Interactive Process Improvement Using Simulation of Enriched Process Trees

Mahsa Pourbafrani(✉) and Wil M. P. van der Aalst

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

Abstract. Event data provide the main source of information for analyzing and improving processes in organizations. Process mining techniques capture the state of running processes w.r.t. various aspects, such as activity-flow and performance metrics. The next step for process owners is to take the provided insights and turn them into actions in order to improve their processes. These actions may be taken in different aspects of a process. However, simply being aware of the process aspects that need to be improved as well as potential actions is insufficient. The key step in between is to assess the outcomes of the decisions and improvements. In this paper, we propose a framework to systematically compare event data and the simulated event data of organizations, as well as comparing the results of modified processes in different settings. The proposed framework could be provided as an analytic service to enable organizations in easily accessing event data analytics. The framework is supported with a simulation tool that enables applying changes to the processes and re-running the process in various scenarios. The simulation step includes different perspectives of a process that can be captured automatically and modified by the user. Then, we apply a state-of-the-art comparison approach for processes using their event data which visually reflects the effects of these changes in the process, i.e., evaluating the process improvement. Our framework also includes the implementation of the change measurement module as a tool.

Keywords: Process mining · Business process improvement · Process simulation · Earth mover's distance · Performance spectrum

1 Introduction

Process owners use data-driven process mining techniques to improve their processes. The discovered process models, their performance states, and hidden problems, such as deviations and bottlenecks, are critical to process improvement. The process mining techniques in the process discovery and conformance checking areas are widely used to illustrate the current states of processes and their potential problems [1]. However, before taking any action based on process mining diagnostics, one wants to have an estimation of the impact. To do so, it is required to play out the processes with the process owners' adjustments

H. Hacid et al. (Eds.): ICSOC 2021 Workshops, LNCS 13236, pp. 61–76, 2022.
https://doi.org/10.1007/978-3-031-14135-5_5

and then assess the effects of the actions. To improve processes in an evidence-based manner, *forward-looking* process mining techniques such as prediction and simulation are needed. They enable what-if and scenario-based analyses of business processes. However, the validity of the generated results, as well as their clear interpretation, are two determining factors when employing these techniques. The model's reliability can be improved by incorporating process mining insights, e.g., the designed simulation model is derived directly from the process's historical event data [2].

Techniques such as generating CPN models [12,14,22] and BPMN Models [3] have been proposed for generating simulation models of processes based on event logs. Simulation approaches in process mining are also useful for other applications. In [23], for example, process model simulations are used to estimate the alignment value. The gap that we aim to fill is not only providing a platform for users to easily re-run their processes using the automatically generated simulation models but also a more accurate technique for measuring improvement/changes w.r.t. the process owners' interactions with the process. The conventional comparison of two processes includes conformance checking between the event logs and the corresponding process models. In addition, for the purpose of performance comparison, general performance metrics are usually considered. Most of the current approaches are not detailed enough in both aspects, i.e., conformance checking and performance analysis. These techniques do not measure and reflect the effect of changes at the detailed level. For instance, the existing conformance checking techniques only return a value such as the fitness of two event logs, or one event log and the corresponding process model [4]. These techniques also neglect the importance of the frequency of process instances. The detailed distance between the original event log and the regenerated event log is critical for determining their similarity [19].

In this paper, we propose an approach to systematically compare the event data of a process with its simulated event data to assess the reliability of the simulation model, i.e., the accuracy of the simulation. As a result, the simulated processes in different settings can be compared. The simulation module is implemented as a new software capturing different process perspectives, in which the event logs are used to enrich the process models (trees) with existing aspects. The enriched process trees generate process behaviors in the form of event logs with/without applied changes to the process. The state-of-the-art comparison framework is then applied to the results of the simulation. It measures the effects of changes using detailed conformance and performance techniques. To demonstrate a proof of concept of the framework, we use a sample process as an example to illustrate the approach steps. Then, we employ a real-life event log to evaluate the approach.

The remainder of this paper is structured as follows. We present the related work in Sect. 2. In Sect. 3, we introduce background concepts and notations. In Sect. 4, we present our main approach. We evaluate the approach in Sect. 5 by designing simulation models, and Sect. 6 concludes this work.

2 Related Work

Process mining enables designing data-driven simulation models of processes [2]. Authors in [22] use different aspects of a process using its event data, e.g., process models, resource pooling, and performance metrics, and automatically generate simulation models. This work as a pioneer in the data-driven simulation in process mining translates insights from event data into the process simulation parameters. Other simulation approaches in process mining follow the same direction. For instance, [21] uses *stochastic Petri nets* to simulate processes and determine the duration of instances in business processes. In [18] a business simulation model is generated which is based on the user domain knowledge. Tools based on *Protos* try to reduce modeling efforts by introducing the reference process models [24]. [9] discusses how process mining insights can be exploited in the business process simulation context. As an example, the proposed tool in [3] presents the idea of combining BPMN and process mining for simulation purposes, where indicators for measuring the accuracy of the simulation results are also introduced.

In [11,15], different levels of simulating processes are proposed where all the aspects of a process are extracted at different levels, i.e., not only instance level but also higher-level, e.g., describing processes per day quantitatively. The examples of high-level simulations are presented in [16,17] with the use of the designed tool for the modeling and data extraction steps in [10]. In our approach, the enriched process models, e.g., process trees, accuracy of the performance-related aspects, effortless interaction with users, and social network analysis (resource aspects) are the main criteria for designing simulation models.

On the other side, visualization techniques are powerful tools in process mining analysis in both descriptive and predictive analyses. There are a couple of visualization techniques that are able to represent the process w.r.t. different process aspects for providing visual inspection or process comparison. For instance, the performance spectrum [5] represents the process performance behaviors in detail between every two sets of activities in the process. i.e., process segments. The stochastic conformance checking method used in [20] considers the frequency of the traces in two event logs while comparing their differences. The idea of using *Earth Mover's Distance* for conformance checking and comparing two event logs, or event logs and process models enables assessing the difference of two processes w.r.t. their behaviors in detail.

We provide a platform for regenerating a process in different settings and measure the effects of changes/results using our designed modules based on the presented ideas. The presented tool in [13] is the simulation approach taken in the current work as the intermediate tool for regenerating the process behaviors. The process trees are automatically generated and enriched with the probability and performance information and allow us to change the processes w.r.t. the activity-flow and performance aspects.

Table 1. A part of a sample event log. Each row represents an event.

	Case ID	Activity	Resource	Timestamp
e_1	1	Register request	Pete	12/30/2010 11:02
e_2	2	Register request	Mike	12/30/2010 11:32
	3	Register request	Pete	12/30/2010 14:32
...	1	Examine thoroughly	Sue	12/31/2010 10:06
	2	Decide	Sara	1/5/2011 11:22
	1	Decide	Sara	1/6/2011 11:18
	1	Reject request	Pete	1/7/2011 14:24
e_n	...	...	...	...

3 Preliminaries

In this section, we establish the basic notations for events, event logs, and process trees which are used in the framework.

Definition 1 (Event). *Let $\mathcal{A}$ be the universe of activities, $\mathcal{T}$ be the universe of timestamps, $\mathcal{R}$ be the universe of resources, and $\mathcal{C}$ be the universe of case identifier. An event e is a tuple $e = (c, a, r, t)$ where activity a at time t for case c is performed by resource r. $\mathcal{E} = \mathcal{C} \times \mathcal{A} \times \mathcal{R} \times \mathcal{T}$ is the universe of events. For each $e \in \mathcal{E}$, $\pi_D(e)$ projects e on the attribute from domain D, e.g., $\pi_{\mathcal{A}}(e) = a$.*

Definition 2 (Trace). *Let $\mathcal{E}$ be the universe of events, a trace $\sigma \in \mathcal{E}^*$ is a finite sequence of events. For each $\sigma = \langle e_1, ..., e_n \rangle$, $e_i \in \sigma$ happens at most once and for each $e_i, e_j \in \sigma, \pi_C(e_i) = \pi_C(e_j) \wedge \pi_{\mathcal{T}}(e_i) \leq \pi_{\mathcal{T}}(e_j)$, if $1 \leq i < j \leq n$. For $\sigma = \langle e_1, ..., e_n \rangle \in \mathcal{E}^*$, $\Pi_D(\sigma) = \langle \pi_{\mathcal{D}(e_1)}, \pi_{\mathcal{D}(e_2)}, ..., \pi_{\mathcal{D}(e_n)} \rangle$ is the projection of trace σ on the attribute from domain D, e.g., $\Pi_{\mathcal{A}}(\sigma) = \langle \pi_{\mathcal{A}(e_1)}, \pi_{\mathcal{A}(e_2)}, ..., \pi_{\mathcal{A}(e_n)} \rangle$.*

Definition 3 (Event Log). *Let $\mathcal{E}$ be the universe of events and $\mathcal{E}^*$ be the set of possible traces, we define an event log L as a set of traces, i.e., $L \subseteq \mathcal{E}^*$.*

We denote $L_{\mathcal{A}} = [\Pi_{\mathcal{A}}(\sigma) | \sigma \in L]$ as the multiset of traces projected on the activity attribute. Furthermore, $\widetilde{L_{\mathcal{A}}} = \{\sigma \in L_{\mathcal{A}}\}$ is the set of unique traces (variants) projected on the activity attribute in the event log L. We refer to $\widetilde{L_{\mathcal{A}}}$ as the set of process behaviors presented in L.

Table 1 represents a part of a sample event log, where each row indicates an event, e.g., considering the first row as e_1, $\pi_{\mathcal{C}}(e_1) = 1$ and $\pi_{\mathcal{A}}(e_1) =$ *register request*. Process mining utilizes such event logs to discover running processes inside organizations. The process models are the representative ways of the discovered running processes. The process tree notation is one of the common approaches to present a process, where the nodes of trees are operators and leaves are activities in the process.

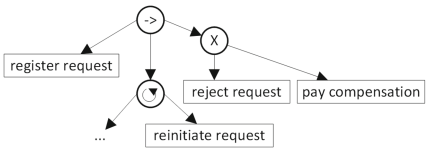

Fig. 1. A part of the discovered process tree for the sample event log.

A part of the process tree representing the example process is shown in Fig. 1. For example, there is a choice, i.e., XOR ($\times$) as a node between activity *reject request* and *pay compensation* indicating that in the process either a request is rejected or the compensation is paid. The root node ($\rightarrow$) indicates that activity *register request* is always followed by a loop ($\circlearrowleft$). A loop represents a redo of works between its children, i.e., activities in the leaves of a loop node may happen multiple times in a trace. Furthermore, the notation of τ is for silent activities which are not visible in the process but used for the representation of process trees.

Definition 4 (Process Tree). *Let L be an event log, $A_L = \{a \in \sigma | \sigma \in \widetilde{L_{\mathcal{A}}}\}$ be the set of activities in L and $Op = \{\rightarrow, \times, \circlearrowleft, +\}$ be the set of process operators. If $a \in A_L \cup \{\tau\}$, then $Q = a$ is a process tree. If $n \geq 1$, $Q_1, Q_2, Q_3, ..., Q_n$ are process trees, and $op \in \{\rightarrow, \times, +\}$, then $Q = op(Q_1, Q_2, ..., Q_n)$ is a process tree. If $n \geq 2$ and $Q_1, Q_2, Q_3, ..., Q_n$ are process trees, then $Q = \circlearrowleft(Q_1, Q_2, ..., Q_n)$ is a process tree. For a process tree Q, we denote Q_a and Q_{op} as the set of activities and the set of operators in Q.*

For a given process tree Q, $Q_w = Q_{op} \times Q_a$ is the set of edges connecting operators to activities. For instance, $(\rightarrow, register\ request)$ is an edge in the example process tree in Fig. 1 where *register request* is child of the tree under parent $\rightarrow$. Note that a process tree may also contain edges from an operator to an operator, which is not relevant in the implementation of our framework.

4 Approach

Our framework enables interactive process improvement inside organizations for designing/improving process models. The current behaviors of processes captured in the form of an event log serve as the starting point for any improvement. To enrich the discovered process models, process discovery, performance analysis, and social network analysis (resource perspective) techniques are used. We use the *Discrete Event Simulation* (DES) technique as a tool to play out the process with the current states, which results in an event log as shown in Fig. 2. The original behavior of the event log w.r.t. activity-flow (process behavior) and performance metrics are compared to ensure that the automatically designed simulation model is reliable and behaves close to reality, *Improvement Measurement* module in Fig. 2. This step allows the user to change the process parameters and re-run the process to generate the new behavior and measure the process improvement, depicted by the dotted lines in Fig. 2. These measurements are presented in a numerical format as well as in a detailed graphical format. The detailed comparative visualization increases the interaction between the framework and the user. First, we explain the automatic generation of the simulation results, including process mining techniques and enriching the process model, and continue with the *Improvement Measurement* module.

4.1 Simulating Process Trees

Enriching Process Trees. The inductive miner algorithm [7] is used to discover the process model since it is capable of capturing all the behaviors in

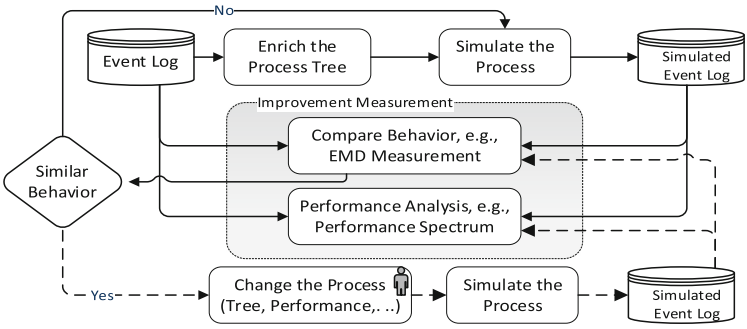

Fig. 2. The overview of the framework to improve the processes interactively. The straight lines show the path to assess the quality of the regenerated behavior by the simulation model w.r.t. activity-flow and performance metrics. The dotted lines illustrate the path that the user is able to change the process and measure and observer the improvement, i.e., the effect of changes, in the process.

a process in the form of a process model. The generated process tree by the inductive miner algorithm is able to represent the traces in the event log. The process tree's limited number of operators as defined in Definition 4 allows for easy understanding and modification of the process. To play out the process accurately, i.e., applying the new changes in the process, more information than the flow of activities provided by the process tree is required.

The tree should be enriched with the probability of activity-flows, performance information of the activities, and the corresponding resource information, e.g., organizations of the resources, the number of resources in each organization, and hand-over of activities between resources, for each activity from the real process. Therefore, the probability of the choices and the possible number of loops should be taken into account for regenerating a similar event log. Furthermore, for a process tree Q and the edge $w = (op, a) \in Q_w$, w_a represents the probability of occurrence of activity a in a generated trace from the process tree. For the edge $w = (op, a) \in Q_w$, if $op \in \{\rightarrow, +, \circlearrowleft\}$, then $w_a = 1$. Note that to avoid the generation of infinite traces due to the loops in the process tree, we limit the execution of loops in the simulation with the probability of the number of occurrences of a loop on average in a trace and the maximum times that a loop happens in a trace. For all activities $a \in Q_a$, there is a binding performance metric, i.e., the average duration of each activity. Moreover, the activities are assigned to the existing automatically discovered organizations and the capacity of the resources.

For the example process shown in Fig. 1, a part of the automatically enriched tree with the activity-flow, performance, and resource information is presented in Fig. 3. For instance, for the process edge $w = (\times, reject\ request)$, $w_{rej} = 0.5$, and the shown loop in the process can be executed at most 2 times in a trace and the probability of its occurrence is 30% which is derived from the event log. Activity *register request* takes on average 43000 s to be performed,

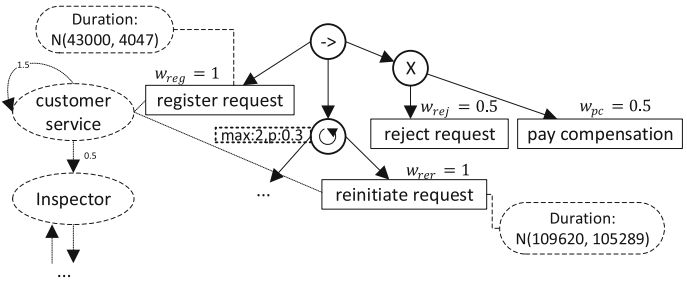

Fig. 3. Enriched tree with the probability information, resource allocation, and duration of each activity. The enriched process tree can be simulated. The hand-over of resources is shown (left) to provide more accurate simulation results (event logs) w.r.t. the resource allocation in the organizations.

Table 2. The general list of automatically discovered insights using process mining techniques to form process simulations. The top row shows what is discovered from event data. The bottom row shows what can be set or change by the user.

	Process mining										Simulation execution parameters	
	Process model (tree)	Arrival rate	Activity duration, deviation	Activities capacity	Activities unique resources (shared resources)	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	+	+	+	+	+	+	+	+	+	+	+	+

and the average is used for simulating its duration using a normal distribution. Also, *register request* and *reinitiate request* belong to the *customer service* organization where the resources in this organization hand over tasks to the *inspector* organization.

The information extracted from event logs is shown in Table 2. This information, along with the discussed information for enriching process trees are the required simulation parameters. Moreover, the changeable aspects for process improvement by the user in the simulation step are specified in detail. The discovery and design of the simulation models including generating event logs as a result of the simulation models are represented in detail in [13].

4.2 Measuring the Process Improvement

To measure the changes in the newly generated process represented with an event log, we have to compare two event logs. For comparing two processes, i.e., event logs, two major aspects of the processes should be considered, activity-flow which generates the behaviors, and the performance aspects. Note that the intermediate regenerator tool can be different from the one that we use in our framework, and yet the *Measuring the Process Improvement* module can be used for measuring the effect of changes in two event logs.

Table 3. A sample example of EMD measurement for two event logs [19]. The reallocation function allocates the 49 traces in L to 49 traces with activity-flow $\langle a, e, c, d\rangle$ and 1 remaining trace to $\langle a, b, c, d\rangle$ in L'. The sum of the value of the table indicates the general EMD value, i.e., the difference between the two event logs. Each cell represents the minimum cost to map its corresponding trace in the original event log (row) into the traces in the simulated event log (column).

$L_{\mathcal{A}}$	$L'_{\mathcal{A}}$			
	$\langle a, b, c, d\rangle$	$\langle a, c, b, d\rangle$	$\langle a, e, c, d\rangle^{49}$	$\langle a, e, b, d\rangle^{49}$
$\langle a, b, c, d\rangle^{50}$	$\frac{1}{100} \times 0$	0×0.5	$\frac{49}{100} \times 0.25$	0×0.5
$\langle a, c, b, d\rangle^{50}$	0×0.5	$\frac{1}{100} \times 0$	0×0.5	$\frac{49}{100} \times 0.25$

Activity-Flow Behaviors. The fact that process trees include silent transitions, loops, and XOR operators makes generating more behavior (new traces) than the existed ones in the original log possible. Therefore, the similarity of behaviors is one of the main indicators in the comparison step.

Given two event logs, the original event log L and the simulated event log L', we show the presented behaviors in each event log using their set of unique traces, i.e., $\widetilde{L}_{\mathcal{A}}$, $\widetilde{L'}_{\mathcal{A}}$. The new generated behaviors in the simulated event log, i.e., not existing in the original event log, and the removed behaviors from the original process are calculated as $\widetilde{L'}_{\mathcal{A}} \backslash \widetilde{L}_{\mathcal{A}}$, and $\widetilde{L}_{\mathcal{A}} \backslash \widetilde{L'}_{\mathcal{A}}$, respectively. Therefore, $\frac{|\widetilde{L'}_{\mathcal{A}} \backslash \widetilde{L}_{\mathcal{A}}|}{|\widetilde{L}_{\mathcal{A}} \cup \widetilde{L'}_{\mathcal{A}}|}$ and $\frac{|\widetilde{L}_{\mathcal{A}} \backslash \widetilde{L'}_{\mathcal{A}}|}{|\widetilde{L}_{\mathcal{A}} \cup \widetilde{L'}_{\mathcal{A}}|}$ are the fraction of the new and removed behaviors, respectively.

These metrics represent the pairwise difference between two event logs. They evaluate whether the simulation of the original log is close to reality, as well as capturing any different behavior added/removed due to the changes in a process tree (flow of activities). In the example process presented in Fig. 1, after regenerating the process without any change multiple times, on average 22% of the generated variants (unique traces) in the simulated logs are newly generated. The sample event log for further experiments with the tools is publicly available[1]. Furthermore, the precise comparison of two event logs should be based on their behavior, taking into account the frequency of the behavior. To determine the difference between the original and the simulated event logs, we employ a stochastic conformance checking approach.

Earth Mover's Distance Conformance Checking. To accurately compare two event logs' behaviors, we use the probability distance of each two traces in two event logs based on Earth Mover's Distance (EMD). To calculate the EMD measurement between two event logs, we use the conformance techniques presented in [6]. For every trace in the original log, we calculate the movement of its frequency to all the traces in the simulated event log using the reallocation function. As the next step, the cost of the movement is considered using the trace distance function.

[1] https://github.com/mbafrani/VisualComparison2EventLogs.

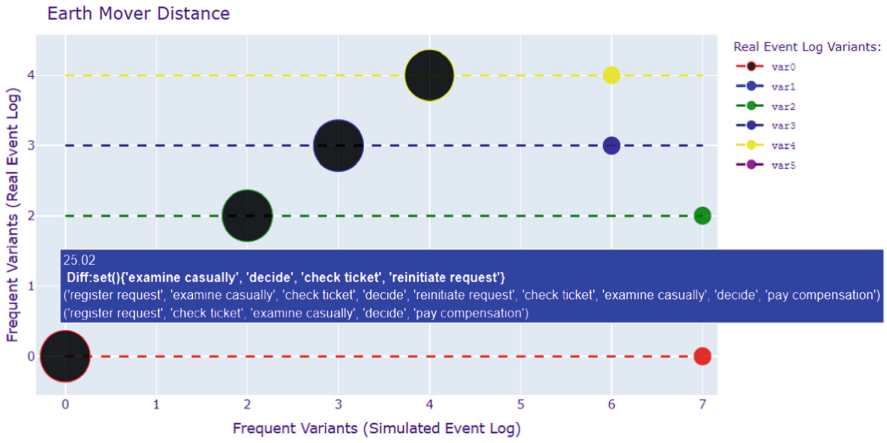

Fig. 4. The detailed comparison of two event logs for the sample process, i.e., the results of EMD measurement. It is the results of the EMD *reallocation* and *trace distance* functions in the form of a table such as Table 3. The points are the proportional cost of moving every trace in one event log to the simulated event log. Each row (color) indicates a trace in the original event log. The black points are similar traces in both event logs. The sizes of the points are the relative costs of movement for each variant (unique traces) in the original event logs.

Reallocation. Let L and L' be the original and the simulated event logs, respectively. Function $r \in \widetilde{L}_{\mathcal{A}} \times \widetilde{L'}_{\mathcal{A}} \rightarrow [0,1]$ returns the relative frequency of $\sigma \in \widetilde{L}_{\mathcal{A}}$ that should be transformed to $\sigma' \in \widetilde{L'}_{\mathcal{A}}$, i.e., $r(\sigma, \sigma')$. Note that for all $\sigma \in \widetilde{L}_{\mathcal{A}}$, $\frac{L_{\mathcal{A}}(\sigma)}{|L_{\mathcal{A}}|} = \sum_{\sigma' \in \widetilde{L'}_{\mathcal{A}}} r(\sigma, \sigma')$, i.e., the frequency of each $\sigma \in \widetilde{L}_{\mathcal{A}}$ is considered properly. The same should be considered for each $\sigma' \in \widetilde{L'}_{\mathcal{A}}$.

Trace Distance. The distance between each two traces in the original log and the simulated logs is calculated based on *normalized string edit distance* (Levenstein) [8]. Function $d \in \mathcal{A}^* \times \mathcal{A}^* \rightarrow [0,1]$ calculates the distance between two traces, where for two similar traces the value is 0 and $d(\sigma, \sigma') = d(\sigma', \sigma)$.

To represent the algorithm clearly, we reduced the sample process and presented a couple of traces in Table 3. The EMD measurement of the two event logs is $EMD(L_{\mathcal{A}}, L'_{\mathcal{A}}) = \min_{r \in R} r.d = \sum_{\sigma \in \widetilde{L}_{\mathcal{A}}} \sum_{\sigma' \in \widetilde{L'}_{\mathcal{A}}} r(\sigma, \sigma') d(\sigma, \sigma')$ where R is the universe of reallocation functions. Table 3 represents a sample EMD measurement for two sample event logs L and L'. For instance, for $\langle a, b, c, d \rangle^{50}$ in L and $\langle a, e, c, d \rangle^{49}$ in L', the trace distance value is 0.25 given the differences between two traces using *normalized string edit distance* (Levenstein). The reallocation value is 0.49, i.e., 49 of 100 traces in L are reallocated to 49 traces with the sequence $\langle a, e, c, d \rangle$ in L'. Therefore, the minimum effort of mapping the one trace to the second one is $0.49 * 0.25 = 0.122$. Besides the EMD value of two event logs that indicates how two event logs are different, we are interested in the required effort for every trace in the original event log to be mapped/transformed into the simulated event log for accurate comparison of the simulation results.

Applying the designed EMD measurement to the complete sample process and its simulated event log without any changes, Fig. 4 illustrates the result. The unique traces in the original event log and the unique traces in the simulated event log are depicted using the x-axis and the y-axis, respectively. If we assume that in our example, $r \in R$ is the reallocation function, the cost of EMD (effort of mapping) for each point of Fig. 4 shows the relative effort, i.e., $\sigma \in \widetilde{L}_{\mathcal{A}}$ and for each $\sigma'_i \in \widetilde{L'}_{\mathcal{A}}$, $effort_{L_{\mathcal{A}},L_{\mathcal{A}}}(\sigma,\sigma'_i) = \frac{d(\sigma,\sigma'_i).r(\sigma,\sigma'_i)}{\sum_{\sigma' \in \widetilde{L'}} d(\sigma,\sigma').r(\sigma,\sigma')}$. The most frequent trace in the original event log (first row) will be converted to the $(74.98\%, 0, 0, 0, 0, 0, 0, 25.02\%)$, i.e., points in the first row. The values indicate that to map the first trace in the original event log (most frequent one) to the simulated event log 74.98% of the effort is to map it to the first (most frequent trace) in the simulated event log, i.e.,$effort_{L_{\mathcal{A}},L_{\mathcal{A}}}(\sigma_1,\sigma'_1) = 75\%$. Also, each row illustrates the minimum required effort to map/transform the traces into the simulated event log.

Performance Behaviors. Performance is the second factor to consider when assessing improvement/changes. However, because the times are abstracted from the real data in prediction and simulation techniques, exact measurements are impossible. It is worth noting that in many cases, time-related parameters such as the duration of simulation events are generated using a random function, e.g., normal distribution in our case. General performance KPIs at a high level of aggregation, e.g., the average waiting time of traces, or average service time are too abstract to represent the effects of the changes in the process. Therefore, besides the usual metrics, we use the performance spectrum, which relies on the structure of the process and directly reflects the effects of changes in specific parts of the process on others. For instance, changing the current service time of the activity *examine thoroughly* in the example process has an impact not only on the overall metrics but also on the duration of the later activities in the traces, e.g., *decide* or *reinitiate request*.

Aggregated Performance Spectrum. Performance Spectrum is a concept introduced to visualize the performance of process steps at the detailed level. A process segment in event log L is a step from activity a to activity b, i.e., $(a,b) \in A_L \times A_L$ is a process segment in L where $A_L = \{a \in \sigma | \sigma \in \widetilde{L}_{\mathcal{A}}\}$. Each occurrence of a segment in a trace allows measuring the time between occurrences of a and b [5]. We define the set of all tuples of events that are directly followed in the traces in L as $SEG^L = \{(e_i, e_{i+1}) | \exists_{\sigma=\langle e_1,e_2,\ldots,e_n\rangle \in L} e_i, e_{i+1} \in \sigma\}$. The projection of the events in SEG^L on their activity attribute provides the multiset of process segments, i.e., $SEG^L_{\mathcal{A}} = [(\pi_{\mathcal{A}}(e_1), \pi_{\mathcal{A}}(e_2)) | (e_1,e_2) \in SEG^L]$. For instance, $[(examine\ thoroughly, decide)^{17}, (examine\ thoroughly, reinitiate\ request)^{20}]$ is the part of the multiset of segments in our example.

We consider two aspects for representing a process segment in an event log: *average time* of the segment and *frequency* of the segment. For $seg = (a,b) \in SEG^L_{\mathcal{A}}$, function $PS(seg, L) = (AvgTime(seg, L), Freq(seg, L))$ represents the frequency of the process segment seg and the corresponding average time difference for the segment. For $seg = (a,b) \in SEG^L_{\mathcal{A}}$, we define

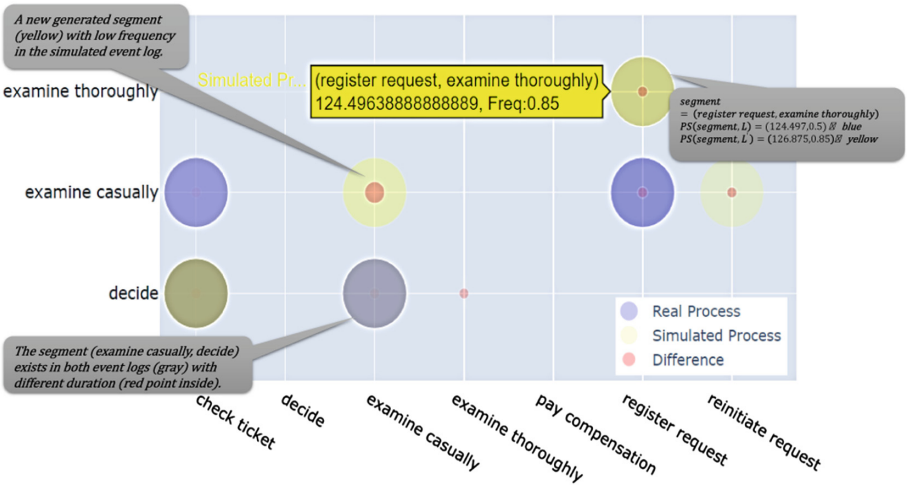

Fig. 5. Part of the performance measurement for the example process based on the aggregated performance spectrum. Each event log is represented by a different color, i.e., blue for the original and yellow for the simulated one. Overlapping segments are represented by the gray color (same duration between segments). Each point's transparency and size indicate the frequency and duration of the segment in the event logs. (Color figure online)

$AvgTime(seg, L) = Avg(\{\pi_{\mathcal{T}}(e_2) - \pi_{\mathcal{T}}(e_1) | (e_1, e_2) \in SEG^L \wedge \pi_{\mathcal{A}}(e_1) = a \wedge \pi_{\mathcal{A}}(e_2) = b\})$ and $Freq(seg, L) = SEG^L_{\mathcal{A}}((a, b))$.

Figure 5 is the result of the introduced performance measurement (PS) for the example process and the regenerated event log. In order to represent different aspects of the results, e.g., new/eliminated segments and different duration, we performed the simulation based on the changed process. For instance, given L and L' as the original and simulated event logs, each segment's colors refer to an event log, the size refers to the average time difference between the segments, and the transparency indicates the frequency (darker means more frequent). The gray color represents the overlapped segment in two event logs with similar performance metrics, and the yellow points represent the new segment generated in the simulated event log as a result of process tree choices. The implementation also includes the option to display only the difference (red points).

5 Evaluation

A real event log representing the process of taking loans by customers inside a financial company, known as the *BPI challenge 2012*, is used in this section. First, we simulate a similar process with different configurations and assess how close they are to the original event log. Following that, we alter the activity-flow of the process model in order to improve the process and evaluate the effect of the applied changes. Having both simulated and original behaviors of the

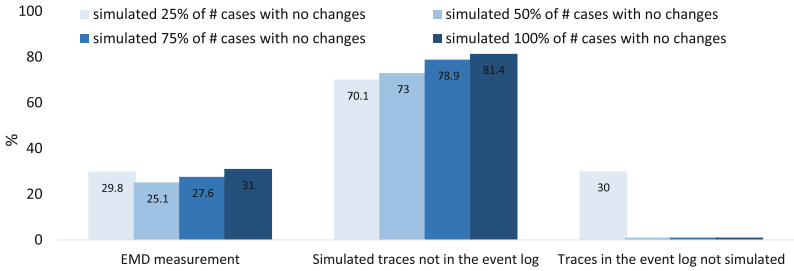

Fig. 6. The comparison of the generated event logs using simulating a specific number of traces in the original event log (BPI Challenge 2012). The EMD measurement indicates, how the original and the simulated event logs are different.

process (with or without modifications) the possibility of comparing between two processes is easily provided. To do so, we used our tool *SIMPT*[2] for simulating the process, and our developed modules for comparing two event logs w.r.t. the detailed performance and control flow aspects[3]. The provided tools make it possible to evaluate the framework for the interactive improvement of different processes for different event logs.

We start with automatically discovering and enriching the underlying process tree before regenerating the process, where the similarity of the two event logs indicates the possibility of using the simulation models for further investigation. Therefore, we simulated the event log multiple times without applying any changes. As shown in Fig. 6, we took a specific percentage of the total number of traces in the process for each round of simulation of the original process. As expected, when the number of simulated traces is small, there is a chance of missing specific process behaviors, e.g., using 25% of the number of traces, we lost 30% of the behaviors (unique traces). On the other hand, increasing the number of simulated traces increases the number of new behaviors. Since the generation of the traces (activity-flow) is based on probability and the process tree includes both XOR choices and silent transitions, the new behaviors are expected to be generated.

Afterward, in the process tree of the original process, we changed the optional activity *preaccepted* to be a mandatory activity for all the traces that are going to be *accepted* in Fig. 7. The structure of the process tree (activity-flow) is changed from $\rightarrow (submitted, partlysubmitted, \times(\tau, preaccepted), \times(\tau, accepted), \times(\tau, finalized), \times(declined, cancelled))$ to $\rightarrow (submitted, partlysubmitted, \times(\tau, \rightarrow (preaccepted, accepted)), \times(\tau, finalized), \times(declined, cancelled))$. Note that these changes are possible in different aspects of the process such as the process model, performance metrics, e.g., activity duration, arrival rate of the traces, or capacity of the resources.

Based on the shown results of simulating the original event log without any changes in Fig. 8, we simulated the changed process model with 50% of original

[2] https://github.com/mbafrani/SIMPT-SimulatingProcessTrees.
[3] https://github.com/mbafrani/VisualComparison2EventLogs.

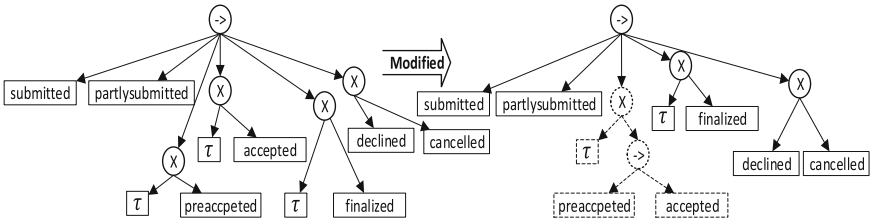

Fig. 7. The process tree for handling application in the BPI challenge 2012 event logs (left). To evaluate the approach, the optional activity *preaccepted* is changed to be mandatory in the flow of activities for all the traces (right) in the process. Dotted lines indicate the parts of the tree that have changed.

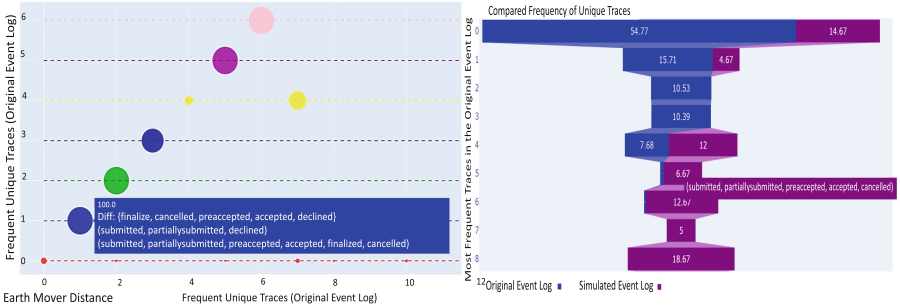

Fig. 8. The detailed comparison of the changed process and the original process model. The detailed EMD diagram (left) shows the differences of the two event logs w.r.t. the activity-flow and the comparing frequent chart (right) represents the preserved and removed behavior in the simulated process as the effect of the applied changes.

traces. In the proposed scenario (changed process tree), 63.6% of generated behaviors (unique traces) are new. However, it is less than the behaviors in the simulated event log without any modifications, since we removed one of the XOR choices limiting the possibilities of producing new behaviors. On the other hand, 23% of the behaviors due to the change in the process tree are eliminated, i.e., the traces that skipped the activity *preaccepted* in the original process. Also, in Fig. 6, the pairwise comparison of the traces (right), as well as the detailed EMD companions for the cost of the mapping of two event logs (left) after the changes, are shown. The applied changes in the process model not only affected the process behavior but also these changes affected the performance of the later segments in the process, e.g., the duration for the process segment *accepted* and *finalized* increased while activity *finalized* was not changed. The provided detailed comparison along with the intermediate simulation tool enables the possibility of capturing these types of unexpected insights. Note that the reliability of the simulation techniques such as the presented ones in Sect. 2 can be assessed using the measurement modules.

6 Conclusion

Process mining supports organizations in finding running processes, as well as identifying challenges or possible areas for improvement. The process improvement should be supported with process knowledge. We use process mining insights and simulation models as an intermediate method to regenerate processes in various scenarios. The framework begins with an event log, discovers a process tree, and enriches it with all the knowledge needed to regenerate the process. The similarity of the simulated results and the original process behavior in the form of an event log is then measured in the next step. The degree of similarity reflects the accuracy of our model. As a result, the improvement of the change in the process can be played out, and the impact of changes can be tracked using the same measurement module in both the activity-flow and performance aspects of the process. The advantage of our framework in both generating simulation models and enriching them based on event logs automatically, and the new representation of the comparing of the event logs. Furthermore, the intermediate simulation technique described in this paper can be replaced with other simulation techniques capable of generating event logs for the specified changes.

Acknowledgments. Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy-EXC-2023 Internet of Production - 390621612. We also thank the Alexander von Humboldt (AvH) Stiftung for supporting our research.

References

1. van der Aalst, W.M.P.: Process Mining - Data Science in Action, 2nd edn. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49851-4
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. Camargo, M., Dumas, M., Rojas, O.G.: Simod: a tool for automated discovery of business process simulation models, pp. 139–143 (2019)
4. Carmona, J., van Dongen, B.F., Solti, A., Weidlich, M.: Conformance Checking - Relating Processes and Models. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-99414-7
5. Denisov, V., Fahland, D., van der Aalst, W.M.P.: Unbiased, fine-grained description of processes performance from event data. In: Weske, M., Montali, M., Weber, I., vom Brocke, J. (eds.) BPM 2018. LNCS, vol. 11080, pp. 139–157. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-98648-7_9
6. Leemans, S.J.J., Syring, A.F., van der Aalst, W.M.P.: Earth movers' stochastic conformance checking. In: Hildebrandt, T., van Dongen, B.F., Röglinger, M., Mendling, J. (eds.) BPM 2019. LNBIP, vol. 360, pp. 127–143. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26643-1_8
7. Leemans, S.J.J., Fahland, D., van der Aalst, W.M.P.: Discovering block-structured process models from incomplete event logs. In: Ciardo, G., Kindler, E. (eds.) PETRI NETS 2014. LNCS, vol. 8489, pp. 91–110. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-07734-5_6

8. Levenshtein, V.I.: Binary codes capable of correcting deletions, insertions, and reversals. In: Soviet Physics Doklady, vol. 10, pp. 707–710. Soviet Union (1966)
9. Martin, N., Depaire, B., Caris, A.: The use of process mining in business process simulation model construction. Bus. Inf. Syst. Eng. **58**(1), 73–87 (2016). https://doi.org/10.1007/s12599-015-0410-4
10. Pourbafrani, M., van der Aalst, W.M.P.: PMSD: data-driven simulation in process mining. In: Proceedings of the Demonstration Track at BPM 2020 co-located with 18th International Conference on Business Process Management, BPM, pp. 77–81 (2020). http://ceur-ws.org/Vol-2673/paperDR03.pdf
11. Pourbafrani, M., van der Aalst, W.M.P.: Extracting process features from event logs to learn coarse-grained simulation models. In: La Rosa, M., Sadiq, S., Teniente, E. (eds.) CAiSE 2021. LNCS, vol. 12751, pp. 125–140. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-79382-1_8
12. Pourbafrani, M., Balyan, S., Ahmed, M., Chugh, S., van der Aalst, W.M.P.: GenCPN: automatic generation of CPN models for processes (2021)
13. Pourbafrani, M., Jiao, S., van der Aalst, W.M.P.: SIMPT: process improvement using interactive simulation of time-aware process trees. In: Cherfi, S., Perini, A., Nurcan, S. (eds.) RCIS 2021. LNBIP, vol. 415, pp. 588–594. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-75018-3_40
14. 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)
15. 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: Dobbie, G., Frank, U., Kappel, G., Liddle, S.W., Mayr, H.C. (eds.) ER 2020. LNCS, vol. 12400, pp. 77–91. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-62522-1_6
16. 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: Abramowicz, W., Klein, G. (eds.) BIS 2020. LNBIP, vol. 389, pp. 249–263. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-53337-3_19
17. Pourbafrani, M., van Zelst, S.J., van der Aalst, W.M.P.: Supporting decisions in production line processes by combining process mining and system dynamics. In: Ahram, T., Karwowski, W., Vergnano, A., Leali, F., Taiar, R. (eds.) IHSI 2020. AISC, vol. 1131, pp. 461–467. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-39512-4_72
18. Pufahl, L., Weske, M.: Extensible BPMN process simulator. In: Proceedings of the BPM Demo Track and BPM Dissertation Award co-located with 15th International Conference on Business Process Modeling (BPM) (2017)
19. Rafiei, M., van der Aalst, W.M.P.: Towards quantifying privacy in process mining. In: Leemans, S., Leopold, H. (eds.) ICPM 2020. LNBIP, vol. 406, pp. 385–397. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-72693-5_29
20. Rafiei, M., Schnitzler, A., van der Aalst, W.M.P.: PC4PM: a tool for privacy/confidentiality preservation in process mining. In: Proceedings of the Demonstration Track at BPM co-located with 19th International Conference on Business Process Management (BPM), vol. 2973, pp. 106–110. CEUR-WS.org (2021)
21. Rogge-Solti, A., Weske, M.: Prediction of business process durations using non-Markovian stochastic petri nets. Inf. Syst. **54**, 1–14 (2015)
22. Rozinat, A., Mans, R.S., Song, M., van der Aalst, W.M.P.: Discovering simulation models. Inf. Syst. **34**(3), 305–327 (2009)

23. Sani, M.F., Gonzalez, J.J.G., van Zelst, S.J., van der Aalst, W.M.: Conformance checking approximation using simulation. In: 2020 2nd International Conference on Process Mining (ICPM), pp. 105–112 (2020)
24. Verbeek, E., van Hattem, M., Reijers, H., de Munk, W.: Protos 7.0: simulation made accessible. In: Ciardo, G., Darondeau, P. (eds.) ICATPN 2005. LNCS, vol. 3536, pp. 465–474. Springer, Heidelberg (2005). https://doi.org/10.1007/11494744_27