Process-Aware Bayesian Networks for Sequential Event Log Queries

Simon Rauch[*†‡], Christian M. M. Frey[†], Ludwig Zellner[*] and Thomas Seidl[*†‡]
[*]Database Systems and Data Mining, LMU Munich, Munich, Germany
Email: {rauch, zellner, seidl}@dbs.ifi.lmu.de
[‡]Munich Center for Machine Learning, Munich, Germany
[†]Center for Applied Research on Supply Chain Services, Fraunhofer IIS, Nuremberg, Germany
Email: christianmaxmike@gmail.com

2024 6th International Conference on Process Mining (ICPM) | 979-8-3503-6503-0/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICPM63005.2024.10680678

***Abstract*—Business processes from many domains like manufacturing, healthcare, or business administration suffer from different amounts of uncertainty concerning the execution of individual activities and their order of occurrence. As long as a process is not entirely serial, i.e., there are no forks or decisions to be made along the process execution, we are - in the absence of exhaustive domain knowledge - confronted with the question whether and in what order activities should be executed or left out for a given case and a desired outcome. As the occurrence or non-occurrence of events has substantial implications regarding process key performance indicators like throughput times or scrap rate, there is ample need for assessing and modeling that process-inherent uncertainty. We propose a novel way of handling the uncertainty by leveraging the probabilistic mechanisms of Bayesian Networks to model processes from the structural and temporal information given in event log data and offer a comprehensive evaluation of uncertainty by modelling cases in their entirety. In a thorough analysis of well-established benchmark datasets, we show that our Process-aware Bayesian Network is capable of answering process queries concerned with any unknown process sequence regarding activities and/or attributes enhancing the explainability of processes. Our method can infer execution probabilities of activities at different stages and can query probabilities of certain process outcomes. The key benefit of the Process-aware Query System over existing approaches is the ability to deliver probabilistic, case-diagnostic information about the execution of activities via Bayesian inference.**

***Index Terms*—Process Query Systems, Bayesian Networks, Predictive Process Monitoring**

I. Introduction

To assess future realizations of process key performance indicators (KPIs) like the execution time of a process, its successful completion, or the quality of a produced good, the actual path a process takes is a crucial bit of information. A process instance leaving some activities untouched can drastically affect its duration if the skipped activities are time-intensive or influence a successful process completion, e.g., if important process steps like inspections of workpieces or the delivery of essential notifications are left out. Moreover, the execution order and execution frequency of activities or partial traces can also have implications on the likelihood of a case being faulty. This highlights the need for a far-sighted handling of uncertainty in highly variable processes to be able to assess future realizations of different process KPIs either qualitatively or by applying additional data-driven methods.

In this work, we use Bayesian Networks in a diagnostic way for data-based queries of best practices and process improvements in terms of control-flow for reaching desired outcomes. For that, we propose Process-aware Bayesian Networks (PBNs) providing capabilities to answer process queries regarding (sub-)trace and outcome predictions in a single framework. The properties of Bayesian Networks as a tool for graphical probabilistic modeling reflect both, the omnipresent structural component of business processes and the ability to model the uncertainty that is apparent in real-world processes. Most real-world processes, e.g., from the manufacturing industry, involve ambiguous order, concurrency, and loops of activities. To define valid Bayesian Networks fulfilling the properties of directed acyclic graphs (DAGs), we annotate the activities with their position number within a trace of an event log and represent activity loops as a serial sequence. The parameter learning on the sequentially designed structure of the Bayesian Networks serves as an alternative to modeling activity transitions with traditional techniques, e.g., mining algorithms or simple (Directly-)Follows Graphs. The novelty of our approach allows for performing inference on manipulated case scenarios, e.g., a model query can tell us how and in which order the beginning of a process would have to be carried out to reach a particular desired activity, subtrace or outcome at a later position of the case. A thorough evaluation of various process queries justifies the proposed Bayesian inference model on different real-world event logs going beyond the structural limitations being imposed by traditional and state-of-the-art process models.

In summary, our contributions to the field of Process Query Systems are the following:

- Translation of event information into a PBN for sequential probabilistic inference
- A model that allows for querying process information systems with regard to uncertain control-flow or attribute information
- A framework for subsequently deriving best practices in process executions such as most probable prefix paths or most influential activities for specific outcomes

This work was supported by the Bavarian Ministry for Economic Affairs, Infrastructure, Transport and Technology through the Center for Analytics Data Applications (ADA-Center) within framework "BAYERN DIGITAL II".

II. Related Work

We put our approach into context by analyzing related work from the field of Bayesian Networks along with their advantages as well as neural-network-based approaches for predictive and prescriptive process monitoring. As this area comprises the usage of prediction models, we start by covering related work from the area of Next Activity Prediction. We then proceed with articles from the more general field of trace inference for process treatment.

In the past years, various approaches have been developed for predicting next activities and remaining traces. Pauwels & Calders [1] are using Dynamic Bayesian Networks (DBNs) to predict next activities with high accuracy. However, predicting remaining traces remains a tough task for their DBN and they are not able to perform full-case probabilistic queries. Moreover, Prasidis et al. [2] handle uncertainty in event logs by combining a DAG derived from a process model with conditional probability tables directly learned from log data.

Lu et al. [3] follow a similar approach and try to detect abnormal behavior by probabilistically inferring it from a Petri Net-Based Bayesian Network. However, neither of the Bayesian Network approaches is suitable for complete trace queries as our sequential network structure allows for.

Savickas & Vasilecas [4] focus on the extraction of probabilistic models such as Bayesian belief networks to facilitate the decision-making process, offering a system similar to ours for querying. The authors create these networks by transforming mined business processes and comparing their results to the output of different mining algorithms. Their approach only works semi-automated, so an improvement on that end is necessary. Additionally, this idea cannot produce an ordered result of multiple events.

Lin et al. [5] address Next Activity Prediction and additionally achieve the prediction of corresponding attributes by adding them as input in a recurrent neural network (RNN). Tax et al. [6], Camargo et al. [7], and Gunnarsson et al. [8] accomplish remaining trace prediction using LSTM neural networks which are a popular tool for sequential pattern analysis. As the architectures of these approaches show, Remaining Trace Prediction is only made possible by successive predictions of next activities. In contrast, the sequential structure of our approach resembling whole traces is capable of inferring the complete remaining traces at once. Our approach adopts a declarative style, where we articulate outcomes as queries and identify the most probable paths that align with those queries. Therefore, we do not prioritize predicting the next activities or trace suffixes but carry out predictions of whole remaining traces as a precondition for meaningful query results.

A closely related area addresses the problem of trace inference. Sutrisnowati et al. [9] first introduce the idea of deriving a Bayesian Network from an event log tailored to process analysis and inference about lateness probabilities of specific activities in port logistics. In contrast, we address Process-aware Bayesian Network queries for business processes in a more general manner, offering a broader application and allowing for more versatile and comprehensive analysis. Additionally, the time-based information in the event logs is not utilized, making it difficult to infer sequential data. Moreira et al. [10] made a significant contribution by addressing missing data. They translate event logs into both classical and quantum-like Bayesian Networks and prune the network structure to eliminate occurring activity loops. However, their model only yields a binary outcome for occurrences of activities. While the capability to predict (non-)occurrences is beneficial, it does not address the specific completion of traces nor does it consider sequential trace information. Bozorgi et al. [11] actively contribute to the field of causal inference for process treatments, focusing on case-level treatment recommendations to improve the rate of positive outcomes. Subsequently [12], they identify the optimal starting point for treatments. Here, a *treatment* refers to specific actions that modify the process, leading to more desirable outcomes. Unlike our approach, which emphasizes deriving process paths that most likely lead to a certain outcome, the work of Bozorgi et al. [11], [12] centers on optimizing processes through modification.

Weinzierl et al. [13] and De Leoni et al. [14] address a similar task: recommending activities to maximize a specific KPI using distinct approaches. This resembles a constrained perspective on querying the system as it presupposes the existence of a KPI to be optimized. As opposed to this, our approach offers the flexibility to define queries for results freely. These queries can encompass typical information, such as the most probable activity or attribute outcome for a given prefix, as well as possible intermediate process executions.

As previously mentioned, our primary focus is on performing predictive and diagnostic event log queries. By concentrating on this aspect, we aim to provide comprehensive and reliable information to support decision-making processes and enhance overall system apprehension.

III. Preliminaries and Problem Formulation

We provide preliminary definitions necessary for our framework and formulate the querying tasks we are focusing on.

A. Process Mining

In Process Mining a sequence of *Events* describes a *Trace*. A collection of traces is referred to as an *Event Log*.

Definition 1 (Event). An event is defined as a tuple $\epsilon = (c, a, t, h_1, \ldots, h_n)$ describing the case-id c out of a set of case identifiers $\mathcal{I}$, the executed activity a from an activity set $\mathcal{A}$, the recorded timestamp t from a set of timestamps $\mathcal{T}$ as well as additional case or event attributes $h_1, \ldots, h_n$ from a set of attributes $\mathcal{H}$.

Definition 2 (Trace). A trace T is an ordered, finite sequence of events $\langle \epsilon_1, \ldots, \epsilon_{|T|} \rangle$ that belong to the same process instance, i.e., case-id. Let $\#_{act}(T) = \langle a_1, \ldots, a_{|T|} \rangle$ denote the projection of a trace T to its sequence of activities, and $\#_{att}(T)_j = \langle h_{j1}, \ldots, h_{j|T|} \rangle$ refer to the sequence of the j-th attribute of trace T. We define the *place* of an activity and the realizations of different event or case attributes as $p(a_i) = i$ and $p(h_{ji}) = i$, respectively.

Definition 3 (Event Log). An Event Log L is a set of traces $\{T_1, \ldots, T_{|L|}\}$ that are recorded during the execution of a given business process. We refer to unique sequences of activities inside an event log as *variants*.

Definition 4 (Prefix/Suffix/Outcome). A prefix ρ of an event $\epsilon_k \in T$ is a sub-sequence of T with events $\langle \epsilon_1, \ldots, \epsilon_{k-1} \rangle$ that occur prior to ϵ_k. A suffix ϕ of an event $\epsilon_k \in T$ is a sub-sequence of T with events $\langle \epsilon_{k+1}, \ldots, \epsilon_{|T|} \rangle$ that occur after ϵ_k. An outcome o is the evaluation of event $\epsilon_{|T|}$.

B. Problem Formulation

Problem Statement. *Process Querying describes the task of inferring about any unknown events* $\epsilon_i^{(u)} \in T$ *of a process instance given an empty or non-empty set of evidence* E *of known events* $\epsilon_j^{(k)} \in T$ *about a trace* T *with* $i + j = |T|$*. The evidence, i.e., the process information specified as available, as well as the respective unknown information, can be concerned with the control-flow perspective of the process event log, additional case or event attributes or case outcomes.*

We tackle the problem with PBNs for probabilistic and sequential Bayesian inference regarding unknown activities and attributes of a trace T conditional on evidence E consisting of known activity and attribute information. We can use the method to bi-directionally infer the state of any unknown activity or attribute occurring at different positions of T.

IV. Methodology

This section introduces our mapping procedure of an event log to the nodes of a Bayesian Network for activity inference. An implementation is available online[1]. Section IV-A introduces the structure of a *Process-aware Bayesian Network* and Section IV-B elaborates on the learning, inference, and querying mechanisms. The proposed framework is visualized in Fig. 1 and described in the following sections.

A. Process-aware Bayesian Network

We introduce our method with a mapping of activities and attributes of an event log to nodes of a Bayesian Network relying on the definition of directed acyclic graphs:

Definition 5 (Directed Acyclic Graph). A directed acyclic graph is defined as a pair $G = (\mathcal{N}, \mathcal{R})$ of *nodes* $\mathcal{N}$ and *relations* $\mathcal{R}$. Relations $\mathcal{R} \subseteq \mathcal{N} \times \mathcal{N}$ are defined as a set of (ordered) tuples. A directed graph is called acyclic iff there are no directed cycles, i.e., no sequence of nodes $v_1, v_2, \ldots, v_n$ such that $v_1 = v_n$ and $\forall i \in [1, n-1] : (v_i, v_{i+1}) \in \mathcal{R}$.

Nodes in Bayesian Networks represent random variables, edges represent probabilistic dependencies between them:

[1]https://gitlab.cc-asp.fraunhofer.de/iis-scs-a-publications/pbn4seq

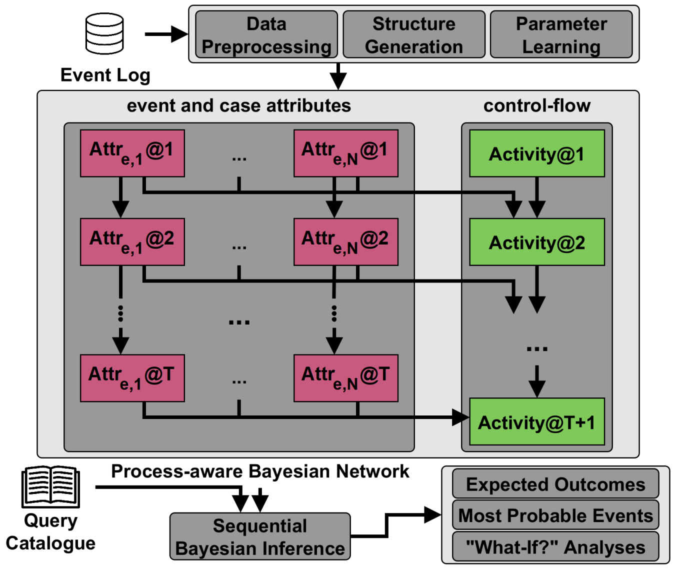


Fig. 1: Our framework for Bayesian Sequential Inference with a generic Process-aware Bayesian Network including control-flow nodes (green), time-variant event attributes and time-invariant case attributes (red). The Query Catalogue for predictive and diagnostic querying is discussed in Section V.

Definition 6 (Parent/Child). Considering two nodes $u, v \in \mathcal{N}$ connected via an edge $(u, v) \in \mathcal{R}$, node u is called a *parent node* of v and node v is a *child node* of node u. The set of parent nodes of a given node x is defined as $pa(x) := \{u | (u, x) \in \mathcal{R}\}$, the set of child nodes of node x as $ch(x) := \{v | (x, v) \in \mathcal{R}\}$. A node with in-degree zero is called a *root*, and a node with out-degree zero is called a *leaf*.

Our (discrete) PBN models the occurring activities $\mathcal{A}$ in L and their sequential transitions:

Definition 7 (Process-aware Bayesian Network) Given an Event Log L, a (discrete) Process-aware Bayesian Network B for probabilistic trace analysis is a triple $B = (\mathcal{X}, G, \mathcal{P})$ where a DAG $G = (\mathcal{N}, \mathcal{R})$ defines the dependencies between random variables $\mathcal{X}$, i.e., each node $v \in \mathcal{N}$ corresponds one-to-one with a discrete random variable $X_v \in \mathcal{X}$ and $\mathcal{P}$ denotes a set of conditional probability distributions. The depth of G, resp., B is given by $|T|_{max} + 1$ denoting the maximal trace length in L plus an additional terminal node.

- **(Structure)** We define two disjoint sets $\mathcal{X}_a \cup \mathcal{X}_h = \mathcal{X}$ denoting the random variables for activities and (temporal) attributes. We use $X_a^{(r)}$ and $X_h^{(r)}$ as a short-hand notation to refer to a node occurring at a specific level r in G that refers to the place of the modeled trace where $r = 1$ corresponds to the first place. For binding the activity of an event ϵ with its respective attributes, it holds that $pa(X_a^{(r)}) = \{X_{h_j}^{(r-1)} | h_j \in \epsilon\} \cup \{X_a^{(r-1)}\}$ and

$pa(X_{h_j}^{(r)}) = \{X_{h_j}^{(r-1)} | h_j \in \epsilon\}$, where $pa(X^{(1)}) = \emptyset$.

- **(Parameters)** The conditional probability distributions $P(X_a|pa(X_a)), P(X_h|pa(X_h)) \in \mathcal{P}, \quad \forall X_a, X_h \in \mathcal{X}$ describe the conditional transition probabilities between the network nodes for their case-specific configurations with respect to the occurring activities and attributes.

The network structure contains connected nodes for all possible places inside a process. For recording that a case has ended, we pad all traces with an additional activity end until we reach $|T|_{max} + 1$ with $|T|_{max}$ being the maximal trace length of an event log. The network structure concerned with the control-flow information of the log carries a total of $(|T|_{max}+1)$ nodes. For each attribute or additional feature, we add an additional $(|T|_{max} + 1)$ nodes to account for possible changes over time in the respective attribute or feature. Hence, the total number of nodes inside the resulting PBN is bounded by $(|T|_{max} + 1) \cdot (1 + |X_h|))$. We map each network node, i.e., random process variable to its respective place within a trace. Due to the place-annotation of the trace information and, especially, the activities, we not only can predict the occurrence or non-occurrence but also the most probable position of all remaining activities (cf. Problem Statement).

For better comprehensibility of the structural design of the PBN, we provide a visualization of a generic network structure in Figure 1. Given an input event log, we preprocess the data as explained in Section IV-B. The PBN models the control flow on the one hand (right) and attributes (left) on the other. For capturing the control-flow information of the process, we introduce serially connected, place-specific nodes carrying the activity occurrence information $X_a^{(r)}$ at a certain place r inside a trace (*control-flow*). Due to a drastic increase in model complexity when generally including higher-order dependencies between the nodes, we leave edges that extend beyond a distance of one as potential future work.

As the recorded event and case attributes in the event log can also contribute to the predictive power of the model – either in terms of the future control-flow or the overall outcome – we add the respective nodes to the parent node set of the succeeding control-flow node (*event* and *case attributes*).

We honor the temporal dimension by adding well-established temporal features from the field of Predictive Process Monitoring [6], [7], [12] to the set of integrated attributes: the time since the last event (*TSLE*), since the start of the case (*TSCS*) and since midnight (*TSMN*) to account for the current time of day. The trained PBN serves as a Process Query System for diverse demands which are further evaluated in Section V-B.

B. Parameter Learning

To perform parameter learning of the generated PBN structure, the event log data needs to be preprocessed.

Event Log Preprocessing. When matching each activity and its attributes with their position within a trace, the columns of our training data need to resemble that structure and necessarily contain the relevant occurrence and attribute information. See Tables Ia, Ib for an exemplary preprocessing of trace data in a standardized form where we arrange the information about the occurrence and order of events (*act, place*) as multicategorical input values. The control-flow information is extended by a global ending activity *end* marking the end of a trace. All additional attributes record a global attribute *NO* (short for *no occurrence*). We discretize any additional attribute information used as parent input for the place-wise activity nodes to remain in a fully discrete setting and not shift into the realms of Conditional Linear Gaussian Bayesian Networks in which continuous nodes as parents for discrete nodes are not allowed [15]. After all preprocessing steps are finished, the Bayesian model can learn conditional transition probabilities between activities occurring at different places with respect to case and event attributes over time.

Parameter Learning. The conditional probability tables (CPTs) of the PBN are estimated via maximizing the network likelihood function l:

$$l(\mathcal{P}, L) = \prod_{k=1}^{|L|} \prod_{i=1}^{|\mathcal{X}|} P(X_i = x_i^{(k)} | pa(X_i) = x_{pa(X_i)}^{(k)}, T^{(k)}),$$

where $x_i^{(k)}$ and $x_{pa(X_i)}^{(k)}$ are the realizations of the respective random variables for the k-th trace $T^{(k)} \in L$. See Table Ic for a learned CPT by the PBN illustrated in Fig. 2 from the running example.

C. Bayesian Inference

We can use the network parameters, i.e., CPTs, to draw samples from the PBN regarding any random variable or set of random variables conditional on informational evidence E about the process, i.e., the realizations of a set of random variables $X_E \subseteq \mathcal{X}$. Under usage of Bayesian inference, the network samples can be used to calculate the probabilities of events, i.e., realizations of a set of query nodes $X_Q \subseteq \mathcal{X}/\{X_E\}$, conditional on the specified set of evidence:

$$P((X_{Q,1} = x_{q,1}) \cap \ldots \cap (X_{Q,|X_Q|} = x_{Q,|X_Q|})|E),$$

with evidence E – similar to the formulation of the query event – being specified as a probabilistic expression

$$(X_{E,1} = x_{E,1}) \cap \ldots \cap (X_{E,|X_E|} = x_{E,|X_E|}).$$

The queries can either be used for predictive actions, i.e., for a running process instance, or for case diagnostics, i.e., by analyzing completed or even counterfactual instances with respect to optimization potential.

Activity Queries. The sequential network structure allows us to use information about the beginning (end) of a trace to query the network for probabilities at different places for all remaining (preceding) activities. The number of possible states of each discrete variable $X_a^{(i)} \in \mathcal{X}_a$ of B (cf. Def. 7) equals the number of unique activities within L, i.e., $|X_a| = |\mathcal{A}|$ and an instantiation refers to a specific activity $x_a = a$ with $a \in \mathcal{A}$. For inference regarding a given activity at a position r we query the conditional probability for the control-flow variable corresponding to that place r given by:

$$P_{X_a^{(r)}}(X_a^{(r)} = a|E)$$

TABLE I: Preprocessing of an example trace for parameter learning

ID	act	$timestamp$	res	$place$
1	B	02:20:15	Betty	1
1	A	02:27:36	Betty	2
1	B	02:30:39	Betty	3
2	B	02:35:25	Betty	1
2	B	02:37:37	Alex	2
...	...	...	...	
8	B	03:37:29	Betty	2
8	A	03:43:11	Alex	3

(a) Original example log data

ID	act_1	act_2	act_3	act_4	res_1	res_2	res_3	res_4
1	B	A	B	end	Betty	Betty	Betty	NO
2	B	B	end	end	Betty	Alex	NO	NO
3	A	B	end	end	Betty	Betty	NO	NO
4	A	A	end	end	Betty	Betty	NO	NO
5	B	B	end	end	Alex	Alex	NO	NO
6	A	B	B	end	Alex	Betty	Betty	NO
7	A	A	end	end	Alex	Alex	NO	NO
8	A	B	A	end	Alex	Betty	Alex	NO

(b) Preprocessed log data

		$P(act_2 \mid act_1, res_1)$		
act_1	res_1	A	B	end
A	Alex	0.33	0.66	0
A	Betty	0.50	0.50	0
A	NO	–	–	–
B	Alex	0	1	0
B	Betty	0.50	0.50	0
B	NO	–	–	–
end	Alex	–	–	–
end	Betty	–	–	–
end	NO	–	–	–

(c) Conditional probability table (CPT) from example log data. Dashes (–) mark activity-attribute combinations that are not present in the data. Probabilities for those cases are undefined.

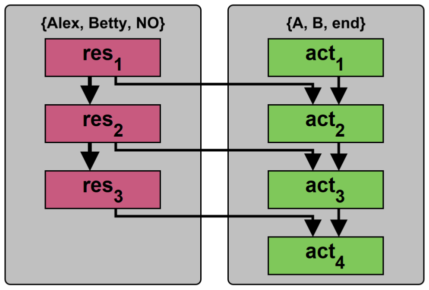


Fig. 2: Network structure from example log data. Possible node values are specified above.

for a set of evidence E containing the prefix (suffix) information available when the query is performed. The predicted activity for place r, i.e., the realization of $X_a^{(r)}$, is then:

$$a_{pred}^{(r)} = argmax_{a \in \mathcal{A}} P_{X_a^{(r)}}(X_a^{(r)} = a|E).$$

For queries about the remaining trace, with a given evidence set of occurrence information up to a given place $l < |T|$, we conduct queries regarding random variables $X_a^{(r)}$ for all places $r \in \{(l+1), \ldots, |T|\}$ for most probable activities.

Outcome Queries. To enable queries for outcomes, the type of outcomes $o_i \in \mathcal{O}$, i.e., the respective attribute, and the different discrete outcome classes $\{o_1, \ldots, o_{|\mathcal{O}|}\}$ have to be provided to the PBN in terms of the training data used for learning the network parameters. We extend the activity space by appending the outcome classes as ending activities:

$$\mathcal{A}_{outcome} := \mathcal{A} \cup \{o_1, \ldots, o_{|\mathcal{O}|}\}.$$

By that, we are able to formulate our outcome queries in the same way as the activity queries described before and are able to infer those outcomes with the proposed network structure in Figure 1. We determine the most probable outcome given a set of evidence E with prefix information via an activity query with respect to $\mathcal{A}_{outcome}$ for the node corresponding to the last possible ending activity $X_a^{(|T|+1)}$. Suppose we want to query possible process instances for temporal outcomes in terms of an overall case duration of, e.g., higher or lower than 24 hours. The respective outcome activities for queries added to $\mathcal{A}$ could be *end_below_24hours* and *end_above_24hours*. The trained model then can infer overall case duration classes via querying for the respective ending activity.

V. Evaluation

In the following, we provide an overview of *'What kind of process queries can be answered by applying Process-aware Bayesian Networks?'* in Section V-A. In Section V-B, we evaluate the predictive capabilities of our approach on sequential activity and outcome queries, and conclude with a discussion of the current limitations in Section V-C.

A. Process Queries with PBNs

As a backbone to the following evaluation for the Bayesian sequential process queries, we offer a sample of possible questions and queries that can be analyzed in Table II. Depending on the process and the goals of the process owners, those queries can be more specific or cover other attribute or outcome dimensions. We analyze these questions to shape the potential of a Bayesian sequential query system incorporating the sequential information of a process.

TABLE II: Catalogue of selected PBN queries

	Query	Category
S1	What is the most probable trace length given prefix ρ?	suffix
P1	For a given suffix ϕ, what is the most probable path leading to ϕ?	prefix
P2	For a given prefix ρ and a desired outcome o, what is the most probable intermediate process path?	prefix/ suffix
O1	What is the most probable outcome o given prefix ρ?	outcome
O2	Which activity is the most influential for a set of outcomes $o_i \in \mathcal{O}$?	outcome
...	...	...

The possible outcomes specified in Table II are dependent on the process and business domain at hand as well. They can range from a more generic outcome, such as the overall process duration, to domain-specific attributes, such as the quality of a processed good in production processes, or the conversion or retention status of customer leads in customer relationship management. The querying procedure is performed by specifying the desired trace elements as evidence, i.e., fixed information. All remaining, uncertain nodes and their realizations can then be queried conditional on that evidence.

B. Capabilities of process outcome and activity queries

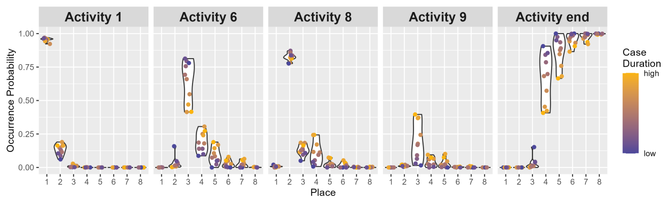
We split the evaluation into two parts. First, we perform specific queries from the query catalogue and showcase the process insights practitioners can gather from applying the PBN to their process data. We follow up with an assessment of the model performance for activity and outcome prediction.

Datasets. We evaluate our approach in terms of potentials and performance on the Helpdesk (HD)[2] event log [16], different variations of the BPI'12 event log [17] used throughout the literature on Predictive Process Monitoring, the BPI'20 event log for the travel permits process (TP) [18] and the Road Traffic Fines (RF) [19] log. Regarding BPI'12, we use the event log as a whole with all events (BPI'12 - Full), the whole event log excluding self-loops of length 1 (BPI'12 - Sub; corresponding to analyses in [8]) and the workflow event log (W) with *COMPLETE* events (BPI'12 - WC).

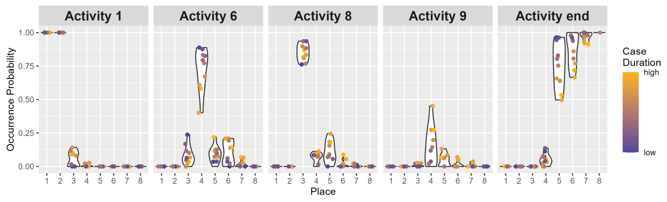
Implementation Details. For parameter learning and Bayesian inference, we make use of the R-package BNLEARN [20]. The underlying network structures are generated as described in Section IV. The CPTs are learned by employing Maximum Likelihood Estimation. Bayesian inference is carried out via Likelihood weighting - a form of particle-based approximate inference [21].

Model Setting. To showcase the sequential queries (cf. PBN Process Queries), we train a PBN using all available traces as we do not evaluate the queried process paths with a fixed set of testing instances and specify probabilistic statements derived from the specific queries process owners could ask regarding their business process. For performance assessment of the model in terms of predictive power (cf. Predictive Performance), we randomly pick 80% of the complete traces to generate the structural model and for parameter learning. We test on the remaining 20% of traces. As the time-features $TSLE$ and $TSCS$ are typically skewed towards zero, we perform a log-transformation of those features. We discretize each continuous attribute by applying quantile binning.

PBN Process Queries. For the queries in Table II, we analyzed temporal outcomes in terms of overall case duration in context of the respective most probable process paths. Figure 3a shows the query results for the Helpdesk event log (Query P1/O2). We used 10 different outcome classes translating to the deciles of the recorded overall durations. For each outcome class (purple=*low duration*, yellow=*high duration*), we can see the queried most probable paths our approach yields so that the outcome is achieved with the highest probability. The spread of occurrence probabilities for the different activities at certain positions is an indicator that the (non-)occurrence of those activities at the respective places has a high influence on the recorded outcome. For instance, *Activity 6* exhibits notably low occurrence probabilities as the third element of a trace for outcomes associated with longer overall durations. Conversely, it demonstrates a preference for executing the activity in the third position for outcomes with shorter overall durations. This is also in line with the fact that the probability of finishing the process instance increases drastically between the third and fourth executed activity of a trace, as can be seen from the difference in the occurrence probability of *Activity end* between the third and fourth position.



(a) Query with ten outcome classes (Query P1/O2).



(b) Query with ten outcome classes and given prefix information $\rho = \langle Activity\ 1, Activity\ 1\rangle$ (Query P2).

Fig. 3: Temporal outcome queries for the Helpdesk log with ten different outcome classes (color-coded). Each dot represents the queried conditional probability of occurrence for an activity with respect to a given class. Rarely occurring activities were excluded from the graph for better visibility.

Switching from the ex-ante point of view to the analysis of running processes, we can query the most probable intermediate process path for a given prefix ρ and a desired outcome (Query P2). We select an arbitrary prefix $\rho = \langle Activity1, Activity1\rangle$ and carry on with querying for all possible outcomes. In a real-world application of the approach, this procedure corresponds to specifying the as-is execution of a running process as the evidence for a probabilistic query with the PBN. In a direct comparison between Figure 3a and Figure 3b, we first notice an overall shift in the occurrence probabilities of activities at the respective places. The most critical decision based on the prefix is now whether *Activity 6* or *Activity 9* is executed as the fourth activity in the trace. The choice for the third activity seems to be rather unanimous and not deeply connected to shifts in the temporal outcome, as we can recognize a high occurrence probability queried

[2]We use the anonymized version of the Helpdesk event log with only control-flow and timestamp information (no other additional attributes) for evaluating our approach as well as benchmark approaches.

for *Activity 8* at the third place with a rather small spread for the different outcomes. Using such a probabilistic query, practitioners can greatly benefit from the PBN's sequential modeling capabilities to be able to control future process execution with certain goals, i.e., a desired outcome, in mind. For both queries we can also infer the most probable trace length (Query S1 combined with given prefix and/or outcome). Figure 3a points to a more probable case termination after three events, whereas the prefix $\rho = \langle Activity\,1, Activity\,1\rangle$ in Figure 3b leads to a case of at least four events.

Predictive Performance. For activity and outcome predictions we split all traces T_i in the test set into prefixes ρ_j with $j = \{0, \ldots, |T_i|\}$ and perform the inference task for all unknown nodes inside the network with the prefix information as evidence E. Activity and duration outcome predictions are evaluated in terms of classification accuracy and remaining trace predictions in terms of the already established Damerau-Levenshtein-Similarity [1], [5]–[8], [22], [23]. We report the results for the best-performing parameter combinations for our approach per dataset in Table III. Competing approaches were executed with the parameters recommended in the respective publications. As a direct competitor to our approach, we identified the Dynamic Bayesian Network (DBN) from Pauwels & Calders [1] and also report performance metrics of a prominent deep learning approach [7]. The reproduced benchmark models were executed with the same split of training and test data without filtering of traces.

In comparison with the DBN model from Pauwels & Calders, we recognize that our model is outperformed on the analyzed event logs in terms of Next Activity Prediction. Due to its place-agnostic structure, the DBN approach can handle smaller neighborhood transitions better. However, we record a much better overall performance for Remaining Trace Prediction with our approach as can be achieved with the DBN approach. Also, for the less complex logs (Helpdesk, BPI'2012 - W, RF), we achieve higher similarity values than the considered deep learning approach. We attribute those improvements to the mapping of complete traces into the sequential network structure. Conversely, the holistic sequential network structure appears to be the reason for the poorer performance in predicting directly subsequent activities. There, the direct connection between activities and their respective places inside the model seems to be prone to false classifications when we analyze logs with longer traces, various activities, and high fluctuations between those activities and their respective positions (BPI'12 variations, TP). Although other approaches follow a similar trend for more complex logs, we put those results into perspective and highlight a key difference of the PBN: Other approaches forecast activities a_{t+1} for each prefix and use greedily chosen predictions to extrapolate a remaining trace for $t+2, ..., |T_i|$. As we make use of Bayesian belief propagation for the prediction, we always generate fully predicted traces carrying forth all probabilistic uncertainty and all possible paths until the end of the trace instead of mapping predictive uncertainty, e.g., two activities with a predicted occurrence probability of ≈50% each, into completely certain control-flow information as input for succeeding predictions. We want to refrain from iterative predictions because this would oppose the general mechanisms of the approach for complete trace querying at each inference step. Due to this mechanism we can transform our approach into the Process Query System we showcased in the preceding section.

The two-class predictions of total case durations in Figure 4 show high accuracies for the BPI'12 - Full and RF logs. Accuracies for the other evaluated logs (HD, TP) are on a lower level. However, we record a quite steady performance across all datasets even when only low amounts of evidence are available (≈ 20–40%). We attribute this to the sequential network structure which allows the propagation of conditional probabilities through the whole network. In turn, we attribute the deteriorating accuracy for almost full evidence information to the same aspects as the lower accuracy when predicting directly subsequent activities.

C. Limitations and future work

Generally, for logs with numerous possible activities, lengthy traces, and a large number of potential variants, the number of nodes and parameters in the network rapidly increases, as discussed in Section IV. Moreover, an overall higher amount of possible transitions increases the complexity of the model during the training phase and makes a distinct decision regarding occurrence or non-occurrence between many activities at a given place harder as their occurrence

TABLE III: Performance comparison for Next Activity Prediction (NAP - Accuracy) and Remaining Trace Prediction (RTP - Damerau Levenshtein Similarity) . The best and second best approaches are in bold and underlined, respectively.

		BPI'12					
	HD	Full	Sub	WC	TP	RF	
Camargo et al. [7]	.686	.634	.723	**.717**	**.747**	.770	NAP
Pauwels & Calders [1]	**.848**	**.857**	**.849**	.708	.746	**.933**	NAP
Our Approach	.801	.681	.784	.676	.723	.811	NAP
Camargo et al. [7]	.780	**.502**	**.711**	.440	**.779**	.571	RTP
Pauwels & Calders [1]	.579	.222	.195	.354	.156	.654	RTP
Our Approach	**.849**	.276	.464	**.500**	.632	**.712**	RTP

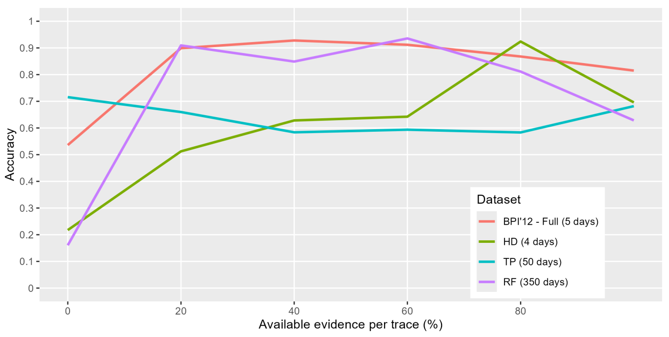


Fig. 4: Accuracies for two-class duration outcome prediction for different amounts of evidence regarding the query trace (corresponding to Query O1). The classification boundaries (greather/less than) for the datasets are: 5 days (BPI'12 - Full), 10 days (HD), 50 days (TP) and 350 days (RF).

probabilities may be located in a similar numerical range. This drastically decreases the predictive performance for complex logs, highlighting that the scalability of our approach is subject to further investigation in future works. Additionally, the strict annotation of activities with their places leads to difficulties for element-wise shifted traces. A solution for that could lie in the application of trace alignments or by inducing dynamic mechanisms into the model. However, when considering the possibilities for diagnostics in the sequential query tasks, i.e., path queries for desired outcomes, the framework builds an interesting foundation for future improvements. Moreover, the capabilities of the proposed approach depend on the performed discretization of any additional attributes. As a form of making our approach more robust and tailored to the process at hand, we will investigate the potential of integrating either expert knowledge or ways of a more effective binning for the connected variables. The latter could be achieved by applying different splitting criteria similar to those in tree-based methods [24]. In terms of improving scalability, one approach to managing the complexity issue is to implement log filtering techniques to condense activity sequences (cf. [10]) or to refactor reoccurring structures into single activities (cf. [4]). Furthermore, we also suggest to apply trace clustering methods to identify similar activity-attribute combinations and to train one PBN for each cluster. This would reduce the overall complexity of the model and facilitate the forecasting as well as the diagnostic querying process due to a lower overall number of parameters. In a similar vein, identifying sequential (sub-)trace hierarchies within cases for training a PBN within a hierarchical approach could mitigate scalability issues.

VI. Conclusion

In this work, we provide a Bayesian inference model in the realm of business processes. Whereas state-of-the-art models mainly focus on the prediction of the next activity given historical event data, our approach breaks the limitations by providing a holistic Process Query System capable of inferring any element of a trace including sequences of activities and outcomes. To provide a valid PBN honoring the properties of a DAG, we design a sequential network structure containing the place information of activities as well as attributes inside a trace. This enables the definition of a proper Bayesian Network whilst capturing all possible sequences of activities in a business event log. An in-depth evaluation shows the diagnostic capabilities of Bayesian inference in the scope of business processes, especially for querying process paths for desired outcomes. Moreover, we provide a discussion on the limitations and possible further extensions.

References

[1] S. Pauwels and T. Calders, "Bayesian network based predictions of business processes," in *Lecture Notes in Business Information Processing*. Springer International Publishing, 2020, pp. 159–175.

[2] I. Prasidis, N.-P. Theodoropoulos, A. Bousdekis, G. Theodoropoulou, and G. Miaoulis, "Handling uncertainty in predictive business process monitoring with bayesian networks," in *2021 12th International Conference on Information, Intelligence, Systems & Applications (IISA)*. IEEE, 2021, pp. 1–8.

[3] K. Lu, X. Fang, and N. Fang, "Pn-bbn: a petri net-based bayesian network for anomalous behavior detection," *Mathematics*, vol. 10, no. 20, p. 3790, 2022.

[4] T. Savickas and O. Vasilecas, "Bayesian belief network application in process mining," in *Proceedings of the 15th International Conference on Computer Systems and Technologies*. ACM, Jun. 2014.

[5] L. Lin, L. Wen, and J. Wang, "MM-pred: A deep predictive model for multi-attribute event sequence," in *Proceedings of the 2019 SIAM International Conference on Data Mining*. Society for Industrial and Applied Mathematics, May 2019, pp. 118–126.

[6] N. Tax, I. Verenich, M. L. Rosa, and M. Dumas, "Predictive business process monitoring with LSTM neural networks," in *Advanced Information Systems Engineering*. Springer International Publishing, 2017, pp. 477–492.

[7] M. Camargo, M. Dumas, and O. González-Rojas, "Learning accurate LSTM models of business processes," in *Lecture Notes in Computer Science*. Springer International Publishing, 2019, pp. 286–302.

[8] B. R. Gunnarsson, S. vanden Broucke, and J. D. Weerdt, "A direct data aware LSTM neural network architecture for complete remaining trace and runtime prediction," *IEEE Transactions on Services Computing*, pp. 1–13, 2023.

[9] R. A. Sutrisnowati, H. Bae, and M. Song, "Bayesian network construction from event log for lateness analysis in port logistics," *Computers & Industrial Engineering*, vol. 89, pp. 53–66, 2015.

[10] C. Moreira, E. Haven, S. Sozzo, and A. Wichert, "Process mining with real world financial loan applications: Improving inference on incomplete event logs," *PLOS ONE*, vol. 13, no. 12, p. e0207806, Dec. 2018.

[11] Z. D. Bozorgi, I. Teinemaa, M. Dumas, M. La Rosa, and A. Polyvyanyy, "Process mining meets causal machine learning: Discovering causal rules from event logs," in *2020 2nd International Conference on Process Mining (ICPM)*. IEEE, 2020.

[12] Z. D. Bozorgi, M. Dumas, M. L. Rosa, A. Polyvyanyy, M. Shoush, and I. Teinemaa, *Learning When to Treat Business Processes: Prescriptive Process Monitoring with Causal Inference and Reinforcement Learning*. Springer Nature Switzerland, 2023, p. 364–380.

[13] S. Weinzierl, S. Dunzer, S. Zilker, and M. Matzner, *Prescriptive Business Process Monitoring for Recommending Next Best Actions*. Springer International Publishing, 2020, p. 193–209.

[14] M. De Leoni, M. Dees, and L. Reulink, "Design and evaluation of a process-aware recommender system based on prescriptive analytics," in *2020 2nd International Conference on Process Mining (ICPM)*. IEEE, 2020, pp. 9–16.

[15] R. G. Cowell, "Local propagation in conditional gaussian bayesian networks," *Journal of Machine Learning Research*, vol. 6, no. 52, pp. 1517–1550, 2005.

[16] I. Verenich, "Helpdesk," https://doi.org/10.17632/39BP3VV62T.1, 2016.

[17] B. van Dongen, "BPI Challenge 2012," https://data.4tu.nl/articles/_/12689204/1, 2012.

[18] ——, "BPI Challenge 2020: Travel Permit Data," https://data.4tu.nl/articles/dataset/BPI_Challenge_2020_Travel_Permit_Data/12718178/1, 2020.

[19] M. M. de Leoni and F. Mannhardt, "Road traffic fine management process," https://data.4tu.nl/articles/_/12683249/1, 2015.

[20] M. Scutari, T. Silander, and R. Ness, "bnlearn: Bayesian Network Structure Learning, Parameter Learning and Inference," https://CRAN.R-project.org/package=bnlearn.

[21] D. Koller and N. Friedman, *Probabilistic graphical models*. MIT Press, 2009.

[22] F. Taymouri, M. L. Rosa, S. Erfani, Z. D. Bozorgi, and I. Verenich, "Predictive business process monitoring via generative adversarial nets: The case of next event prediction," in *Lecture Notes in Computer Science*. Springer International Publishing, 2020, pp. 237–256.

[23] F. Taymouri, M. L. Rosa, and S. M. Erfani, "A deep adversarial model for suffix and remaining time prediction of event sequences," in *Proceedings of the 2021 SIAM International Conference on Data Mining (SDM)*. Society for Industrial and Applied Mathematics, Jan. 2021, pp. 522–530.

[24] S. Robben, M. Velikova, P. J. Lucas, and M. Samulski, *Discretisation Does Affect the Performance of Bayesian Networks*. Springer London, Oct. 2010, p. 237–250.