AITIA-PM: Discovering the true causes of events in a process mining context

Peer-reviewed author version

VAN HOUDT, Greg; MARTIN, Niels & DEPAIRE, Benoit (2023) AITIA-PM: Discovering the true causes of events in a process mining context. In: Engineering Applications of Artificial Intelligence, 126 (Part D) (Art N° 107145).

DOI: 10.1016/j.engappai.2023.107145
Handle: http://hdl.handle.net/1942/41433

AITIA-PM: Discovering the True Causes of Events in a Process Mining Context

Greg Van Houdt[a,*], Niels Martin[a], Benoît Depaire[a]

[a] *UHasselt – Hasselt University, Martelarenlaan 42, Hasselt, BE-3500, Belgium*

Abstract

Process mining is a research area that enables businesses to analyze and improve their processes by deriving knowledge from event logs. While pinpointing the causes of, for instance, a negative case outcome can provide valuable insights for business users, only a limited amount of research has been done to uncover causal relations within the process mining field while actively distinguishing between correlation and causality. The AITIA-PM algorithm is one of these research projects. This article updates the AITIA-PM method, which uses causality theory to measure cause-and-effect relationships in event logs. The system uses probabilistic temporal logic (PTL) to formulate hypotheses explicitly and then automatically checks them for causality using available data. More precisely, AITIA-PM is designed for process mining since it operates directly on event logs, giving users access to the information stored there, and increasing the scope for meaningful causal analysis in a process mining setting. With this addition, PTL is emphasized more as a crucial algorithmic component, and the method to control for false discovery rates (FDR) is adjusted for increased practical use. The case study shows that after the domain expert provides the search space of hypotheses, the AITIA-PM algorithm can extract valuable cause-effect insights from an event log. The search space can be flexibly defined, making AITIA-PM a powerful tool for business users. An evaluation on artificial data proves AITIA-PM is capable of extracting the causal relationships, while a demonstration on the Road Traffic Fines Management dataset shows the applicability of the algorithm on real data.

Keywords: Process Mining, Causal Analysis, Data-driven, Probabilistic Temporal Logic, Event Data Analytics

1. Introduction

Process mining is a research domain that enables businesses to analyze and improve their processes by extracting insights from event logs [1], in which the actual execution of a business process is recorded. This information can be used for process discovery [2], but in combination with a normative model of how the process should have been executed, also for conformance checking [3], for example. Merely discovering how the process deviated from the normative model might not be sufficient for the analyst. Insights into the true causes of those deviations or, in more general terms, specific events can be of much more interest to the business user. As such, data-driven root cause analysis in process mining is desirable, despite being a complex task [4].

Previous research has presented methods and techniques to execute data-driven root cause analyses [4, 5, 6, 7]. However, a root cause analysis should be distinct from a correlation analysis. A correlation between a process characteristic and a particular (un)desirable outcome or event does not necessarily imply a cause-effect relation. Many other factors can be in play. Thus, confounding variables must be acknowledged, and spurious relations must be filtered. Also, when a technique is performing an actual root cause analysis, it tends to impose heavy restrictions on the data, e.g., [8] where only linear causal relations are accepted. Recent research tried to tackle these problems [9]. Still, although the paper claims to be capable of tapping into the vast information in event logs, the theoretical foundations and empirical demonstration show room for improvement.

*Corresponding author

Email addresses: `greg.vanhoudt@uhasselt.be` (Greg Van Houdt), `niels.martin@uhasselt.be` (Niels Martin), `benoit.depaire@uhasselt.be` (Benoît Depaire)

Against this background, this paper proposes an update to the AITIA-PM algorithm from Van Houdt et al. [9]. Inspired by [10, 11], the algorithm is a data-driven way to perform causal analytics, tailored explicitly for process mining event logs. By employing PTL, AITIA-PM can create causal hypotheses very flexibly. Additionally, confounding factors are explicitly taken into account in the method. Our contributions are best summarized as follows:

- We update the AITIA-PM algorithm on multiple facets, increasing the potential of AITIA-PM for causal analysis in process mining. This is accomplished by (i) increasing the support of the PTL modeling language to define cause-effect relations, now specifically tailored to XES event logs, and (ii) proposing a new method to determine causal significance based on q-values [12, 13]. The main advantage of q-values, being a state-of-the-art FDR controller, is the better applicability in real-world scenarios where a domain expert might not define a large set of hypotheses.

- The case study shows that, under the assumption that the domain expert defines the entire search space, one can easily identify causes for a given effect in a data-driven fashion based on information stored in the event log while controlling for confounding variables. The domain expert is given full freedom in defining the search space. The applicability of AITIA-PM is shown on a real-world dataset, and the algorithmic output is also validated on synthetic data.

The remainder of this paper is structured as follows. Relevant related work is discussed in Section 2. Section 3 then gives essential background information regarding causality theory and PTL, after which Section 4 explains how the AITIA-PM taps into the knowledge in event logs using PTL. Furthermore, two case studies are executed in Section 5 which also provide insights into the computational complexity, after which Section 6 dives deeper into the differences with other techniques and our limitations. Finally, Section 7 concludes the paper.

2. Related Work

Process mining does not restrict a root cause analysis (RCA) to a single family of approaches [9]. Examples are (i) classification techniques as seen in, for example, [6, 14, 15, 16, 17], and (ii) rule mining algorithms like association rules [18] and subgroup discovery [19]. Unfortunately, the distinction between correlation and causation receives too little attention in most applications.

To identify cause-and-effect relationships by testing for Granger causality [8], Hompes et al. [5] suggested a graph-based method that produced a time series analysis, explicitly taking causation rather than correlation between features into account. It's not ideal either, though. Granger causality, as initially conceived, cannot explain instantaneous or nonlinear causal relationships or confounding effects. Additionally, Maziarz [20] noted that Granger causality makes several significant assumptions about the underlying data that are rarely realized in practice. Recent research by Koorn et al. [21], which proposed a novel method to mine statistical relationships between events, specifically for the healthcare industry, underlined the significance of examining confounding variables. The end-user is explicitly informed of the confounding factors' effects, demonstrating how neglecting them can significantly harm the findings of a causal model.

Qafari and van der Aalst recently presented research on structural equation models for RCA, which was later expanded with counterfactual reasoning [4, 22]. If available, the domain expert can offer the structure of causal links. The counterfactual reasoning extension enables the writers to come up with suggestions for how particular cases should have been handled better to prevent issues in the future [22]. Despite the model's accuracy, the authors acknowledge that adopting a machine learning technique carries the danger of receiving incorrect or inaccurate recommendations or perhaps missing out on the right ones. Narendra et al. [23] demonstrate how to use structural causal models and counterfactual reasoning to address what-if problems, demonstrating the efficacy of the techniques but recognizing their lack of intuitiveness. Causal graphs are further used in the work of Leemans and Tax [24], where the authors specifically stress the difference between association (correlation) and causation. The goal of [24] is to design a causal method to study dependencies between control-flow decision points. Another interesting application is the use of process discovery to identify the causes of abnormal machine behavior based on two event logs: a log of healthy machines and a log of unhealthy machines [25].

In order to establish a definition of causality, several frameworks have been proposed in the literature [26]. The potential outcomes framework by Rubin [27, 28], also called the Rubin Causal Model (RCM) is a statistical framework for analyzing causal effects based on observational data. The model is based on the observation unit getting a *treatment* or not after which the alternative futures can be compared. In short: RCM describes how the causal effect of treatment X versus treatment Y on a particular unit or entity during a specific time interval from t_1 to t_2 can be understood as the difference between (i) what would have happened to the unit at t_2 if it received treatment X starting at t_1, and (ii) what would have happened at t_2 if it received treatment Y starting at t_1 [27]. Confounding can be taken into account by including said confounding variables in the statistical analysis to increase the accuracy of the causal effect under investigation. However, in RCM a *decision* implies a causal effect: one decides to apply a treatment or not.

More recently, Pearl introduced a graphical framework of causality [29, 30]. The graphical framework is based on Bayesian networks capable of representing causal relationships between variables by simulating causal mechanisms operating in the environment. A key advantage of this approach is the ability to move beyond simple associations, and by manipulating the model, different scenarios can be tested. Similarly, confounders can be identified. However, the graphical model assumes all causal relationships are deterministic. This assumption may not hold in all situations [30]. For more details on defining causality, including other existing frameworks, the interested reader is referred to [26].

To establish causality, Kleinberg's causality measure and associated algorithm [10, 11] heavily rely on the philosophical underpinnings of Hume [31] and Suppes [32]. To that end, the algorithm can distinguish genuine causal relations from spurious ones from data. It is accomplished by using PTL to define hypotheses that are subsequently tested using statistical significance and probability theory. Additionally, confounding variables are addressed explicitly by Kleinberg's method. As such, the AITIA-PM algorithm opts for Kleinberg's framework because we retain a stochastic view of causality, which Pearl's framework does not allow for. Second, Kleinberg allows for much richer expressions of causal relations than RCM allows for, given that a cause can also be an observation of multiple events in the past instead of one treatment. Kleinberg (i) allows for a causal effect to be clearly defined, (ii) shows relatively low computational complexity while PTL allows to flexibly define causal hypotheses of varying causal complexity, and (iii) the final output is both statistically significant and highly interpretable. Van Houdt et al. [9] carried out this algorithm's first process mining application. In this process mining context, a precise definition of causality is offered, in contrast to most other existing applications in the domain.

In summary, there are two fundamental issues with the state-of-the-art applications in the process mining literature. First, causal and correlation analyses can be mixed up far too readily. Classification is frequently used to identify relations between data attributes, yet, they inherently look for correlation. A precise positioning is lacking in terms of defining a cause-effect relationship. Second, different strategies have varying degrees of intuitiveness. Machine learning models may be accurate, but interpretability is often crucial for businesses to trust an analysis technique.

3. Causality and Probabilistic Temporal Logic

To fully understand the design choices made for the AITIA-PM algorithm, one must be up to date on the definitions of causality and temporal logic as used in our algorithm. This section provides this essential background information.

3.1. Defining Causality

Throughout philosophy, much has been discussed as possible definitions of causality. One proposal of causality comes from Hume [31]: causality is a regular connection so that C causes E if an event of type E follows every event of type C. Though this is a reasonable definition, counterexamples can be made against it. For example, "*day causes night*" (or vice versa) fits the proposal, yet we would not call day to be a cause of night in reality. This is merely a natural process. On the other hand, multiple instances of C can lead to one observation of the effect. This raises the question if each instance of C is still required.

One main strength of this proposed theory is that it allows verification based on empirical data. More recent theories of causality add a probabilistic view, claiming that a (positive) cause raises the probability of their effects, e.g., Suppes [32], Eells [33]. As such, this work follows the definition of causality as described in Definition 1. For more information, the interested reader is referred to [10].

Definition 1. Two properties must hold to establish a causal relationship: (i) the cause must precede the effect in time (in line with the philosophy of Hume [31]), and (ii) a cause must raise the probability of the effect (in line with Suppes [32]). The second property is also known as the prima facie condition. The prima facie condition is critical when working on sample data.

A pitfall to consider is that in theory, not each causal relationship (significantly) increases the probability of the effect in reality. For example, a webshop is observing an overwhelming amount of orders (O), which seems to demotivate their employees, as their productivity (P) has dropped significantly. However, the reality is that the holiday break (H) is approaching. If the information about the holidays were missing, one would conclude from the data that getting more orders would cause the decrease of productivity of the employees. When considering the holidays, that probability increase would no longer be tied to the increasing number of orders the company got. In other words, there might be a rise in the probability of the effect P without it being caused by O. This is the concept of *spurious causal relations*, as a third factor H was at play. Fig. 1 illustrates this concept.

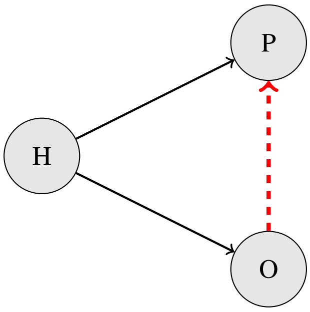

Figure 1: When keeping the common cause fixed, the causal relationship between the increasing number of orders arriving, leading to less productive employees, would disappear.

Summarized, a spurious causal relation is a cause-effect relation based on observations frequently made together. However, it is not an actual cause in reality. *Screening off* is thus required to identify those spurious causal relations so that the output of a causal analysis algorithm reflects reality as closely as possible. This pitfall thus illustrates the difference between correlation and causation. A correlation analysis finds associations between data points and represents them in terms of a number showing both the direction and strength of the relationship. Two *attributes* can frequently move in the same direction, but that does not necessarily imply that one causes the other to move along with it, as the previous example showed.

The second point of interest is the matter around causal chains. In contrast to spurious relations, where two effects have a common cause, a causal chain describes how one phenomenon C is caused by another B. Yet, the cause of B could be identified as phenomenon A. In this case, as shown in Fig. 2 where $A \to B \to C$, one must ask himself whether to identify B as the cause of C or claim A as the actual (root) cause. AITIA-PM performs a causal analysis; if the analyst wishes to perform a *root* cause analysis, they would have to repeat the procedure.

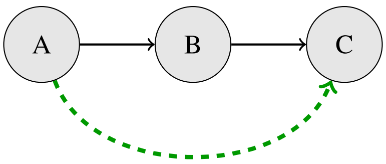

Figure 2: In case of a *causal chain*, it is uncertain how the real cause of C should be defined. One can argue that B is the true cause of C given the direct link, however, when B is caused by A, A is the indirect cause of C.

3.2. Probabilistic Temporal Logic

PTL is a flexible language that allows the definition of causal relationships [34]. More specifically, it allows reasoning on the likelihood of an event within a specific time interval. As such, properties should not hold *eventually*, as they are bound in time, so it can be quantified how likely it is to happen. By allowing to freely define the system states and the kinds of relations between them, PTL is highly flexible in execution.

PTL is based on computational tree logic (CTL) [35]. CTL allows expressing properties such as *"for all paths, the sky is blue"*. The sky being blue is a state on the path. In other words, CTL expressions can describe how system states are related throughout a path.

Definition 2. A **state formula** is a CTL expression that holds true in a specific state [10, 35]. We propose that the *state of the case* can be defined as the sequence of performed activities of the mentioned case, accompanied by case and event attributes that describe these activities. This definition allows for an evaluation of the case state at any given time. Standard logical operators, i.e. the AND operator $\wedge$, are utilized to express a state formula. For instance, to determine if activity A_i was executed by resource R_j, we evaluate $(A_i \wedge R_j)$.

In process mining, each execution of a case can be considered a path. As such, expressions such as *"for each case, the order value is more than €50"* can be evaluated where the order value of €50 is a state formula. However, perhaps the analyst is interested in evaluating an expression under more specific circumstances, like *"when an order is created (O), signing (S) the order must be the next step"*. Temporal operators define the degree to which specific properties must hold throughout a case's execution.

Definition 3. The **temporal operators** describe where along the path the properties must hold [10].

- F, *finally*, at some state on the path, the property will hold.
- G, *globally*, the property will hold along the entire path.
- X, *next*, the property will hold at the next state of the path.
- U, *until*, for two properties, the first holds at every state along the path until at some state the second property holds.
- W, *weak until* (also called unless or release), for two properties, the first holds at every state along the path until a state where the second property holds, with no guarantee that the second property will ever hold (in which case the first must remain forever true).

The expressions created with these temporal operators are evaluated throughout the execution of a case in the event log. Such an expression is then called a **path formula**.

To retake the example of signing S the order O after creating it, this is expressed with the path formula OXS: to track activity O's occurrence to check if activity S directly follows it. Every time this pattern is observed for a case, the expression evaluates to true. However, evaluating expressions describing a state at a specific moment may also be interesting.

PTL, also called probabilistic computational tree logic (PCTL), expands on the CTL formulas by adding a specific time dimension during computations. PTL makes the formulas more specific than "a case exists where a property will hold". PTL expands it to "these are the time bounds when the property must hold for said case". The time bounds allow the computation of probabilities of a state occurring. In other words, PTL enables, by including time dimension, the description and evaluation of expressions such as $(i\,\mathrm{U}^{\leq 14}\, p) \vee r$: a customer receives an invoice (i) and pays the bill (p) within 14 days, or the customer gets a reminder (r).

To conclude, Hansson and Jonsson [36] introduced the *leads-to* operator.

Definition 4. The **leads-to operator** indicates a sequence of events exists, starting from f, leading to g, taking at most time t:

$$f \leadsto^{\leq t} g \equiv [f \rightarrow \mathrm{F}^{\leq t}\ g]$$

where f and g are state formulas, in accordance with [36].

This leads-to operator represents the expectation of one thing leading to another or that there is a series of events connecting one thing to another. This leads-to formula forms the basis of the AITIA-PM algorithm as each causal hypothesis will be expressed in the form of $c \leadsto^{\leq r, \geq s} e$: a logical formula describing the time r by when a potential cause c should trigger an effect e, or describing the time s by which a case c should trigger the effect e if the occurrence of c took longer than s [9]. There are several points to note when using PTL in AITIA-PM specifically: (i) in case the user does not want to restrict the hypothesis in terms of time, the parameter t may be dropped; (ii) the cause and effect can be represented by both state and path formulas and represent the information to extract from the event log; (iii) the probabilities are computed on the data and, as such, are not needed to be provided by the user; and finally (iv) , AITIA-PM makes use of PTL templates. As such, each part of the hypothesis could also be defined by a set of values. For example, given the PTL template $A_i \, \mathrm{U}^{\leq T} A_j \leadsto e$ where $A_i = \{a_1\}$, $A_j = \{a_2, a_3\}$ and $T = \{1, 2, 3\}$, six hypotheses would be extracted: one for each combination of values of A_i, A_j, and T. This enables the business user to quickly define a large number of hypotheses at once.

The primary purpose of PTL in the AITIA-PM algorithm is to serve as an easy-to-use language to represent causal relations. The formulas also allow for testing, including antecedents, consequents, and time bounds. More specifically for process mining, where AITIA-PM operates, PTL also means a straightforward way to tap into the knowledge in event logs. In other words, case attributes, event attributes, activity order, etc., can be exploited to form the hypothesis the business user is genuinely interested in, as PTL provides great freedom. The algorithm design is explained in Section 4, which will clarify the actual tapping into the event log knowledge.

The above has described PTL as required for fully understanding this paper. The interested reader is referred to [10] for additional details about PTL.

4. AITIA-PM – Algorithm Design

The AITIA-PM method guides the data-driven discovery of meaningful causes. In summary, AITIA-PM consists of four different steps, given an event log is loaded: (i) defining the search space, or in other words, defining the antecedents and consequents of the causal hypotheses, (ii) determining which hypotheses are prima facie causes, (iii) computing the causal measure for all prima facie causes, and finally (iv) separating the actual causal relations from the spurious ones utilizing a statistical test. Algorithm 1 summarizes the workflow of the AITIA-PM algorithm and shows how the input event log and domain knowledge is transformed into the genuine causes of the given effect.

***Step 1** – Defining the Search Space*

The search space consists of both the antecedent and consequent of a causal hypothesis as described in Section 3. The domain expert is free in how they are defined. The antecedents are what the domain expert deems as possible causes for the effect. The effect, in turn, is the consequent of a causal relationship. Both antecedents and consequents must be inferrable from the event log. As is in line with PTL, each antecedent and consequent can consist of multiple data points in the event log while giving special attention to the time dimension. In other words, they can be composed of state and path formulas. By joining the formulas of the antecedent and consequent with the leads-to operator, one obtains the causal relation that is ready for testing in step 2.

Recall that each relation describes a *template* of a causal hypothesis. For example, the relation dictates that the combination of activities (A) and the resources (R) which executed them (antecedent) leads to case delay (D): $(A \wedge R) \leadsto D$. A and R represent the sets of all activities and resources as observed in the event log. Assume that both sets have a length of 10. The algorithm would extract all possible combinations in the background, and the analyst would end up with 100 hypotheses based on that one template.

***Step 2** – Testing for Prima Facie Causes*

The hypotheses generated in defining the search space contain all combinations of cause-effect the analyst is interested in. However, they also describe relations that might not meet the prima facie condition. In other words, the hypothesis might not describe a *causal* relation in the sample. For a potential cause to be a prima facie cause of the given effect, it must satisfy the following two conditions:

1. the cause must have occurred before the effect,
2. the cause must increase the probability of the effect occurring.

Algorithm 1: The AITIA-PM algorithm.

Input : The event log E,
P, a set of PTL templates provided by a Domain Expert
Output : A set of genuine cause-effect relations

HypothesesSet $\leftarrow \emptyset$;
PrimaFacieCauses $\leftarrow \emptyset$;
EpsilonAvgSet $\leftarrow \emptyset$;
SignificantCauses $\leftarrow \emptyset$;
`/* Step 1 - Defining the Search Space */`
foreach $p \in P$ **do**
  H $\leftarrow$ `ExtractHypotheses(`p`)`;
  $\forall h \in H$: add h to HypothesesSet ; `/* Each hypothesis consists of a cause` c `and effect` e `*/`
end
`/* Step 2 - Testing for Prima Facie Causes */`
foreach $(c, e) \in$ HypothesesSet **do**
  $p_1 = P(e|c)$;
  $p_2 = P(e)$;
  if $p_1 > p_2$ **then**
    add (c, e) to PrimaFacieCauses;
  end
end
`/* Step 3 - Compute Epsilon Values */`
foreach $(c, e) \in$ PrimaFacieCauses **do**
  $\forall x \in C \setminus \{c\}$: Compute ϵ_x;
  Compute ϵ_{avg} and add to EpsilonAvgSet;
end
`/* Step 4 - Determine Causal Significance */`
foreach $\epsilon_{avg} \in$ EpsilonAvgSet **do**
  Compute z-statistic;
  Compute p-value;
end
Adjust p-values to q-values;
Set α to desired value;
foreach $q \in$ *q-values* **do**
  if $q \leq \alpha$ **then**
    Add related (c, e) to SignificantCauses;
  end
end
return SignificantCauses

These conditions identify the hypotheses which could represent an actual causal relation. However, note that so far, no filtering on spurious relations has been performed. It is relatively straightforward to compute whether or not a cause is a prima facie cause for an effect from the event log. Only the hypotheses fulfilling the above requirements are considered *potential* genuine causes for the effect.

The following details are needed in order to perform this prima facie test: (i) when and for which cases was the cause (c) observed, (ii) when and for which cases was the effect (e) observed, and (iii) how often did the effect occur after the cause given the observations belong to the same case. Then, using the event log L, the prima facie condition is probabilistically verified as follows:

$$P(e|c) > P(e) \tag{1}$$

where

$$P(e) = \frac{|L_e|}{|L|} \tag{2}$$

and

$$P(e|c) = \frac{|L_{cFe}|}{|L_c|} \tag{3}$$

where $|L_c|$, $|L_e|$, and $|L|$ represent the number of traces the cause was observed in the event log, the number of traces the effect was observed in the event log, and the total amount of traces in the event log, respectively. The computation $|L_{cFe}|$ takes the case ID into account, as each case is considered a path. This computation, therefore, checks if there exists a c before e within the same case and if not, the hypothesis is automatically classified as false. For example, resource R_y is only involved after the case has already produced error E. As such, $P(E|R_y) = 0$, meaning that R_y cannot be a prima facie cause of E. More specifically, this count is performed as follows: for every trace, it is checked if the cause was observed before the effect, no matter the frequencies of the cause or effect. For example, $|L_{(cFe)}|$ for the trace $\langle c, c, e, e\rangle$ would return 1.

Step 3 *– Calculation of Epsilon Values*

Having determined all prima facie causes of the effect of interest, the genuine causes are now to be separated from the spurious ones. To that end, we use epsilon values to measure causality that can be statistically tested. When a cause c is present, the measure ϵ_{avg}, introduced by Kleinberg [10], estimates the average change in the probability of an effect e while maintaining a different prima facie cause x constant. In other words, by keeping the other prima facie causes constant, screening off is performed. The resulting value $\epsilon_x(c, e)$ provides a clear interpretation. Assume that $\epsilon_x(c, e) = 0.50$. This is interpreted as "screening off for the other potential cause x, the probability increase of the effect e caused by cause c equals 50 percentage points". Similarly, for a negative value, one would interpret the value as a probability decrease of the effect e given the cause x when screening off. As such, for each other cause x, an ϵ_x is calculated, after which the average describes the general impact of c on e.

Formally, given the set C of potential causes and a hypothesis containing a potential cause c and the effect e, screening off for another potential cause $x \in C\backslash\{c\}$ is performed as follows:

$$\epsilon_x(c, e) = P(e|c \wedge x) - P(e|\neg c \wedge x). \tag{4}$$

where

$$P(e|c \wedge x) = \frac{|L_{(c\wedge x)Fe}|}{|L_{c\wedge x}|} \tag{5}$$

and

$$P(e|\neg c \wedge x) = \frac{|L_{(\neg c\wedge x)Fe}|}{|L_{\neg c\wedge x}|} \tag{6}$$

Similar to Eq. (3), the temporal operator F ensures that observations of c and e are only counted when observed within the same case. $|L_{c\wedge x}$ and $|L_{\neg c\wedge x}|$ count the amount of cases both c and x were observed in and the amount of cases c was not observed but x was, respectively.

Screening off is performed for every $x \in C\backslash\{c\}$ to compute the actual impact c has on the probability of e occurring. These individual epsilon values are then averaged over the entire set of potential causes as shown in Eq. (7).

$$\epsilon_{avg}(c, e) = \frac{\sum_{x\in C\backslash\{c\}} \epsilon_x(c, e)}{|C\backslash\{c\}|}. \tag{7}$$

As is the case for $\epsilon_x(c, e)$, $\epsilon_{avg}(c, e)$ also has a clear interpretation. For the average value, a check is performed for all confounding factors of c. As such, the interpretation is as follows: the average change in the probability of the effect occurring when the potential cause is observed while controlling for alternative causal explanations equals $\epsilon_{avg}(c, e)$ percentage points.

Step 4 *– Determining Causal Significance*

Given the epsilon values, expressing the average probability changes of the effect e occurring given the presence or absence of a prima facie cause c, the real causes and the spurious ones can be distinguished using a statistical method. The AITIA-PM algorithm does this by utilizing FDR. The FDR is the proportion of erroneously falsely rejected null hypotheses in a multiple hypothesis testing problem [37].

Several methods to compute the FDR exist in the literature base [38]. Our previous method [9] employed the FDR controlling technique as described in Efron [39]. Although giving insights into false discoveries, the main limitation is the need for an adequately large set of hypotheses undergoing testing. As such, in a real-world setting where a business user might be interested in only testing a few hypotheses, the method of Efron [39] might fail. Therefore, this work incorporates q-values [12, 13], a Bayesian analog to the p-value, as a replacement to identify the true causes among the spurious ones. In summary, the q-value is a measure of the strength of an observed statistic with respect to the positive FDR [12, 13]. The positive FDR describes the FDR values conditioned on at least one positive finding having occurred [13]. As q-values are Bayesian posterior p-values, they can be computed from the original statistics or their p-values.

Determining the causal significance is then done as follows:

- Compute the z-statistic based on the set of ϵ_{avg}: $z = (\epsilon_{avg} - \mu)/\sigma$ where μ and σ represent the mean and the standard deviation of the set of ϵ_{avg}, respectively;
- Compute the p-value for each z, calculated for an upper-tailed test;
- Compute q-values given the set of p-values;
- Set the significance level to the desired level and separate genuine causes from spurious causes.

The interested reader is referred to the work of Storey [13] for a full explanation of the workings of q-values, and Korthauer et al. [38] for an in-depth analysis of the performance of different state-of-the-art FDR-controlling techniques.

5. Case Studies

In this section, a case study is executed on artificially generated data in Section 5.1 to demonstrate the validity of the AITIA-PM algorithm, followed by a demonstration on real-life data in Section 5.2. Next, Section 5.3 provides insights into the computational complexity of the algorithm. Finally, the artificial data is also fed to other state-of-the-art techniques to compare their capabilities with AITIA-PM in Section 5.4.

The source code of AITIA-PM is publicly available via GitHub[1]. In terms of implementation, steps 1 – 3 are fully implemented in Python, while step 4 is implemented in R[2].

5.1. Case Study 1 – Artificial Data

In order to show that the output of AITIA-PM is reliable, or in other words, that the algorithm is capable of extracting genuine causal relations from data, an artificial dataset has been generated. This allows control over the causes in the business process and, as such, it also ensures the expected output can be known in advance. This is done through the means of business process simulation (BPS). BPS is a technique to model the behavior of a business process and then emulate that behavior through the use of Discrete Event Simulation (DES) [40, 41]. For the implementation of the simulation model, the Arena software is chosen[3] [42].

The simulated process represents a simplified customer complaint-handling process as visualised in Fig. 3. A total of 10.000 traces are simulated, of which Table 1 shows the different activities, their frequencies and the resource class that performs the activity. The activity '2nd Opinion Initial Assessment' is an optional activity executed in each trace with a probability of 90%. There are no case or event attributes present. In total, there are 15 clerks and 10 analysts in the system.

The ground truth is imputed in this business process as follows. The effect under investigation are cases where the activity 'Unresolved Complaint' is performed, which is an indication that the complaint was not handled correctly. The causes inserted in the simulation are the following:

- The activity '2nd Opinion Initial Assessment' was not performed;
- The communication of the initial assessment to the customer was not done within 24 hours of starting the case;

[1] `https://github.com/gregvanhoudt/AITIA-PM`
[2] `https://github.com/StoreyLab/qvalue`
[3] `https://www.rockwellautomation.com/en-us/products/software/arena-simulation.html`

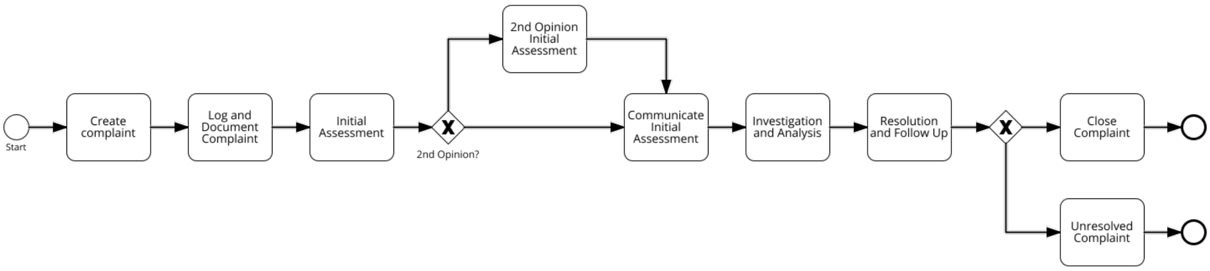

Figure 3: The simplified customer complaint handling process used for BPS.

Activity	Frequency	Resource
Create Complaint	10.000	Automated
Log and Document Complaint	10.000	Clerk
Initial Assessment	10.000	Clerk
2nd Opinion Initial Assessment	9.030	Analyst
Communicate Initial Assessment	10.000	Clerk
Investigation and Analysis	10.000	Analyst
Resolution and Follow Up	10.000	Analyst
Unresolved Complaint	2.102	Clerk
Close Complaint	7.898	Clerk

Table 1: An overview of the activities and their frequencies in the simulated event log.

- Analysts 1 and 2 cannot work well with analysts 9 and 10. If either Analyst 1 or 2 performs ‘Investigation and Analysis’ and either Analyst 9 or 10 subsequently performs ‘Resolution and Follow Up’, the customer would likely not be happy.

The existence of such cause-effect relations in a trace is not a guarantee for the failure of the complaint handling in said trace. If any of the aforementioned causes are present in a trace, there is a 75% probability of the complaint remaining unresolved. If none of these causes are present, on the other hand, then the case always results in success.

The first task is to define the search space. Given the ground truth mentioned above, the search space can be defined to include said cause-effect relations as well as other confounding factors. The PTL statements provided to define the search space are as follows:

- $\mathrm{G}(\neg A_i) \rightsquigarrow e$ where $A_i = \{\text{‘2nd Opinion Initial Assessment’}\}$;
- $\mathrm{F}^{\geq T} A_i \rightsquigarrow e$ where $A_i = \{\text{‘Communicate Initial Assessment’}\}$ and $T = \{12, 24, 48\}$. The lower bounds 12 and 48 are included to introduce confounding factors;
- $(A_i \wedge R_a)\,\mathrm{U}(A_j \wedge R_a) \rightsquigarrow e$ where $A_i = \{\text{‘Investigation and Analysis’}\}$, $A_j = \{\text{‘Resolution and Follow Up’}\}$, and $R_a = \bigcup_{i=1}^{10} \text{Analyst}_i$. All analyst resources are included to introduce confounding factors;
- For additional confounding, we check for every activity-resource combination: $(A \wedge R) \rightsquigarrow e$ where A represents the complete set of activities and R is the complete set of resources in the simulation model.

These templates return 194 unique hypotheses of causal effects observed in total. Next, these hypotheses must be tested for the prima facie conditions to filter out the potential causes which do not increase the probability of the effect in general. From those 194 hypotheses, 52 are deemed prima facie causes of the effect. To recap, a prima facie cause is a cause that increases the probability of the effect occurring. However, at this stage, no controlling for confounding factors is done. Confounding is taken into account during the next step, computing epsilon values. Remember that each ϵ_{avg} quantifies the average impact of each potential cause on the probability of the effect, controlling for confounding variables. The set of ϵ_{avg} also forms the basis of determining causal significance through FDR controlling in accordance to step 4 of the method.

Table 2 shows the output obtained from the algorithm. As expected, the genuine causes put into the simulation model fill the top six of the results. As such, one can be sure AITIA-PM will detect and present the genuine causes to the business user. Furthermore, only the q-values of the six genuine causes are not equal to one. Strictly speaking, only 'Communicate Initial Assessment not observed within 24 hours' and '2nd Opinion Initial Assessment not observed' are genuine causes according to the algorithm when using a significance level $\alpha < 83.7\%$. This can be explained by their much larger epsilon values, mainly. To give a concrete interpretation of the first causal effect found in the data: when the communication of the initial assessment is not done within 24 hours, the average increase in the probability of the customer being unhappy and the complaint remaining unresolved is 62.73 percentage points. To conclude this experiment, the results show AITIA-PM is capable of separating the genuine causes and the confounding factors as spurious causes.

cause	ϵ_{avg}	q
Communicate Initial Assessment not observed within 24 hours	0.6273	0.0001
2nd Opinion Initial Assessment not observed	0.5066	0.0040
Investigation and Analysis by Analyst 1 - Resolution and Follow Up by Analyst 10	0.2391	0.8370
Investigation and Analysis by Analyst 2 - Resolution and Follow Up by Analyst 9	0.2360	0.8370
Investigation and Analysis by Analyst 2 - Resolution and Follow Up by Analyst 10	0.2348	0.8370
Investigation and Analysis by Analyst 1 - Resolution and Follow Up by Analyst 9	0.2330	0.8370
Resolution and Follow Up - Analyst 10	0.0886	1.0000
Investigation and Analysis by Analyst 9 - Resolution and Follow Up by Analyst 10	0.0880	1.0000
Investigation and Analysis - Analyst 2	0.0767	1.0000
Investigation and Analysis - Analyst 1	0.0666	1.0000
Initial Assessment - Clerk 1	0.0639	1.0000
Initial Assessment - Clerk 1	0.0639	1.0000
Communicate Initial Assessment - Clerk 1	0.0639	1.0000
Communicate Initial Assessment - Clerk 1	0.0639	1.0000
Resolution and Follow Up - Analyst 9	0.0581	1.0000

Table 2: The top 15 causal hypotheses leading to the effect in the artificial data setting, sorted on decreasing epsilon.

As mentioned before, the domain expert is responsible for defining the search space. The assumption is made that the expert is an all-knowing oracle who does so in a complete fashion. However, in real-life scenarios, it is possible the real causes are omitted from the search space. In such an event, it would be undesirable to get insights that could be false in reality. In other words, it is undesired behavior for AITIA-PM to then detect genuine causes which are not actually genuine.

To that end, the experiment is repeated with one alteration: the six genuine causes are omitted from the search space. As such, the algorithm is executed on all 10.000 traces, using the same PTL statements, but now with 188 causal hypotheses to start with. From those 188 hypotheses, 46 were now deemed prima facie causes. The results of this new experiment are shown in Table 3. The table shows that no genuine causes are detected unless the business user employs a significance level $\alpha > 48.08\%$. As such, one can conclude no true causes are found in the data, which in this setting, is the expected result.

5.2. *Case Study 2 – Road Traffic Fine Management*

In this case study, the applicability of AITIA-PM is demonstrated based on the Road Traffic Fine Management dataset[4]. In contrast to [9], this case study includes more complex hypotheses for the root-cause analysis. However, do note that the algorithm is not limited to said hypotheses.

[4]`https://data.4tu.nl/articles/dataset/Road_Traffic_Fine_Management_Process/12683249`

cause	ϵ_{avg}	q
Resolution and Follow Up - Analyst 10	0.0990	0.4808
Investigation and Analysis by Analyst 9 - Resolution and Follow Up by Analyst 10	0.0882	0.4808
Investigation and Analysis - Analyst 2	0.0873	0.4808
Investigation and Analysis - Analyst 1	0.0768	0.6752
Resolution and Follow Up - Analyst 9	0.0675	0.7194
Initial Assessment - Clerk 1	0.0648	0.7194
Initial Assessment - Clerk 1	0.0648	0.7194
Communicate Initial Assessment - Clerk 1	0.0648	0.7194
Communicate Initial Assessment - Clerk 1	0.0648	0.7194
Investigation and Analysis by Analyst 9 - Resolution and Follow Up by Analyst 6	0.0613	0.7426
Investigation and Analysis by Analyst 3 - Resolution and Follow Up by Analyst 4	0.0502	0.9470
Investigation and Analysis by Analyst 5 - Resolution and Follow Up by Analyst 6	0.0419	0.9470
Initial Assessment - Clerk 12	0.0406	0.9470
Initial Assessment - Clerk 12	0.0406	0.9470
Communicate Initial Assessment - Clerk 12	0.0406	0.9470

Table 3: The top 15 outputs of AITIA-PM when omitting the true causes put into the data through simulation. Data is sorted on descending epsilon values.

The Road Traffic Fine Management dataset is extracted from an Italian police force. The information system supported the management of handling road traffic fines. Some fines are paid on time correctly. For others, legal actions must be taken. In this case study, we are interested in why fines are sent for credit collection (SCC). Since the event log spans cases between January 2000 and June 2013, it seems appropriate only to perform the analysis on recent data. Although data pre-processing is not required for AITIA-PM, log filters are recommended depending on the context of the study. For this study, we filtered for cases occurring between 1 January 2011 and 18 June 2013 (the date of the last event).

First, the domain expert needs to define the search space. Again, AITIA-PM assumes the domain expert is an all-knowing oracle who defines the search space as complete. In other words, no other confounding variables are left out. For this case study, the domain expert is emulated by the authors. The effect has already been decided as the activity "Send for Credit Collection". An analysis of the process in Disco[5] has shown that different sequence flows can be followed, leading to the credit collection activity. For example, not all fines are sent for an appeal to the prefecture. As such, it may be interesting to know what executed activity most likely leads to credit collection. The executing resources are added to this, even though a resource is only registered for the "Create Fine" event. Summarized, the first template is defined as

$$(A \wedge R) \rightsquigarrow SCC$$

where A represents the set of activities, and R represents the set of resources found in the event log. A second theory is that drivers of a certain vehicle class tend not to pay their fines correctly. In PTL, this template is expressed as

$$VC \rightsquigarrow SCC$$

where VC equals the set of vehicle class attribute values. Third, the drivers are not necessarily to be blamed if the fine is sent too late. The analysis in Disco showed that, on average, it takes 55,8 days to send out the fine after creating it. To test this time dimension, an additional template is provided, namely

$$A_{CF}\, \mathrm{U}^{\leq T} A_{SF} \rightsquigarrow SCC$$

where $A_{CF} = \{\text{'Create Fine'}\}$, $A_{SF} = \{\text{'Send Fine'}\}$, and $T = \{30, 60\}$. The set T represents an upper time bound of 30 days and 60 days, respectively. Finally, it seems that, after Insert Fine Notification, three possible paths can follow.

[5] `https://fluxicon.com/disco`

As such, we are interested to know which path of legal action causes credit collection. For that, the following template is defined:

$$A_{IFN} \, \text{X} \, A_p \rightsquigarrow SCC$$

where $A_{IFN} = \{$'Insert Fine Notification'$\}$ and $A_p = \{$'Appeal to Judge', 'Insert Date Appeal to Prefecture', 'Add penalty'$\}$.

All the templates mentioned above are filled out and then tested simultaneously. Given the sets already described, there are 91 unique causal hypotheses. Each of these 91 hypotheses is then tested for the prima facie condition. 80 hypotheses remain, meaning that 80 of the hypotheses' causes increase the probability of the effect, "send for credit collection", occurring. The next step, calculating the epsilon values, quantifies the impact of each potential cause on said probability increase, used to separate the true causes from the spurious ones through the q-value FDR statistic. To recap, the FDR computes the proportion of erroneously falsely rejected null hypotheses. As such, the raw epsilon values ϵ are transformed into q-values in accordance to step 4 of the method.

Table 4 summarizes the results on the Road Traffic Fine Management dataset following the above specifications. Despite the small number of prima facie causes (80), interpreting the q-values allows us to separate genuine discoveries from spurious ones. Results show that there are three genuine causes (with a significance level $\alpha = 5\%$) for a fine to be sent for credit collection, namely (i) that the penalty is added straight after the activity "insert fine notification", (ii) that "insert fine notification" was executed, and finally (iii) that a penalty was added. As such, the true causes of the effect are strongly linked to these two activities. Looking beyond the FDR, other strong candidates were that the fine was sent later than 30 days after being created, and apparently, drivers of vehicles with class A generally declined payment. Again, the epsilon values have a clear interpretation. For the first and primary cause, the interpretation is as follows: when the activity "Add penalty" directly follows the activity "Insert Fine Notification", the probability of the effect "Send for Credit Collection" increases by 69.56 percentage points on average after controlling for confounding variables.

cause	ϵ_{avg}	q
Add penalty directly follows Insert Fine Notification	0.6956	0.0044
Insert Fine Notification	0.6445	0.0059
Add penalty	0.6445	0.0059
Send Fine not observed within 720 hours	0.5409	0.0520
Vehicle Class A	0.5297	0.0529
Send Fine	0.5124	0.0631
Send Fine not observed within 1440 hours	0.4041	0.3830
Vehicle Class C	0.2489	0.9782
Vehicle Class M	0.1796	0.9782
Notify Result Appeal to Offender	0.1508	0.9782
Receive Result Appeal from Prefecture	0.1321	0.9782
Create Fine - 855	0.1261	0.9782
Create Fine - 29	0.1155	0.9782
Create Fine - 46	0.1153	0.9782
Create Fine - 860	0.1048	0.9782

Table 4: Output of the AITIA-PM algorithm as executed on the Road Traffic Fine Management event log. Results are ordered on increasing q-value. Only the first 15 rows out of 80 are shown.

Table 4 also shows the importance of checking for false discoveries. By solely looking at the epsilon values, it is not straightforward to define the boundary of genuine cause versus spurious cause. Even though the fine being sent late still increases the probability of the effect with 54 percentage points on average, giving the impression it has quite a large impact, the test appoints it as spurious given the other factors at play.

5.3. The Driving Factors of Computation Time

Analyzing factors that influence the computation time of an algorithm is an essential step in understanding its performance. Highly relevant drivers of the computation time of the algorithm are (i) the number of traces in the event log, and (ii) the size of the hypotheses set. Controlling for confounding factors, where each cause-effect relation is tested controlling for each other potential cause, combined with a hypothesis being checked on the trace level, introduces some computational complexity to the algorithm. This subsection provides an insight into that computational complexity, more specifically by measuring the computation times under varying numbers of traces (varying from 100 tot 10 000) present in the source data on the one hand, and by altering the size of the search space (varying from 2 to 194) on the other hand. This is presented using the data from the artificial setting in Section 5.1. The computation times are based on an AMD Ryzen 7 PRO 3700U w/ Radeon Vega Mobile Gfx.

Fig. 4 shows the impact of the amount of traces in the event log. The search space was set to the total set of 194 causal hypotheses already discussed in Section 5.1. As such, creating the search space and performing the inference is only affected by the total number of traces. One can observe the relation between the computation time and the number of traces the algorithm is performed on approximates a linear relation. This relationship can be attributed to the fact that each trace is individually checked for meeting a causal hypothesis or not. As such, the more traces, the longer the computation takes.

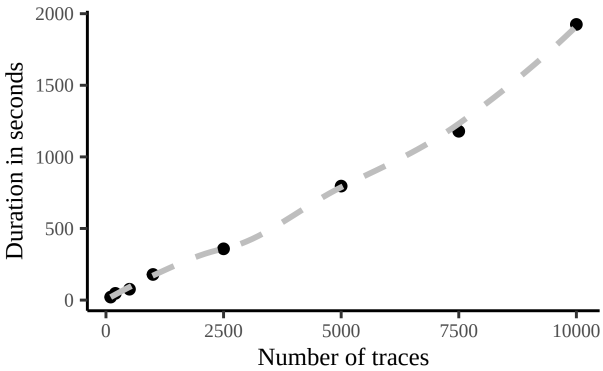

Figure 4: Keeping the size of the search space constant at 194, a linear relation is discovered between the number of traces in the event log and the computation time of AITIA-PM.

Similarly, Fig. 5 shows the relation between the search space size and the computation time. Again, an upwards trend is identified. That can be attributed to the fact that, for each additional hypothesis included, there are additional confounding factors to control for as well. It is worth noting that the complexity of each causal hypothesis impacts the computation time, as different hypotheses may require varying levels of computational resources. For example, detecting a specific activity performed by a given resource is less computationally demanding than detecting a directly-follows relation with time and resource constraints. Consequently, it is expected to observe fluctuations in the upwards trend of computation time as the search space increases.

5.4. An Empirical Comparison with Related Work

In order to show the strengths and novelties of AITIA-PM compared to the related work in the domain, the artificial data will also be fed to comparable causal inference techniques. The starting point in identifying comparable techniques is the collection of applications discussed in Section 2: the machine learning algorithms [6, 14, 15, 16, 17, 18], process discovery technique [25], causal graph techniques [4, 24], subgroup discovery method [19], and counterfactual reasoning methods [22, 23].

Several criteria must be met in order to be included in the comparative study:

- The technique must clearly bring insights into causal effects. As such, techniques returning correlations are not taken into account;
- A clear interpretation of the causal effect is required. The effect of the cause on the effect must be quantified. In other words, conformance scores or common characteristics do not suffice, as they do not measure the magnitude of impact of a causal effect;

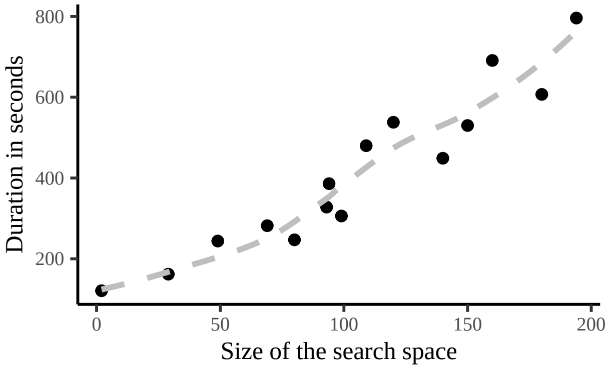

Figure 5: Setting the number of traces to a fixed value of 5000, a linear relation is discovered between the size of the search space, in terms of the number of hypotheses, and the computation time of AITIA-PM.

- The technique must be publicly available.

After taking the above criteria into account, three techniques remain: the causal graph methods of Qafari and van der Aalst [4] and Leemans and Tax [24], and the counterfactual reasoning mechanism of Qafari and van der Aalst [22].

With respect to the counterfactual reasoning mechanism of Qafari and van der Aalst [22], the implementation could only operate with numerical data. Given our artificial setting, which resembles a business process as one would encounter in practice, the data was incompatible with the technique. Consequently, it was no longer taken into account. Regarding the structural equation modeling technique of Qafari and van der Aalst [4], the technique meets all selection criteria, however, no output is generated when used on our artificial event log. Consequently, this lack of output excludes it from the analysis.

The method of Leemans and Tax [24] only requires the event log as input. From the event log, the Visual Miner [43] is applied first to discover a process model. This process model is shown in Fig. 6. Two decision points are identified, between which the technique can discover a dependency. It does so by generating a collection of causal graphs consistent with the observations in the event log. Only two decision points are present in the model, so only one dependency could be identified. More specifically, the dependency between performing or skipping '2nd Opinion Initial Assessment' and the case outcome. Table 5 shows the quantification of this dependency: when the 2nd opinion is skipped in the process, the probability of closing the complaint decreases by 56.8 percentage points. That is also one of the genuine causes imputed in the event log through BPS, and as such, the algorithm has found one genuine cause. However, the other imputed causes are not detected by this algorithm, simply because it is limited to detecting dependencies between decision points in the process.

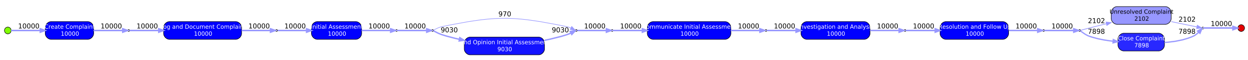

Figure 6: The resulting process model of applying the Visual Miner on the artificially generated data, in which the ground truth is known.

From choice	To choice	Causal strength
2nd Opinion Initial Assessment, [skip 5]	Close Complaint, Unresolved Complaint	-0.568

Table 5: Output of Leemans and Tax [24] performed on the artificial data of the customer complaint process. One dependency was found, which is valid.

6. Discussion

The case study illustrates the ease of setting up and executing the AITIA-PM algorithm and the interpretability of results that business users are looking for. While Granger causality imposes heavy assumptions on the data, our understanding of causality implies that any process mining event log can be used. In contrast to Granger causality, AITIA-PM can deal with confounding variables and non-stationary data. This makes AITIA-PM a very practically feasible solution. Additionally, it is shown that AITIA-PM can also tackle a much richer set of hypotheses than other techniques in the process mining literature.

A more detailed comparison of the potential output sets of different causal inference algorithms in process mining also shows the advancements made by AITIA-PM in the domain. First, the distinction between causal and correlation analysis plays a vital role. Classification techniques and machine learning algorithms, i.e. Bozorgi et al. [14], Ferreira and Vasilyev [15], Böhmer and Rinderle-Ma [18], typically identify correlations between attributes. Although these techniques can theoretically identify a broad range of potential relationships between attributes, it is difficult to claim the presence of a genuine causal relation between said attributes. Techniques producing causal graphs, i.e. [4, 24], do follow a notion of causality theory. However, they are typically more limited in the variety of causal relations they can discover. Typically, causal graphs can model the causal relations between control-flow decision points [24] and process features such as activities performed, case attributes, and event attributes [4]. AITIA-PM can go beyond investigating dependencies between attributes, as different (time) dependencies can be chained together to form a very complex causal hypothesis, which is one of the main strengths of our technique.

Second, AITIA-PM benefits from domain knowledge, similar to structural equation models [4, 22], for example. Without this domain knowledge, however, the authors of [4, 22] acknowledge that using machine learning has downsides. For example, incorrect recommendations may be obtained. Given the probabilistic approach AITIA-PM utilizes in combination with the FDR statistic, the odds of identifying wrong causes are minimized while remaining intuitive. This test for significant causes is also an advantage over mining causal graphs. It must be acknowledged that if the search space is not defined in full or even defined incorrectly, the algorithm cannot identify the true causes at all. However, it is shown that AITIA-PM does not return spurious causes as genuine in case the true causes are omitted from the search space. Naturally, this issue of omitting real causes can be solved by using all information stored in the event log as the search space, but again, this could be potentially computationally infeasible depending on the size of the event log and the size of the search space.

When comparing our technique with counterfactual reasoning algorithms, i.e. [22], we note that these methods are capable of answering what-if questions on the case level. While AITIA-PM provides the causal relations and the impact of the case on the probability of the effect occurring, this insight originates from the log level. Given the clear interpretation, the insights are also actionable: one knows which issue to tackle to halt the effect from occurring, similar to how counterfactual reasoning can answer the what-if question.

As such, AITIA-PM is highly flexible in finding different types of causes of all kinds of complexity through the use of PTL. The domain expert can steer the inference process by indicating the assumed causal relations along with confounding factors. Because of the use of PTL and the varying complexity of PTL statements, the potential search space can be quite vast.

Finally, it should be remembered that event logs carry a case notion, yet different process instances can influence each other. For example, one case can be put on hold because scarce resources are working on other cases. AITIA-PM assumes that each case is defined by the events that can cause the effect within that specific case. However, one must remember that these influences exist. How to handle those influences in such a probabilistic approach requires further research.

7. Conclusion

This paper extends previous work [9] by updating the causal analysis method in process mining named AITIA-PM. The update's contributions mainly lie in (i) including more explicit support for PTL formalisms, (ii) including a new method to compute the causal significance of cause-effect relations through the use of q-values, making the algorithm more applicable in real-world scenarios, and (iii) evaluating the algorithm on synthetic data in addition to merely demonstrating it as was the cause in [9]. AITIA-PM, as such, complements state-of-the-art causal analysis techniques as it follows philosophical causality theory and can evaluate causal effects from varying levels of complexity. In contrast to other well-known systems, AITIA-PM imposes reasonable assumptions about the necessary data, making it very adaptable to the needs of a business user and highly practical in actual scenarios. The technique also effectively addresses confounding factors that can lead to misleading connections by adopting a probabilistic approach and averaging out the probability changes, with the added benefit of being incredibly interpretable, as seen in the case study. As a result, AITIA-PM continues to be an industry standard for process mining causal analysis.

Several future research challenges are identified in this article. First, a domain expert must define the hypotheses of interest the algorithm will check. Automatic hypothesis generation could bring insights the domain expert might need to consider, or even provide the true causal relations the domain expert would discard himself. Second, state formulas in their current form are binary as they evaluate to true or false. Future work could bring an extension that supports continuous variables. Third, additional research must be performed on the influences different cases can have on each other.

Acknowledgment

The authors would like to thank prof. dr. S. J. J. Leemans (RWTH Aachen) to provide us with the results of the causal graph mining technique [24] and M. S. Qafari to provide a look into the source code of the counterfactual reasoning method discussed in [22]. This research was funded by the UHasselt BOF under grant number BOF19OWB19.

References

[1] W. M. P. van der Aalst, Process Mining: Data Science in Action, Springer, Berlin, Heidelberg, 2016.
[2] W. M. P. van der Aalst, T. Weijters, L. Maruster, Workflow mining: discovering process models from event logs, IEEE Transactions on Knowledge and Data Engineering 16 (2004) 1128–1142.
[3] J. Carmona, B. van Dongen, A. Solti, M. Weidlich, Conformance checking, Springer International Publishing, 2018.
[4] M. S. Qafari, W. van der Aalst, Root Cause Analysis in Process Mining Using Structural Equation Models, in: International Conference on Business Process Management, Springer, 2020, pp. 155–167.
[5] B. F. A. Hompes, A. Maaradji, M. La Rosa, M. Dumas, J. C. A. M. Buijs, W. M. P. van der Aalst, Discovering Causal Factors Explaining Business Process Performance Variation, in: E. Dubois, K. Pohl (Eds.), Advanced Information Systems Engineering, volume 10253, Springer International Publishing, 2017, pp. 177–192. Series Title: Lecture Notes in Computer Science.
[6] N. Gupta, K. Anand, A. Sureka, Pariket: Mining business process logs for root cause analysis of anomalous incidents, in: International Workshop on Databases in Networked Information Systems, Springer, 2015, pp. 244–263.
[7] P. Delias, A. Lagopoulos, G. Tsoumakas, D. Grigori, Using multi-target feature evaluation to discover factors that affect business process behavior, Computers in Industry 99 (2018) 253–261.
[8] C. W. Granger, Some recent development in a concept of causality, Journal of econometrics 39 (1988) 199–211.
[9] G. Van Houdt, B. Depaire, N. Martin, Root cause analysis in process mining with probabilistic temporal logic, in: J. Munoz-Gama, X. Lu (Eds.), Process Mining Workshops, Springer International Publishing, Cham, 2022, pp. 73–84.
[10] S. Kleinberg, Causality, Probability, and Time, Cambridge University Press, 2012.
[11] S. Kleinberg, P. N. Kolm, B. Mishra, Investigating causal relationships in stock returns with temporal logic based methods, 2010. arXiv:1006.1791.
[12] J. D. Storey, A direct approach to false discovery rates, Journal of the Royal Statistical Society: Series B (Statistical Methodology) 64 (2002) 479–498.
[13] J. D. Storey, The positive false discovery rate: a Bayesian interpretation and the q-value, The Annals of Statistics 31 (2003).
[14] Z. D. Bozorgi, I. Teinemaa, M. Dumas, M. La Rosa, A. Polyvyanyy, Process mining meets causal machine learning: Discovering causal rules from event logs, in: 2020 2nd International Conference on Process Mining (ICPM), 2020, pp. 129–136.
[15] D. R. Ferreira, E. Vasilyev, Using logical decision trees to discover the cause of process delays from event logs, Computers in Industry 70 (2015) 194–207.
[16] S. Suriadi, C. Ouyang, W. M. van der Aalst, A. H. ter Hofstede, Root cause analysis with enriched process logs, in: International Conference on Business Process Management, Springer, 2012, pp. 174–186.
[17] E. Vasilyev, D. R. Ferreira, J. Iijima, Using inductive reasoning to find the cause of process delays, in: 2013 IEEE 15th Conference on Business Informatics, IEEE, 2013, pp. 242–249.
[18] K. Böhmer, S. Rinderle-Ma, Mining association rules for anomaly detection in dynamic process runtime behavior and explaining the root cause to users, Information Systems 90 (2020). Advances in Information Systems Engineering Best Papers of CAiSE 2018.
[19] M. F. Sani, W. van der Aalst, A. Bolt, J. García-Algarra, Subgroup discovery in process mining, in: International Conference on Business Information Systems, Springer, 2017, pp. 237–252.
[20] M. Maziarz, A review of the granger-causality fallacy, The journal of philosophical economics: Reflections on economic and social issues 8 (2015) 86–105.
[21] J. J. Koorn, X. Lu, H. Leopold, N. Martin, S. Verboven, H. A. Reijers, Mining statistical relations for better decision making in healthcare processes, in: 3rd International Conference on Process Mining (ICPM) (In Press), 2022.

[22] M. S. Qafari, W. M. van der Aalst, Case level counterfactual reasoning in process mining, in: International Conference on Advanced Information Systems Engineering, Springer, 2021, pp. 55–63.

[23] T. Narendra, P. Agarwal, M. Gupta, S. Dechu, Counterfactual reasoning for process optimization using structural causal models, in: T. Hildebrandt, B. F. van Dongen, M. Röglinger, J. Mendling (Eds.), Business Process Management Forum, Springer International Publishing, 2019, pp. 91–106.

[24] S. J. J. Leemans, N. Tax, Causal Reasoning over Control-Flow Decisions in Process Models, in: X. Franch, G. Poels, F. Gailly, M. Snoeck (Eds.), Advanced Information Systems Engineering, volume 13295, Springer International Publishing, Cham, 2022, pp. 183–200.

[25] J. Maeyens, A. Vorstermans, M. Verbeke, Process mining on machine event logs for profiling abnormal behaviour and root cause analysis, Annals of Telecommunications 75 (2020) 563–572.

[26] S. Kleinberg, G. Hripcsak, A review of causal inference for biomedical informatics, Journal of Biomedical Informatics 44 (2011) 1102–1112.

[27] D. B. Rubin, Estimating causal effects of treatments in randomized and nonrandomized studies., Journal of educational Psychology 66 (1974) 688.

[28] D. B. Rubin, Causal inference using potential outcomes: Design, modeling, decisions, Journal of the American Statistical Association 100 (2005) 322–331.

[29] J. Pearl, Graphical models for probabilistic and causal reasoning, Quantified representation of uncertainty and imprecision (1998) 367–389.

[30] J. Pearl, Causality: Models, Reasoning and Inference, 2nd ed ed., Cambridge University Press, Cambridge, 2009.

[31] D. Hume, A Treatise of Human Nature, 1739.

[32] P. Suppes, A probabilistic theory of causality, British Journal for the Philosophy of Science 24 (1973).

[33] E. Eells, Probabilistic causality, volume 1, Cambridge University Press, 1991.

[34] F. Ciesinski, M. Größer, On probabilistic computation tree logic, in: Validation of Stochastic Systems, Springer, 2004, pp. 147–188.

[35] E. M. Clarke, O. Grumberg, D. A. Peled, Model checking, The MIT Press, Cambridge, MA (1999).

[36] H. Hansson, B. Jonsson, A logic for reasoning about time and reliability, Formal Aspects of Computing 6 (1994).

[37] Y. Benjamini, Y. Hochberg, Controlling the false discovery rate: a practical and powerful approach to multiple testing, Journal of the Royal statistical society: series B 57 (1995) 289–300.

[38] K. Korthauer, P. K. Kimes, C. Duvallet, A. Reyes, A. Subramanian, M. Teng, C. Shukla, E. J. Alm, S. C. Hicks, A practical guide to methods controlling false discoveries in computational biology, Genome Biology 20 (2019) 118.

[39] B. Efron, Large-scale simultaneous hypothesis testing: the choice of a null hypothesis, Journal of the American Statistical Association 99 (2004) 96–104.

[40] N. Melão, M. Pidd, Use of business process simulation: A survey of practitioners, Journal of the Operational Research Society 54 (2003) 2–10. doi:10.1057/palgrave.jors.2601477.

[41] J. Banks, J. S. C. II, B. L. Nelson, D. M. Nicol, Discrete-Event System Simulation, 5th Edition, Pearson Education, 2014.

[42] T. Altiok, B. Melamed, Simulation modeling and analysis with Arena, Elsevier, 2010.

[43] S. J. Leemans, D. Fahland, W. M. Van Der Aalst, Process and deviation exploration with inductive visual miner., BPM (demos) 1295 (2014).