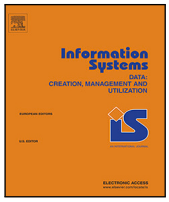
Information Systems 127 (2025) 102447

Contents lists available at ScienceDirect

Information Systems

journal homepage: www.elsevier.com/locate/is



A framework for measuring the quality of business process simulation models

David Chapela-Campa [a,*], Ismail Benchekroun [b], Opher Baron [c,d], Marlon Dumas [a], Dmitry Krass [c,d], Arik Senderovich [e]

[a] University of Tartu, Tartu, Estonia
[b] Department of Statistics, Faculty of Arts & Science, University of Toronto, Canada
[c] Rotman School of Management, University of Toronto, Canada
[d] SiMLQ Inc., Toronto, Canada
[e] School of Information Technology, York University, Canada

ARTICLE INFO

Keywords:
Business process simulation
Process mining
Quality measures

ABSTRACT

Business Process Simulation (BPS) is an approach to analyze the performance of business processes under different scenarios. For example, BPS allows us to estimate the impact of adding one or more resources on the cycle time of a process. The starting point of BPS is a process model annotated with simulation parameters (a BPS model). BPS models may be manually designed, based on information collected from stakeholders and from empirical observations, or automatically discovered from historical execution data. Regardless of its provenance, a key question when using a BPS model is how to assess its quality. In particular, in a setting where we are able to produce multiple alternative BPS models of the same process, this question becomes: How to determine which model is better, to what extent, and in what respect? In this context, this article studies the question of how to measure the quality of a BPS model with respect to its ability to accurately replicate the observed behavior of a process. Rather than pursuing a one-size-fits-all approach, the article recognizes that a process covers multiple perspectives. Accordingly, the article outlines a framework that can be instantiated in different ways to yield quality measures that tackle different process perspectives. The article defines a number of concrete quality measures and evaluates these measures with respect to their ability to discern the impact of controlled perturbations on a BPS model, and their ability to uncover the relative strengths and weaknesses of two approaches for automated discovery of BPS models. The evaluation shows that the proposed measures not only capture how close a BPS model is to the observed behavior, but they also help us to identify the sources of discrepancies.

1. Introduction

Business Process Simulation (BPS) is a technique for estimating the performance of business processes under different scenarios [1]. BPS is commonly used to estimate the impact of business process changes prior to their implementation [2]. For example, BPS enables analysts to address questions such as "what would be the cycle time of a process if one or more resources became unavailable?", "how many more resources are needed to maintain the current cycle time of the process if the number of new cases per day increases by 20%", or "what would be the impact of automating an activity on the waiting times of other activities in the process?".

The starting point of BPS is a process model, e.g. in the Business Process Model and Notation (BPMN),[1] enhanced with simulation parameters [3] (herein, a BPS model). These simulation parameters capture, for example, the processing times of each activity or the rate at which new process instances (cases) are created.

BPS models may be manually created based on information collected via interviews or empirical observations, or they may be automatically discovered from execution data recorded in process-aware information systems (event logs) [4–7]. Regardless of its provenance, a key question when using a BPS model is how to assess its quality. This question is particularly relevant for the automated discovery of BPS models, when tuning the simulation parameters of a discovered BPS model to obtain the version that better reflects the observed behavior of the process. Indeed, each time we perturb the parameters of a simulation model, we effectively derive a new simulation model from

* Corresponding author.
E-mail addresses: david.chapela@ut.ee (D. Chapela-Campa), ismail.benchekroun@mail.utoronto.ca (I. Benchekroun), opher.baron@rotman.utoronto.ca (O. Baron), marlon.dumas@ut.ee (M. Dumas), dmitry.krass@rotman.utoronto.ca (D. Krass), sariks@yorku.ca (A. Senderovich).

[1] https://www.bpmn.org/

https://doi.org/10.1016/j.is.2024.102447
Received 20 January 2024; Received in revised form 24 June 2024; Accepted 20 August 2024
Available online 22 August 2024

an existing one. The question is then: Is the perturbed simulation model better than the unperturbed one, to what extent, and in what respect?

Several approaches have been proposed to address this problem. However, these approaches are either manual and qualitative [7] or they produce a single number that does not allow one to identify the source(s) of deviations between the BPS model and the observed reality [4,8].

In this context, this article studies the problem of measuring the quality of a BPS model. More concretely, we tackle the problem of determining whether a BPS model's output has the accuracy required for its intended purpose, a problem also known as "operational validation" [9]. The article approaches this problem using a *historical data validation* approach [9], which seeks to measure the ability of a simulation model to replicate the actual behavior of a system (in our context, a process). To this end, we assume that we are given as input a historical dataset of executions of the process (an *event log*), and we measure to what extent the events produced by a run of the simulation model (a *simulated log*) are similar to those in the given event log.

Rather than seeking to produce a single quality measure, the article starts from the premise that the quality of a simulation model should be measured from different perspectives. For example, a BPS model may be better than another from a control-flow perspective (i.e. the sequences of activity labels it produces), but worse from a temporal perspective (i.e. the timing of the activity instances). Accordingly, the article proposes a framework that can be instantiated to define quality measures that capture different perspectives of a process. The overarching idea of the framework is that a BPS quality measure based on historical execution data needs to first filter and project this historical data to retain the data that is relevant to a given process perspective, then transform the data into an abstract representation capturing the behavior we seek to measure, and finally compare the abstract representation of the simulated log with the ground-truth event log, for example, via a distance function. Individual quality measures can be derived from the proposed framework by selecting concrete functions for filtering, projecting, and abstracting (transforming) the event logs, and for comparing the resulting abstractions.

We conduct a two-pronged evaluation of the measures using synthetic and real-life datasets. In the synthetic evaluation, we study the ability of the proposed measures to discern the impact of perturbations to a BPS model. This evaluation corresponds to the use-case where the quality measure is used to calibrate a simulation model by perturbing an initial one. In the second evaluation, based on real-life datasets, we analyze the ability of the proposed measures to uncover the relative strengths and weaknesses of two approaches for automated discovery of BPS models. This latter evaluation corresponds to the use case where the evaluation measures are used to compare BPS models generated by different approaches.

This article is an extended and revised version of a previous conference paper [10]. The conference version outlined a number of quality measures of BPS models.[2] While the proposed measures covered different perspectives of a BPS model, each quality measure was defined in isolation. The article extends the work presented in the conference paper with the following contributions:

- It proposes a unified framework for the definition of BPS model quality measures. Individual quality measures (including those proposed in the conference paper) are obtained by instantiating this framework with specific functions.
- It provides formal definitions of the proposed quality measures.
- It proposes an additional quality measure to cover the resource (workforce) perspective of business processes, and evaluates its applicability.

[2] The proposed measures are publicly available as a Python package (log-distance-measures) installable from `pip`. More details about the source code can be found in the "*Reproducibility*" section at the end of this article.

The rest of the article is structured as follows. Section 2 gives an overview of prior research related to the discovery and evaluation of BPS models. Section 3 introduces relevant process mining concepts and distance measures. Section 4 analyzes the problem and proposes a framework for measuring the quality of BPS models. Section 5 presents a set of measures obtained by instantiating the proposed framework. Section 6 discusses the empirical evaluation. Section 7 reflects on the BPS quality assessment problem and the role of the proposed framework and measures. Section 8 describes the limitations of our proposal and of the presented evaluation, and Section 9 draws conclusions and sketches future work.

2. Background

2.1. Business process simulation models

A BPS model consists of (i) a stochastic control-flow model, (ii) an activity performance model, and (iii) an arrival and congestion model. The stochastic model is composed of a process model (e.g., a BPMN model or a Petri net) and a stochastic component capturing the probability of occurrence of each path in the model (branching probabilities). In a BPS model, the stochastic model is enhanced by adding an activity performance model, which determines the duration of the activity instances (e.g., by associating a parametric distribution to each activity in the model). Finally, in a BPS model, an arrival and congestion model determines when new cases arrive in the system, and when the execution of each enabled activity instance starts, given the available resource capacity.

Traditionally, BPS models are constructed manually by experts. Recent approaches advocate for the automated discovery of BPS models from event logs. Below, we consider two such approaches. The first one, namely *SIMOD* [4], starts by constructing a stochastic process model by applying the SplitMiner algorithm [11] to discover a BPMN model from the input log, and replaying the traces of the log to calculate the branching probabilities. Next, SIMOD discovers the activity performance model (activity duration distributions) and a congestion model consisting of: (i) a case inter-arrival time distribution; (ii) a set of resources, their availability timetables, and the activities they perform; and (iii) the distribution of extraneous waiting times between activities (i.e. waiting times not attributable to congestion) [12]. Once a BPS model is discovered, its parameters are tuned to fit the data using a Bayesian hyper-parameter optimizer.

The second BPS model discovery technique we consider is *ServiceMiner*©. ServiceMiner operates in three steps: (i) data preprocessing, where techniques for data cleaning and categorical feature encoding are applied; (ii) data enhancement, where new data attributes that capture trend, seasonality, and system congestion are created using methods described in [13]; and (iii) model learning, where the BPS model is created by combining process discovery, queue mining (learning of queueing building blocks from data), and machine learning (to boost the accuracy of arrival and activity time generation). For process discovery, ServiceMiner mines a Markov chain, estimating the case routing probabilities between consecutive activity pairs. An abstraction mechanism allows for filtering out rare activities, paths, and transitions. Next, using queue mining, the various queueing building blocks are fitted from data, by using techniques described in [14]. Lastly, ServiceMiner applies a machine learning technique that uses congestion features that come from queueing theory, which, via cross-validation, leads to accuracy improvements when generating inter-arrival times and activity durations.

While the evaluation reported below focuses on BPS models discovered by SIMOD and ServiceMiner, the proposed measures can be used to assess the quality of any model that generates event logs. For example, the proposal can also be used to evaluate generative deep learning models of business processes [5]. On the other hand, it cannot be used to assess coarse-grained BPS models, e.g., based on system dynamics [15], unless these are refined to generate event logs.

2.2. Quality assessment of generative models

BPS models are, in essence, generative models specialized in producing data of a specific business process. For this reason, we first analyze the field of Machine Learning (ML), in particular previous studies on how to compare or assess the quality of generative models. However, there is no wide consensus on how to tackle such a problem [16], and ML literature is highly populated with domain-dependent solutions. For example, the evaluation of generative models in computer vision typically focuses on both the quality and the quantity of the generated data [16], e.g., through human annotation and statistical properties comparisons, respectively. These approaches are domain-dependent and not relevant in the world of business processes.

As the event logs generated by BPS models can be interpreted as time-series data, we focus on ML studies [17] that propose several comparison techniques for time series produced by generative models. Some of these approaches advocate for the comparison of two generative models by comparing their generated time-series samples. There are several methods to address qualitative comparisons, e.g., through visualization techniques or human psycho-perceptual evaluations based on expert judgments. In this article, we focus on automatable evaluation measures and, thus, take inspiration from quantitative comparison methods. Some proposals that stand out among them are (i) two-sample tests that statistically compare the distributions of real and generated data by measuring the distance between the two distributions, e.g., using the Maximum Mean Discrepancy (MMD); (ii) predictive scores methods that use generated data to train a predictive model and then evaluate its performance on a real validation set with metrics such as Mean Squared Error (MSE); and (iii) TimeGAN-Specific Evaluation [17], a combination of methods tailored specifically for time-series data generated by a state-of-the-art model dubbed *TimeGAN*. Although these methods are specific to the field of ML, they serve as inspiration for a comparative framework of BPS models, that may be addressed by comparing their generated data.

2.3. Quality measures for business process simulation models

Regarding the field of BPM, prior studies have considered the evaluation of BPS models. Rozinat et al. [7] perform an evaluation of BPS models following manual comparisons. However, they do not propose concrete and automatable evaluation measures. On the other hand, Camargo et al. propose in [4] an automated measure to assess the quality of a BPS model that combines the control-flow and the temporal perspectives. Nevertheless, this latter measure is not scalable, and it does not identify the sources of discrepancies between BPS models. To overcome these shortcomings, we propose an approach that views the process from different perspectives and provides a separation of concerns between BPS model components.

To the best of our knowledge, there is no prior work regarding perspectives such as the temporal or workforce. However, some studies have presented and analyzed measures to assess the quality of BPS models w.r.t. control-flow perspective. Burke et al. studied in [18] the evaluation of stochastic process models, empirically analyzing measures originally designed to assess different characteristics of the model, e.g., simplicity [19], generalization [20], precision [21], etc. In their study, they identify three stochastic quality dimensions: (i) *adhesion*, representing the effort needed to transform one stochastic language (e.g., the model's) into another (e.g., the event log's); (ii) *entropy*, identifying the amount of information in the system; and (iii) *simplicity*, measuring the structural complexity of the model. In this article, we are interested in assessing the ability of the BPS model to replicate the behavior of the process, and thus, we focus on measures related to the adhesion dimension [22].

Regarding this property, Leemans et al. [22] and Camargo et al. [5] independently proposed two measures that work on similar principles, i.e., comparing the language of the stochastic process model with the language recorded in the event log. To generate the language of the stochastic process model, Leemans et al. propose in [22] to unfold the stochastic model, obtaining all supported cases – i.e., activity sequences – with their associated stochastic probability.[3] Then, considering the activity sequences recorded in the event log and their frequency of occurrence as the language of the process, they use linear programming [23] to compute the minimum effort needed to transform the language of the stochastic model into the language of the process. In this setting, the transformation corresponds to the pairing of probability mass from each case in one language, to the probability of one or more cases in the other language, considering the cost of each pairing as the Damerau–Levenshtein distance between the two cases. Although this proposal precisely generates the language of the stochastic model, its unfolding requires an exponential complexity. Furthermore, solving the language transformation as a linear programming problem also presents a worst-case exponential complexity.

Reducing the exponential complexity, Camargo et al. [5] propose to perform a simulation of the stochastic model, limiting the represented language to the number of cases present in the event log. In this way, potentially sacrificing language coverage in complex stochastic models, the worst-case complexity of the language generation is reduced to linear complexity. Consequently, as both languages have the same number of elements, the transformation becomes an assignment problem – i.e., finding the bijection between the two sets of cases – and can be solved by the Hungarian algorithm with a worst-case cubic complexity.

Although more associated with the entropy dimension, Leemans et al. propose in [21] a different measure that evaluates the language of the stochastic model without limiting its representativeness in the case of infinite languages. Their proposal consists of generating the stochastic deterministic finite automatons (SDFA) of the stochastic model and event log, and projecting one SDFA into the other to compute the supported common behavior. However, it presents a quadratic worst-case complexity in the size of the SDFAs state space, which, for the stochastic model, grows exponentially in the number of nodes.

3. Preliminaries

3.1. Event logs

Modern enterprise systems store records of business process executions, which can be used to extract *event logs*: sets of timestamped events capturing the execution of the activities in a process [1]. In this article, we assume that each entry recorded in an event log relates to a case, an activity, a resource, a start timestamp, and an end timestamp (as in Fig. 1(b)). However, the methodology of this article can be generalized to include other life-cycle information (e.g., activity enablement).

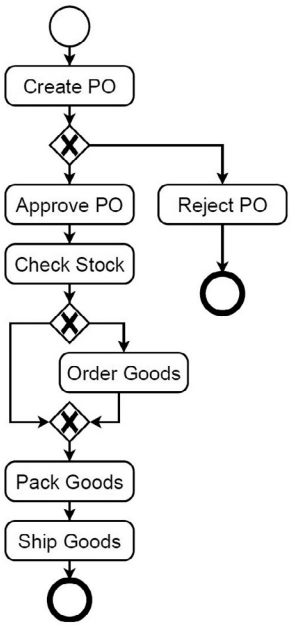
Let $\mathcal{E}$ be the universe of events, $\mathcal{C}$ be the universe of case identifiers, $\mathcal{A}$ be the set of possible activity labels, $\mathcal{R}$ the universe of resource identifiers, and $\mathcal{T}$ be the time domain.

Definition 1 (*Event Log*)**.** An *event log* L is a pair $L = (E, \sigma)$, with $E \subseteq \mathcal{E}$ being a set of executed events, and $\sigma = \{\xi, \alpha, \rho, \tau_s, \tau_e\}$ being a schema (a set of attributes) assigning the following attribute values to events:

- $\xi : \mathcal{E} \to \mathcal{C}$ assigns a case identifier,
- $\alpha : \mathcal{E} \to \mathcal{A}$ assigns an activity label,
- $\rho : \mathcal{E} \to \mathcal{R}$ assigns a resource type,
- $\tau_s : \mathcal{E} \to \mathcal{T}$ assigns the start timestamp of the executed activity, and,
- $\tau_e : \mathcal{E} \to \mathcal{T}$ assigns the end timestamp of the executed activity.

The event log L resides in a universe of logs denoted by $\mathcal{L}$.

[3] In practice, Leemans et al. propose to use a user-defined value to limit the probability covered by the language of the stochastic model in case of infinite languages, i.e., loops.



(a) Process model (BPMN).

Trace	Activity	Start Time	End Time	Resource
		⋮		
512	Create PO	03/11/2021 08:00:00	03/11/2021 08:31:11	DIO
513	Create PO	03/11/2021 08:34:21	03/11/2021 09:02:09	DIO
514	Create PO	03/11/2021 09:11:11	03/11/2021 09:49:51	DIO
		⋮		
512	Approve PO	03/11/2021 12:13:06	03/11/2021 12:44:21	Joseph
513	Reject PO	03/11/2021 12:30:51	03/11/2021 13:15:50	Jolyne
514	Approve PO	03/11/2021 12:59:11	03/11/2021 13:32:36	Joseph
512	Check Stock	03/11/2021 14:22:10	03/11/2021 14:49:22	DIO
514	Check Stock	03/11/2021 15:11:01	03/11/2021 15:46:12	DIO
		⋮		
514	Order Goods	04/11/2021 09:46:12	04/11/2021 10:34:23	Joseph
512	Pack Goods	04/11/2021 10:46:50	04/11/2021 11:18:02	Joseph
		⋮		
512	Ship Goods	04/11/2021 18:17:01	04/11/2021 19:27:45	Giorno
515	Ship Goods	04/11/2021 18:17:01	04/11/2021 19:57:43	Giorno
		⋮		

(b) Log fraction with 12 activity instances storing the trace identifier, the executed activity, the resource, and the start and end times of four different executions of the process depicted in Figure 1a.

Fig. 1. Running example: process model and fraction of an activity instance log of a Procure-to-pay process.

Note that the notion of "event log" typically represents the time information of each event as independent entries, with one entry recording its start, and another entry recording its end. Furthermore, there are cases in which event logs also store entries recording other execution stages of an activity, e.g., schedule. In this article, we group the information related to all stages of each activity execution in a single element, i.e., an event. The transformation from a typical event log to our definition of event log (usually referred to as "activity instance log") is straightforward, and it has already been described in other research [24]. Along the article, we will use both terms: *event log* and *activity instance log*, to refer to the recorded process information as in Definition 1; and *event* and *activity instance*, to refer to the recording of an activity execution as in Definition 1.

In some scenarios, event logs may also store information regarding varied process data. For example, objects associated with an activity instance or trace (i.e., object-centricity) or attributes storing information about the process (e.g., the amount of a loan offer or the total price of a purchase order). Although measuring the quality of a BPS model considering data attributes is out of the scope of this publication, Definition 1 can be extended in order to support such information. In addition, the proposed framework establishes a base for future research on quality measures considering this extended data.

3.2. Measures for time-series and histogram comparison

To analyze the temporal performance of a process, an event log can be mapped to a variety of time series (e.g., activity starts, activity ends). Accordingly, we consider the use of techniques to quantify the distance between two time series, $x = (x_1, x_2, \dots, x_n)$, and $y = (y_1, y_2, \dots, y_m)$, of (potentially different) lengths n and m, respectively.[4] To this end, one may employ various measures, such as computing $\|x-y\|_l$ in any of the standard norms (i.e., $l = 1, 2, \infty$).[5] However, such comparisons would only be possible after padding the shorter time series. In addition, standard norms do not capture the temporal differences between the two time series. For example, a temporal shift in x w.r.t. y may only produce a slight change in $l1$ or $l2$ norm, but represents a significant failure in the model to capture time-series patterns properly. To overcome the two limitations, namely (i) the need for padding and (ii) ignoring temporal differences, a natural measure is the Wasserstein Distance (WD) [26]. In this work, we consider two variations of WD.

- *Earth Mover's Distance (EMD)* [22] computes the effort needed to balance two vectors x and y of different lengths, treating each entry x_i, y_j as 'masses' to move from location to location until the two time series are equal. EMD does not assume that $\sum_i x_i = \sum_j y_j$, i.e., the sum of the 'earth mass' to be moved can be different. In such cases, we add a penalty for creating redundant mass to fill in gaps. Herein, we consider the EMD problem with absolute distance measure [27].
- *1st Wasserstein Distance (1WD)* [27] is a computationally efficient variation of the EMD. It introduces the constraint that the sum of masses must be the same in x and y. In other words, the 1WD problem is similar to the optimization problem solved by the EMD, with the addition that the constraint $\sum_i x_i = \sum_j y_j$ holds. 1WD is suitable for comparing empirical distribution functions (histograms), since the sum of the mass in each is 1.

When comparing two histograms, we let $f = (f_1, f_2, \dots, f_n)$ be the n normalized frequency values of the first histogram, and let $g = (g_1, g_2, \dots, g_m)$ be the m normalized frequencies of the second histogram.[6] We treat the two histograms f and g similarly to the two time series x and y, and employ 1WD distance, since the sum of masses is 1 (EMD and 1WD lead to the same results).

[4] Throughout the article, we will represent time series with temporal frequency arrays – e.g., $x = (x_1, x_2, \dots, x_n)$ –, a discrete representation of events over time where, for $i = 1, \dots, n$, i indicates a specific instant in time and x_i denotes the number of events recorded at that instant.

[5] See Sec. 2.2 in [25] for a survey on time-series comparison measures.

[6] Throughout the article, we will represent histograms with frequency distribution arrays, e.g., $x = (x_1, x_2, \dots, x_n)$, where each element $x_i \in x$, $i = 1, \dots, n$ corresponds to the frequency count of the ith bin in the histogram.

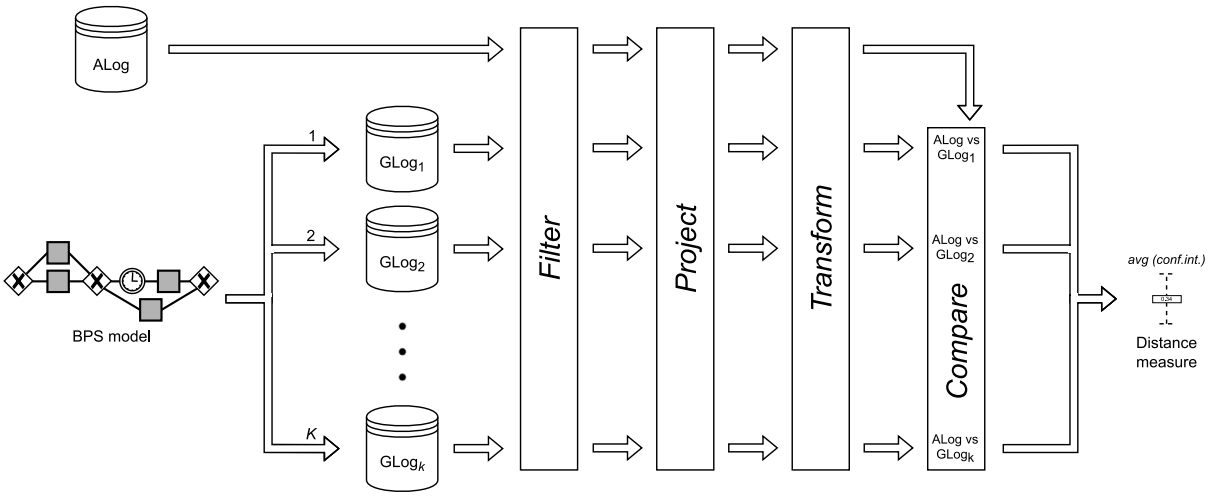


Fig. 2. Structure of the proposed framework.

4. Framework for measuring BPS model quality

The starting point of the proposed framework is that we can assess the quality of the simulation model by computing a distance (or a similarity) function between the event logs that the simulation model generates, and an event log of the actual process being simulated. In other words, we do not assess the quality of the BPS model by analyzing the structure of the model itself, but rather by comparing the event logs that the simulation generates against a ground-truth.

There are two reasons why evaluating a BPS model based solely on the model's own structure and information would be impractical: (i) The 'true' BPS model of the process is typically unavailable (and often does not exist in the first place); thus, we cannot perform a model-to-model comparison. (ii) Different simulation engines (e.g., BIMP [28], Prosimos [29]) support different BPS model formats, hindering a generic evaluation of BPS models. Therefore, in this work, we propose to generate a collection of logs simulated using the BPS model under evaluation, and compare them to event log available from the actual system (i.e., the system that the model aims to mimic). Consequently, one can apply a 'transitive argument': the 'closer' the simulated logs are to the actual data, the better the model is. In other words, we treat the (test) data as our 'ground truth', since useful models are supposed to be faithful generators of 'reality'. Clearly, we assume that the event log is trustworthy, and has a high level of alignment to the actual process (true process behavior is 'well-represented' in the log).

Two additional challenges arise when measuring the quality of a BPS model: (i) a model can be very close to the data in one aspect (e.g., control-flow), yet very different in another (e.g., the arrival of new cases), and, (ii) a model can generate many realities as it is probabilistic in nature (e.g., durations and decisions are stochastic), while the data consists of a single realization of the reality. To overcome the first limitation, we propose a framework that projects the process onto several dimensions in order to quantify the distance across multiple process perspectives. As for the second limitation, our proposal is to first generate multiple event logs (generated logs, or GLogs) simulating the same scenario as the 'ground-truth' event log (e.g., starting from the same instant in time and simulating the same number of cases). Then, we propose to use the GLogs to construct confidence intervals for each of our measures by comparing them with the 'ground-truth' event log (actual event log, or ALog).

Fig. 2 depicts the main structure of the proposed framework. The starting point comprises the event log of the original process (ALog) and the BPS model under evaluation. For each measure, we propose to consider a collection of GLogs resulting from K simulation runs. We compare each GLog to the ALog and construct confidence intervals based on the individually measured distances.

The framework decomposes each comparison between a GLog and the Alog into four steps. First, the *filter* (Section 4.1) and *project* (Section 4.2) operations preprocess the event log, retaining only the information needed to evaluate the desired distance measure. Afterward, this preprocessed data is *transformed* into a format describing the behavior that the desired measure is aiming to capture (Section 4.3). Finally, a *compare* operation (Section 4.4) structures the transformed data into a suitable format (e.g., a histogram) and measures the distance between the two samples. The modularity of this framework allows us to define new measures by altering the operations performed in each step (e.g., filter and project). For example, given a measure that focuses on the temporal distribution of events in calendar time (absolute time), we can define a measure that captures the distribution of events in relative time (relative to the start of each case) by changing the transformation operation, so that instead of comparing absolute times, we compare relative times.

Below, we describe and formalize each of these four operations.

4.1. Filter

Not all the *events* recorded in the event log are necessary when comparing two event logs w.r.t. a specific perspective. Specifically, some of the observations can be essential when analyzing one dimension of the process, yet unnecessary for others. For example, only the first and last events of each case are needed when evaluating the distribution of cycle times in the process. Similarly, the first recorded event of each case is sufficient to analyze the distribution of case arrival times in the process.

The objective of the log filtering, or simply *filter*, operation is to remove observations (i.e., events) that are unnecessary for the distance measure that is being evaluated. To this end, we define a predicate function $\phi : \mathcal{E} \times \mathcal{L} \rightarrow \{T, F\}$ that takes an event and an event log and returns T (True) if the event satisfies the predicate or F (False) otherwise.[7]

Definition 2 (*Log Filtering (Filter)*)**.** Given an event log $L = (E, \sigma)$ and a predicate function ϕ, a filtered log L' is defined as,

$$L' = \{e \in E \mid \phi(e, L) = T\}.$$

[7] This operation is akin to the `WHERE` clause in the SQL language.

Table 1
Filtered event log F' result of applying the filtering operation in Definition 2 with the predicate given by Eq. (1) to the event log in Fig. 1(b).

ξ	α	τ_s	τ_e	ρ
		⋮		
1	Create PO	3/11/21 08:00:00	3/11/21 08:31:11	DIO
2	Create PO	3/11/21 08:34:21	3/11/21 09:02:09	DIO
3	Create PO	3/11/21 09:11:11	3/11/21 09:49:51	DIO
		⋮		

For example, Table 1 depicts an example of a filtering operation performed over the event log in Fig. 1(b). In this example, the filtering operation retains the first activity of each process case (the one with the lowest τ_s). The predicate for can be written as

$$\phi(e, L) = \tau_s(e) == \min\{\tau_s(e') \mid e' \in L \wedge \xi(e) = \xi(e')\}. \quad (1)$$

This predicate returns T only for the first (earliest) event in each corresponding case, assuming the granularity of the recorded timestamps is capturing the precise instant at which they are occurring. Otherwise, e.g., timestamps recorded at a coarse level of granularity, the filter operation should retain only one event per case. Thus, the filtered log, L', will only contain the earliest event(s).

4.2. Project

The objective of the projection step is to retain only the attributes necessary for the analyzed perspective. To this end, we propose a projection step, or simply *project*, that consists of attribute removal, i.e., only the attributes needed for the dimension that is being analyzed are kept.[8]

Definition 3 (*Projection*)**.** Given an event log $L = (E, \sigma)$, *projection* is a function $P_\lambda : \mathcal{L} \to \mathcal{L}$, with the resulting log $L^\lambda = (E, \lambda)$ where $\lambda \subseteq \sigma = \{\xi, \alpha, \rho, \tau_s, \tau_e\}$ is a subset of the attributes retained from σ.

Fig. 3(a) depicts an example of the projection operation $P_{\{\tau_s\}}(L')$ of the filtered event log from Table 1 (L'). In this example, all attributes except the start time (τ_s) are discarded.

4.3. Transform

The two first preprocessing steps, namely filtering and projection, remove information that is irrelevant to measure the specific dimension under analysis. However, in order to analyze each perspective, the remaining data must be transformed into a structure that can be adequately used for the comparison that we target. For example, when analyzing the distribution of cycle times in the process, essential information involves case start and end events. Yet, the comparison is not performed on the raw timestamps, but rather, the cycle time of each case must be computed instead.

The objective of the third step is to apply a *transformation* to the filtered and projected data in order to obtain the information strictly related to the dimension being analyzed. To that end, we propose a *transform* operation to remodel the information received from the previous steps into a unified format for future comparison.

Definition 4 (*Transformation*)**.** Given a projected event log L^λ, a transformation operation $T : \mathcal{L} \to \mathbb{B}(\mathbb{R}^n)$ maps the log into a multiset[9] of n-dimensional real-valued vectors, namely $T(L^\lambda) = \{\{v_1, v_2, \ldots, v_l\}\}$ where $v_i = (v_{i,1}, v_{i,2}, \ldots, v_{i,n})$ with $v_{i,j} \in \mathbb{R}$.

[8] This operation is akin to the SELECT clause in the SQL language.

[9] Throughout the article, we will use {{ }} to represent a multiset, ( ) to represent a vector, ⟨ ⟩ to represent a sequence, and ⌢ to represent the concatenation of vectors or sequences such that $(1,2,3)^\frown(4,5) = (1,2,3,4,5)$ and $\langle 1,2,3\rangle^\frown\langle 4,5\rangle = \langle 1,2,3,4,5\rangle$.

τ_s
⋮
3/11/21 08:00:00
3/11/21 08:34:21
3/11/21 09:11:11
⋮

(a) Projected event log example.

$v_{i,1}$	$v_{i,2}$	$v_{i,3}$	$v_{i,4}$
⋮			
2021	11	03	08
2021	11	03	09
2021	11	03	09
⋮			

(b) Multiset of 4-dimensional vectors.

Fig. 3. Example of a projected event log L^λ, result of applying the projection operation in Definition 3 with $\lambda = \{\tau_s\}$ to the filtered event log in Table 1 (left); and a multiset of 4-dimensional vectors, result of applying the transformation operation in Definition 4 that separates the year, month, day, and hour of a timestamp (right).

Fig. 3(b) shows an example of a transformation operation. In this example, the transformation converts each start timestamp in the set of observations received as input into their corresponding date and hour. Thus, the result is a multiset $\{\{v_1, v_2, \ldots, v_l\}\}$ of 4-dimensional vectors v_i where each of them stores, respectively, the year, month, day, and hour of each case arrival.

4.4. Compare

At this stage, the Alog and each GLog have been individually preprocessed and transformed into multisets of n-dimensional vectors $\{\{v_1, v_2, \ldots, v_l\}\}$. The content and number of dimensions of these vectors depend on the transformation operation. For example, when measuring the distribution of arrivals, each vector must contain the information of the year, month, day, and hour of a case arrival. In order to assess the quality of the BPS model, we propose to compute a 'distance' between pairs of GLog and ALog. Therefore, the objective of the comparison step is to evaluate that distance.

Since not all functions that can be employed to measure the distance between two event logs work with the same data format, we must accommodate these differences by defining a prepare function. For example, two time series are compared when measuring the EMD distance, but a list of paired observations is needed to compute the Mean Squared Error. Hence, we propose a 2-step comparison operation: it first prepares the list of vectors into a format suitable for comparison (e.g., a time series) and only then measures their distance using a distance measure (e.g., EMD).[10]

Let $\pi : \mathbb{B}(\mathbb{R}^n) \times \mathbb{B}(\mathbb{R}^n) \to \mathcal{W} \times \mathcal{W}$ be a prepare function that maps the two multisets (GLog and ALog) to data structures of choice (e.g., time series, or a histogram). The co-domain of π, $\mathcal{W}$, is the universe of elements that the subsequent distance function takes as an input. Next, let $\delta : \mathcal{W} \times \mathcal{W} \to \mathbb{R}$ be a real-valued distance function (e.g., EMD or 1-WD).

Definition 5 (*Comparison*)**.** A comparison operation applies a pair of functions (π, δ) (prepare and compare) to a pair of multisets of n-dimensional vectors, V_1 and V_2, and returns a real value C, $C = \delta(\pi(V_1, V_2))$.

Figs. 4 and 5 depict an example of this operation, where the prepare function (Fig. 4) aggregates the vectors on each multiset (V_1 and V_2) to obtain the number of arrivals in each hour of the process; and the compare function (Fig. 5) measures the distance between these two time series.

[10] Note that, by defining "Comparison" in this generic way, the proposed framework is open to new quality measures employing other distance function (e.g., MSE).

hourID	# *obs*
⋮	
2021-11-03 06h	0
2021-11-03 07h	0
2021-11-03 08h	1
2021-11-03 09h	2
2021-11-03 10h	2
⋮	

hourID	# *obs*
⋮	
2021-11-03 06h	0
2021-11-03 07h	0
2021-11-03 08h	6
2021-11-03 09h	4
2021-11-03 10h	4
⋮	

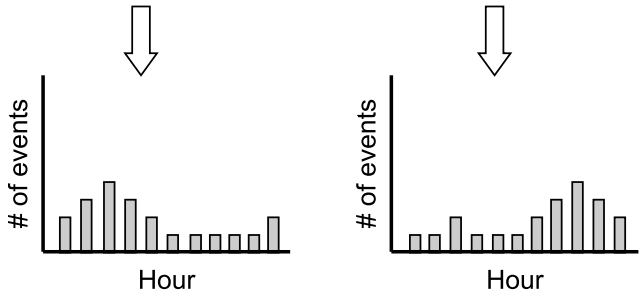


Fig. 4. Example of two time series, result of the prepare function of the comparison operation in Definition 5, for the multiset of 4-dimensional vectors of Fig. 3(b) (left) and a different multiset (right), being π an aggregation function that counts the number of vectors per hour.

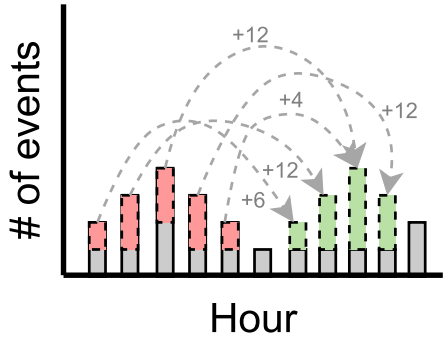


Fig. 5. Example of distance measurement, result of the compare function of the comparison operation in Definition 5, for the two time series of Fig. 4, being δ the Earth Mover's Distance.

Finally, once the distance between each pair GLog-ALog is measured, our proposal is to report the average distance (and confidence interval) of these computed distances.

5. BPS model quality measures

Processes are multi-dimensional. A process comprises the execution of a set of activities following a particular structure. These activities are executed by resources, at specific times of the day, and take different amounts of time. Furthermore, the execution of some activities may generate and consume data along the process. Following with the reasoning presented in Section 4, a BPS model can excel in one of these aspects, yet underperform in another.

To accommodate this multidimensional view of a process, we propose to analyze the quality of a BPS Model w.r.t. four dimensions: control-flow, temporal, congestion, and resource. Accordingly, we propose a set of measures for each perspective. Below, we discuss the rationale for each measure and describe how they are implemented within our framework (see Section 4).[11]

5.1. Control-flow

The control-flow perspective describes the names and ordering of the activities that are executed in the process. As one of the main perspectives of a process, it has played a central role since the beginning of process mining research. As a result, several approaches to evaluate the quality of process models have been proposed [30–32]. Nevertheless, these approaches focus solely on the quality of the discovered structure, discarding the stochastic component. For BPS, it is important that the stochastic process model captures both the behavior of the process – i.e., what paths can be executed – and its stochasticity – i.e., how often each path is executed.

To evaluate the quality of a BPS model w.r.t. the control-flow perspective, i.e., its capability to represent the event sequences in the actual event log, we present one of the state-of-the-art measures (see Section 2.3), and propose a more efficient alternative. As a representative of the-state-of-the-art, we adapted the proposal of Camargo et al. in [5] by inverting its value – transforming it from a similarity into a distance measure – and formalizing it following our proposed framework. This measure, namely *control-flow log distance* (CFLD), penalizes the differences in the control-flow by first computing the Damerau–Levenshtein distance between each case in the GLog and each case in the ALog, and then finding the bijection between the GLog and the ALog – i.e., pairing each case in the GLog to a different case in the ALog – that minimizes the sum of their distances. Additionally, due to the steep computational complexity of CFLD (as well as of the other state-of-the-art measures), we propose a measure that approaches the problem in a more efficient way, the n-gram distance (NGD).

Control-flow Log Distance (CFLD, adapted from [5]). Given two logs L_1 and L_2, the CFLD computes the average distance to transform each case in L_1 to another case in L_2 (see [5] for a description of a similarity version of this measure).

Filter. Given an event log L, do not filter out any event in the log. Thus, the predicate for the filtering operation is defined as follows:

$$\phi(e, L) = T, \tag{2}$$

i.e., always return true, indicating that all events are kept in the filtered event log.

Project. Given a filtered event log result of the previous operation, project only the case id, activity label, and end timestamp of each event, i.e., $\lambda = \{\xi, \alpha, \tau_e\}$.

Transform. Given a projected event log $L^{\lambda} = \{E, \sigma\}$ where each event follows the schema $\sigma = \{\xi, \alpha, \tau_e\}$, and a size $v \in \mathbb{N}^+$ indicating the maximum case length. Let $\Omega \subseteq C$ be the set of case identifiers of the process, let $\zeta : \mathcal{A} \to \mathbb{N}^+$ be a function that maps each activity label to a unique positive natural number, and let $|\xi'|$ denote the number of events $e \in E$ such that $\xi(e) = \xi'$, we define the following transform operation:

$$\begin{aligned}\mathcal{T}(L^{\lambda}) = \{\{\ (\ \zeta(\alpha(e_1)),\ \zeta(\alpha(e_2)),\ \dots,\\ \zeta(\alpha(e_{|\xi'|}))\ )^{\frown}\vec{0}_{v-|\xi'|}\ \mid\ \tau_e(e_i) \le \tau_e(e_{i+1}) \wedge\\ i \in \{1, \dots, |\xi'|\} \wedge \xi(e_i) = \xi' \wedge\ \xi' \in \Omega\ \}\}\end{aligned} \tag{3}$$

This operation creates a multiset of v-dimensional vectors where each $v \in V$ stores the sequence of activity identifiers corresponding to a process case $\xi' \in \Omega$, sorted by end timestamp, and padded with $\vec{0}_{v-|\xi'|}$, i.e., a zero vector with the number of dimensions needed to complete v to length v.

[11] For clarity, each proposed measure is described as a distance between two event logs. However, as explained in Section 4, the proposed framework reports the average and confidence interval of K individual comparisons (each GLog against the ALog).

Compare. Given two multisets V_1 and V_2 of v-dimensional vectors, result of the transformation in the previous operation. The compare operation first equalizes, if needed, the number of vectors (corresponding to the number of cases) in each multiset with the following prepare function:

$$\begin{aligned}\pi(V_1, V_2) = (V_1', V_2') \mid \\ V_1' = V_1 \cup \{\{\ \overrightarrow{0_v}\ \}\}^{\max(|V_2|-|V_1|,0)}, \\ V_2' = V_2 \cup \{\{\ \overrightarrow{0_v}\ \}\}^{\max(|V_1|-|V_2|,0)}.\end{aligned} \tag{4}$$

In essence, we add v-dimensional zero vectors ($\overrightarrow{0_v}$) to the event log that has fewer cases until their size is the same. Subsequently, let $dl : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ be the Damerau–Levenshtein (DL) distance [33] between two n-dimensional vectors of real numbers, $s : \mathbb{R}^n \to \mathbb{N}$ be the function that computes the number of elements in an n-dimensional vector that differ from 0 (the number of activities of that case), and $p : V_1' \to V_2'$ be a bijective function pairing every $v_i \in V_1'$ to a unique $v_j \in V_2'$. The compare operation measures the distance between V_1' and V_2' as,

$$\delta(V_1', V_2') = \min_p \frac{1}{|V_1'|} \sum_{v_i \in V_1'} \frac{dl(v_i, p(v_i))}{\max(s(v_i), s(p(v_i)))}, \tag{5}$$

which corresponds to the minimum average of the normalized DL distances between each case in V_1' and their corresponding pair in V_2'.

As mentioned in Section 2.3, CFLD requires pairing each case in the simulated log with a case in the original log, minimizing the total sum of distances. The computational complexity of computing the DL distance for all possible pairings is $O(N^2 \times MTL^3)$ where N is the number of cases in the logs, and MTL is the maximum case length. Finally, the optimal alignment of cases using the Hungarian algorithm has a cubic complexity in the number of cases. Furthermore, since all pairings are put into a matrix to compute the optimal alignment of cases (the one that minimizes the total sum of distances), CFLD's memory complexity is quadratic on the number of cases.

N-Gram Distance (NGD). Taking inspiration from research on efficient event log storage in graph databases, particularly through projecting the event log into the histogram of its Directly-Follows Graph (DFG) [34], we note that the histogram of 2-grams of an event log is equal to the histogram of its DFG.[12] Given this observation, we propose a general approach based on n-grams, noting that the histogram of n-grams of a log is equal to the $(n-1)^{\text{th}}$-Markovian abstraction of the log [32]. In other words, the histogram of 2-grams is the 1^{st}-order Markovian abstraction (the DFG), the histogram of 3-grams is the 2^{nd}-order Markovian abstraction, and so on.

Given two logs L_1 and L_2, and a positive integer $n \in \mathbb{N}^+$, the NGD computes the difference in the frequencies of the n-grams observed in $L_1 \cup L_2$.

Filter. Given an event log L, do not filter out any event in the log. Thus, the predicate for the filtering operation is defined by Eq. (2).

Project. Given a filtered event log result of the previous operation, project only the case id, activity label, and end timestamp of each event in the log, i.e., $\lambda = \{\xi, \alpha, \tau_e\}$.

Transform. Given a size $n \in \mathbb{N}^+$ and a projected event log $L^\lambda = \{E, \sigma\}$ where each event follows the schema $\sigma = \{\xi, \alpha, \tau_e\}$. Let $\Omega \subseteq C$ the set of case identifiers of the process, and Θ a sequence of n-1 events of a dummy activity θ, let

$$\Psi = \bigcup_{\xi' \in \Omega} \{\{\ \Theta^\frown \langle e | e \in L^\lambda \wedge \xi(e) = \xi' \rangle^\frown \Theta\ \}\}.$$

be the multiset of process cases, where each case stores a sequence of its events padded (at the start and end) with n-1 dummy events, and sorted by their end timestamp such that

$$\forall_{\psi \in \Psi}\ \tau_e(e_i) \leq \tau_e(e_{i+1}) \mid i \in \{n, \dots, (|\psi|-n)\}.$$

Then, let $\zeta : \mathcal{A} \cup \{\theta\} \to \mathbb{N}$ be a function that maps each activity label to a unique natural number, we define the following transform operation:

$$\begin{aligned}\mathcal{T}(L^\lambda) = \bigcup_{\psi \in \Psi} \{\{\ (\zeta(\alpha(e_{i-n-1})), \dots, \zeta(\alpha(e_i)))\ \mid \\ \psi = \langle e_1, e_2, \dots, e_{|\psi|} \rangle \wedge i \in \{n, \dots, |\psi|\}\}\}\end{aligned} \tag{6}$$

This operation creates a multiset of n-dimensional vectors where each $v \in V$ stores n activities observed as a sequence in L^λ.

Compare. Given two multisets V_1 and V_2 of n-dimensional vectors, result of the transformation in the previous operation, and a set $V_o = \{w_i \mid w_i \in V_1 \cup V_2\}$ storing the unique vectors from both V_1 and V_2. The compare operation first builds two histograms x, y with the following prepare function:

$$\begin{aligned}\pi(V_1, V_2) = &((x_1, x_2, \dots, x_{|V_o|}), (y_1, y_2, \dots, y_{|V_o|}))\ \mid \\ &i = 1, \dots, |V_o|, \\ &x_i = |\{\{v \in V_1 \mid v = w_i\}\}|\ /\ (|V_1| + |V_2|), \\ &y_i = |\{\{v \in V_2 \mid v = w_i\}\}|\ /\ (|V_1| + |V_2|).\end{aligned} \tag{7}$$

For each observed n-gram, we count the number of times it appears in each V, relative to the total number of n-grams (in both V_1 and V_2). Then, we measure the distance between x and y using,

$$\delta(x, y) = \sum_{i=1}^{|V_o|} |x_i - y_i|, \tag{8}$$

which corresponds to the Sum of Absolute Errors (SAE) between two histograms $x = (x_1, \dots, x_B)$ and $y = (y_1, \dots, y_B)$.

For example, consider L_1 having three cases A−B−C−D, and L_2 having three cases A−B−E−D. Given $n = 2$, the observed n-grams are 0−A, A−B, B−C, C−D, and D−0 in L_1; and 0−A, A−B, B−E, E−D, and D−0 in L_2 (each one with a frequency of three). The n-grams B−C, C−D, B−E, and E−D have a frequency of 3 in one log, and 0 in the other, thus, the NGD between L_1 and L_2 is 0.4 (12 divided by 30). By adding dummy activities, all activity instances have the same weight in the measure, as each of them is present in n n-grams. Otherwise, the first and last activity instances of each case would be present only in one n-gram. Note that we do not use the EMD to compute the NGD, because the order of the n-grams in the histogram is irrelevant and EMD would take this order into account.

NGD is considerably more efficient than CFLD, as the construction of the histogram of n-grams is linear in the number of events in the log, and the same goes for computing the differences between the n-gram histograms.

5.2. Temporal

Following with the reasoning presented earlier, we propose to consider the time dimension as the second perspective. Independently of whether the activities arriving in the system are closer to the real process or not, a simulation technique could be interested in replicating the temporal distribution of events through the system in the best way possible. For this reason, we propose three temporal measures to assess the ability of a BPS model to capture the temporal performance

[12] An n-gram is a vector of n consecutive activities in a case of a log. A 2-gram is a pair of consecutive activities in a log. Every arc in the DFG of a log is a 2-gram of the log and vice-versa.

perspective, based on the idea that the time series of events generated by a BPS model should be similar to the time series of the test data, with respect to seasonality, trend, and time-to-event.

The first two measures come from time-series analysis, where most approaches in the literature (e.g., SARIMA) decompose the time series into components of trend, seasonality, and noise [35]. We follow a similar path by analyzing the trend (comparing the absolute distribution of events) and the seasonality (comparing the circadian distribution of events). The third measure comes from time-to-event (or survival) analysis [36], a field in statistics that analyzes the behavior of individuals from some point in time until an event of interest occurs. Specifically, we are interested in analyzing the capability of the simulator to correctly reconstruct the occurrence of events (and their timestamps) from the beginning of the corresponding case to its end. Below, we provide the details of the three aforementioned measures.

Absolute Event Distribution (AED). Given two event logs L_1 and L_2, the AED computes the difference in the temporal distribution of events throughout the process timeline. This metric specifically evaluates how the occurrence of events differs over time.

Filter. Given an event log L, do not filter out any event in the log. Thus, the predicate for the filtering operation is defined by Eq. (2).

Project. Given a filtered event log result of the previous operation, project only the start and end timestamps of each event in the log, i.e., $\lambda = \{\tau_s, \tau_e\}$.

Transform. Let $y(\tau)$, $m(\tau)$, $d(\tau)$, and $h(\tau)$ be functions returning, respectively, the year, month, day, and hour of a timestamp τ. Given a projected event log $L^\lambda = \{E, \sigma\}$ where each event follows the schema $\sigma = \{\tau_s, \tau_e\}$, we define the following transform operation:

$$\begin{aligned}\mathcal{T}(L^\lambda) = \{\{\ (y(\tau_i), m(\tau_i), d(\tau_i), h(\tau_i)) \mid \tau_i = \tau_s(e) \wedge \\ e \in E\ \}\} \cup \{\{\ (y(\tau_i), m(\tau_i), d(\tau_i), h(\tau_i)) \mid \\ \tau_i = \tau_e(e) \wedge e \in E\ \}\}.\end{aligned} \tag{9}$$

This operation creates a multiset of four-dimensional vectors where each $v \in V$ stores the year, month, day, and hour of a start or end timestamp from L^λ.

Compare. Given two multisets V_1 and V_2 of four-dimensional vectors, result of the transformation in the previous operation, and the sorted sequence $H_o = \langle w_1, w_2, \dots, w_B \rangle$ storing all the hours in the timeline of the log (i.e., from the first until the last timestamp in $L_1 \cup L_2$) such that each $w_i = (y_i, m_i, d_i, h_i) \in H_o$ stores the year, month, day and hour. The compare operation first builds two time series x, y with the following prepare function:

$$\begin{aligned}\pi(V_1, V_2) = &((x_1, x_2, \dots, x_B), (y_1, y_2, \dots, y_B)) \mid \\ &i = 1, \dots, B, \\ &x_i = |\{\{v \in V_1 \mid v = w_i\}\}|, \\ &y_i = |\{\{v \in V_2 \mid v = w_i\}\}|.\end{aligned} \tag{10}$$

i.e., counts the number of events occurring in each hour within the timeline of the log; and then, measures the distance between x and y with:

$$\delta(x, y) = \mathrm{EMD}(x, y) \tag{11}$$

i.e., the Earth Movers' Distance between the two time series $x = (x_1, \dots, x_B)$ and $y = (y_1, \dots, y_B)$.

Circadian Event Distribution (CED). Given two event logs L_1 and L_2, the CED computes the average difference in the temporal distribution of events throughout the process timeline, by day of the week. This metric specifically evaluates how the occurrence of events differs over time within each day of the week.

Filter. Given an event log L, do not filter out any event in the log. Thus, the predicate for the filtering operation is defined by Eq. (2).

Project. Given a filtered event log result of the previous operation, project only the start and end timestamps of each event in the log, i.e., $\lambda = \{\tau_s, \tau_e\}$.

Transform. Let $wd(\tau)$ and $h(\tau)$ be two functions returning, respectively, the day of the week and hour of a timestamp τ. Given a projected event log $L^\lambda = \{E, \sigma\}$ where each event follows the schema $\sigma = \{\tau_s, \tau_e\}$, we define the following transform operation:

$$\begin{aligned}\mathcal{T}(L^\lambda) = &\{\{\ (wd(\tau_i), h(\tau_i)) \mid \tau_i = \tau_s(e) \wedge e \in E\ \}\} \\ &\cup \\ &\{\{\ (wd(\tau_i), h(\tau_i)) \mid \tau_i = \tau_e(e) \wedge e \in E\ \}\}\end{aligned} \tag{12}$$

This operation creates a multiset of two-dimensional vectors where each $v \in V$ stores the day of the week and hour of a start or end timestamp from L^λ.

Compare. Given two multisets V_1 and V_2 of two-dimensional vectors, result of the transformation in the previous operation. The compare operation first builds two matrices of 7 rows, x and y, where the d^{th} row contains the time series for the d^{th} day of the week, with the following prepare function:

$$\begin{aligned}\pi(V_1, V_2) = &((x_{1,1}, x_{1,2}, \dots, x_{7,24}), (y_{1,1}, y_{1,2}, \dots, y_{7,24})) \mid \\ &d = 1, \dots, 7,\ i = 1, \dots, 24, \\ &x_{d,i} = |\{\{(d, i) \in V_1\}\}|, \\ &y_{d,i} = |\{\{(d, i) \in V_2\}\}|.\end{aligned} \tag{13}$$

The function counts the number of events occurring in each hour for each day of the week. To measure the distance between x and y we use the average of the Earth Movers' Distance between $x_{d,1}, \dots, x_{d,24}$ and $y_{d,1}, \dots, y_{d,24}$ with $d \in \{1, \dots, 7\}$, i.e.,

$$\delta(x, y) = \frac{1}{7} \sum_{d=1}^{7} \mathrm{EMD}(x_d, y_d). \tag{14}$$

Relative Event Distribution (RED). Given two event logs L_1 and L_2, RED computes the difference in the temporal distribution of events w.r.t. the origin of their case (i.e., the case arrival). This metric specifically evaluates how the occurrence of events over time differs within each case.

Filter. Given an event log L, do not filter out any event in the log. Thus, the predicate for the filtering operation is defined by Eq. (2).

Project. Given a filtered event log result of the previous operation, project only the case identifier, start, and end timestamps of each event in the log, i.e., $\lambda = \{\xi, \tau_s, \tau_e\}$.

Transform. Given a projected event log $L^\lambda = \{E, \sigma\}$ where each event follows the schema $\sigma = \{\xi, \tau_s, \tau_e\}$, and let

$$\mu(\xi') = \min_{\tau'} \{\tau' \mid \tau' = \tau_s(e) \wedge \xi(e) = \xi' \wedge e \in E\}, \tag{15}$$

be the arrival time of a case ξ',[13] and being $\lfloor\ \rfloor_h$ the *floor* operator that returns the number of complete hours from a time

[13] Throughout the article, we approximate the arrival of a case with the start time of its first recorded event. It must be noted that our proposed measures are modular w.r.t. the arrival computation. Thus, more complex approximations like the ones presented in [37,38] can be easily used.

duration. We define the following transform operation:

$$\mathcal{T}(L^{\lambda}) = \{\{(\ \lfloor \tau_s(e) - \mu(\xi(e)) \rfloor_h\ ) \mid e \in E\}\} \cup \{\{(\ \lfloor \tau_e(e) - \mu(\xi(e)) \rfloor_h\ ) \mid e \in E\}\}, \tag{16}$$

This operation creates a multiset of one-dimensional vectors, where each $v \in V$ stores, for each (start or end) timestamp in the projected log, the number of complete hours w.r.t. the arrival of its case.

Compare. Given two multisets V_1 and V_2 of one-dimensional vectors, result of the transformation in the previous operation. The compare operation first builds two time series x, y with the following prepare function:

$$\pi(V_1, V_2) = ((x_1, x_2, \dots, x_B), (y_1, y_2, \dots, y_B)) \mid i = 0, \dots, B,\ B = \max(V_1 \cup V_2),\ x_i = |\{\{v \in V_1 \mid v = i\}\}|,\ y_i = |\{\{v \in V_2 \mid v = i\}\}|. \tag{17}$$

In essence, the compare function counts the number of events occurring i hours after their case arrival; and then, measures the distance between x and y using the Earth Movers' Distance between $x_1, \dots, x_B$ and $y_1, \dots, y_B$, i.e.,

$$\delta(x, y) = \text{EMD}(x, y). \tag{18}$$

5.3. Workforce

For the third dimension, we focus on analyzing the workforce of the process. As mentioned earlier, the available evidence is recorded in the event log. In this case, in the form of resources performing activities. To analyze this, we propose to focus on the number of active resources along the process. As work in processes tends to follow circadian cycles (every day/week following the same pattern), our proposal is to develop on the circadian distribution distance, computing the number of active resources over time, instead of computing the number of executed events.

Circadian Workforce Distribution (CWD). Given two event logs L_1 and L_2, the CWD computes the average difference in the distribution of active resources throughout the process timeline, by day of the week. This metric specifically evaluates how the number of active resources differs over time within each day of the week.

Filter. Given an event log L, do not filter out any event in the log. Thus, the predicate for the filtering operation is defined by Eq. (2).

Project. Given a filtered event log result of the previous operation, project only the start timestamp, end timestamp, and resource of each event in the log, i.e., $\lambda = \{\tau_s, \tau_e, \rho\}$.

Transform. Let $wd(\tau)$ and $h(\tau)$ be two functions returning, respectively, the day of the week and hour of a timestamp τ. Given a projected event log $L^{\lambda} = \{E, \sigma\}$ where each event follows the schema $\sigma = \{\tau_s, \tau_e, \rho\}$, and being $\zeta : \mathcal{R} \to \mathbb{N}$ a function that maps each resource identifier to a unique natural number, we define the following transform operation:

$$\mathcal{T}(L^{\lambda}) = \{\{(wd(\tau_i), h(\tau_i), \zeta(\rho(e))) \mid \tau_i = \tau_s(e) \wedge e \in E\}\} \cup \{\{(wd(\tau_i), h(\tau_i), \zeta(\rho(e))) \mid \tau_i = \tau_e(e) \wedge e \in E\}\}. \tag{19}$$

This operation creates a multiset of three-dimensional vectors, where each $v \in V$ stores the day of the week, hour, and resource of a start or end timestamp from L^{λ}.

Compare. Given two multisets V_1 and V_2 of three-dimensional vectors, result of the transformation in the previous operation. The compare operation first builds two matrices of 7 rows, x and y, where the dth row contains the time series for the dth day of the week, with the following prepare function:

$$\pi(V_1, V_2) = ((x_{1,1}, x_{1,2}, \dots, x_{7,24}), (y_{1,1}, y_{1,2}, \dots, y_{7,24})) \mid d = 1, \dots, 7,\ i = 1, \dots, 24,\ x_{d,i} = |\{z \mid (d, i, z) \in V_1\}|,\ y_{d,i} = |\{z \mid (d, i, z) \in V_2\}|. \tag{20}$$

The function counts the number of unique resources performing events recorded in each hour for each day of the week; Then, we measure the distance between x and y using the average of the Earth Movers' Distance between $x_{d,1}, \dots, x_{d,23}$ and $y_{d,1}, \dots, y_{d,23}$ with $d \in \{1, \dots, 7\}$, i.e.,

$$\delta(x, y) = \frac{1}{7} \sum_{d=1}^{7} \text{EMD}(x_d, y_d). \tag{21}$$

5.4. Congestion

The last perspective of a process that we propose to consider is the congestion in the system. To measure the capability of a model to represent congestion, we rely on queuing theory, a field in applied probability that studies the behavior of congested systems [39]. In steady state, Little's Law states that

$$WiP = \lambda \times CT, \tag{22}$$

where WiP denotes the work in progress in the system (i.e., the number of active work items at a specific point in time), λ denotes the case-arrival rate, and CT represents the average cycle time of a case. To cover this, we propose two distance measures to analyze the performance of a BPS model w.r.t. the case-arrival rate, and the case cycle times. Regarding the work in process in the system, it can be analyzed as the combination of the aforementioned components, as stated in Eq. (22).

Case Arrival Rate (CAR). This measure compares case arrival patterns (shape) and counts (number of arrivals per bin). Given two event logs L_1 and L_2, the CAR computes the difference in the temporal distribution of case arrivals throughout the process timeline. This metric specifically evaluates how the arrival of new process cases differs over time.

Filter. Retain only, for each case, the event with the earliest start timestamp:

$$\phi(e, L) = \tau_s(e) == \min\{\tau_s(e') \mid e' \in L \wedge \xi(e) = \xi(e')\} \tag{23}$$

Project. Given a filtered event log result of the previous operation, project only the start timestamp of each event in the filtered log, i.e., $\lambda = \{\tau_s\}$.

Transform. Let $y(\tau)$, $m(\tau)$, $d(\tau)$, and $h(\tau)$ be functions returning, respectively, the year, month, day, and hour of a timestamp τ. Given a projected event log $L^{\lambda} = \{E, \sigma\}$ where each event follows the schema $\sigma = \{\tau_s\}$, we define the following transform operation:

$$\mathcal{T}(L^{\lambda}) = \{\{(y(\tau_i), m(\tau_i), d(\tau_i), h(\tau_i)) \mid \tau_i = \tau_s(e) \wedge e \in E\}\}. \tag{24}$$

This operation creates a multiset of four-dimensional vectors where each $v \in V$ stores the year, month, day, and hour of a case arrival timestamp from L^{λ}.

Compare. Given two multisets V_1 and V_2 of four-dimensional vectors, result of the transformation in the previous operation, and the sorted sequence $H_o = \langle w_1, w_2, \dots, w_B \rangle$ storing all the hours in the timeline of the vectors V_1 and V_2, such that each $w_i = (y_i, m_i, d_i, h_i) \in H_o$ stores the year, month, day and hour. The compare operation first builds two time series x, y with the prepare function defined in Eq. (10), i.e., by counting the number of events occurring in each hour within the timeline; then, we measure the distance between x and y using the Earth Movers' Distance between $x_1, \dots, x_B$ and $y_1, \dots, y_B$, i.e.,

$$\delta(x, y) = \text{EMD}(x, y). \tag{25}$$

Cycle Time Distribution (CTD). Here, we seek to measure the ability of the BPS model to capture the end-to-end cycle time of the process. Given two event logs L_1 and L_2, the CTD collects all cycle times of the process into a single histogram (per event log), which depicts their empirical probability distribution functions (PDF). Then, it compares the distance between these two histograms.

Filter. Retain only, for each case, the event(s) with the earliest start timestamp and latest end timestamp.

$$\phi(e, L) = \nexists(e' \in L)\ (\tau_s(e') < \tau_s(e) \vee \tau_e(e') > \tau_e(e)) \mid \xi(e') = \xi(e). \tag{26}$$

Project. Given a filtered event log result of the previous operation, project only the case identifier, start, and end timestamps of each event in the log, i.e., $\lambda = \{\xi, \tau_s, \tau_e\}$.

Transform. Given a projected event log $L^{\lambda} = \{E, \sigma\}$ where each event follows the schema $\sigma = \{\xi, \tau_s, \tau_e\}$, and being $\lfloor\ \rfloor_h$ the *floor* operator that returns the number of complete hours from a time duration. We define the following transform operation:

$$\mathcal{T}(L^{\lambda}) = \{\{(\ \lfloor \tau_e(e') - \tau_s(e) \rfloor_h\ ) \mid \tau_s(e) \leq \tau_s(e') \wedge \tau_e(e') \geq \tau_e(e) \wedge \xi(e) = \xi(e') \wedge e, e' \in E\}\}. \tag{27}$$

This operation creates a multiset of one-dimensional vectors, where each $v \in V$ stores the duration (in complete hours) of a case in the process.

Compare. Given two multisets V_1 and V_2 of one-dimensional vectors, result of the transformation in the previous operation. The compare operation first builds two histograms x, y with the prepare function defined in Eq. (17), i.e., by counting the number of cases with a duration of i hours; and then, measures the distance between x and y using the 1-Wasserstein Distance (1-WD)[14] between $x_1, \dots, x_B$ and $y_1, \dots, y_B$, i.e.,

$$\delta(x, y) = 1WD(x, y). \tag{28}$$

6. Evaluation

We report on a two-fold experimental evaluation. The first part aims to validate the applicability of the proposed measures by testing the following evaluation question: *are the proposed measures able to discern the impact of different known modifications to a BPS model?* **(EQ1)**. Given the potential efficiency issues of CFLD, the first part of the evaluation also aims to answer the question: *is the N-Gram Distance's performance significantly different from the CFLD's performance?* **(EQ2)**. The second part of the evaluation is designed to test if: *given two BPS models discovered by existing automated BPS model discovery techniques in real-life scenarios, are the proposed measures able to identify the strengths and weaknesses of each technique?* **(EQ3)**. Given the complexity of the EMD (cf. Section 3), the second part of this evaluation also focuses on answering: *does the 1-WD report the same insights in real-life scenarios as the EMD?* **(EQ4)**.

In the case of the NGD, we report on this measure for a size $N = 2$.[15] The distance computed by the EMD is not directly interpretable, as it is an absolute number on a scale that depends on the range of values of the input time series. Accordingly, we divide the raw EMD by the number of observations in the original log. In this way, we can interpret the resulting scaled-down EMD as the average number of bins that each observation of the original log must be moved to transform it into the simulated log. For example, a value of 10 implies that, on average, each observation had to be moved 10 bins. This interpretation is, in fact, the same as the interpretation of the 1-WD computation.

6.1. Synthetic evaluation

We designed two experiments to assess different aspects of EQ1. The first one (EQ1.1) aims to evaluate if the proposed measures are able to detect the impact of a known modification to a BPS model. The second one (EQ1.2) focuses on evaluating if the proposed measures are sensible to different levels of perturbations, i.e., if they not only detect the impact of a modification, but also its strength.

Datasets and Setup. To assess EQ1.1, we manually created the BPS model of a loan application process based on the examples from [1, Chapter 10.8]. The process comprises 12 activities (with one loop, a 3-branch parallel structure, 3 exclusive split gateways, and 3 possible endings) and 6 different resource types (performing different activities with a working schedule from Monday to Friday, from 9am to 5pm). We simulated a log of 1000 cases as the log recording the process (i.e., the ALog). We created 7 modifications of the original BPS model: (i) altering the control-flow by arranging the parallel activities as a sequence (Loan_{SEQ}); (ii) altering, on top of the previous modification, the branching probabilities ($\text{Loan}_{S\text{-}G}$); (iii) modifying the rate of case arrivals (Loan_{ARR}); (iv) increasing the duration of the activities of the process (Loan_{DUR}); (v) halving the available resources to create resource contention (Loan_{RC}); (vi) changing the resource working schedules from 9am–5 pm to 2pm–10 pm (Loan_{CAL}); and *(vii)* adding timer events to simulate extraneous waiting time [12] delaying the start of 4 of the activities (Loan_{EXT}).

To assess EQ1.2, we created 15 additional modifications of the original BPS model: (i) altering the control-flow by arranging the two parallel activities as a sequence, (additionally) interchanging the order of two sequential activities, and (additionally) renaming one activity present in all cases (designed to analyze NGD); (ii) increasing the duration between each two case arrivals from 30 to 60, 90, and 120 min (designed to analyze AED and CAR); (iii) increasing the duration of some activities of the process in two levels, and (additionally) injecting timer events to delay each execution of two activities (designed to analyze RED and CTD); (iv) incrementally increasing the number of available resources (designed to analyze CED); and (v) displacing the resource working schedules in 3, 6, and 9 h (designed to analyze CWD).

In order to consider the impact of different levels of structuredness in a process when assessing EQ2, we used both the aforementioned BPS model – "lasagna-like" process – and a BPS model from a procure-to-pay process that we manually modified to create a "spaghetti-like" structure. This process comprises 13 activities with three single-entry single-exit substructures sorted in a sequence: two "flower-like" structures of five and four activities, and one sequence of four activities with

[14] In this measure, we propose to compute the distance between x and y with the 1-WD because we assume $|x| = |y|$, i.e., both event logs have the same number of cases. Thus, they can be relativized into their CDF without distorting the error.

[15] Augusto et al. [32] found that, for models with no duplicate activities, the size of $N = 2$ captures enough information to compare processes from a control-flow perspective.

Table 2
Results (average and 95% confidence interval) of the proposed measures for the original and modified BPS models of a loan application process.

	NGD	CFLD	AED	CED
$Loan_{GT}$	0.02 (±0.00)	0.02 (±0.00)	2.39 (± 0.51)	0.05 (±0.01)
$Loan_{SEQ}$	0.34 (±0.00)	0.20 (±0.00)	2.58 (± 0.34)	0.05 (±0.01)
$Loan_{S\text{-}G}$	0.46 (±0.00)	0.32 (±0.00)	44.17 (± 9.59)	0.24 (±0.01)
$Loan_{ARR}$	0.04 (±0.01)	0.03 (±0.00)	35.66 (±14.46)	0.08 (±0.01)
$Loan_{DUR}$	0.03 (±0.01)	0.02 (±0.00)	4.29 (± 1.34)	0.05 (±0.01)
$Loan_{RC}$	0.23 (±0.03)	0.15 (±0.01)	23.67 (± 8.81)	0.06 (±0.01)
$Loan_{CAL}$	0.02 (±0.00)	0.02 (±0.00)	6.27 (± 0.37)	3.51 (±0.02)
$Loan_{EXT}$	0.02 (±0.01)	0.02 (±0.00)	5.43 (± 1.64)	0.09 (±0.01)
	RED	CWD	CAR	CTD
$Loan_{GT}$	0.22 (± 0.05)	0.04(±0.00)	0.00 (± 0.00)	7.06 (± 1.41)
$Loan_{SEQ}$	1.91 (± 0.26)	0.04(±0.00)	0.00 (± 0.00)	42.38 (± 3.83)
$Loan_{S\text{-}G}$	235.36 (±12.93)	0.20(±0.01)	0.00 (± 0.00)	7,667.13 (±443.38)
$Loan_{ARR}$	0.62 (± 0.16)	0.06(±0.01)	42.39 (±14.12)	11.89 (± 2.42)
$Loan_{DUR}$	7.09 (± 0.51)	0.04(±0.00)	0.09 (± 0.04)	200.53 (± 15.66)
$Loan_{RC}$	31.66 (± 8.62)	0.32(±0.00)	0.03 (± 0.02)	759.45 (±210.93)
$Loan_{CAL}$	0.26 (± 0.06)	3.50(±0.01)	6.24 (± 0.00)	7.51 (± 1.94)
$Loan_{EXT}$	8.02 (± 0.18)	0.08(±0.00)	0.00 (± 0.00)	262.30 (± 6.19)

different decision choices, allowing the execution flow to skip some of the activities and go back in the sequence. To evaluate the effective unstructuredness of this model, we performed a simulation of 1000 cases, producing an event log with more than 900 case variants.[16] We then followed the procedure mentioned above, creating 7 modified BPS models.

For each modified BPS model, we simulated $K = 10$ logs with 1000 cases (as the GLogs). According to the proposed framework, to assess their quality in each dimension, we computed the distance of each GLog with the proposed measures, and reported the average and 95% confidence interval. We used the t-score method to compute the confidence interval, as it offers a more appropriate and cautious approach for smaller samples of unknown distribution, ensuring better statistical reliability and validity.

Results and Discussion. Regarding EQ1.1, Table 2 shows the results of the proposed measures for each modified scenario of the loan application process, and for the original BPS model as ground truth ($Loan_{GT}$) to measure the distance associated with the stochastic nature of the simulation. The results show how the proposed measures appropriately penalize the BPS models for the modifications affecting their corresponding perspectives. In the control-flow measures, the BPS models showing significant differences w.r.t. the ground truth are those with control-flow modifications. The distances of $Loan_{RC}$ and $Loan_{SEQ}$ are explained by the parallel activities being executed more frequently in a specific order. In the first BPS model, due to resource contention, which delays the execution of one of the parallel activities in some cases. In the second one, due to the control-flow modification. Finally, $Loan_{S\text{-}G}$ reports the highest distance as, in addition to the modification in $Loan_{SEQ}$, it also alters the frequency of each process variant.

For temporal measures, the AED distance captures the difference in the distribution of events along the entire process. However, to identify the sources of these differences. We require a combination of the penalties incurred by CED, RED, and CAR. Thus, we must analyze them to find the root causes for the discrepancies in AED. Starting from the seasonal aspects captured by CED, only $Loan_{S\text{-}G}$ and $Loan_{CAL}$ report significant differences, being the latter the only BPS model altering seasonal aspects. $Loan_{S\text{-}G}$'s distance is due to the change in the gateway probabilities, which in turn impacts the overall distribution of executed events. As expected, $Loan_{CAL}$ presents the highest CED distance due to the change in schedules that displaces executed events from morning to evening.

Moving to RED, which reports the distance in the distribution of events over time within each case, we observe that all modifications except $Loan_{CAL}$ should affect this perspective. The slightly higher penalization of $Loan_{ARR}$ is due to the higher case arrival rates, which delay the start activities due to resource contention. $Loan_{SEQ}$ presents a higher distance (close to a displacement of 2 h per event) as the three parallel activities are executed as a sequence, delaying subsequent activities. Similarly, in $Loan_{DUR}$, $Loan_{EXT}$, and $Loan_{RC}$, activity delays are caused by longer durations, extraneous delays, and resource contention waiting times, respectively. Finally, $Loan_{S\text{-}G}$ presents the highest RED distance due to the high frequency differences in each process variant.

Switching to CAR, we do not observe significant differences in BPS models that exhibit the same arrival rate, except for $Loan_{CAL}$. The latter is explained by the change in schedules, as cases cannot start until the resources start their working period (which skews effective start times). Unsurprisingly, for $Loan_{ARR}$, the difference in CAR is due to the change in the arrival model.

Concerning CWD, it proved its ability to analyze the distribution of resources more precisely than CED. The analysis of the circadian cycles in CED detects changes in the availability calendars of the resources ($Loan_{CAL}$), but not in the number of available resources ($Loan_{RC}$). This is because the temporal distribution of recorded events within each day is displaced by altering the availability calendars, but not by changing the number of available resources. On the contrary, the analysis of the circadian workforce of CWD, which focuses on the number of active resources per hour, detects both the change in the availability calendars and the number of resources ($Loan_{CAL}$ and $Loan_{RC}$). In this case, the high CWD distance of $Loan_{S\text{-}G}$ is explained by the abovementioned reasons.

Finally, the last proposed measure is CTD, which reports the distance in case duration among all the cases. The results of CTD follow a similar pattern to RED, yet with different values, since cycle times correspond to the time distance between the first and last events of the case. However, this correlation might not hold across all scenarios. Specifically, if the distribution of executed activities in the middle of each case is different, but the last event does not change, RED would detect discrepancies that CTD would not (as the cycle time would remain the same). Thus, CTD is most relevant when the analysis revolves around total cycle times, while disregarding the temporal distribution of events within the case.

With respect to EQ1.2, Fig. 6 shows the evolution of each proposed measure for the ground truth and the three modified BPS models designed to impact it with different levels of intensity. The graphics show how all the distance metrics grow according to the level of intensity of the perturbation. In the case of the NGD, the modifications to the

[16] A case variant of an event log is a unique sequence of activities associated with a particular case, and its frequency corresponds to the number of cases in the event log that follows that sequence.

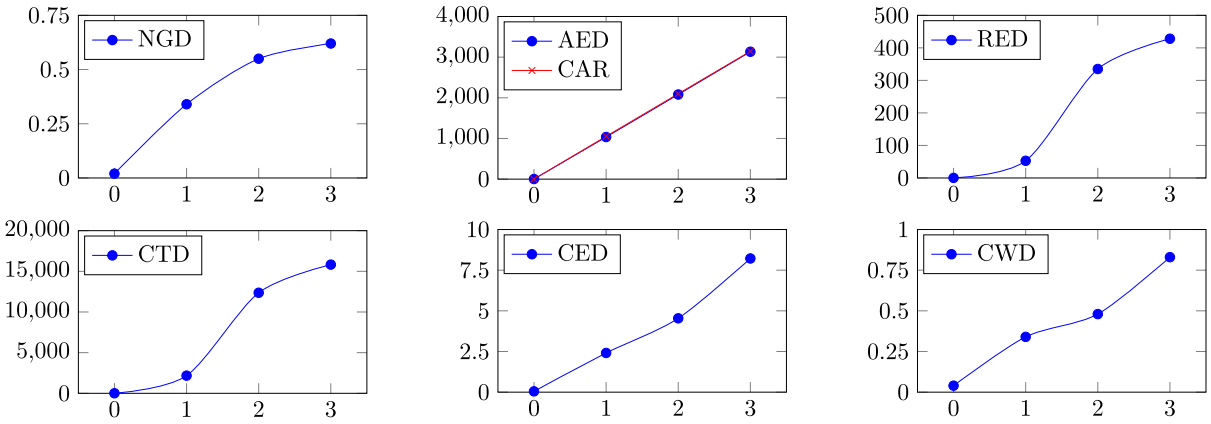


Fig. 6. Results of the proposed measures for BPS models with different levels of perturbations, where the y axis denotes the distance value, and the x axis the level of intensity of the perturbation.

Table 3
NGD and CFLD (average and 95% confidence interval) for the original and modified BPS models of the ("lasagna-like") loan application and the ("spaghetti-like") procure-to-pay processes sorted in ascending order by their NGD value.

	NGD	CFLD
Loan_{GT}	0.02 (±0.00)	0.02 (±0.00)
Loan_{CAL}	0.02 (±0.00)	0.02 (±0.00)
Loan_{EXT}	0.02 (±0.01)	0.02 (±0.00)
Loan_{DUR}	0.03 (±0.01)	0.02 (±0.00)
Loan_{ARR}	0.04 (±0.01)	0.03 (±0.00)
Loan_{RC}	0.23 (±0.03)	0.15 (±0.01)
Loan_{SEQ}	0.34 (±0.00)	0.20 (±0.00)
$\text{Loan}_{S\text{-}G}$	0.46 (±0.00)	0.32 (±0.00)
P2P_{CAL}	0.03 (±0.00)	0.30 (±0.00)
P2P_{EXT}	0.03 (±0.00)	0.30 (±0.00)
P2P_{GT}	0.04 (±0.00)	0.30 (±0.00)
P2P_{ARR}	0.04 (±0.00)	0.30 (±0.00)
P2P_{RC}	0.04 (±0.00)	0.30 (±0.00)
P2P_{DUR}	0.04 (±0.00)	0.30 (±0.00)
P2P_{SEQ}	0.31 (±0.00)	0.37 (±0.00)
$\text{P2P}_{S\text{-}G}$	0.39 (±0.00)	0.40 (±0.00)

control-flow increase the number of mismatching n-grams, increasing the total distance value. For all the other measures, the use of the Earth Movers' Distance (or Wasserstein Distance) provides them with the ability to capture the intensity of the modifications. As explained in Section 3, these distance metrics capture not only the difference in the mass, but also the distance between such differences.

To answer EQ2, Table 3 shows the results of NGD and CFLD for both the structured and unstructured processes. We computed the Kendall rank correlation coefficient between NGD and CFLD, and we obtained a correlation of 1.0. Thus, in light of the complexity of CFLD (cf. Section 5), we recommend using NGD to assess the quality of a BPS model from the control-flow perspective.

6.2. Real-life evaluation

Datasets and Setup. To evaluate EQ3 and EQ4, we selected four real-life logs of different complexities: (i) a log from an academic credentials' management process (AC_CRE), containing a high number of resources exhibiting low participation in the process. (ii) a log of a loan application process from the Business Process Intelligence Challenge (BPIC) of 2012 [40] – we preprocessed this log by retaining only the events corresponding to activities performed by human resources (i.e., only activity instances that have a duration). (iii) the log from the BPIC of 2017 [41] – we pre-processed this log by following the recommendations reported by the winning teams participating in the competition.[17] And (iv) a log from a call centre process (CALL) containing numerous cases of short duration – on average, two activities per case. To avoid data leakage, we split the log of each dataset into two sets (*training* and *testing*). These datasets correspond to disjoint (non-overlapping) intervals in time with similar case and event intensity. The training dataset contains cases that are fully contained in the training period, and same for the testing dataset. Table 4 shows the characteristics of the four training and four testing event logs.

For each dataset, we ran two automated BPS model discovery techniques (SIMOD and ServiceMiner) on the training log, and evaluated the quality of the discovered BPS models on the test log following the same procedure as in the assessment of EQ1 and EQ2 (i.e., $K = 10$ and a t-score confidence interval of 95%).

Results and Discussion. Answering EQ3, Table 5 shows the results of the proposed measures for the BPS models automatically discovered by SIMOD and ServiceMiner (henceforth M_{SIMOD} and M_{SerMin}, respectively). From the control-flow perspective, M_{SerMin} performs closer to the original log than M_{SIMOD} for all four datasets. The reason lies in the methods that the approaches use to model the control-flow. SIMOD is designed to discover an interpretable process model to support modification for *what-if?* analyses. To this end, SIMOD uses a model discovery algorithm that applies multiple pruning techniques to simplify the discovered model. Conversely, ServiceMiner discovers a Markov chain, which yields more accurate results, yet can lead to complex 'spaghetti models'.

Two main differences are reported w.r.t. the temporal and congestion aspects. First, for Case Arrival Rate (CAR), M_{SerMin} presents better results in BPIC12, BPIC17, and CALL, while M_{SIMOD} outperforms it in AC_CRE. To model the arrival of new cases, ServiceMiner splits the timeline into one-hour windows, and bootstraps the arrivals per time window. SIMOD computes the inter-arrival times (i.e., the time between each arrival and the next one) and estimates a parametrized distribution to model them. The complexity of ServiceMiner's arrival model allows it to capture better the arrival rate in scenarios where the density of case arrivals per hour is high, and/or the rate of arrivals varies through time (BPIC12, BPIC17, and CALL). On the contrary, if cases are scattered over time (AC_CRE), SIMOD's approach presents a better result.

The second main difference lies in the Relative Event Distribution (RED) and the Cycle Time Distribution (CTD) distances. Here, M_{SIMOD} obtains better results in both measures except in one case. In the CALL dataset, M_{SerMin} obtains a smaller RED distance (both methods perform well w.r.t. the original log). SIMOD outperforms ServiceMiner due to a high amount of extraneous activity delays (i.e., waiting times not

[17] https://www.win.tue.nl/bpi/2017/challenge.html

Table 4
Characteristics of the real-life logs used in the evaluation.

Event log	Cases	Activity instances	Variants	Activities	Resources
AC_CRE_TR	398	1,945	54	16	306
AC_CRE_TE	398	1,788	35	16	281
BPIC12_TR	3,030	16,338	735	6	47
BPIC12_TE	2,976	18,568	868	6	53
BPIC17_TR	7,402	53,332	1,843	7	105
BPIC17_TE	7,376	52,010	1,830	7	113
CALL_TR	260,889	445,567	1,689	19	2,712
CALL_TE	260,890	454,807	1,573	19	2,711

Table 5
Distance measures for the BPS models discovered by SIMOD and ServiceMiner on the logs in Table 4. The CFLD ran out of memory (48 GB of allocated memory) on the CALL dataset after > 2 h, thus no values are reported in those cells.

		AC_CRE	BPIC12	BPIC17	CALL
NGD	SIMOD	0.24 (±0.01)	0.56 (±0.00)	0.37 (±0.00)	0.08 (±0.00)
	ServiceMiner	0.13 (±0.01)	0.13 (±0.01)	0.06 (±0.00)	0.04 (±0.00)
CFLD	SIMOD	0.21 (±0.00)	0.55 (±0.00)	0.34 (±0.00)	–
	ServiceMiner	0.18 (±0.01)	0.16 (±0.00)	0.06 (±0.00)	–
AED	SIMOD	91.72 (± 16.66)	61.57 (±10.80)	192.18 (±33.03)	48.13 (±0.11)
	ServiceMiner	298.17 (±105.87)	29.22 (± 8.33)	51.24 (±11.90)	1.67 (±0.13)
CAR	SIMOD	110.38 (± 16.94)	336.42 (±42.98)	390.04 (±43.39)	61.68 (±0.00)
	ServiceMiner	327.85 (±118.33)	153.25 (±24.76)	121.90 (±23.60)	3.57 (±0.20)
CED	SIMOD	2.22 (±0.09)	20.55 (±0.82)	10.72 (±0.50)	18.17 (±0.04)
	ServiceMiner	1.63 (±0.19)	28.52 (±1.24)	9.87 (±0.42)	0.70 (±0.01)
CWD	SIMOD	2.27 (±0.23)	5.92 (±0.26)	7.29 (±0.39)	38.86 (±0.08)
	ServiceMiner	–	–	–	–
RED	SIMOD	9.96 (±6.46)	3.99 (±0.95)	62.80 (±2.16)	0.10 (±0.00)
	ServiceMiner	70.49 (±4.08)	45.91 (±2.09)	149.60 (±0.42)	0.03 (±0.01)
CTD	SIMOD	62.23 (±1.65)	93.45 (±0.71)	102.85 (±0.84)	8.18 (±0.08)
	ServiceMiner	99.88 (±0.30)	124.21 (±0.33)	112.83 (±0.16)	12.04 (±0.07)

related to the resource allocation or activity performance) exhibited in these processes. Specifically, SIMOD includes a component to discover extraneous delays, which improves the distribution of the events within the case. Both techniques perform close to the original log in the CALL dataset because extraneous delays are rare for the call centre process.

For seasonality, the Circadian Event Distribution (CED) reports slight differences between the two methods for AC_CRE and BPIC17, and a moderately better result for M_{SIMOD} in BPIC12. The CALL dataset presents the highest difference, where M_{SerMin} obtains better results, which can be attributed to its highly accurate arrival model. The CALL dataset has mostly cases with one or two events. Hence, case execution depends more on the arrival time of the case, than on the activity performance and congestion models.

Combining all the temporal perspectives in one measure, the results of Absolute Event Distribution (AED) follow the same distribution as CAR, where M_{SerMin} presents better results in BPIC12, BPIC17, and CALL, while M_{SIMOD} performing better in AC_CRE. Although this measure summarizes all the temporal performance in one, it is highly affected by the performance of the arrival model. A wrong arrival rate propagates the error to all the events per case, displacing them even if their relative distribution is accurate.

Finally, regarding the CWD, we only report values of this measure for M_{SIMOD} due to the need for explicit resource information. SIMOD produces an event log where each activity instance has an associated resource, while ServiceMiner uses a queuing system that focuses on the congestion aspects of the simulation, and does not produce explicit information about the resources.

The proposed measures detected key differences between the considered BPS model discovery techniques. Additionally, our results can help to identify potential improvements in these techniques. SIMOD's inferior performance in the control-flow perspective is expected, given that it takes a simplified process model as input. Moreover, there is a natural fit between the control-flow measures (e.g., NGD) and the Markovian approach of ServiceMiner – as a Markov chain is, in essence, a generative 2-gram model. The results also highlight the benefits of SIMOD's extraneous waiting time discovery component (a feature that ServiceMiner does not have). Finally, although ServiceMiner's arrival model achieved the best results in most of the scenarios, the evaluation in AC_CRE point toward an improvement opportunity in the situation where cases arrive at a slow rate.

To evaluate EQ4, Table 6 shows the result of AED, CAR, CED, CWD, and RED measures when computing the distance with 1WD, instead of EMD. The results follow the same distribution in most of the cases. The only difference resides in the CED measure on the BPIC17 dataset, and the RED measure on the CALL dataset. In both cases, the small differences shown by EMD are reduced to a similar value by both techniques. Finally, for arrivals, as explained in Section 3, computing the distance with EMD and 1WD provide the same result, as the number of observations in both samples is the same (i.e., the number of cases). In conclusion, computing the distance using 1WD leads to similar conclusions at a lower computational cost. Thus, we recommend using 1WD when the masses of both time series are close to each other, and when the number of observations (amount of mass) is large.

7. Discussion

In this article, we studied the quality assessment of BPS models. Due to the existent variety of BPS model formats and BPS engines, we advocate for a generic approach – relying on the comparison through simulated event logs – rather than a specific approach that requires the BPS model to adjust to a certain format. In this way, the proposed approach can be used for a wide spectrum of BPS model formats, as long as they produce an event log recording the simulated behavior. Within this assessment, we studied the intrinsic complexity of BPS models, identifying different components that interact and influence

Table 6
Results of the proposed measures for the BPS models discovered by SIMOD and ServiceMiner with the real-life logs in Table 4.

		AC_CRE	BPIC12	BPIC17	CALL
AED	SIMOD	117.32 (± 18.85)	313.30 (±41.50)	314.92 (±43.02)	61.76 (±0.08)
	ServiceMiner	315.88 (±115.60)	79.19 (±16.12)	65.56 (±12.55)	3.47 (±0.19)
CAR	SIMOD	110.38 (± 16.94)	336.42 (±42.98)	390.04 (±43.39)	61.68 (±0.00)
	ServiceMiner	327.85 (±118.33)	153.25 (±24.76)	121.89 (±23.60)	3.57 (±0.20)
CED	SIMOD	3.11 (±0.18)	2.10 (±0.07)	1.65 (±0.03)	4.72 (±0.00)
	ServiceMiner	2.50 (±0.27)	2.34 (±0.03)	1.65 (±0.02)	1.03 (±0.03)
CWD	SIMOD	3.17 (±0.17)	2.12 (±0.06)	1.59 (±0.04)	4.50 (±0.00)
	ServiceMiner	–	–	–	–
RED	SIMOD	48.19 (±1.72)	96.82 (±1.61)	132.31 (±1.68)	0.00 (±0.00)
	ServiceMiner	74.50 (±0.18)	150.83 (±0.57)	150.26 (±0.28)	0.00 (±0.00)

each other. Given that the main reason for this quality assessment is to identify the BPS model that better reflects the behavior of the process – in order to perform *what-if?* analysis –, there is an interest in identifying the source of discrepancies w.r.t. the real process. For this reason, even though our framework allows for the proposal of a generic measure that summarizes all perspectives into one distance value, we propose to tackle the problem by individually analyzing each perspective.

The idea of assessing the quality of BPS models from multiple perspectives, opens an opportunity for BPS model optimization. For example, an automated BPS model discovery technique can optimize the discovery of each of its components independently or jointly. For example, the component of the BPS model discovery technique that discovers the case arrival model can be optimized using Case Arrival Rate (CAR) distance, while the component that discovers the control-flow perspective (the process model) can be optimized against the N-Gram Distance, for example. The developer of the technique may opt to optimize these two components separately or jointly.

Furthermore, the proposed measures can guide an analyst when manually tuning an automatically discovered BPS model. In this context, the analyst can identify potential deviations from the behavior of the process, e.g., an imprecise resource model detected by a big distance in this perspective (CWD). Then, they can adjust the discovered resource model and evaluate its performance to assess if the adjusted BPS model reflects better the behavior of the process.

In our study, we identify four main perspectives when analyzing state-of-the-art BPS techniques. The control-flow perspective is one of the most important dimensions in the process, modeling the path – i.e., the sequence of activities – that each case follows in the process. To assess the quality w.r.t. this dimension, we present one state-of-the-art measure adapted to our framework (CFLD), and propose a new measure with better computational efficiency (NGD). The CFLD measure is more precise and sensible to long-term dependencies due to its complete comparison trace-to-trace, while NGD focuses on local relations by comparing the distribution of n-grams along the log. Both measures showed similar tendencies in both structured and unstructured processes. Accordingly, we propose to use NGD in situations where efficiency is important – e.g., when automatically discovering the best control-flow model – or when the size of the log exceeds the few thousand cases.

Regarding the temporal dimension, we first presented a measure that gives a global overview of the temporal distribution of events along the process (AED). This measure showed the ability to assess the quality of BPS models w.r.t. the temporal dimension, but not to detect the source of discrepancies. We then identified different aspects that influence this dimension, and proposed a set of measures that focus on each of them. The first one focuses on the rate of arrival of new cases (CAD). Although under the umbrella of the congestion perspective, it has shown to be highly related to the temporal perspective. A wrong case arrival rate can totally change the temporal distribution of events in the process. For example, a simulation of cases arriving more frequently than in reality might provoke a high resource contention, delaying the execution of all future activities of the process. The second proposed measure (RED) focuses on the temporal distribution of events within each case (i.e., when they do occur since their case arrived). This measure showed ability to identify discrepancies in the simulation of processing and waiting times. A difference in the processing times of an activity, or in the waiting times previous to it, will displace their execution and of the future activities in the case, being reflected in the RED. The last proposed temporal measure is CED, focusing on the distribution of events within each day of the week. CED showed the lower impact on the overall temporal performance, and proved to be more useful when analyzing the modeled resource availability (i.e., their working calendars).

The CED measure focuses on the availability of the resources w.r.t. their working calendars, but is not sensitive to the workforce of the process, i.e., the number of available resources. After identifying the importance of the workforce perspective in BPS, we proposed a measure to estimate the quality of a BPS model w.r.t. this perspective (CWD). This measure focuses on the average number of active resources that showed to be active along the week. Accordingly, it showed usefulness to detect not only how close the simulated availability of the resources (i.e., their working calendars) is to reality, but also the number of resources.

Finally, as the other measure related to the congestion perspective, we proposed CTD. This is a KPI-based measure, focusing on the distribution of cycle times (i.e., case durations) along the process. This measure showed a high correlation with RED, and can be seen as a simplified version of it, only considering the first and last events of each case. The usefulness of this measure is more related to situations where only the overall performance of the BPS model is important. In the context of optimizing the discovered BPS model, its usefulness is limited, especially if RED is available.

Through this study, we discovered a high complexity inherent to the simulation of business processes. The execution of a process depends on many components: the (control-flow) paths followed, how congested the system is, the number of available resources at each point in time, etc. All these interconnected aspects influence each other, making it impractical to analyze each of them in isolation. For example, an inaccurate control-flow model may not only alter the number of executed activities in a case, but also which activities are executed. This might also impact the temporal distribution of events within a case, as the number and duration of the executed activities differ. In a different setting, an arrival model that simulates a higher arrival rate may increase the congestion in the system, delaying the execution of the pending activities, and also impacting the cycle times of the process. Given these interconnections, altering a BPS model in one dimension might also affect others. Clearly, an optimization based on a generic metric is undesirable, as it provides no guidance on which parts of the BPS model to focus the changes. A set of measures focusing on different perspectives, as the ones researched in this paper, offer a better understanding of the discrepancies between the BPS model and the real process. However, the optimization (either manual or automated) of a BPS model should not be a linear process in which each component is individually optimized after each other. Instead, each

component should be optimized according to a measure that specializes on it, but later revisiting previously optimized models to ensure the changes made did not worsen their performance.

8. Limitations and validity

This section introduces the limitations of the proposed framework, as well as the threats to validity associated with this publication.

8.1. Limitations

This article proposes a framework to assess the quality of a BPS model by generating a set of simulated event logs, and comparing such logs with the ground truth – i.e., the actual log of the system. This allows for a generic comparison of BPS models, independently of their composition – e.g., white-box vs black-box components. However, the comparison through simulated logs presents three main limitations. First, it narrows the – potentially – infinite grammar of the model to a finite sample, limiting the evaluated representativeness of the model. For example, uncommon paths or resources performing activities in unlikely – but possible – situations may not be represented in the simulated logs. To mitigate this limitation, this framework proposes to generate a set of K simulated logs (see Section 4). Second, the need for the BPS model to generate an event log restricts the quality assessment to discrete event simulation models. Accordingly, as mentioned in Section 2, this framework is not suitable to assess the quality of BPS models based on system dynamics [15], unless these are refined to generate event logs. Third, assessing the quality of the BPS model through event logs impedes the evaluation of its structuredness.

Considering the actual event log of the process as the ground truth also presents some limitations. First, it assumes all the behavior of the process is reflected in the actual event log. Although this is a strong assumption, typically, the event log of a process is the only standardized data available. Thus, this limitation is necessary to avoid a quality assessment method that depends on specific uncommon process data, and/or on manual interactions. Second, our proposal is based on the availability of an event log with information such as start and end times. This limitation is inherited from business process simulation itself, in which the duration of the activities has to be known. However, techniques to estimate the start of each activity instance [8,42] could be used for event logs containing only end timestamps.

The presented framework generalizes the comparison of a simulated event log with the actual event log of the system as a pipeline consisting of four operations. This modularization presents different advantages. For example, structuring each measure as a set of four modules makes it more intuitive to modify one measure – e.g., Absolute Event Distribution – in order to obtain measures that focus on different perspectives – e.g., Relative Event Distribution or Case Arrival Distribution – while working on the same principles. On the other hand, we acknowledge that the framework may require substantial effort to be instantiated, particularly when a new BPS quality measure requires its own transformation or comparison operator. For example, if we wanted to instantiate the framework to define new quality measures that take into account domain-dependent data attributes in a business process (e.g. attributes such as the "sales order amount" or "product type" or "production status" in an order-to-cash process), we would need to define transformation operators to encode those attributes in a way that is suitable for the comparison step.

Finally, using the EMD or 1WD to measure the distance between two time series produces an absolute value that depends on the length of both time series. The longer the two time series are, the higher the value can be. Thus, this distance cannot be used to perform inter-process comparisons, i.e., assessing the quality difference between BPS models of different processes. We studied two strategies to normalize the EMD and 1WD distances into a value between 0 and 1 that enabled inter-process comparisons. First, we considered dividing the raw distance by the maximum distance possible. However, the range of the simulated event log affects this normalization. We observed that longer ranges (even caused by outliers) end in a greater denominator for the normalization, reducing the absolute error. In order to avoid this problem when the number of bins can be different, we also considered adjusting the number of bins (buckets) of each sample, so the range is always the same. However, this introduces another problem: a higher range in the simulated sample causes the bins to be longer. The comparison between the actual log and a simulated one could be matching bins of one hour with bins of ten minutes for one simulated log, and with bins of ten hours for another. For these reasons, and given that the distance is useful to perform intra-process comparisons, i.e., comparing two BPS models of the same process, we opted not to normalize the distance.

8.2. Threats to validity

The evaluation reported above is potentially affected by the following threats to the validity. First, regarding *internal validity*, the experiments rely on 29 BPS models of two synthetic processes, and 8 automatically discovered BPS models from 4 real-life processes. The results could be different for other datasets. Second, regarding *external validity*, the evaluation was assessed with real-life event logs from processes of different domains. However, the results could not be generalized for processes of domains presenting specific unseen characteristics. Third, regarding *construct validity*, we proposed a set of measures of goodness based on discretized distributions and time series. The results could be different for other measures.

Finally, regarding *ecological validity*, the evaluation compares the BPS models against the original log. While this allows us to measure how well the simulation models replicate the as-is process, it does not allow us to assess the goodness of the simulation models in a *what-if?* setting, e.g., predicting the performance of the process after an intervention.

9. Conclusion

We proposed a framework to measure the ability of a BPS model to replicate the behavior recorded in an event log. The framework relies on functions to filter, project, transform event logs, and a distance function, which collectively allow us to compare a log produced by a simulation run (a simulated log) against the log given as ground truth. The filter and project steps allow us to select the portion of the data in a log that is relevant for a given purpose (e.g. to measure the quality from a control-flow perspective). The transform step allows us to abstract the data in the log into a desired structure, such as a time series or histogram, capturing the relevant behavior. Finally, the distance function allows us to measure how far or close is the simulated log relative to the ground-truth log. The article defines several instantiations of the proposed framework, designed to capture four perspectives of a process: control-flow, temporal, resource, and congestion.

We evaluated the adequacy of the proposed measures with respect to two use cases. First, we evaluated the ability of the proposed measures to discern the impact of perturbations to a BPS model. This corresponds to a use case where the measures are used to calibrate a simulation model. The results show that the measures are able to detect the expected effects of perturbations affecting the above four perspectives of a process. In particular, we found that the N-Grams Distance (NGD) with an N=2 effectively captures the effect of perturbations along the control-flow perspective, achieving similar results as the Control-Flow Log Distance (CFLD) measure proposed in [4], while being much more computationally efficient. We also found that the Circadian Workforce Distance (CWD) is able to capture the effect of perturbations on the resources of the process, both perturbations on the availability calendars of resources, and perturbations on the number of available resources.

Furthermore, we used real-life logs to evaluate the adequacy of the metrics when it comes to comparing two techniques for automated discovery of BPS models. The findings show that, in addition to capturing the overall quality of a BPS model and pinpointing sources of discrepancies, the proposed measures can also assist in eliciting areas for improvement in these techniques.

At its core, the proposed framework allows us to compare two event logs with overlapping activities and resources. Accordingly, we foresee that the framework could be applied in other settings where event logs need to be compared. In future work, we will explore the applicability of the proposed measures to other process mining problems, e.g., *concept drift detection* [43], where the goal is to compare the behavior of a process during two timeframes, and *variant analysis* [44], which seeks to diagnose differences between event logs of multiple variants of the same process.

In this article, we focused on four perspectives of business processes: the control-flow, temporal, resource, and congestion perspectives. BPS models may also cover the data perspective, by incorporating variables (e.g., case variables), update operations (e.g., the completion of a task changing the value of a variable), and branching conditions based on these variables. Another direction for future work is to investigate measures of BPS model quality along the data perspective, for example, measures that could find out that a BPS model does not capture the differences in the behavior of a process equally across different countries, product types, or customer types.

Finally, the proposed framework focuses on assessing the quality of simulation models of individual business processes. Oftentimes, processes are connected, for example, because they share common data artifacts or objects [45]. Another avenue for future work is to study how to assess the quality of BPS models in the context of such collections of inter-related processes.

Reproducibility

The scripts to reproduce the experiments, the datasets, and the results are publicly available at Zenodo.[18] The measures have been implemented as a Python package (log-distance-measures) installable from pip, and the code is publicly available at GitHub.[19]

CRediT authorship contribution statement

David Chapela-Campa: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Data curation, Conceptualization. **Ismail Benchekroun:** Writing – review & editing, Investigation, Data curation. **Opher Baron:** Writing – review & editing, Supervision, Resources, Project administration, Funding acquisition, Conceptualization. **Marlon Dumas:** Writing – review & editing, Writing – original draft, Supervision, Resources, Project administration, Funding acquisition, Formal analysis, Conceptualization. **Dmitry Krass:** Writing – review & editing, Supervision, Resources, Project administration, Funding acquisition, Conceptualization. **Arik Senderovich:** Writing – review & editing, Writing – original draft, Supervision, Project administration, Methodology, Funding acquisition, Formal analysis, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

[18] https://doi.org/10.5281/zenodo.12126071

[19] https://github.com/AutomatedProcessImprovement/log-distance-measures

Data availability

The used data is publicly available through a DOI link in the article.

Acknowledgments

This work has been funded by the European Research Council (PIX Project) and the National Science and Engineering Research Council (NSERC) grants held by Opher Baron, Dmitry Krass, and Arik Senderovich.

References

[1] M. Dumas, M.L. Rosa, J. Mendling, H.A. Reijers, Fundamentals of Business Process Management, Second Ed., Springer, 2018.
[2] M.T. Wynn, M. Dumas, C.J. Fidge, A.H.M. ter Hofstede, W.M.P. van der Aalst, Business process simulation for operational decision support, in: Workshops of the 5th International Conference on Business Process Management, BPM Workshops 2007, in: LNCS, vol. 4928, Springer, 2007, pp. 66–77.
[3] K. Rosenthal, B. Ternes, S. Strecker, Business process simulation on procedural graphical process models, Bus. Inf. Syst. Eng. 63 (5) (2021) 569–602.
[4] M. Camargo, M. Dumas, O. González, Automated discovery of business process simulation models from event logs, Decis. Support Syst. 134 (2020) 113284.
[5] M. Camargo, M. Dumas, O.G. Rojas, Discovering generative models from event logs: Data-driven simulation vs deep learning, PeerJ Comput. Sci. 7 (2021) e577.
[6] N. Martin, B. Depaire, A. Caris, The use of process mining in business process simulation model construction - structuring the field, Bus. Inf. Syst. Eng. 58 (1) (2016) 73–87.
[7] A. Rozinat, R.S. Mans, M. Song, W.M.P. van der Aalst, Discovering simulation models, Inf. Syst. 34 (3) (2009) 305–327.
[8] C. Fracca, M. de Leoni, F. Asnicar, A. Turco, Estimating activity start timestamps in the presence of waiting times via process simulation, in: Proceedings of the 34th International Conference on Advanced Information Systems Engineering, CAiSE 2022, in: LNCS, vol. 13295, Springer, 2022, pp. 287–303.
[9] R.G. Sargent, Verification and validation of simulation models, in: Proceedings of the 2010 Winter Simulation Conference, WSC 2010, Baltimore, Maryland, USA, 5-8 December 2010, IEEE, 2010, pp. 166–183.
[10] D. Chapela-Campa, I. Benchekroun, O. Baron, M. Dumas, D. Krass, A. Senderovich, Can I trust my simulation model? Measuring the quality of business process simulation models, in: Proceedings of the 21st International Conference on Business Process Management, BPM 2023, in: LNCS, 14159, Springer, 2023, pp. 20–37.
[11] A. Augusto, R. Conforti, M. Dumas, M.L. Rosa, A. Polyvyanyy, Split miner: Automated discovery of accurate and simple business process models from event logs, Knowl. Inf. Syst. 59 (2) (2019) 251–284.
[12] D. Chapela-Campa, M. Dumas, Enhancing business process simulation models with extraneous activity delays, Inf. Syst. 122 (2024) 102346.
[13] A. Senderovich, J.C. Beck, A. Gal, M. Weidlich, Congestion graphs for automated time predictions, in: Proceedings of the 33rd AAAI Conference on Artificial Intelligence, AAAI 2019, AAAI Press, 2019, pp. 4854–4861.
[14] A. Senderovich, M. Weidlich, L. Yedidsion, A. Gal, A. Mandelbaum, S. Kadish, C.A. Bunnell, Conformance checking and performance improvement in scheduled processes: A queueing-network perspective, Inf. Syst. 62 (2016) 185–206.
[15] M. Pourbafrani, W.M.P. van der Aalst, Discovering system dynamics simulation models using process mining, IEEE Access 10 (2022) 78527–78547.
[16] E. Brophy, Z. Wang, Q. She, T. Ward, Generative adversarial networks in time series: A systematic literature review, ACM Comput. Surv. 55 (10) (2023) 199:1–199:31.
[17] J. Yoon, D. Jarrett, M. van der Schaar, Time-series generative adversarial networks, in: Proceedings of the 2019 Annual Conference on Advances in Neural Information Processing Systems 32, NeurIPS 2019, 2019, pp. 5509–5519.
[18] A. Burke, S.J.J. Leemans, M.T. Wynn, W.M.P. van der Aalst, A.H.M. ter Hofstede, Stochastic process model-log quality dimensions: An experimental study, in: Proceedings of the 4th International Conference on Process Mining, ICPM 2022, IEEE, 2022, pp. 80–87.
[19] J. Mendling, G. Neumann, W.M.P. van der Aalst, Understanding the occurrence of errors in process models based on metrics, in: On the Move To Meaningful Internet Systems 2007: CoopIS, DOA, ODBASE, GADA, and IS (OTM 2007), in: LNCS, vol. 4803, Springer, 2007, pp. 113–130.
[20] W.M.P. van der Aalst, Relating process models and event logs - 21 conformance propositions, in: Proceedings of the International Workshop on Algorithms & Theories for the Analysis of Event Data 2018, ATAED 2018, in: CEUR Workshop Proceedings, vol. 2115, CEUR-WS.org, 2018, pp. 56–74.
[21] S.J.J. Leemans, A. Polyvyanyy, Stochastic-aware conformance checking: An entropy-based approach, in: Proceedings of the 32nd International Conference on Advanced Information Systems Engineering, CAiSE 2020, in: LNCS, vol. 12127, Springer, 2020, pp. 217–233.

[22] S.J.J. Leemans, A.F. Syring, W.M.P. van der Aalst, Earth movers' stochastic conformance checking, in: Proceedings of the 4th Business Process Management Forum, BPM Forum 2019, in: LNBIP, vol. 360, Springer, 2019, pp. 127–143.
[23] G. Sierksma, Linear and integer programming - theory and practice, in: Pure and Applied Mathematics, vol. 198, Dekker, 1996.
[24] N. Martin, L. Pufahl, F. Mannhardt, Detection of batch activities from event logs, Inf. Syst. 95 (2021) 101642.
[25] T.W. Liao, Clustering of time series data - A survey, Pattern Recognit. 38 (11) (2005) 1857–1874.
[26] M. Muskulus, S. Verduyn-Lunel, Wasserstein distances in the analysis of time series and dynamical systems, Physica D 240 (1) (2011) 45–58.
[27] E. Levina, P.J. Bickel, The earth mover's distance is the mallows distance: Some insights from statistics, in: Proceedings of the 8th International Conference on Computer Vision (ICCV-01), vol. 2, IEEE Computer Society, 2001, pp. 251–256.
[28] M. Abel, Lightning Fast Business Process Simulator (Master's thesis), Institute of Computer Science, University of Tartu, 2011.
[29] O. López-Pintado, M. Dumas, Business process simulation with differentiated resources: Does it make a difference? in: Proceedings of the 20th International Conference on Business Process Management, BPM 2022, in: LNCS, vol. 13420, Springer, 2022, pp. 361–378.
[30] A. Rozinat, W.M.P. van der Aalst, Conformance testing: Measuring the fit and appropriateness of event logs and process models, in: C. Bussler, A. Haller (Eds.), in: Business Process Management Workshops, BPM 2005 International Workshops, BPI, BPD, ENEI, BPRM, WSCOBPM, BPS, Nancy, France, September 5, 2005, Revised Selected Papers, vol. 3812, 2005, pp. 163–176.
[31] N. Tax, X. Lu, N. Sidorova, D. Fahland, W.M.P. van der Aalst, The imprecisions of precision measures in process mining, Inform. Process. Lett. 135 (2018) 1–8.
[32] A. Augusto, A. Armas-Cervantes, R. Conforti, M. Dumas, M.L. Rosa, Measuring fitness and precision of automatically discovered process models: A principled and scalable approach, IEEE Trans. Knowl. Data Eng. 34 (4) (2022) 1870–1888.
[33] C. Zhao, S. Sahni, String correction using the Damerau-Levenshtein distance, BMC Bioinform. 20-S (11) (2019) 277:1–277:28.
[34] A. Jalali, Graph-based process mining, in: Workshops of the 2nd International Conference on Process Mining, ICPM 2020, in: LNBIP, vol. 406, Springer, 2020, pp. 273–285.
[35] P.J. Brockwell, R.A. Davis, Introduction to Time Series and Forecasting, Springer, 2002.
[36] J.D. Kalbfleisch, R.L. Prentice, The Statistical Analysis of Failure Time Data, John Wiley & Sons, 2011.
[37] G. Berkenstadt, A. Gal, A. Senderovich, R. Shraga, M. Weidlich, Queueing inference for process performance analysis with missing life-cycle data, in: Proceedings of the 2nd International Conference on Process Mining, ICPM 2020, IEEE, 2020, pp. 57–64.
[38] N. Martin, B. Depaire, A. Caris, Using event logs to model interarrival times in business process simulation, in: Workshops of the 13th International Conference on Business Process Management, BPM Workshops 2015, in: LNBIP, vol. 256, Springer, 2015, pp. 255–267.
[39] M.U. Thomas, Queueing systems. volume 1: Theory (leonard kleinrock), SIAM Rev. 18 (3) (1976) 512–514.
[40] B. van Dongen, BPI challenge 2012, 2012, http://dx.doi.org/10.4121/uuid:3926db30-f712-4394-aebc-75976070e91f, URL https://data.4tu.nl/articles/dataset/BPI_Challenge_2012/12689204/1.
[41] B. van Dongen, BPI challenge 2017, 2017, http://dx.doi.org/10.4121/uuid:5f3067df-f10b-45da-b98b-86ae4c7a310b, URL https://data.4tu.nl/articles/dataset/BPI_Challenge_2017/12696884/1.
[42] A. Wombacher, M. Iacob, Start time and duration distribution estimation in semi-structured processes, in: Proceedings of the 28th Annual ACM Symposium on Applied Computing, SAC 2013, ACM, 2013, pp. 1403–1409.
[43] D.M.V. Sato, S.C.D. Freitas, J.P. Barddal, E.E. Scalabrin, A survey on concept drift in process mining, ACM Comput. Surv. 54 (9) (2022) 189:1–189:38.
[44] F. Taymouri, M.L. Rosa, M. Dumas, F.M. Maggi, Business process variant analysis: Survey and classification, Knowl.-Based Syst. 211 (2021) 106557.
[45] R. Hull, Artifact-centric business process models: Brief survey of research results and challenges, in: Proceedings of the OTM Conferences 2008 (CoopIS, DOA, GADA, IS, and ODBASE), in: LNCS, vol. 5332, Springer, 2008, pp. 1152–1163.