Root Cause Analysis in Process Mining Using Structural Equation Models

Mahnaz Sadat Qafari and Wil van der Aalst

Rheinisch-Westfälische Technische Hochschule Aachen(RWTH), Aachen, Germany
m.s.qafari@pads.rwth-aachen.de,wvdaalst@pads.rwth-aachen.de

Abstract. Process mining is a multi-purpose tool enabling organizations to monitor and improve their processes. Process mining assists organizations to enhance their performance indicators by helping them to find and amend the root causes of performance or compliance problems. This task usually involves gathering process data from the event log and then applying some data mining and machine learning techniques. However, using the results of such techniques for process enhancement does not always lead to any process improvements. This phenomenon is often caused by mixing up correlation and causation. In this paper, we present a solution to this problem by creating causal equation models for processes, which enables us to find not only the features that cause the problem but also the effect of an intervention on any of the features. We have implemented this method as a plug-in in ProM and we have evaluated it using two real and synthetic event logs. These experiments show the validity and effectiveness of the proposed method.

Keywords: Process mining · Root cause analysis · Causality inference.

1 Introduction

One of the main purposes of using process mining is to enhance process performance indicators leading to reduced costs and response times and better quality. To enhance the performance of a process, we first need to identify friction points in the process. The second step is finding the root causes of each friction point and estimating the possible effect of changing each factor on the process. The final step is planning process enhancement actions and then re-engineering the process. While there are different techniques that help to find the friction points in processes, there is little work on root cause analysis. So, the focus of this paper is on the second step.

The task of finding the root cause of a problem in a given process is quite intricate. Each process involves many steps and in each step, many factors may be of influence. Also, the steps or the order of the steps that are taken for each case may vary. Another obstacle arises when using a classifier, which is basically designed for prediction and not for interventions, for finding the root cause of the problem. By judging the causal relationships among the features merely based on the findings of a classifier, we may fall into the trap of considering correlation as causation.

Consider a scenario where in an online shop, some of the package deliveries get delayed and there is a high correlation between the delayed orders and the resources that were responsible for them. We can infer causal relationships based on the observed

correlations and declare these resources that were in charge of delivering those delayed items as the reason for delays. However, there may be a non-causal correlation between them. Changing the process based on an observed correlation may aggravate the problem (or create new problems). Two correlated events may have a confounder, i.e., a common unmeasured (hidden) cause. In this scenario, the delayed cases are related to the packages with a bigger size which are usually assigned to specific resources.

Two general frameworks for finding the causes of a problem and anticipating the effect of any intervention on the process are random experiments and the theory of causality [10, 11]. Applying random experiments, i.e., randomly setting the values of those features that have a causal effect on the problem of interest and monitoring their effect, is usually too expensive (and sometimes unethical) or simply impossible. The other option is modeling the causal relationships between different features of the process using a *structural causal model* [10, 11] and then studying the effect of changes imposed on each process feature on the process indicators.

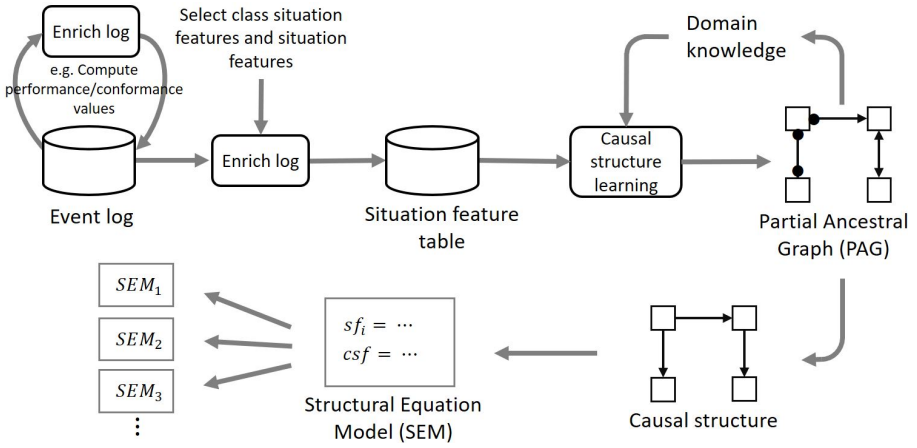

Fig. 1. The general structural causal equation discovery.

In this paper, we propose a framework based on the second approach for root cause analysis, which includes several steps. First, the event log is enriched by several process-related features derived from different data sources like the event log, the process model, and the conformance checking results. Then, depending on the identified problem and features that have a possible direct or indirect causal effect on it, a specific data table, which we call it *situation feature table*, is created. In the next step, a graph encoding the structure of causal relationships among the process features is provided by the customer. The other option for creating such a graph is using a causal structure learning algorithm, also called *search algorithm*, on the data. The resulting graph can be modified by adding domain knowledge as an input to the search algorithm or by modifying the discovered graph. Finally, the strength of each causal relationship and the effect of an intervention on any of the process features on the identified problem are estimated. The general overview of the proposed approach is shown in Figure 1.

2 Motivating Example

Consider an IT company, that implements software for its customers, but does not maintain the implemented software after it has been released. The Petri-net model of this company is depicted in Figure 2. Each trace, which is corresponding to the process of implementation of one project, has an associated attribute *priority*, indicating how urgent the software is for the customer. The manager of the company is concerned about the duration of the implementation phase of projects. By *implementation phase*, we mean the sub-model including two transitions "development" and "test" (marked with a blue rectangle in Figure 2). For simplicity, we use the abbreviations mentioned in Table 1 in this paper.

The manager believes that P, NT, and PBD are the process features that might have a causal effect on IPD. The question is *which features have a causal effect on IPD and to what extent*. Also, the manager needs to know what is the effect of an intervention on any of the features on IPD. Mentioned questions are valid ones to be asked before planning for re-engineering and enhancing a process. Also, we consider C (the *complexity* of a project) as a feature that is not recorded in the event log and may have a causal effect on IPD. The answers to such questions are highly influenced by the

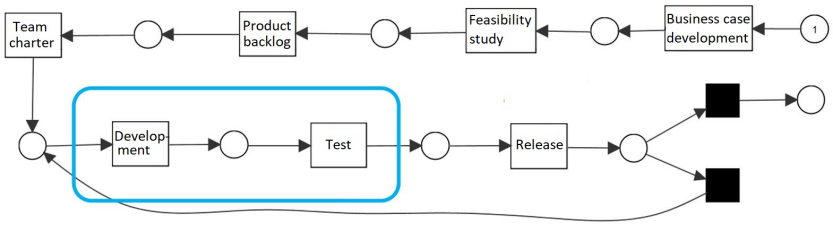

Fig. 2. The process of IT company described in Section 2.

list of abbreviations			
FS feasibility study	CID case ID	BC business case development	REL release
PB product backlog	DEV development	NT number of employees in team	TE test
DD development duration	TD test duration	PBD product backlog duration	P priority
TC team charter	C complexity	IPD implementation phase duration	

Table 1. The list of abbreviations based on IT example, Section 2.

structure of the causal relationship among the features. Some of the possible structures are depicted in Figure 3. According to 3.a), the high correlation between IPD and PBD is not a causation. So, changing PBD does not have any effect on IPD. According to 3.b), one may conclude that all P, PBD, and NT play a role in the determination of the value of IPD and by changing each of these three features, one can influence the value of IPD. According to 3.c), the existence of a hidden feature in the model, captured by C (and depicted by the gray dashed oval in the model), has been considered that causally influences both IPD and PBD. So, their significant correlation relationship is due to

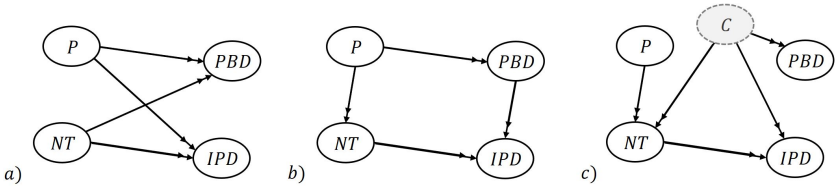

Fig. 3. Three possible causal structures.

having a common cause. If this is the case, then forcing *PB* activity to take a shorter or longer amount of time does not have any effect on the value of *IPD*.

It is worth noting that it is not possible to intervene on all the features. For example, the manager can assign more or fewer people to a project; but he cannot change the complexity of a project. So, it is possible to intervene on *NT*, but not on *C*. Just using common sense and domain knowledge we can judge whether a feature can be intervened.

The remainder of the paper is organized as follows. In Section 3, we briefly present some related work. In Section 4, an overview of the problem and the proposed approach is presented. The details and assumptions of the implemented plugin and the experimental results of applying it on synthetic and real event logs are presented in Section 5. Finally, in Section 6, we summarize our approach and its applications.

3 Related Work

The main approaches in process mining to find the root causes of a performance or compliance problem are classification [4, 6], and rule mining [13]. Although the theory of causality has been studied deeply [11] and its methods have been successfully applied on a variety of domains (e.g. [7, 18]), their application in the area of process mining is limited. However, there are some works in this field that use causality theory. For example, in [5] an approach based on time-series analysis has been proposed for discovering causal relationships between a range of business process characteristics and process performance indicators. The idea is to look at the performance indicators values as time-series, and investigating the causality relationship between them using the Granger causality test [3]. The problem with this approach is that the Granger test can only find predictive causality which might not be a true cause-and-effect relationship.

In [8], a methodology for structural causal model discovery in the area of process mining has been proposed. They propose discovering structural causal models using the event log and the BPMN model of a process. One of the assumptions in this work is that the BPMN model is an accurate model of a process, which is not always the case.

4 Approach

In the proposed method, we assume that we already know that a problem, such as a bottleneck or deviation, exists in the process. Several features may have a causal effect

on the problem, which some of them may not exist in the event log. As the first step, the event log is enriched by adding some new derived features computed from the event log or possibly other sources. The derived features can be related to any of the process perspectives; the time perspective, the data flow-perspective, the control-flow perspective, the conformance perspective, or the resource/organization perspective of the process. The enriched event log is then used for creating the situation feature table. Finally, we discover the *Structural Equation Model (SEM)* of the situation features using the situation feature table. In the sequel, we explain the details of situation feature table creation and SEM inference. But first, we need to define an (enriched) event log.

An event log is a collection of traces where each trace itself is a collection of events. Let $\mathcal{U}_{act}$ be the universe of all *activity names*, $\mathcal{U}_{time}$ the universe of all *time stamps*, $\mathcal{U}_{att}$ the universe of all *attribute names*, and $\mathcal{U}_{val}$ the universe of all *values*. Also, consider $\mathcal{U}_{map} = \mathcal{U}_{att} \not\mapsto \mathcal{U}_{val}$ and $dom : \mathcal{U}_{att} \mapsto \mathbb{P}(\mathcal{U}_{val})$[1], the function that returns the set of all possible values of a given attribute name. We define an event log as follows:

Definition 1 (Event Log). *Each element of $\mathcal{U}_{act} \times \mathcal{U}_{time} \times \mathcal{U}_{map}$ is an* event *and $\mathcal{E}$ is the universe of all possible events. Let $\pi_{act}(e) = a$, $\pi_{time}(e) = t$, and $\pi_{map}(e) = m$ for a given event $e = (a, t, m) \in \mathcal{E}$. Each element of $\mathbb{P}(\mathcal{U}_{map} \times \mathcal{E}^+)$ is called an* event log *where $\mathcal{E}^+$ is the set of all non empty sequences of events such that for each $\langle e_1, \ldots, e_n \rangle \in \mathcal{E}^+$ we have $\forall_{1 \leq i < j \leq n} \pi_{time}(e_i) \leq \pi_{time}(e_j)$. The universe of all possible logs is denoted by $\mathcal{L}$ and each element (m, E) of an event log is called a* trace.

We assume that in a given log L each event has a unique time stamp. Also, given $E = \langle e_1, \ldots, e_n \rangle \in \mathcal{E}^+$, we define $tail(E) = e_n$, $prfx(E) = \{\langle e_1, \ldots, e_i \rangle | 1 \leq i \leq n\}$ which is the set of all the nonempty prefixes of E, and $set(E) = \{e_1, \ldots, e_n\}$.

4.1 Situation Feature Table Creation

An observed problem in the process might be related to either a trace or a specific activity. We assume that, given a problem, in each trace only that part of the data that has been recorded before the occurrence of the problem can have a causal effect on it. So the relevant part of a trace to a given problem is a prefix of it which we call a *situation*. More formally, a situation is an element of $\mathcal{U}_{map} \times \mathcal{E}^+$ and we use $\mathcal{U}_{sf}$ to denote the universe of all possible situations. Given an event log $L \in \mathcal{L}$, we define the set of all situations of L as $S_L = \bigcup_{(m,E) \in L} \{(m, E') | E' \in prfx(E)\}$. Considering $ActNames_L$ as the set of the activity names of all the events that appear in the traces of L, we define an *a-based situation subset* of L as $S_{L,a} = \{(m, E) \in S_L | \pi_{act}(tail(E)) = a\}$, where $a \in ActNames_L$ and *trace-based situation subset* of L as $S_{L,\perp} = L$.

There are two types of attributes in a given event log, attributes linked to the traces and attributes linked to the events. When extracting the data from event log, we need to distinguish these two levels of attributes. For that, we use *situation feature* function which is defined over the situations and is identified by an activity, a (possibly $a = \perp$), and an attribute, at. If $a = \perp$, $sf_{a,at}((m, E))$ returns the value of at in trace level (i.e. $m(at)$). However, if $a \neq \perp$, then $sf_{a,at}(s)$ returns the value of at in $e \in set(E)$ with the maximum time stamp for which $\pi_{act}(e) = a$. More formally:

[1] We define $\mathbb{P}(A)$ as the set of all non-empty subsets of set A.

Definition 2 (Situation Feature). *Let $L \in \mathcal{L}$, $a \in ActNames_L \cup \{\perp\}$, and $at \in \mathcal{U}_{att}$. We define a* situation feature *as $sf_{a,at} : \mathcal{U}_{sit} \not\mapsto \mathcal{U}_{val}$. Given $(m, E) \in L$, we define,*

$$sf_{a,at}((m,E)) = \begin{cases} m(at) & a = \perp \\ \pi_{map}(\arg\max_{e \in \{e \in set(E) | \pi_{act}(e) = a\}} \pi_{time}(e))(at) & a \in ActNames_L \end{cases}.$$

We denote the universe of all possible situation features by $\mathcal{U}_{sf}$. With slight abuse of notation, we define $dom(sf_{a,at}) = dom(at)$. Also, for a given $n \in \mathbb{N}$, $\boldsymbol{SF} \in \mathcal{U}_{sf}^n$ is a situation feature extraction plan *of size n, where $\mathcal{U}_{sf}^n$ is defined as $\underbrace{\mathcal{U}_{sf} \times \cdots \times \mathcal{U}_{sf}}_{n \text{ times}}$.*

We can interpret a situation feature extraction plan as the schema composed of those situation features that are relevant to the given problem in the process. In the sequel, we call the situation feature that captured the existence of the problem (or represents the quantity or quality of interest) in the process the *class situation feature* and denote it as *csf*. Moreover, in case of no ambiguity, we remove the subscripts of situation features to increase readability.

Now, we can concretely specify the problem in the process and the set of situation features that we need to investigate their causal effect on the occurrence of the problem.

Definition 3 (Causal Situation Specification). *A causal situation specification is a tuple $CSS = (\boldsymbol{SF}, csf)$ in which $\boldsymbol{SF} = (sf_1, \ldots, sf_n) \in \mathcal{U}_{sf}^n$ for some $n \in \mathbb{N}$, $csf \in \mathcal{U}_{sf}$, and $csf \notin set(\boldsymbol{SF})$ where with slight abuse of notation $set(\boldsymbol{SF}) = \{sf_1, \ldots, sf_n\}$.*

Here, $\boldsymbol{SF}$ is the tuple that includes all of the situation features that we expect them to have a potential causal effect on the class situation feature. Here, we assume that there is no hidden common confounder exists any subset of situation features in $set(\boldsymbol{SF}) \cup \{csf\}$. Using the causal situation specification, we can define a situation feature table as follows:

Definition 4 (Situation Feature Table). *Given an event log $L \in \mathcal{L}$ and a causal situation specification $CSS = (\boldsymbol{SF}, csf)$, where $\boldsymbol{SF} = (sf_1, \ldots, sf_n)$ and $csf = sf_{a,at}$, a* situation feature table *is a multi-set which is defined as:*

$$T_{CSS,L} = [(sf_1(s), \ldots, sf_n(s), csf(s)) | s \in S_{L,a} \wedge csf(s) \neq \perp].$$

We call $T_{CSS,L}$ a situation feature table of *L.*

4.2 SEM Inference

Given a log L, if we consider a trace or activity based situation subset of L as a sample and each situation feature as a random variable, then we can define a structural equation model for a given causal structure specification as follows[2]:

Definition 5 (Structural Equation Model (SEM)). *Given $CSS = (\boldsymbol{SF}, csf)$, a structural equation model is defined as a collection of assignments $\mathcal{S} = \{S_{sf_1}, \ldots, S_{sf_n}, S_{csf}\}$ such that*

$$S_{sf} \quad : \quad sf = f_{sf}(\boldsymbol{PA}_{sf}, N_{sf}), \quad sf \in set(\boldsymbol{SF}) \cup \{csf\},$$

[2] Definition 5 and 8 are based on [11].

where $\boldsymbol{PA}_{sf} \subseteq set(\boldsymbol{SF}) \cup \{csf\} \setminus \{sf\}$ *is called* parents *of sf and and* $N_{sf_1}, \ldots, N_{sf_n}, N_{csf}$ *are distributions of the noise variables, which we require to be jointly independent.*

Note that these equations are not normal equations but a way to determine how to generate the observational and the interventional distributions. The set of parents of a situation feature is the set of situation features that have a direct causal effect on it.

The structure of the causal relationships between the situation features in a SEM can be encoded as a directed acyclic directed $\mathcal{G} = (V, \twoheadrightarrow)$ which is called a *causal structure*. In this graph, $\boldsymbol{V} = set(\boldsymbol{SF}) \cup \{csf\}$, $\twoheadrightarrow = \{(sf_1, sf_2) \in \boldsymbol{V} \times \boldsymbol{V} | sf_1 \in \boldsymbol{PA}(sf_2)\}$.

The first step of inferring a SEM is discovering its causal structure and the second step is estimating a set of equations describing how each situation feature is influenced by its immediate causes. In the sequel, we describe these two steps.

Causal Structure Discovery. The causal structure can be determined by an expert who possesses the domain knowledge about the underlying process and the causal relationships between its features. But having access to such knowledge is quite rare. Hence, we support discovering the causal structure in a data driven manner.

Several search algorithms have been proposed in the literature (e.g., [2, 15, 9]). The input of a search algorithm is observational data in the form of a situation feature table (and possibly knowledge) and its output is a graphical object that represents a set of causal structures that cannot be distinguished by the algorithm. One of these graphical objects is *Partial Ancestral Graph (PAG)* introduced in [19].

A PAG is a graph whose vertex set is $\boldsymbol{V} = set(\boldsymbol{SF}) \cup \{csf\}$ but has different edge types, including $\rightarrow, \leftrightarrow, \bullet\!\!\rightarrow, \bullet\!\!-\!\!\bullet$[3]. Each edge type has a specific meaning. Let $sf_1, sf_2 \in \boldsymbol{V}$. $sf_1 \rightarrow sf_2$ indicates that sf_1 is a direct cause of sf_2, $sf_1 \leftrightarrow sf_2$ means that neither sf_1 nor sf_2 is an ancestor of the other one, even though they are probabalistically dependent (i.e., sf_1 and sf_2 are both caused by one or more hidden confounders), $sf_1 \bullet\!\!\rightarrow sf_2$ means sf_2 is not a direct cause of sf_1, and $sf_1 \bullet\!\!-\!\!\bullet sf_2$ indicates that there is a relationship between sf_1 and sf_2, but nothing is known about its direction. The formal definition of a PAG is as follows [19]:

Definition 6 (Partial Ancestral Graph (PAG)). *A PAG is a tuple* $(\boldsymbol{V}, \rightarrow, \leftrightarrow, \bullet\!\!\rightarrow, \bullet\!\!-\!\!\bullet)$ *in which* $\boldsymbol{V} = set(\boldsymbol{SF}) \cup \{csf\}$ *and* $\rightarrow, \leftrightarrow, \bullet\!\!\rightarrow, \bullet\!\!-\!\!\bullet \subseteq \boldsymbol{V} \times \boldsymbol{V}$ *such that* $\rightarrow$, $\leftrightarrow$, $\bullet\!\!\rightarrow$, *and* $\bullet\!\!-\!\!\bullet$ *are mutually disjoint. Moreover, there is at most one edge between every pair of situation features in* $\boldsymbol{V}$.

The discovered PAG by the search algorithm represents a class of causal structures that satisfies the conditional independence relationships discovered in the situation table and ideally, includes the true causal structure of the causal situation specification. Now, it is needed to modify the discovered PAG to a compatible causal structure. As we assume no hidden common confounder exists, we expect that in the PAG, relation $\leftrightarrow$ be empty[4]. We can define the compatibility of a causal structure with a PAG as follows:

Definition 7 (Compatibility of a Causal Structure With a Given PAG). *Given a PAG* $(\boldsymbol{V}, \rightarrow, \leftrightarrow, \bullet\!\!\rightarrow, \bullet\!\!-\!\!\bullet)$ *in which* $\leftrightarrow = \emptyset$, *we say a causal structure* $(\boldsymbol{U}, \twoheadrightarrow)$ *is compatible with*

[3] We usually use $a \bullet b$ instead of $(a, b) \in \bullet$ for $\bullet \in \{\rightarrow, \leftrightarrow, \bullet\!\!\rightarrow, \bullet\!\!-\!\!\bullet\} \cup \{\twoheadrightarrow\}$.

[4] If $\leftrightarrow \neq \emptyset$, the user can restart the procedure after adding some more situation features to the causal situation specification.

the given PAG if $\boldsymbol{V} = \boldsymbol{U}$, $(sf_1 \rightarrow sf_2 \vee sf_1 \bullet\!\!\rightarrow sf_2) \implies sf_1 \twoheadrightarrow sf_2$, *and* $sf_1 \bullet\!\!-\!\!\bullet sf_2 \implies (sf_1 \twoheadrightarrow sf_2 \oplus sf_2 \twoheadrightarrow sf_1)$, *where* $sf_1, sf_2 \in \boldsymbol{V}$.

To transform the output PAG to a compatible causal structure, which represents the causal structure of the situation features in the situation feature specification, domain knowledge of the process and common sense can be used. These information can be use to directly modify the discovered PAG or by adding them to the search algorithm, as an input, in the form of *required directions* or *forbidden directions* denoted as D_{req} and D_{frb}, respectively. $D_{req}, D_{frb} \subseteq \boldsymbol{V} \times \boldsymbol{V}$ and $D_{req} \cap D_{frb} = \emptyset$. If $(sf_1, sf_2) \in D_{req}$ then $sf_1 \rightarrow sf_2$ or $sf_1 \bullet\!\!\rightarrow sf_2$ in the output PAG. However, if $(sf_1, sf_2) \in D_{frb}$, then in the discovered PAG it should not be the case that $sf_1 \rightarrow sf_2$.

Causal Strength Estimation. The final step of discovering the causal model is estimating the strength of each direct causal effect using the observed data. Suppose $\mathcal{G}$ is the causal structure of a causal situation specification $CSS = (\boldsymbol{SF}, csf)$. As $\mathcal{G}$ is a directed acyclic graph, we can sort its nodes in a topological order γ. Now, we can statistically model each situation feature sf_2 as a function of the noise terms N_{sf_1} of the situation features sf_1 for which $\gamma(sf_1) \leq \gamma(sf_2)$, where $sf_1, sf_2 \in set(\boldsymbol{SF}) \cup \{csf\}$. The result is $sf_2 = f_2((N_{sf_1})_{sf_1:\gamma(sf_1)\leq\gamma(sf_2)})$ [11]. The set of these functions, for all $sf \in set(\boldsymbol{SF}) \cup \{csf\}$, is the SEM of $CSS = (\boldsymbol{SF}, csf)$.

Finally, we want to answer questions about the effect of an intervention on any of the situation features on the class situation feature. Here we focus on *atomic interventions* which are defined as follows:

Definition 8 (Atomic Intervention). *Given an SEM* $\mathcal{S}$, *an intervention is obtained by replacing* $S_{sf} \in \mathcal{S} \backslash \{S_{csf}\}$ *by* $sf = c$ *where* $c \in \mathbb{R}$.

Note that the corresponding causal structure of an SEM after intervention on sf is obtained from the original causal structure of $\boldsymbol{M}$ by removing all the incoming edges to sf [11]. When we intervene on a situation feature, we just replace the equation of that situation feature in the SEM and the others do not change as causal relationships are autonomous under interventions [11].

If in a given causal structure of a causal situation specification $CSS = (\boldsymbol{SF}, csf)$, there is no directed path between $sf \in set(\boldsymbol{SF})$ and csf, they are independent and consequently, intervening on sf by forcing $sf = c$ has no effect on csf. Otherwise, we need to estimate the effect of that intervention on the class situation feature which is the function estimating (the distribution of) csf condition on $sf = c$, while controlling for situation features in $\boldsymbol{PA}_{sf}$.

5 Experimental Results

To validate the proposed approach, we implemented it as a plugin in ProM [16], an open-source and extensible platform for process mining. The implemented plugin takes the event log, the Petri-net model of the process, and, the conformance checking results of replaying the given event log on the given model as input. The implemented plugin is available in the nightly-build of ProM under the name *causality inference in process mining*. In the following, we briefly mention some of the implementation details and the results of applying the plugin on both synthetic and real event logs.

5.1 Implementation Notes

As the search algorithm, we use the Greedy Fast Causal Inference (GFCI) algorithm [9]. GFCI is a hybrid search algorithm where its inputs are the situation feature table and possibly background knowledge and its output is a PAG with the highest score on the input data. In [9], it has been shown that if the assumptions mentioned in Section 4.2 hold, then under the large sample limit each edge in the PAG computed by GFCI is correct. Also, using empirical results on simulated data, it has been shown that GFCI has the highest accuracy among several other search algorithms [9]. In this plugin, we use the Tetrad [14] implementation of the GFCI algorithm. In the experiments, we use the following settings for the GFCI algorithm: cutoff for p-values = 0.05, maximum path length = -1, maximum degree = -1, and penalty discount = 2.

For estimating the effect of an intervention on the class situation feature, in the case of continuous data, we assume linear dependencies among the situation features and additive noise. We can represent the SEM graphically by considering the coefficient of sf_1 in S_{sf_2} as the weight of the edge from sf_1 to sf_2 in its corresponding causal structure. Thus, to estimate the magnitude of the effect of sf on the csf, it is enough to sum the multiplication of the weights of the edges for each directed path from sf to csf.

5.2 Synthetic Data

We have created the Petri-net model of the process described in Section 2 using CPN Tools [12] and generate an event log with 1000 traces. The log is enriched by adding IPD as a derived trace level attribute that indicates the duration of the sub-model including DEV and TE transitions in person-day. We generate the log with the following settings:

$$
\begin{array}{ll}
sf_{\perp,C} = N_{sf_{\perp,C}} & N_{sf_{\perp,C}} \sim Uniform(1,10) \\
sf_{BC,P} = N_{sf_{BC,P}} & N_{sf_{BC,P}} \sim Uniform(1,3) \\
sf_{PB,PBD} = 10sf_{\perp,C} + N_{sf_{PB,PBD}} & N_{sf_{PB,PBD}} \sim Uniform(-2,4) \\
sf_{TD,NT} = 5sf_{\perp,C} + 3sf_{BC,P} + N_{sf_{TD,NT}} & N_{sf_{TD,NT}} \sim Uniform(-1,2) \\
sf_{\perp,IPD} = 50sf_{\perp,C} + 5sf_{TD,NT} + N_{sf_{\perp,IPD}} & N_{sf_{\perp,IPD}} \sim Uniform(10,20)
\end{array}
$$

Then, we use the proposed approach for $CSS = ((sf_{\perp,P}, sf_{PB,PBD}, sf_{TC,TN}), sf_{\perp,IPD})$. The discovered causal structure is as depicted in Figure 4.a. This causal structure does not say much about the direction of discovered potential causal relationships. Regarding the types of edges, we can guess that there might be another influential attribute that acts as a hidden common cause. If we consider $sf_{\perp,C}$ as one of the independent situation features, then the discovered causal structure is the one depicted in Figure 4.b, which is more accurate. If the complexity of a project is not recorded in the event log, we can assume that the PB takes longer in more complex projects and compute it as the floor of the value of PBD divided by 10. Now, using domain knowledge, we can turn this PAG to the one depicted in Figure 4.c. By doing the estimation, the SEM shown in Figure 4.d is obtained. By comparing the estimated coefficients of situation features in Figure 4.d (the weights of edges), and those in the mentioned equations, it is clear the estimated and real strengths of causal relationships are quite close which proves that the implemented plugin is capable of discovering the true SEM.

Now, if we want to see how each situation feature affects the class situation feature, we just need to click on its corresponding node in the causal structure to see

the estimated interventional effect. Suppose c_1 is a constants. For example, we have $sf_{\perp,IPD} = 75.0004 \times sf_{\perp,C} + c_1 \times sf_{TD,NT} + noise$ which means that if we could enforce the complexity of the projects to be one unit more complex, then the implementation phase will take approximately 75 more person-days. The other estimated interventional effect is $sf_{\perp,IPD} = 0.0 \times sf_{PB,PBD}$ that means intervention on $sf_{PB,PBD}$ has no effect on $sf_{\perp,IPD}$.

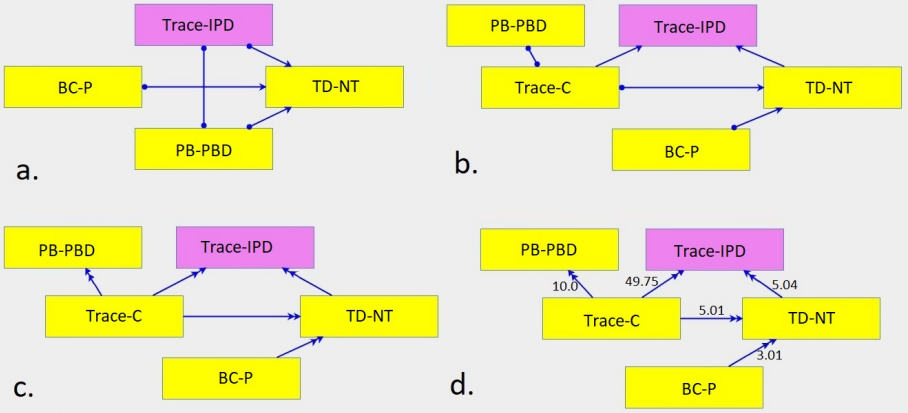

Fig. 4. a) The PAG of causal situation specification of Section 5.2, discovered by the implemented plugin. Using this PAG, we can guess that there are other influential situation features not recorded in the data. b) The resulting PAG after considering $sf_{\perp,C}$ as one of the influential situation features, c) The causal structure obtained by modifying the previous one using common sense and domain knowledge. d) The causal structure model generated by doing estimation on the strength of each discovered causal relationship.

5.3 Real Data

For the real data, we use *receipt phase of an environmental permit application process (WABO) CoSeLoG project* [1] event log (receipt log for short), which includes 1434 traces and 8577 activities. As a problem, we consider the delay in some of the cases where the threshold for the delay is set to 3% of the maximum duration of all traces. Note that the average trace duration in this event log is about 2% of the maximum duration of its traces. So, the class situation feature is $sf_{\perp,delay}$. Here, we want to investigate the effect of three situation features indicating the choice of recourse working on three activities "Confirmation of receipt" denoted as $Conf$, "T02 Check confirmation of receipt" denoted as $T02$, and "T04 Determine confirmation of receipt" denoted as $T04$, on $sf_{\perp,delay}$. Thus, the causal situation specification that we are interested in is $CSS = ((sf_{Conf,Resource}, sf_{T02,Resource}, sf_{T04,Resource}), sf_{\perp,delay})$. Note that using the Chi-square test, there is a high correlation between every pair of situation features.

The output of applying the implemented plugin on the created situation feature table for this problem is shown in Figure 5.a). Using the temporal ordering of the activities (in this process, $Conf$ happens before $T02$, and $T02$ happens before $T04$ in all the traces) and common sense (the choice of the resource of an activity has no effect on the choice of the resource of another activity that happened before) we can infer that

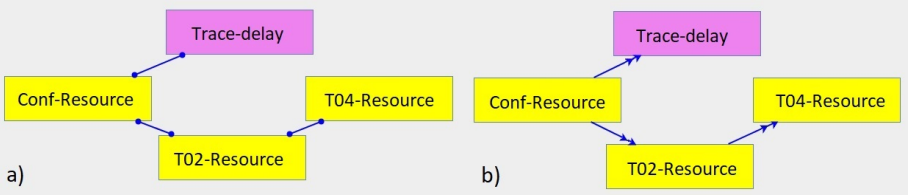

Fig. 5. a) The PAG of causal situation specification of receipt event log example in Section 5.3, discovered by the implemented plugin. b) The causal structure obtained by modifying the previous one using common sense and domain knowledge.

the true causal structure is the one shown in Figure 5.b). This causal structure indicates that $sf_{Conf,Resource}$ has a causal effect on $sf_{\perp,delay}$ and there is no causal relationship between $sf_{T02,Resource}$ and $sf_{\perp,delay}$ and also between $sf_{T04,Resource}$ and $sf_{\perp,delay}$. Moreover, it indicates that the choice of $sf_{T02,Resource}$ is influenced by $sf_{Conf,Resource}$. By doing the estimation and clicking on each node of the graph we can see the interventional distributions of $sf_{\perp,delay}$ caused by intervention on the corresponding situation feature of that node. For example, we can see that the the probability of $sf_{\perp,delay} = delayed$ under an intervention which enforce $sf_{Conf,Resource} = Resource14$ is almost 0.159.

6 Conclusion

If an organization has performance and/or conformance problems, then it is vital to uncover the causes of these problems and the strength of their effect. It is also needed to investigate the effect of an intervention on the process. This information is essential to design and order the process enhancement and re-engineering steps. This is a very fundamental and influential step toward process enhancement. Process interventions based on correlations that are not causalities may lead to more problems. By using the framework proposed in this paper, the stakeholders can incorporate both domain knowledge and potential statistically supported causal effects to find the SEM of the features and indicators of the process. Using SEM in this framework enables us to estimate to what extent each feature contributes to the given problem using the observational data.

As mentioned in Section 4.2, the search algorithm assumes strong assumptions such as the independence and identically distribution of situations. One of the main drawbacks of applying this framework is that the features of different traces are not independent. Future research aims to address these limitations.

Acknowledgement

We thank the Alexander von Humboldt (AvH) Stiftung for supporting our research.

References

1. Buijs, J.: Receipt phase of an environmental permit application process (‘wabo’), coselog project. Eindhoven University of Technology (2014)

2. Chickering, D.M.: Optimal structure identification with greedy search. Journal of machine learning research **3**(Nov), 507–554 (2002)
3. Granger, C.W.: Some recent development in a concept of causality. Journal of econometrics **39**(1-2), 199–211 (1988)
4. Gupta, N., Anand, K., Sureka, A.: Pariket: Mining business process logs for root cause analysis of anomalous incidents. In: Proceedings of Databases in Networked Information Systems - 10th International Workshop. vol. 8999, pp. 244–263. Springer (2015). https://doi.org/10.1007/978-3-319-16313-0_19
5. Hompes, B.F.A., Maaradji, A., Rosa, M.L., Dumas, M., Buijs, J.C.A.M., van der Aalst, W.M.P.: Discovering causal factors explaining business process performance variation. In: Proceedings of Advanced Information Systems Engineering. vol. 10253, pp. 177–192. Springer (2017). https://doi.org/10.1007/978-3-319-59536-8_12
6. de Leoni, M., van der Aalst, W.M., Dees, M.: A general process mining framework for correlating, predicting and clustering dynamic behavior based on event logs. Inf. Syst. **56**(C), 235–257 (2016). https://doi.org/10.1016/j.is.2015.07.003
7. Mothilal, R.K., Sharma, A., Tan, C.: Explaining machine learning classifiers through diverse counterfactual explanations. In: Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency. pp. 607–617 (2020), http://arxiv.org/abs/1905.07697
8. Narendra, T., Agarwal, P., Gupta, M., Dechu, S.: Counterfactual reasoning for process optimization using structural causal models. In: Proceedings of Business Process Management Forum. vol. 360, pp. 91–106. Springer (2019). https://doi.org/10.1007/978-3-030-26643-1_6
9. Ogarrio, J.M., Spirtes, P., Ramsey, J.: A hybrid causal search algorithm for latent variable models. In: Proceedings of Probabilistic Graphical Models - Eighth International Conference. pp. 368–379 (2016), http://proceedings.mlr.press/v52/ogarrio16.html
10. Pearl, J.: Causality. Cambridge university press (2009)
11. Peters, J., Janzing, D., Schölkopf, B.: Elements of causal inference: foundations and learning algorithms. MIT press (2017)
12. Ratzer, A.V., Wells, L., Lassen, H.M., Laursen, M., Qvortrup, J.F., Stissing, M.S., Westergaard, M., Christensen, S., Jensen, K.: CPN tools for editing, simulating, and analysing coloured petri nets. In: Proceedings of Applications and Theory of Petri Nets. vol. 2679, pp. 450–462. Springer (2003). https://doi.org/10.1007/3-540-44919-1_28
13. Sani, M.F., van der Aalst, W.M.P., Bolt, A., García-Algarra, J.: Subgroup discovery in process mining. In: Proceedings of Business Information Systems. pp. 237–252. Springer (2017). https://doi.org/10.1007/978-3-319-59336-4_17
14. Scheines, R., Spirtes, P., Glymour, C., Meek, C., Richardson, T.: The tetrad project: Constraint based aids to causal model specification. Multivariate Behavioral Research **33**(1), 65–117 (1998)
15. Spirtes, P., Glymour, C.N., Scheines, R., Heckerman, D.: Causation, prediction, and search. MIT press (2000)
16. Verbeek, H., Buijs, J., Van Dongen, B., van der Aalst, W.M.: Prom 6: The process mining toolkit. Proc. of BPM Demonstration Track **615**, 34–39 (2010)
17. Verenich, I., Dumas, M., La Rosa, M., Nguyen, H.: Predicting process performance: A white-box approach based on process models. Journal of Software: Evolution and Process **31**(6), e2170 (2019)
18. Wang, Y., Liang, D., Charlin, L., Blei, D.M.: The deconfounded recommender: A causal inference approach to recommendation. CoRR **abs/1808.06581** (2018), http://arxiv.org/abs/1808.06581
19. Zhang, J.: On the completeness of orientation rules for causal discovery in the presence of latent confounders and selection bias. Artif. Intell. **172**(16-17), 1873–1896 (2008). https://doi.org/10.1016/j.artint.2008.08.001