

# SIMPT: Process Improvement Using Interactive Simulation of Time-aware Process Trees <sup>\*</sup>

Mahsa Pourbafrani<sup>1</sup>, Shuai Jiao<sup>2</sup>, and Wil M. P. van der Aalst<sup>1</sup>

<sup>1</sup> Chair of Process and Data Science, RWTH Aachen University, Germany  
{mahsa.bafrani,wvdaalst}@pads.rwth-aachen.de

<sup>2</sup> RWTH Aachen University, Germany  
shuai.jiao@rwth-aachen.de

**Abstract.** Process mining techniques including process discovery, conformance checking, and process enhancement provide extensive knowledge about processes. Discovering running processes and deviations as well as detecting performance problems and bottlenecks are well-supported by process mining tools. However, all the provided techniques represent the past/current state of the process. The improvement in a process requires insights into the future states of the process w.r.t. the possible actions/changes. In this paper, we present a new tool that enables process owners to extract all the process aspects from their historical event data automatically, change these aspects, and re-run the process automatically using an interface. The combination of process mining and simulation techniques provides new evidence-driven ways to explore "what-if" questions. Therefore, assessing the effects of changes in process improvement is also possible. Our Python-based web-application provides a complete interactive platform to improve the flow of activities, i.e., process tree, along with possible changes in all the derived activity, resource, and process parameters. These parameters are derived directly from an event log without user-background knowledge.

**Keywords:** process mining, process tree, interactive process improvement, simulation, event log, automatic simulation model generation.

## 1 Introduction

The real value of providing insights by process mining emerges when these insights can be put into action [1]. Actions include the improvement of discovered running processes, performance problems, deviations, and bottlenecks. Process owners should be able to take some actions based on this information with a certain level of confidence. To do so, they need to improve/change their processes interactively. Therefore, simulation and prediction techniques are taken into account to foresee the process after changes and improvement. Simulation techniques are capable of replaying processes with different scenarios.

---

<sup>\*</sup> Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy – EXC 2023 Internet of Production- Project ID: 390621612. We also thank the Alexander von Humboldt (AvH) Stiftung for supporting our research.

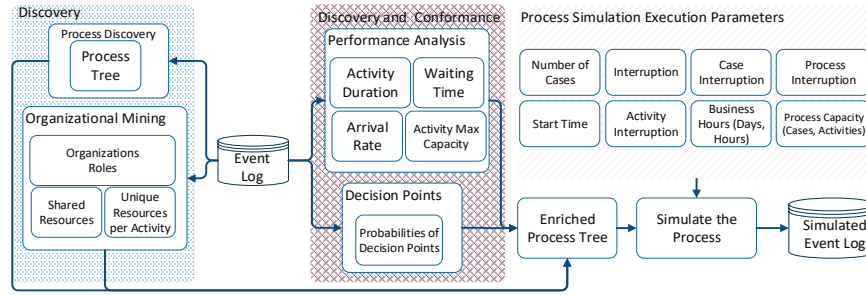


Fig. 1: The general framework of our tool for generating a process tree using process mining techniques and enriching the process tree with the possible information from an event log. The resulting process model can be executed to re-run the process with the user-provided configuration.

Process mining also enables designing data-driven simulation models of processes [2]. However, in the current tools for simulation in process mining either interaction with the user and user knowledge is a prerequisite of designing a simulation model or the tools are highly dependent on the interaction of multiple simulation tools. In [5], an external tool, i.e., ADONIS for simulating the discovered model and parameters are used. The combination of BPMN and process mining is presented in [4] in which BIMP is used as a simulation engine. However, the possibility of interaction for changing the process model is not available for the user. Also, the authors in [14] propose a Java-based discrete event simulation of processes using BPMN models and user interaction where the user plays a fundamental role in designing the models. Generating a CPN model based on CPN tools [18] is presented in [17]. The user needs to deal with the complexity of CPN tools and *SML*. In [7], the focus is also to generate CPN models and measuring performance measurements using the *Protos* models which can be used easier but more restricted than CPN tools. [16] performs simulation on top of discovered Petri nets and measure the performing metrics and not re-generating the complete behavior of a process. The *Monte Carlo* simulation technique, as well as generating sequences of activities based on process trees, are also proposed in Python [3]. However, the simulation results are not in the form of an event log, and they lack the time perspective. Also, the tool in [10] as a python library simulates the Petri nets of processes based on their event logs. In [13], aggregated simulations which are useful for what-if analyses in high-level decision-making scenarios are introduced. The *PMSD* tool represents the aggregated approach and generates a simulation model at a higher level of detail [8] based on the approach in [11]. Different process variables are introduced in [9] that makes the different levels of process simulations possible, e.g., simulating the daily behavior of a process instead of simulating every event in the process.

The interactive improvement/changes of processes is not a straightforward task when it comes to changing the process models and parameters by the user. Therefore, we use the discovered process tree and provide an interface for the

Table 1: The general parameters in the tool are listed. Most can be derived using process mining. Also, the required execution parameters. All the parameters discovered using process mining from event logs by default and are filled in the tool with the real values automatically. The execution values guaranteed the default values in case that users do not change/provide the parameters. Note that the handover matrix is used for logging resources.

|                          | Process Mining       |              |                              |                     |                                     |                                  |              |                |                           |                          |   | Simulation Execution Parameters |                 |
|--------------------------|----------------------|--------------|------------------------------|---------------------|-------------------------------------|----------------------------------|--------------|----------------|---------------------------|--------------------------|---|---------------------------------|-----------------|
|                          | Process Model (Tree) | Arrival Rate | Activity Duration, Deviation | Activities Capacity | Unique Resources (Shared Resources) | Social Network (Handover Matrix) | Waiting Time | Business Hours | Activity-flow Probability | Process Capacity (cases) | Interruption (Process, Cases, Activities) | Start Time of Simulation        | Number of Cases |
| Automatically Discovered | +                    | +            | +                            | +                   | +                                   | +                                | +            | +              | +                         | +                        | -   | -                               |                 |
| Changeable by User       | +                    | +            | +                            | +                   | +                                   | -                                | +            | +              | +                         | +                        | +   | +                               |                 |

user to discover and design a new process model including all the performance and environmental attributes, e.g., changing business hours, or resource capacity. All of these changes are supported by the information derived from event logs of processes using process mining techniques as shown in Figure 1. In this paper, we present our tool which is designed to support process improvement using simulation and process mining. Our tool is implemented as an integrated Python web-application using *Django* framework, where all the modules are also accessible outside the user interface for the users to re-use or add their desired modification to the code. Moreover, to the best of our knowledge, it is the first tool that runs the possible changes in the process tree while considering performance aspects such as resource, activity, and process capacity, accurate time, as well as business hours. These capabilities are directly used to address interactive process discovery as well as process improvement.

## 2 SIMPT

As shown in Figure 1, our tool automatically extracts all the existing aspects of a process based on general event log attributes. These quantitative aspects are used for running the process tree w.r.t. different scenarios. The main aspect is the discovered process tree which is used for interaction with users since it is limited to 4 operations. The user-interaction using process trees is easier compared to the complexity of Petri nets. The sequence, parallel, loop, and XOR operators are easy to be represented and understandable by the user for further changes. The process structure and the flow of activities are represented using these operators. We extend the implementations of the process tree method in [3] to generate a comprehensive tree including the probability of choices (XOR) and loop restrictions, e.g., the maximum length of traces and execution of loops. For instance, measuring the performance KPIs of a process in case that activity  $a$  and  $b$  are parallel instead of being sequential is possible. Not only the possible traces are generated but also generating the complete event log gives all the performance aspects of the new process, e.g., the length of the queues, the service time of cases, and other possible aspects from an event log. The provided insights also include the possibility of checking the newly generated behaviors by the new process structure (process tree) and configuration for conformance checking too.

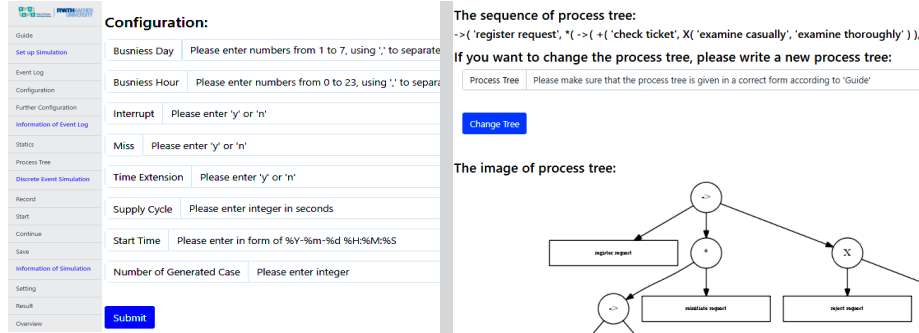


Fig. 2: A part of the tool parameters configuration to apply the changes to the basic performance elements as well as the process tree structure tab for the possible changes.

Table 1 shows the general overview of the parameters and the possibility for the user to interact with the process and reproduce the new process behavior (event log) w.r.t. the changes in that parameters. Here, we explain some of the main modules. All the details along with the tool source code and a representing video are presented extensively.<sup>3</sup>

The tool has three main modules. The first module runs the discovery, extracts the current parameters in the event logs, and presents the model and the parameters to the user to decide on the possible changes, e.g., process tree, deviations, or waiting time. For the performance analysis, both event logs with start and complete timestamps and only one timestamp can be considered. The activities' durations are also taken from their real averages and deviations. The second module is configuring the new process parameters for simulating and running the simulation for the given parameters, e.g., the number of cases to be generated or the start time of the simulation. Furthermore, the interruption concept for process, activities, and traces is introduced. The user can define whether in specific circumstances, e.g., when a running case or activities is passing the business hours, the interruption can happen and it is logged for the user. The last module is running and simulating the defined and configure process tree in which the results are presented as an overview as well as the possibility for downloading as an event log. The handover matrix of the process is also used to log the resources based on reality in the generated event log. The simulation is generating new events, e.g., the arrival of a new case or the start/end of an activity for a case, based on the clock of the system and configured properties, e.g., available resources. A part of the tool interface is shown in Figure 2, the *Guide* section in the tool provides features, possibilities, required steps, and information, extensively. The python library *simpy*<sup>4</sup> is used for discrete event simulation and handles the required system clock to generate new events.

<sup>3</sup> <https://github.com/mbafrani/SIMPT-SimulatingProcessTrees.git>

<sup>4</sup> [simpy.readthedocs.io](http://simpy.readthedocs.io)

### 3 Tool Maturity

The tool has been used in different projects to design/improve a process model interactively in different situations and generate different event logs. In the IoP project<sup>5</sup>, the tool is used to simulate multiple production lines to estimate the effect of the capacity of activities on the production process, e.g., average production speed. Moreover, [12] exploits the tool for car company production analyses, different arrival rates and activities' duration for the same process has been selected and the tool event logs are generated. Also, we use the tool as the base of the interactive design of the job-shop floor. The possible flow of jobs in the job-shop floor, i.e., the flow of activities, is presented as a process tree. These trees omit forbidden actions in the production line using the knowledge of the production manager and simulate the production line with the desired setting.

Figure 3 presents a sample scenario of changing the process structure and measuring the differences after the changes. Note that the inserted behaviors are generated based on the choices and loops in the rest of the process. As mentioned, having both simulated and original behavior of the process (with or without modifications) creates the possibility of the comparison between two processes which is available using the existing process mining tools and techniques. To demonstrate the tool functionality and validity of the re-generated event log, we used the BPI Challenge 2012 event log. We assessed the similarity of the original event log with the re-generated one using *Earth-Mover Distance* (EMD) technique as presented in [6] using the Python implementation in [15]. EMD calculates the shortest distance between the two event logs w.r.t. the minimum movement over the minimum distance between traces. The process is re-run without any changes in performance parameters to check the process behavior w.r.t. the flow of activities. The value of 0.34 as the EMD measure indicates the similarity of the two event logs. Note that the choices in the process model are the reason to have more behavior than the real event log which is expected w.r.t. *precision metrics* of discovery algorithm. Given the closeness of the simulation results and the original event log, the next changes can be applied to the process tree and other aspects of the process with enough confidence to reproduce the new event log including the effects of changes.

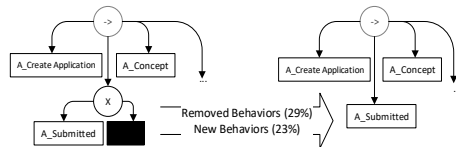


Fig. 3: A sample scenario for the process model of the BPI Challenge 2017 event log (application requests) in the tool. Activity *A-Create Application* can be skipped in the discovered process tree (left). By changing the choice to a sequence (right), i.e., this activity is required for all the cases, the removed and new inserted behaviors in the process can be measured.

<sup>5</sup> [www.iop.rwth-aachen.de](http://www.iop.rwth-aachen.de)

## 4 Conclusion

Given the fact that process improvement is the next step in the process mining path, simulation techniques will become more important. The combination of process mining and simulation techniques makes it possible to improve and re-design the processes w.r.t. the discovered possible change points in processes. Our tool is designed and implemented with the purpose of making process improvement using process mining and simulating the processes with the user's changes possible. The process tree notation along with all the performance and execution aspects of the process make the generated new behavior of the process w.r.t. user possible improvement reliable. Based on the provided reliable platform, the possibility of recommending the best process model interactively with the user considering both performance and activity flow is the next step.

## References

1. van der Aalst, W.M.P.: "Process mining - data Science in Action", second edition. Springer (2016)
2. van der Aalst, W.M.P.: Process mining and simulation: a match made in heaven! In: Computer Simulation Conference. pp. 1–12. ACM Press (2018)
3. Berti, A., van Zelst, S.J., van der Aalst, W.M.P.: Process mining for python (pm4py): bridging the gap between process- and data science. CoRR **abs/1905.06169** (2019)
4. Camargo, M., Dumas, M., Rojas, O.G.: Simod: a tool for automated discovery of business process simulation models. In: Demonstration Track at BPM (2019)
5. Gawin, B., Marcinkowski, B.: How close to reality is the as-is business process simulation model? Organizacija **48**(3), 155 – 175 (2015)
6. Leemans, S.J.J., Syring, A.F., van der Aalst, W.M.P.: Earth movers' stochastic conformance checking. In: BPM Forum 2019. pp. 127–143 (2019)
7. Netjes, M., Reijers, H., Aalst, W.M.P.: The PRICE tool kit: tool support for process improvement. CEUR Workshop Proceedings **615** (01 2010)
8. Pourbafrani, M., van der Aalst, W.M.P.: PMSD: data-driven simulation using system dynamics and process mining. CoRR **abs/2010.00943** (2020), <https://arxiv.org/abs/2010.00943>
9. Pourbafrani, M., van der Aalst, W.M.P.: Extracting process features from event logs to learn coarse-grained simulation models. In: Advanced Information Systems Engineering. Springer International Publishing, Cham (2021)
10. Pourbafrani, M., Vasudevan, S., Zafar, F., Xingran, Y., Singh, R., van der Aalst, W.M.P.: A python extension to simulate petri nets in process mining. CoRR **abs/2102.08774** (2021)
11. Pourbafrani, M., van Zelst, S.J., van der Aalst, W.M.P.: Scenario-based prediction of business processes using system dynamics. In: On the Move to Meaningful Internet Systems: COOPIS 2019 Conferences, 2019. pp. 422–439 (2019). [https://doi.org/10.1007/978-3-030-33246-4\\_27](https://doi.org/10.1007/978-3-030-33246-4_27)
12. Pourbafrani, M., van Zelst, S.J., van der Aalst, W.M.P.: Semi-automated time-granularity detection for data-driven simulation using process mining and system dynamics. In: Conceptual Modeling - 39th International Conference, ER 2020. vol. 12400, pp. 77–91. Springer (2020). [https://doi.org/10.1007/978-3-030-62522-1\\_6](https://doi.org/10.1007/978-3-030-62522-1_6)

13. Pourbafrani, M., van Zelst, S.J., van der Aalst, W.M.P.: Supporting automatic system dynamics model generation for simulation in the context of process mining. In: 23rd International Conference, BIS 2020, Proceedings. pp. 249–263 (2020). [https://doi.org/10.1007/978-3-030-53337-3\\_19](https://doi.org/10.1007/978-3-030-53337-3_19)
14. Pufahl, L., Wong, T., Weske, M.: Design of an extensible BPMN process simulator. In: Proceedings of Demonstration Track at BPM 2017. pp. 782–795
15. Rafei, M., van der Aalst, W.M.P.: Towards quantifying privacy in process mining. In: International Conference on Process Mining - ICPM (2020), International Workshops. pp. 1–13 (2020)
16. Rogge-Solti, A., Weske, M.: Prediction of business process durations using non-markovian stochastic Petri nets. *Inf. Syst.* **54**, 1–14 (2015)
17. Rozinat, A., Mans, R.S., Song, M., van der Aalst, W.M.P.: Discovering simulation models. *Inf. Syst.* **34**(3), 305–327 (2009)
18. Westergaard, M.: CPN Tools 4: Multi-formalism and extensibility. In: 34th International Conference, Petri net 2013. Proceedings. pp. 400–409 (2013)