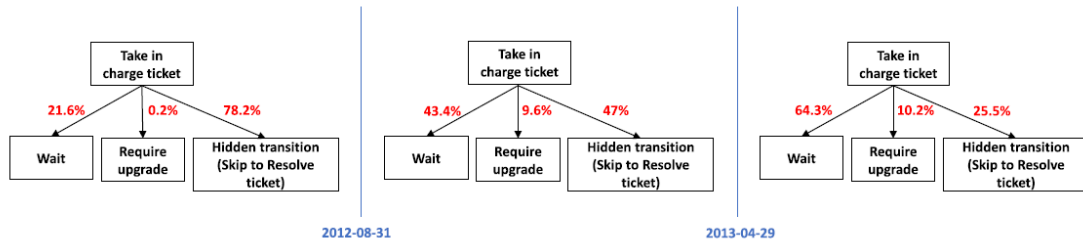


Figure 4: Visualisation of the detected branching frequency changes in the "Italian Help Desk" log

discovery algorithms can be extended to handle complex behaviours. On the other hand, stand-alone tools can also be constructed for the detection of specific types of behaviours.

Although we have provided some solutions for the research problem, there are still some challenges. Firstly, there is the lack of methods to validate the results. For example, in [17], although a process model with higher precision value can be obtained from the publicly available event log, it may still be insufficient to conclude that switch behaviours actually exist in the process. We plan to evaluate this work empirically to evaluate its capabilities in the future. Secondly, some methods heavily rely on user-defined parameters. For example, In [21], a window size is required from users. The window size defines the number of events in each sample when performing statistical tests. Different process drift points can be reported with different window sizes. In [28], the detection of drift points relies on an external change detection algorithm. The detection results are dependent on the parameters of the external algorithm.

Finally, we also plan to develop methods to discover other complex behaviours such as the automatic detection of cancellation behaviours and non-free-choice behaviours in event logs.

Acknowledgments

This research project is supervised by Associate Professor Simon K. Poon from the School of Computer Science, the University of Sydney.

References

- [1] W. van der Aalst, *Process Mining*, Springer Berlin Heidelberg, Berlin, Heidelberg, 2016. doi:10.1007/978-3-662-49851-4.
- [2] A. Augusto, R. Conforti, M. Dumas, M. La Rosa, F. M. Maggi, A. Marrella, M. Mecella, A. Soo, Automated discovery of process models from event logs: review and benchmark, *IEEE transactions on knowledge and data engineering* 31 (2018) 686–705.
- [3] W. M. van der Aalst, What makes a good process model?, *Software & Systems Modeling* 11 (2012) 557–569.
- [4] F. Zandkarimi, J.-R. Rehse, P. Soudmand, H. Hoehle, A generic framework for trace clustering in process mining, in: *2020 2nd International Conference on Process Mining (ICPM)*, 2020, pp. 177–184. doi:10.1109/ICPM49681.2020.00034.
- [5] W. M. van der Aalst, On the representational bias in process mining, in: *2011 IEEE 20th International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises*, IEEE, 2011, pp. 2–7.
- [6] M. Dees, B. Hompes, W. M. van der Aalst, Events put into context (epic), in: *2020 2nd International Conference on Process Mining (ICPM)*, 2020, pp. 65–72. doi:10.1109/ICPM49681.2020.00020.
- [7] D. M. V. Sato, S. C. De Freitas, J. P. Barddal, E. E. Scalabrin, A survey on concept drift in process mining, *ACM Computing Surveys (CSUR)* 54 (2021) 1–38.
- [8] H. M. Marin-Castro, E. Tello-Leal, Event Log Preprocessing for Process Mining: A Review, *Applied Sciences* 11 (2021) 10556. doi:10.3390/app112210556.
- [9] W. Van der Aalst, T. Weijters, L. Maruster, Workflow mining: Discovering process models from event logs, *IEEE transactions on knowledge and data engineering* 16 (2004) 1128–1142.
- [10] A. A. De Medeiros, B. F. Van Dongen, W. M. Van der Aalst, A. Weijters, Process mining: Extending the α -algorithm to mine short loops (2004).
- [11] Q. Guo, L. Wen, J. Wang, Z. Yan, P. S. Yu, Mining invisible tasks in non-free-choice constructs, in: *International Conference on Business Process Management*, Springer, 2016, pp. 109–125.
- [12] L. Wen, J. Wang, W. M. van der Aalst, B. Huang, J. Sun, Mining process models with prime invisible tasks, *Data & Knowledge Engineering* 69 (2010) 999–1021.
- [13] L. Wen, W. M. Van Der Aalst, J. Wang, J. Sun, Mining process models with non-free-choice constructs, *Data Mining and Knowledge Discovery* 15 (2007) 145–180.
- [14] S. J. J. Leemans, D. Fahland, W. M. P. van der Aalst, Discovering block-structured process models from event logs - a constructive approach, in: J.-M. Colom, J. Desel (Eds.), *Application and Theory of Petri Nets and Concurrency*, Springer Berlin Heidelberg, Berlin, Heidelberg, 2013, pp. 311–329.
- [15] S. J. J. Leemans, D. Fahland, W. M. P. van der Aalst, Discovering block-structured process models from event logs containing infrequent behaviour, in: N. Lohmann, M. Song, P. Wohed (Eds.), *Business Process Management Workshops*, Springer International Publishing, Cham, 2014, pp. 66–78.
- [16] M. Leemans, W. M. P. van der Aalst, Modeling and discovering cancelation behavior, in: H. Panetto, C. Debruyne, W. Gaaloul, M. Papazoglou, A. Paschke, C. A. Ardagna, R. Meersman (Eds.), *On the Move to Meaningful Internet Systems. OTM 2017 Conferences*,

- Springer International Publishing, Cham, 2017, pp. 93–113.
- [17] Y. Lu, Q. Chen, S. Poon, A novel approach to discover switch behaviours in process mining, in: S. Leemans, H. Leopold (Eds.), *Process Mining Workshops*, Springer International Publishing, Cham, 2021, pp. 57–68.
 - [18] M. Leemans, W. M. Van Der Aalst, M. G. Van Den Brand, Recursion aware modeling and discovery for hierarchical software event log analysis, in: *2018 IEEE 25th international conference on software analysis, evolution and reengineering (SANER)*, IEEE, 2018, pp. 185–196.
 - [19] X. Lu, A. Gal, H. A. Reijers, Discovering hierarchical processes using flexible activity trees for event abstraction, in: *2020 2nd International Conference on Process Mining (ICPM)*, 2020, pp. 145–152. doi:10.1109/ICPM49681.2020.00030.
 - [20] S. J. Leemans, K. Goel, S. J. van Zelst, Using multi-level information in hierarchical process mining: Balancing behavioural quality and model complexity, in: *2020 2nd International Conference on Process Mining (ICPM)*, 2020, pp. 137–144. doi:10.1109/ICPM49681.2020.00029.
 - [21] Y. Lu, Q. Chen, S. Poon, A robust and accurate approach to detect process drifts from event streams, in: A. Polyvyanyy, M. T. Wynn, A. Van Looy, M. Reichert (Eds.), *Business Process Management*, Springer International Publishing, Cham, 2021, pp. 383–399.
 - [22] A. Ostovar, A. Maaradji, M. La Rosa, A. H. M. ter Hofstede, B. F. V. van Dongen, Detecting drift from event streams of unpredictable business processes, in: I. Comyn-Wattiau, K. Tanaka, I.-Y. Song, S. Yamamoto, M. Saeki (Eds.), *Conceptual Modeling*, Springer International Publishing, Cham, 2016, pp. 330–346.
 - [23] A. Yeshchenko, C. Di Ciccio, J. Mendling, A. Polyvyanyy, Visual drift detection for sequence data analysis of business processes, *IEEE Transactions on Visualization and Computer Graphics* (2021).
 - [24] X. Lu, D. Fahland, F. J. H. M. van den Biggelaar, W. M. P. van der Aalst, Handling duplicated tasks in process discovery by refining event labels, in: M. La Rosa, P. Loos, O. Pastor (Eds.), *Business Process Management*, Springer International Publishing, Cham, 2016, pp. 90–107.
 - [25] Y. Dubinsky, P. Soffer, Detecting the “split-cases” workaround in event logs, in: A. Augusto, A. Gill, S. Nurcan, I. Reinhartz-Berger, R. Schmidt, J. Zdravkovic (Eds.), *Enterprise, Business-Process and Information Systems Modeling*, Springer International Publishing, Cham, 2021, pp. 47–61.
 - [26] A. R. Hevner, S. T. March, J. Park, S. Ram, Design science in information systems research, *MIS Quarterly* 28 (2004) 75–105.
 - [27] A. Augusto, R. Conforti, M. Dumas, M. La Rosa, A. Polyvyanyy, Split miner: automated discovery of accurate and simple business process models from event logs, *Knowledge and Information Systems* 59 (2019) 251–284.
 - [28] Y. Lu, Q. Chen, S. Poon, Detecting and understanding branching frequency changes in process models, in: A. Augusto, A. Gill, S. Nurcan, I. Reinhartz-Berger, R. Schmidt, J. Zdravkovic (Eds.), *Enterprise, Business-Process and Information Systems Modeling*, Springer International Publishing, Cham, 2021, pp. 39–46.