Discovering Causal Factors Explaining Business Process Performance Variation

Bart F.A. Hompes[1,2(✉)], Abderrahmane Maaradji[3], Marcello La Rosa[3], Marlon Dumas[4], Joos C.A.M. Buijs[1], and Wil M.P. van der Aalst[1]

[1] Eindhoven University of Technology, Eindhoven, The Netherlands
{b.f.a.hompes,j.c.a.m.buijs,w.m.p.v.d.aalst}@tue.nl
[2] Philips Research, Eindhoven, The Netherlands
[3] Queensland University of Technology, Brisbane, Australia
{abderrahmane.maaradji,m.larosa}@qut.edu.au
[4] University of Tartu, Tartu, Estonia
marlon.dumas@ut.ee

Abstract. Business process performance may be affected by a range of factors, such as the volume and characteristics of ongoing cases or the performance and availability of individual resources. Event logs collected by modern information systems provide a wealth of data about the execution of business processes. However, extracting root causes for performance issues from these event logs is a major challenge. Processes may change continuously due to internal and external factors. Moreover, there may be many resources and case attributes influencing performance. This paper introduces a novel approach based on time series analysis to detect cause-effect relations between a range of business process characteristics and process performance indicators. The scalability and practical relevance of the approach has been validated by a case study involving a real-life insurance claims handling process.

Keywords: Process mining · Performance analysis · Root cause analysis

1 Introduction

Improving process performance can lead to significant cost and time savings, and to better service levels (e.g. better response times). Accordingly, process performance analysis and optimization has been an active field of research in recent years [1,2]. Business process performance is generally affected by a plethora of factors. For example, the waiting time for a procedure in a hospital may depend on the amount of scheduled staff; the duration of a credit check in a credit approval process might depend on the number of clients waiting to be approved; the waiting time for a payment receipt might depend on the time of day, etc. It is often not known to process owners which factors affect which performance indicators. Consequently, it is hard to identify the best actions to be taken when performance is unsatisfactory. For instance, when process owners have a limited set of resources available, it is often not known to which tasks these resources

E. Dubois and K. Pohl (Eds.): CAiSE 2017, LNCS 10253, pp. 177–192, 2017.
DOI: 10.1007/978-3-319-59536-8_12

should be allocated in order to redress the performance issues. The latter is especially important in processes that have a high level of variability and do not follow a fixed process model.

Although several techniques have been proposed to automatically discover process performance bottlenecks and deviations based on event data (e.g. [1,2]), little research has gone into the automated discovery of causal factors of business process performance. As a result, a number of hypotheses typically have to be tested manually in order to identify causal factors for process performance issues. Additionally, factors can have both a direct and indirect effect on process performance. For example, factors may influence other factors that in the end influence performance. Hence, new analysis techniques are required that are able to discover such chains of causal relations between causal factors and the performance indicators of interest.

In this paper, we propose a technique that, given an event log of a business process, generates a graph of causal factors explaining process performance. The technique identifies causal relations between a range of business process characteristics and process performance indicators such as case duration (a.k.a. cycle time) and activity waiting time. In order to detect causal relations, we test for Granger causality [3], a statistical test that is widely used for causal analysis of time series in a range of fields, e.g. economics and neuroscience [4,5]. The idea is that values for performance indicators are seen as time series. A factor is said to be causal to another when past values of this factor provide information that can help predict the other factor above and beyond the information contained in the past values of the latter factor alone. This idea is illustrated in Fig. 1.

Given the large number of factors that may affect process performance and their possible combinations, one of the main bottlenecks when extracting a causal graph is to prune down the number of causal relations to be tested. To this end, the paper proposes an approach to prune the space of causal relations in order to identify a manageable subset of candidate causal relations. The proposed approach has been validated via a case study involving an insurance claims handling process at a large Australian insurer.

The remainder of this paper is structured as follows. Section 2 discusses related work. Section 3 introduces preliminary definitions. Our causal discovery approach is detailed in Sect. 4 and validated in Sect. 5. The paper is concluded with views on future work in Sect. 6.

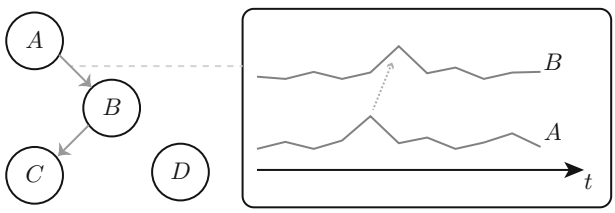

Fig. 1. Factor A causes B which in turn causes C, whereas factor D does not have any observed effect. In order to improve C, A and B should be improved.

2 Related Work

Techniques that exploit process execution data collected by information systems have gained increased interest from both industry and the research community. Model-based techniques such as alignments compare the observed behavior with either a discovered or manually designed process model and can be used for conformance checking as well as performance analysis [1]. An extensive literature review of process measures that can be used in this context can be found in [2].

Other studies addressed the interplay between different perspectives of a business process in order to provide more targeted insights. The method proposed in [6] for example aims at identifying cases that might exceed certain deadlines based on predefined process risk indicators such as activity duration. If a case contains at least one outlier value of the defined indicators, the case is labeled as being at risk. A framework for inferring new event and case attributes is proposed in [7]. Inferred attributes are subsequently used as cause-effect variables in a decision tree classifier in order to discover business rules. A related technique is proposed in [8], where process characteristics are correlated using decision tree learning. This approach is extended in [9] with a technique for recommending business decisions based on risk. The authors of [10] use decision tree learning to find process paths and contexts that lead to improved performance. In [11], the notion of process cubes is proposed. Process cubes are based on the OLAP data cube concept, and define a set of actions and operations that can be used to explore event data based on different business process perspectives. A similar framework is presented in [12].

Though the methods described above have their individual merits and applications, they provide limited insight into what cause-effect relationships might exist in event data. In [13], the authors establish the need for techniques that are able to provide actionable insights, rather than merely showing low-level analytical insights, and provide a framework to aid in this translation. In our work, we hypothesize that causal factors can be discovered for and between performance indicators, and test this hypothesis statistically using an established technique from the time series domain.

Whereas existing techniques focus mainly on finding differences in performance and on general statistics, in this paper, we focus on supporting business process decision making by discovering and providing the causal factors for business process performance. Actionable insights can then be obtained by looking at causal factors. We base our idea on methods proposed in [14] where we proposed a technique that automatically discovers statistically significant differences in performance between different contexts. This paper specifically considers the time dimension, which allows for more elaborate analysis, including the cause-effect relations focused on in this paper.

3 Preliminaries

The executed *events* of multiple *cases* of a *process* are usually recorded by some *information system*. These so-called event logs serve as input for any process

mining technique. Typically, different attribute values are recorded for these events, such as the time they took place, which activity was performed, and which resources were involved. Definitions for universes and event bases used in this paper are based on those in [11].

Definition 1 (Universes). *$\mathcal{U}_V$ is the universe of possible attribute values (e.g. strings, numbers, etc.). $\mathcal{U}_S = \mathcal{P}(\mathcal{U}_V)$ is the universe of value sets*[1]*. $\mathcal{U}_H = \mathcal{P}(\mathcal{U}_S)$ is the universe of value set collections (set of sets), and $\mathcal{T} \subseteq \mathcal{U}_V$ is the universe of time stamps.*

Note that $v \in \mathcal{U}_V$ is a single value (e.g. $v = 10$), $S \in \mathcal{U}_S$ is a set of values (e.g. $S = \{gold, silver, bronze\}$), and $H \in \mathcal{U}_H$ is a collection of sets. For example, $H = \{\{Bob, John\}\{Mary, Sue\}\}$, or $H = \{\{x \in \mathbb{N} \mid x < 12\}, \{x \in \mathbb{N} \mid 12 \leq x < 55\}, \{x \in \mathbb{N} \mid x \geq 55\}\}$. Any $t \in \mathcal{T}$ represents a unique time stamp (e.g. 2016-1-4 9:15). Time stamps can have different levels of granularity (e.g. week, hour, millisecond).

Definition 2 (Event base). *An event base $EB = (E, P, \pi)$ defines a set of events E, a set of event properties P, and a function $\pi \in P \rightarrow (E \nrightarrow \mathcal{U}_V)$. For any property $p \in P$, $\pi(p)$ (denoted π_p) is a partial function mapping events onto values. If $\pi_p(e) = v$, then event $e \in E$ has a property $p \in P$ and the value of this property is $v \in \mathcal{U}_V$. If $e \notin dom(\pi_p)$, then event e does not have property p and we write $\pi_p(e) = \perp$.*

The set E refers to individual events, recorded by some information system. The event base can either consist of a single event log, or, alternatively, multiple event logs can be combined to create an aggregated event base. Note that $e \in E$ is a unique identifier and function π is needed to attach meaning to e. P is the set of properties that events may or may not have. For example, $P = \{$*case, age, type, activity, instance, time, resource, transition, cost*$\}$ corresponds to the columns in Table 1. Here, $\pi_{case}(1) = 1$, $\pi_{activity}(1) = A$, $\pi_{resource}(1) = John$, etc. An execution of an activity in the process is represented by one or more events that are associated with a lifecycle state transition for the activity instance. These states are used to calculate performance information such as activity durations and waiting times. Events belonging to the same activity instance have the same value for the instance property.

Given an event base, one can derive additional event properties. For example, different event properties can be aggregated together to form new properties, e.g. $\pi_{ar} = (\pi_{activity}, \pi_{resource})$. Alternatively, functions that operate on other properties can be defined. For example, function $\pi_{agegroup}(e) = \frac{(\pi_{age}(e) \ - \ \pi_{age}(e) \ div \ 20)}{20}$ can be used to group events for cases in age groups of 20 years, etc. Such derived event properties may also be based on other events. For example, $\pi_{case\ start}(e) = min\{\pi_{time}(e') \mid e' \in E \wedge \pi_{case}(e') = \pi_{case}(e)\}$. We use these calculated properties to create specific projections of the event base, in order to define potential causal factors for business process performance.

[1] $\mathcal{P}(Y)$ denotes the powerset of a set Y, i.e. $X \in \mathcal{P}(Y) \iff X \subseteq Y$.

Table 1. Example event log L_1. Events can be characterized by multiple properties.

Case id	Case attributes		Event id	Event attributes				
	Age	Type		Time	Activity	Transition	Resource	Instance
1	33	Gold	1	2016-1-4 8:00	A	Start	John	1
			2	2016-1-4 9:15	A	Complete	John	1
			3	2016-1-4 10:12	B	Complete	Bob	2
			4	2016-1-4 14:00	C	Start	Sue	3
			5	2016-1-4 14:05	C	Complete	Sue	3
2	27	Silver	6	2016-1-6 10:43	A	Start	Bob	4
			7	2016-1-6 11:00	A	Complete	Bob	4
			8	2016-1-7 09:33	B	Complete	John	5
			9	2016-1-7 09:35	C	Start	Sue	6
			10	2016-1-7 09:35	C	Complete	Sue	6
3	18	Silver	11	2016-1-7 9:27	A	Start	John	7
			12	2016-1-7 10:40	A	Complete	John	7
			13	2016-1-7 15:03	B	Complete	Bob	8
4	$\perp$	Gold	14	2016-1-7 12:10	A	Start	Bob	9
			15	2016-1-7 12:24	A	Complete	Bob	9
			16	2016-1-8 08:47	B	Complete	John	10
5	41	Silver	17	2016-1-8 15:32	A	Start	Bob	11
			18	2016-1-8 15:51	A	Complete	Bob	11
...	...	...	...	...	...	...	...	...

4 Method

Our causal factor detection approach consists of three main steps, as shown in Fig. 2. In the first step, the event base is systematically decomposed into a directed acyclic graph, in which each node represents a collection of events that share certain business process characteristics and can be considered a potential causal factor (Subsect. 4.1). Nodes in this so-called *decomposition graph* are connected by an edge when the target node is the result of further decomposition of the source node using any (additional) process characteristic. In the second step, this decomposition graph is converted into a so-called *inclusion graph* (Subsect. 4.2). The edges of the inclusion graph represent candidate causal relations between factors (nodes), and will be used in the third and final step. In the causal discovery step, we test for causality between factors in a pair-wise manner (Subsect. 4.3). For every pair of connected nodes in the inclusion graph, the performance values for events in each node are converted into time series and tested for causality. This three-step approach results in a graphical causal model referred to as a *causality graph*.

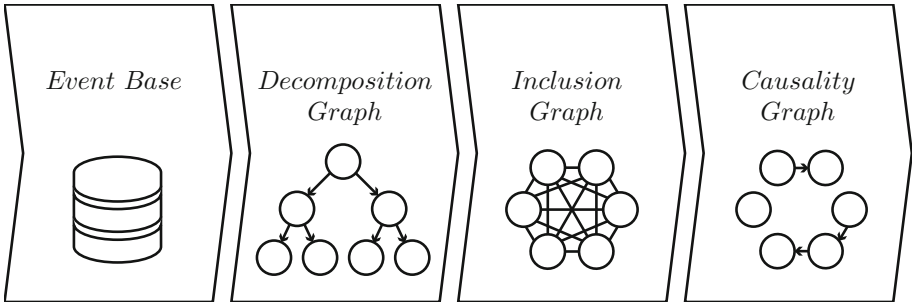

Fig. 2. The steps of our approach (from left to right). A decomposition graph is created from the event base (step 1). Next, it is converted into an inclusion graph (step 2). From the inclusion graph, a causality graph is discovered (step 3).

4.1 Systematic Decomposition

The first step in the approach takes as input the complete event base and returns as output a so-called decomposition graph in which each node represents a set of events that share certain process characteristics. For example, a decomposition can be made by differentiating between activity names, the resource that was responsible for the execution of an event, the type of case, etc. It also possible to differentiate by any combination of properties, as discussed in Sect. 3.

The decomposition step works as follows. We decompose the event collection E in an event base $EB = (E, P, \pi)$ by the set of event properties P using function π. As such, any combination of values for the event properties in P is considered to be a unique process characteristic. Conceptually, our goal is to test whether process performance of a certain set of events that share one or multiple process characteristics causes process performance of another set of events sharing other process characteristics. For example, one such test could test whether the waiting time of all *Pay invoice* activities with a cost greater than *100* causes the case duration for *Gold* customers. If causality is confirmed, it can be said that the former is a causal factor of the latter. Consequently, when the case duration for *Gold* customers is unsatisfactory, process optimization efforts should be directed towards improving the waiting time of activities *Pay invoice* in which the cost was greater than *100*. Formally, decomposition graphs are defined as follows.

Definition 3 (Decomposition graph). *Let $EB = (E, P, \pi)$ be an event base. $G_D(EB) = (N, R_D)$ denotes a decomposition graph over EB, where:*

- $N = \left\{ \left(E', \{P_1, \ldots, P_n\} \right) \in \mathcal{P}(E) \times \mathcal{P}(P) \mid E' \neq \varnothing \wedge \right.$
  $\left. \exists_{v_1,\ldots,v_n \in \mathcal{U}_V} E' = \{e \in E \mid \forall_{1<i<n} \pi_{p_i}(E) = v_i\} \right\}$ *is the set of nodes, and*
- $R = \left\{ \left((E_1, P_1), (E_2, P_2) \right) \in N \times N \mid P_1 \subseteq P_2 \wedge E_2 \subseteq E_1 \right\}$ *the set of edges.*

Note that each decomposition graph is a directed acyclic graph with a root node $(E, \varnothing)$. All events in a node in the decomposition graph share a common value for

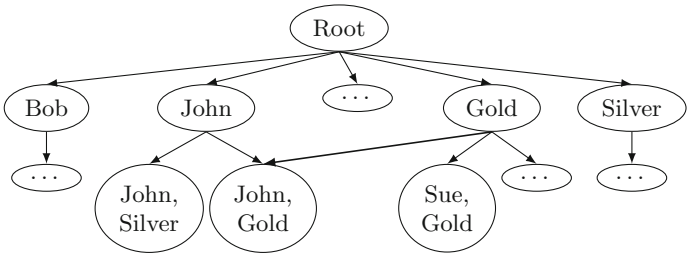

Fig. 3. The decomposition graph created from the event base EB_1. Only selected nodes and edges are shown for sake of simplicity.

each property defined for that node. Additionally, directed edges exist between any pair of nodes for which it holds that the set of properties defined for the source node is included in the set of properties of the target node, and the set of events of the target node is a subset of the set of events of the source node. Note that as only observed values are considered, the decomposition graph is finite. Additionally, context functions can be used to discretize continuous values.

For example, consider the event base $EB_1 = (L_1, P, \pi)$ created from event log L_1 in Table 1, the event properties $P = \{resource, type\}$, and function π. Applying the decomposition step for this event base leads to a decomposition graph as illustrated in Fig. 3. Here, the node labeled *"John, Gold"* holds all events that were performed by resource *John* for cases of type *Gold*, i.e. events 1, 2, and 16. These events are in the intersection of the sets of events in the nodes labeled "*John*" (events $1, 2, 8, 11, 12, 16$) and "*Gold*" (events $1, 2, 3, 4, 5, 14, 15, 16$).

4.2 Candidate Causal Factor Selection

The second step in the approach takes as input the decomposition graph and produces as output a so-called inclusion graph in which each edge represents a candidate causal relation. Formally, inclusion graphs are defined as follows.

Definition 4 (Inclusion graph). *Let $EB = (E, P, \pi)$ be an event base. $G_I(G_D(EB)) = (N, R_I)$ denotes an inclusion graph over a decomposition graph $G_D(EB) = (N, R_D)$, where:*

- $N = \left\{ \left(E', \{P_1, \ldots, P_n\}\right) \in \mathcal{P}(E) \times \mathcal{P}(P) \;\middle|\; E' \neq \varnothing \wedge \exists_{v_1,\ldots,v_n \in \mathcal{U}_V} E' = \{e \in E \;|\; \forall_{1<i<n} \pi_{p_i}(E) = v_i\} \right\}$ *is the set of nodes, and*
- $R_I = \left\{ \left((E_1, P_1), (E_2, P_2)\right) \in N \times N \;\middle|\; \left((E_1, P_1), (E_2, P_2)\right) \notin R_D^+ \wedge \left((E_2, P_2), (E_1, P_1)\right) \notin R_D^+ \right\}$ *the set of edges.*

Every edge in the inclusion graph represents one candidate causal relation in the data. The performance related to events of the source node of such an edge is a potential causal factor for the performance related to events of the

target node. Since our approach is automated, in order to test all combinations of factors, all pairs of nodes are initially connected. However, edges between those pairs of nodes that have an ancestry relationship in the decomposition graph are removed from the inclusion graph, as for those pairs, neither node can be a causal factor of the other. To illustrate this, take the decomposition graph of the running example (depicted in Fig. 3). Any causal relation between the performance related to events performed by *John* for cases of type *Gold*, with the performance related to all events performed by *John* would not have any logical meaning. Note that we are testing a time-lagged causal relationship rather than a compositional relationship (i.e. we do not aim to find which factor contributes most), rather we look at which factor has predictive power over another.

In order to reduce the risk of discovering spurious causalities, and in order to optimize the performance of the causality detection technique, the inclusion graph can be pruned further by removing nodes and/or edges that do not make sense from a business point of view. Multiple such pruning techniques can be constructed, from domain knowledge-based manual selection to automatic clustering and filtering of the data represented by the nodes. Any further pruning of the inclusion graph however falls beyond the scope of this paper.

4.3 Discovering Causality

Once the inclusion graph has been created, it serves as input for the causality discovery step, where individual pairs of nodes in the inclusion graph are checked for cause-effect relationships.

For many years, the concept of causality has received continuing interest in various domains of research. Over the years, the concept has evolved, and as a result, a variety of definitions have been proposed, many of which have a statistical foundation. Techniques such as structural equation modeling [15] and Bayesian networks [16] have been widely used to assess cause-effect relationships between a set of observable and latent variables. In [17], for example, dynamic Bayesian network inference is used to discover causal relations in biological data. However, these techniques are generally more applicable for confirmatory causality analysis based on predefined hypotheses.

Other techniques have been proposed to find causal relationships in case no a-priori knowledge is available about the causal structure in the data [18,19]. These techniques often return a set of causal models which are either hard to interpret, assume the input data to be of a certain restrictive form, or do not consider the time perspective in the data. For business process performance analysis however, the time perspective is of particular importance. In the context of econometric models, Granger has introduced a framework for testing predictive causality that can be used to discover causality between two time series and can be used to create graphical models of causality [3–5,20,21]. In this paper, in order to detect causal factors for business process performance, we test for Granger causality between time series that represent business process performance of different potential causal factors.

Business Process Performance. We define business process performance indicators as functions over events. Different performance functions can have different input and output. Whereas most performance analysis techniques will take only a collection of events as input, other functions can be constructed that take additional input as well. For example, the fitness of a specific case to a process model [1] can be a useful performance function when finding causal factors for non-conformance or when looking for root causes for protocol violations. In this paper, we limit the domain of performance functions to a set of events and the range to timed real values. However, our approach can easily be extended and integrated in situations that require performance functions with different input and output.

Definition 5 (Performance function). *I defines a set of performance indicators. A performance function is a function $\theta \in I \rightarrow (E \nrightarrow \mathbb{R} \times \mathcal{T})$ where for any performance indicator $i \in I$, $\theta(i)$ (denoted θ_i) is a partial function mapping events onto timed real values. If $\theta_i(e) = (r,\ t)$ (denoted r_t), then the performance of event $e \in E$ is $r \in \mathbb{R}$ and the associated time stamp is $t \in \mathcal{T}$. If $e \notin dom(\theta_i)$, then event e does not have a value for performance indicator i and we write $\theta_i(e) = \perp$.*

Typical performance functions are case duration, activity duration, activity waiting time, activity sojourn time, etc. Below, we give definitions for the case duration (Eq. 1), activity sojourn time (Eq. 2) and activity duration (Eq. 3). Other performance functions can be defined analogously.

$$\begin{aligned}\theta_{caseduration}(e) = \Big(&max\{\pi_t(e') \mid e' \in E \wedge \pi_c(e') = \pi_c(e)\} - \\ &min\{\pi_t(e') \mid e' \in E \wedge \pi_c(e') = \pi_c(e)\}, \\ &max\{\pi_t(e') \mid e' \in E \wedge \pi_c(e') = \pi_c(e)\}\Big)\end{aligned} \tag{1}$$

$$\begin{aligned}\theta_{activityduration}(e) = \Big(&max\{\pi_t(e') \mid e' \in E \wedge \pi_i(e') = \pi_i(e)\} - \\ &min\{\pi_t(e') \mid e' \in E \wedge \pi_i(e') = \pi_i(e)\}, \\ &max\{\pi_t(e') \mid e' \in E \wedge \pi_i(e') = \pi_i(e)\}\Big)\end{aligned} \tag{2}$$

$$\begin{aligned}\theta_{activitysojourntime}(e) = \Big(&max\{\pi_t(e') \mid e' \in E \wedge \pi_i(e') = \pi_i(e)\} - \\ &max\{\pi_t(e') \mid e' \in E \wedge \pi_c(e') = \pi_c(e) \wedge \pi_i(e') \neq \pi_i(e) \\ &\pi_t(e') \leq min(\pi_t(e'') \mid e'' \in E \wedge \pi_i(e'') = \pi_i(e))\}, \\ &max\{\pi_t(e') \mid e' \in E \wedge \pi_i(e') = \pi_i(e)\}\Big)\end{aligned} \tag{3}$$

where $\pi_t = \pi_{time}$, $\pi_c = \pi_{case}$ and $\pi_i = \pi_{instance}$.

Time Series. By applying a performance function to a collection of events we obtain a set of timed real values. These values can be represented as a time series, which form the basis of the Granger causality detection technique.

Definition 6 (Time series). *Let $\mathcal{U}_{TS}$ be the universe of time series. Any time series $S \in \mathcal{U}_{TS} = \{s_t \mid s \in \mathbb{R} \wedge t \in \mathcal{T}\}$ defines a time-ordered collection of real values.*

In business processes, the measurements of most process performance indicators arrive at irregular time intervals. In order to perform the Granger causality test, time series regularization needs to be performed. Therefore, we regularize the time series by re-sampling to a common measurement interval. To this end, values for time intervals that do not have any recorded values are imputed (e.g. by linear interpolation), and values for intervals with multiple values are aggregated (e.g. averaged). During analysis, a threshold must be set to avoid a high number of imputed values relative to the number of actual values.

Causality Detection. Each edge in the inclusion graph indicates a candidate causal relation between the source and target nodes. This relation is tested for Granger causality. A time series $S \in \mathcal{U}_{TS}$ is said to Granger cause another time series $S' \in \mathcal{U}_{TS}$ if the past values of S help predict future values of S' better than the past values of S' can predict itself. Three steps are needed to perform the Granger test (denoted $G_{S \to S'}$). First, a linear univariate autoregressive model of S' is fitted.

$$s'_t = \sum_{k=1}^{L} a'_k \cdot s'_{t-k} + \epsilon'_t, \tag{4}$$

Here, L is the lag of Granger test, $t = L+1, \ldots, |S'|$, a' is a vector of parameters for Eq. 4, and ϵ' is the residual. Next, a bivariate linear autoregressive model for S' including the past values of S is fitted as well:

$$s'_t = \sum_{k=1}^{L} a_k \cdot s'_t - k + \sum_{k=1}^{L} b_k \cdot s_{t-k} + \epsilon_t, \tag{5}$$

Here, a and b are vectors of parameters for Eq. 5, and ϵ is the residual. The residuals of Eqs. 4 and 5 can be estimated using a maximum likelihood estimator (in this paper we use the ordinary least squares estimator). Finally, the Granger-Sargent statistic is computed as follows.

$$G_{S \to S'} = \frac{(\epsilon' - \epsilon)/L}{\epsilon/(|S'| - 2L)}. \tag{6}$$

Informally, a large value for $G_{S \to S'}$ indicates that the past information in S is useful for predicting the future values of S'. The Granger-Sargent test is performed to test the null hypothesis of no causality. If the returned p-value is less than the test threshold (typically 5%), S is said to "Granger cause" S'.

4.4 Interpretation

It is worth mentioning here that any statistical causality technique, including ours, discovers only statistically plausible causal structures, and causal factors that are extraneous to the event data cannot be detected.

Complexity. The time-complexity of the approach is bounded by the number of nodes in the inclusion graph times the number of pair-wise tests performed, i.e. $O\left(\sum_{i=0}^{|P|}\binom{|P|}{i}\cdot P_{2|I|}^2\right)$. This can be reduced to $O\left(|P|^{2|P|+2}\cdot|I|!\right)$. However, it should be noted that as explained in Subsect. 4.2, many combinations will not be tested due to their ancestry relationship. Therefore, this upper bound is purely theoretical, and not representative for the real-world complexity.

5 Case Study

The approach presented in this paper has been implemented in the process mining tool ProM[2], and evaluated with a case study using a dataset provided by one of Australia's largest insurance providers. The obtained results were interpreted and validated by a domain expert from this company who is involved in process standardization and optimization efforts. The results were found to provide sensible and actionable insights related to business process performance.

The process that was analyzed is a variant of an automotive claims handling process for which events are recorded by a claims handling system. The provided dataset consists of 17,474 events that have been recorded for 2,577 claims (cases), spanning a total of 13 months. There are 14 distinct activities in the process, and information is recorded about which of the 739 resources was involved in the execution of activities. The total runtime of our technique on this real-life dataset is in the order of several minutes (on modern hardware).

The following subsections correspond to the different steps involved in our causality detection approach, as described in Sect. 4. We complete this section with a discussion on the result.

5.1 Systematic Decomposition

In order to decompose the event collection into a decomposition graph, the following process characteristics were used. We used the activity name as different activities can clearly have different influences on process performance. Additionally, in the process in question, not every activity is mandatory, i.e. not every activity was recorded for every case. As many resources are involved in this process, their performance is an interesting potential cause for the selected performance indicators. Thus, besides the activity name, we used the resource that executed the activity as a process characteristic. In order to have enough measurements per causality analysis, nodes that contained less than 250 events or less than 250 values in their respective time series were filtered out. In total, the obtained decomposition graph contains 25 nodes and 27 edges (after filtering).

5.2 Candidate Causal Factor Selection

From the decomposition graph obtained in the previous step, an inclusion graph was created by applying the technique described in Subsect. 4.2. No additional

[2] See http://promtools.org and the *RootCauseAnalysis* package for more information.

graph pruning techniques were used other than the (automatic) removal of edges between nodes for which an ancestry relation exists in the decomposition graph. The resulting inclusion graph contained 1,161 edges, representing 1,161 candidate causal relations per performance function.

5.3 Discovering Causality

In the dataset provided by the insurance company, only events referring to the completion of activities were available. Consequently, the activity duration could not be calculated, as no events representing the beginning of activities were recorded. We selected the case duration and the activity sojourn time, as defined in Sect. 4.3, as business process performance indicators.

Considering the time-granularity of the recorded data, the time series for the two performance functions for each candidate factor were re-sampled to daily intervals in order to obtain regular time series. When multiple values were available for any given period, the average value was taken. Missing values were replaced by linearly interpolated values. As a filtering step, time series for which more than one out of ten values were imputed were not considered in the analysis. In the Granger causality test, the maximum lag value was set to 7, in order to incorporate time-lagged effects of up to one week. The resulting causality graph showed a total of 16 causal relations involving 11 factors. Out of these, 11 relations between 10 factors were selected for further analysis and explanation in this paper, and can be seen in Fig. 4.

5.4 Discussion of Results

Visual analysis of the selected causal relations discovered by our technique showed five main observations.

Observations 1: One part of the causality graph consists of a set of three activities for which the sojourn times are all causal factors for the sojourn time of the *close claim* activity.

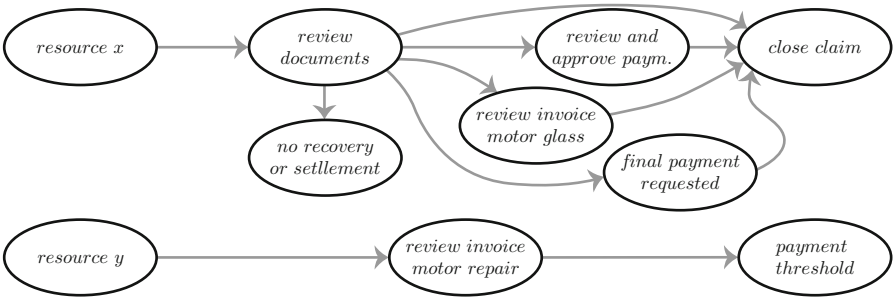

Fig. 4. The causality graph obtained by applying the technique on the insurance claims dataset. Five main observations are found.

Observations 2: The sojourn time of these three activities resulted to be caused by the sojourn time of an activity representing the *review of documents*.

Observations 3: In turn, the causal factor of the sojourn time for this activity was the sojourn time of any activity performed by a *specific resource* (resource x in Fig. 4). On close inspection of the event log, we found that the *close claim* activity was the last activity in about 30% of cases, and that the identified resource was in the top five resources that frequently performed this activity.

Observations 4: Remarkably, the *review document* node also caused the sojourn time of a fifth activity ('no recovery or settlement'), which did not seem to cause the performance of the *close claim* activity.

Observations 5: Finally, one node in the causality graph involves the sojourn time of an activity representing the *review invoice* for the vehicle repair. This factor seems to be caused by the sojourn time of a *specific resource* (resource y in Fig. 4), and is itself a causal factor for the sojourn time of an activity representing some *payment threshold being reached*.

The domain expert was presented with these five observation and was asked to validate the results. With respect to the observations 1, 2 and 3, the explanation given by the domain expert is that claims can only be closed once a checklist of other activities has been completed. The activities on this list correspond to the activities for which the sojourn time was found to be a direct causal factor for the sojourn time of the *close claim* activity. Since the resource names were made anonymous in our dataset, the specific resource could not be identified. However, the domain expert suggested that the identified resource could be an over-utilized person with a validation role, hence the effect on the sojourn time (which includes both waiting and processing time).

Based on the identified factors and the explanation provided by the domain expert, we have suggested (i) to make use of an early-knockout strategy rather than waiting until the claim is about to be closed to check all activities [22], and (ii) to allocate more resources for the validations, or to remove workload from the resources involved in the validation.

Additionally, the activity of which the performance was caused by the sojourn time of the *review document* activity but that did not cause the performance of the *close claim* activity (observation 4), was found to be an activity in which repair costs were recovered from a third-party insurance company. In such cases, the claim may be closed (i.e. this activity is not on the checklist).

Finally, with respect to observation 5, the domain expert explained that the repair of vehicles in this process is performed by a third party. As such, occasionally, an invoice needs to be reviewed. For cases in which this invoice exceeds a certain threshold, a resource having the manager role is involved. Since the performance of this resource is a causal factor, it might indicate an over-utilization. Discussing these cause-effect factors with the domain expert lead to the following recommendations: (i) allocate more resources to review invoices, and/or (ii) increase the threshold for the total amount on the invoice, in order to decrease the number of invoices that need to be reviewed by a manager.

6 Conclusions and Future Work

We proposed a novel technique to automatically discover root causes for business process performance issues such as bottlenecks from event data. To the best of our knowledge, this is the first technique of its kind. The technique supports a range of business process characteristics to perform the analysis and information from additional inputs such as process models may be used to provide specific performance insights. A case study on a real-life dataset showed that the technique has practical relevance and can be used to provide actionable insights to analysts.

One limitation of the current implementation of our technique is that the original definition for Granger causality does not account for latent confounding effects and does not capture instantaneous and non-linear causal relationships. In future work, we would like to explore extensions or alternative causality detection techniques, such as those defined on a structure of relations rather than on pairwise connections. The performance of the technique can be improved by further pruning the inclusion graph by means of clustering and filtering techniques. We would also like to investigate how obtained insights can be used for monitoring, prediction and recommendation of business process performance optimization strategies.

Acknowledgments. This research is funded by the Australian Research Council (grant DP150103356), the Estonian Research Council (grant IUT20-55) and the RISE_BPM project (H2020 Marie Curie Program, grant 645751).

References

1. van der Aalst, W.M.P., Adriansyah, A., van Dongen, B.F.: Replaying history on process models for conformance checking and performance analysis. Wiley Interdisc. Rev.: Data Min. Knowl. Discov. **2**(2), 182–192 (2012)
2. González, L.S., Rubio, F.G., González, F.R., Velthuis, M.P.: Measurement in business processes: a systematic review. Bus. Process Manag. J. **16**(1), 114–134 (2010)
3. Granger, C.W.J.: Some recent development in a concept of causality. J. Econometrics **39**(1), 199–211 (1988)
4. Kamiński, M., Ding, M., Truccolo, W.A., Bressler, S.L.: Evaluating causal relations in neural systems: Granger causality, directed transfer function and statistical assessment of significance. Biol. Cybern. **85**(2), 145–157 (2001)
5. Roebroeck, A., Formisano, E., Goebel, R.: Mapping directed influence over the brain using Granger causality and fMRI. Neuroimage **25**(1), 230–242 (2005)

6. Pika, A., van der Aalst, W.M.P., Fidge, C.J., ter Hofstede, A.H.M., Wynn, M.T.: Predicting deadline transgressions using event logs. In: Rosa, M., Soffer, P. (eds.) BPM 2012. LNBIP, vol. 132, pp. 211–216. Springer, Heidelberg (2013). doi:10.1007/978-3-642-36285-9_22
7. Suriadi, S., Ouyang, C., van der Aalst, W.M.P., ter Hofstede, A.H.M.: Root cause analysis with enriched process logs. In: Rosa, M., Soffer, P. (eds.) BPM 2012. LNBIP, vol. 132, pp. 174–186. Springer, Heidelberg (2013). doi:10.1007/978-3-642-36285-9_18
8. de Leoni, M., van der Aalst, W.M.P., Dees, M.: A general process mining framework for correlating, predicting and clustering dynamic behavior based on event logs. Inf. Syst. **56**, 235–257 (2016)
9. Conforti, R., de Leoni, M., La Rosa, M., van der Aalst, W.M.P., ter Hofstede, A.H.M.: A recommendation system for predicting risks across multiple business process instances. Decis. Support Syst. **69**(1), 1–19 (2015)
10. Ghattas, J., Soffer, P., Peleg, M.: Improving business process decision making based on past experience. Decis. Support Syst. **59**, 93–107 (2014)
11. van der Aalst, W.M.P.: Process cubes: slicing, dicing, rolling up and drilling down event data for process mining. In: Song, M., Wynn, M.T., Liu, J. (eds.) AP-BPM 2013. LNBIP, vol. 159, pp. 1–22. Springer, Cham (2013). doi:10.1007/978-3-319-02922-1_1
12. Vogelgesang, T., Appelrath, H.-J.: PMCube: a data-warehouse-based approach for multidimensional process mining. In: Reichert, M., Reijers, H.A. (eds.) BPM 2015. LNBIP, vol. 256, pp. 167–178. Springer, Cham (2016). doi:10.1007/978-3-319-42887-1_14
13. Tan, S., Chan, T.: Defining and conceptualizing actionable insight: a conceptual framework for decision-centric analytics. In: Australasian Conference on Information Systems (2015)
14. Hompes, B.F.A., Buijs, J.C.A.M., van der Aalst, W.M.P.: A generic framework for context-aware process performance analysis. In: Debruyne, C., et al. (eds.) OTM 2016. LNCS, vol. 10033, pp. 300–317. Springer, Cham (2016). doi:10.1007/978-3-319-48472-3_17
15. Bollen, K.A.: Structural Equations with Latent Variables. Wiley, Hoboken (2014)
16. Jensen, F.V.: An Introduction to Bayesian networks, vol. 210. UCL Press, London (1996)
17. Yu, J., Smith, V.A., Wang, P.P., Hartemink, A.J., Jarvis, E.D.: Advances to bayesian network inference for generating causal networks from observational biological data. Bioinformatics **20**(18), 3594–3603 (2004)
18. Shimizu, S., Hoyer, P.O., Hyvarinen, A., Kerminen, A.: A linear non-gaussian acyclic model for causal discovery. J. Mach. Learn. Res. **7**, 2003–2030 (2006)
19. Richardson, T.: A polynomial-time algorithm for deciding Markov equivalence of directed cyclic graphical models. In: Proceedings of the Twelfth International Conference on Uncertainty in Artificial Intelligence, pp. 462–469. Morgan Kaufmann Publishers Inc., Burlington (1996)
20. Granger, C.W.J.: Investigating causal relations by econometric models and cross-spectral methods. Econometrica **37**(3), 424 (1969)

21. Dahlhaus, R., Eichler, M.: Causality and graphical models in time series analysis. In: Oxford Statistical Science Series, pp. 115–137 (2003)
22. van der Aalst, W.M.P.: Re-engineering knock-out processes. Decis. Support Syst. **30**(4), 451–468 (2001)