University Library

A gateway to Melbourne's research publications

Minerva Access is the Institutional Repository of The University of Melbourne

Author/s:
Weidlich, M;Polyvyanyy, A;Desai, N;Mendling, J;Weske, M

Title:
Process compliance analysis based on behavioural profiles

Date:
2011-11-01

Citation:
Weidlich, M., Polyvyanyy, A., Desai, N., Mendling, J. & Weske, M. (2011). Process compliance analysis based on behavioural profiles. Information Systems, 36 (7), pp.1009-1025. https://doi.org/10.1016/j.is.2011.04.002.

Persistent Link:
https://hdl.handle.net/11343/258715

Process Compliance Analysis based on Behavioural Profiles

Matthias Weidlich[a,*], Artem Polyvyanyy[a], Nirmit Desai[b], Jan Mendling[c], Mathias Weske[a]

[a] *Hasso Plattner Institute, University of Potsdam, Prof.-Dr.-Helmert-Str. 2-3, D-14482 Potsdam, Germany*
[b] *IBM India Research Labs, Domlur Ring Road, Bangalore 560071, India*
[c] *Humboldt University, Unter den Linden 6, D-10099 Berlin, Germany*

Abstract

Process compliance measurement is getting increasing attention in companies due to stricter legal requirements and market pressure for operational excellence. In order to judge on compliance of the business processing, the degree of behavioural deviation of a case, i.e., an observed execution sequence, is quantified with respect to a process model (referred to as fitness, or recall). Recently, different compliance measures have been proposed. Still, nearly all of them are grounded on state-based techniques and the trace equivalence criterion, in particular. As a consequence, these approaches have to deal with the state explosion problem. In this paper, we argue that a behavioural abstraction may be leveraged to measure the compliance of a process log – a collection of cases. To this end, we utilise causal behavioural profiles that capture the behavioural characteristics of process models and cases, and can be computed efficiently. We propose different compliance measures based on these profiles, discuss the impact of noise in process logs on our measures, and show how diagnostic information on non-compliance is derived. As a validation, we report on findings of applying our approach in a case study with an international service provider.

Keywords: Process Compliance, Compliance Measurement, Log Conformance, Root Cause Analysis

1. Introduction

Compliance management is becoming increasingly important. Companies seek for a better control of their processes not only to satisfy new legal requirements but also to leverage cost-saving opportunities through standardisation of business operations. Once

*Corresponding author
Email addresses: matthias.weidlich@hpi.uni-potsdam.de (Matthias Weidlich), Artem.Polyvyanyy@hpi.uni-potsdam.de (Artem Polyvyanyy), Nirmit123@in.ibm.com (Nirmit Desai), jan.mendling@wiwi.hu-berlin.de (Jan Mendling), Mathias.Weske@hpi.uni-potsdam.de (Mathias Weske)

non-compliant cases of operations are detected, the company can either update its process model to cover the respective case or it can impose new mechanisms to enforce best practice execution. In this way, compliance management is a central piece in the puzzle of advancing a company towards a higher degree of process maturity.

In order to make compliance management work in practice, it is required to gather detailed information on the execution of business processes. In recent years, process mining has emerged as a technique that automatically reworks process log data such that managerial decision making can be supported [1, 2, 3, 4, 5]. In this paper, we focus on scenarios in which the processing is already described by normative process models. Such a model defines which steps have to be performed, and in which order, to achieve a business value. Once a process log is available, certain key measures of compliance management can be quantified along a set of orthogonal dimensions [6, 7]. *Fitness*, also referred to as *recall*, for instance, measures to which degree the behaviour of a single case (or a complete process log) is captured in a process model. Other dimensions focus on the appropriateness of a process model with respect to a process log. In this case, the degree to which the process model is restricted to the observed behaviour (*precision*) or allows for additional behaviour (*generality*) is quantified, see [8] for a detailed discussion.

To evaluate the compliance of business operations, fitness measures are of utmost importance. These measures provide feedback on cases that do not conform to the normative process model and quantify any behavioural deviation. The latter is then used to compute the degree of compliance for the cases. Precision and generality, in turn, aim at quantifying the quality of a process model with respect to the observed behaviour instead of the quality of the cases of a process log. Hence, these metrics are mainly used to judge on the quality of process models that are discovered by mining algorithms (cf., [9]).

Various compliance measures, in the sense of a fitness measure, have been proposed in the literature [9, 10, 11, 12, 13]. Still, all of these measures rely on state-based techniques that involve replaying the cases of a process log, i.e., the single observed execution sequences. First and foremost, compliance may be computed as the share of cases of a process log that can be replayed in the process model [11, 12]. More fine-granular measures try to replay a case step-wise in the process model and quantify the number of execution steps that are in line with the process model semantics [9, 10, 13]. All these measures are based on the notion of trace equivalence, which is a weak notion in the linear time – branching time spectrum [14]. As a consequence, these approaches have to cope with the state explosion problem in order to achieve efficient computation [15]. That, in turn, leads to the application of heuristics that have to be tuned for a certain setting. In addition, phenomena such as compliance values below one for valid execution sequences of the process model that stem from invisible activities have to be addressed separately [16]. Moreover, the classical fitness measures [9, 10] has been criticised to yield compliance values which are significantly lower than what is considered to be correct by domain experts [17]. These results, along with the inherent complexity of state-space based approaches, suggest to consider an alternative grounding for compliance measurement.

In this paper, we approach the problem from the perspective of relations between pairs of activities or log events, respectively, instead of trying to replay cases according to a rather strict notion of equivalence. That is, we leverage causal behavioural profiles [18, 19] as a base line to compliance measurement. These profiles are a behavioural abstraction that is more relaxed than trace equivalence. As these profiles can be computed efficiently for many classes of process models, we avoid performance issues of existing state-based

measures. This paper is an extended and revised version of our earlier work [20], which proposed a first set of compliance measures based on causal behavioural profiles. While we follow the same idea in this paper, we introduce novel compliance measures on a more general level. In addition, we address two aspects that are of high importance once compliance measures are applied in practice. On the one hand, diagnostic information should be given in case of non-compliance. On the other hand, the impact of noise in a process log on the compliance measures has to be made explicit in order to allow for appropriate interpretation of the computed compliance values. In summary, the contributions of this paper are the following:

- We present novel compliance measures for the cases of a process log. In contrast to our previous work [20], these measures are directly grounded on the utilised behavioural abstraction.
- We give an overview of common noise patterns in a process log and their impact on our compliance measures. This overview helps to pin down the computed compliance values with respect to the noise that can be expected in a certain setting.
- Our approach provides detailed feedback on non-compliance and supports root cause analysis. Diagnostic information is presented for single cases and on the level of a process log.

Finally, we report on findings from an application of our approach in an industry case study. Using a process log from an international service provider, we discuss the results of our compliance analysis.

Against this background, the paper is structured as follows: Section 2 discusses the challenge of measuring compliance by means of an example. Section 3 presents preliminaries for our investigations. Section 4 introduces our compliance measures based on behavioural profiles. Subsequently, Section 5 gives an overview of common noise patterns and their impact on our measures. Section 6 introduces an approach of deriving diagnostic information on non-compliance. Section 7 presents findings from our validation, for which we implemented a prototype and tested it on a real-world log. Section 8 discusses our approach in the light of related work. Finally, Section 9 concludes the paper and identifies topics for future research.

2. Background

This section illustrates the problem of measuring compliance for business processes that are described by normative process models. While there are various process modelling languages with varying expressiveness and notations, a process model can commonly be seen as a graph consisting of nodes and edges. The former depict business activities and split / merge nodes that implement an execution logic beyond simple sequencing of activities. The latter define the control flow structure of the business process. Figure 1 shows an exemplary process model captured in BPMN. It includes 11 activities, all named with a capital letter. The diamonds define the routing behaviour of the BPMN model. Once I and A have been executed, there is a choice being made at the diamond marked with an X. First, the upper branch including B may be taken. After an execution of B, another choice is made. This may result in an execution of H or J, so that B is executed

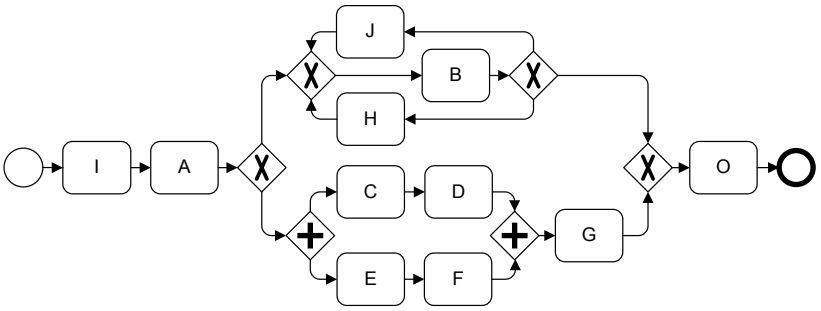

Figure 1: Example of a BPMN process model

again. As an alternative, activity O may be executed directly after the first execution of B. Second, the lower branch leading to the diamond with the plus sign may be taken. In this case, the sequences C, D and E, F are executed in parallel, which are followed by an execution of G. Finally, the alternative branches are merged and control is passed towards the completion of the process after execution of O.

In the context of this paper, we assume that information on the actual processing is available in the form of a process log (or log). A log comprises cases that represent *observed* execution sequences of activities from a process model. Extracting events from an IT-system and relating them to activities of a process model may be a cumbersome task. Depending on the logging facility, events may have to be filtered and aggregated, or even have to be generated based on state changes on the database level, see also [21]. In a perfect setting, the cases of a log completely comply with the behaviour defined by the process model. Then, the cases are *valid* execution sequences. In practice, however, observed execution sequences often deviate from predefined behaviour. This may be a problem of the model when it does not meet validity and completeness requirements [22]. When the process model has a normative character, deviations may be caused by information systems that record a log but do not explicitly enforce the execution order of activities. It is also possible that people deliberately work around the system [2]. This may result in cases for the process model in Figure 1, such as the following.

- Case $c_1 = \langle I, A, E, C, D, F, G, O \rangle$
- Case $c_2 = \langle I, A, C, B, G, F, O \rangle$
- Case $c_3 = \langle I, A, B, J, H, B, O, G \rangle$
- Case $c_4 = \langle I, C, E \rangle$
- Case $c_5 = \langle F, C, D, G \rangle$

From these five cases, only the first one is also a valid execution sequence, i.e., it can be completely replayed by the process model. Still, the other cases capture a certain share of the behaviour defined in the model. Such cases make it necessary to measure compliance *a posteriori*.

As stated above, there are different measures that aim at quantification of the degree to which a single case or a complete process log match a normative process model. While the share of cases that represent valid execution sequences of the process model can be seen as a straight-forward compliance measure, cf., [11, 12], more fine-granular approaches are grounded on state-based concepts from Petri net theory. The basic idea behind the classical fitness measure [10] is to replay the cases through the model. Activities that are

enabled in the model when they appear in the case are counted and related to the overall number of activities. If an activity is not enabled, the Petri net transition representing the activity is forced to fire, which produces a token on each of its output places. In this way, one can quantify compliance of a case against the process model as a ratio of enabled activities to the total number of activities. For example, in Figure 1, case c_1 can be completely replayed and has therefore a compliance value of one. Instead, case c_2 can be replayed solely until B appears in the case. This activity is then fired without being enabled. The same holds for G and F. Therefore, four firings are compliant out of the seven firings altogether, yielding a fitness value of 0.57. In case c_3, activities H, G, and the second occurrence of B are forced to fire although they are not enabled. Thus, altogether, only I, A, J, O, and the first occurrence of B are fired correctly from eight activities. Therefore, the fitness is 0.63. For case c_4, the absence of activity A in the case implies a non-compliant firing of C and E, such that the fitness is 0.33. Finally, for case c_5, we derive a fitness value of 0.5 since the firing of two out of four activities is correct, i.e., D and G.

In this paper, we suggest to base compliance analysis on a set of behavioural constraints that a process model imposes for a pair of activities. Examples for such behavioural constraints would be the exclusiveness (B and C in Figure 1) and order (C and D in Figure 1) of activity execution. Other constraints relate to the obligation to execute a certain activity (A in Figure 1) or to causal dependencies between activity executions (C and G in Figure 1). With taking these constraints as a basis for compliance analysis, a certain degree of behavioural abstraction is assumed. Consequently, we do not postulate total validity and completeness of the process model [22], for which the compliance of a process log is analysed. We show how the preservation of behavioural constraints is leveraged to assess compliance of a given case. To this end, we employ the concept of causal behavioural profiles which provide a behavioural abstraction of process models in terms of behavioural constraints. Based on the compliance values obtained for single cases, conclusions on the compliance of a log can be drawn.

3. Preliminaries

This section gives preliminaries for our work. First, we introduce the notion of a process model used throughout the paper. Second, we define causal behavioural profiles as a behavioural abstraction of a process model.

3.1. Process Models

For our investigations we use a notion of a process model that is based on a graph containing activity nodes, split and merge nodes (alias control nodes). A process model captures the commonalities of process description languages. Thus, the subset of BPMN used in our initial example can be traced back to the following definition of a process model.

Definition 1. *(Process Model)*
A *process model* is a tuple $P = (A, a_i, a_o, C, F, T)$, with

- A as a non-empty set of activity nodes, and C as a set of control nodes, A and C are disjoint,
- $a_i \in A$ as an initial activity, $a_o \in A$ as a final activity,

- $F \subseteq ((A \setminus \{a_o\}) \cup C) \times ((A \setminus \{a_i\}) \cup C)$ as the flow relation, such that $(A \cup C, F \cup \{(a_o, a_i)\})$ is a strongly connected graph, and
- $T : C \mapsto \{and, or, xor\}$ as a function that assigns types to control nodes.

Our notion of a process model postulates the existence of two dedicated activities that represent the start of the process (a_i) and the end of the process (a_o), respectively. We assume these activities to carry no business semantics except for the creation or closing of a process instance. In case a process model has multiple activities without outgoing or incoming control flow, normalisation is applied to derive a model of the respective structure, cf., [23]. In the remainder of this paper, the identity relation for activities is denoted by id_A, i.e., $(a, a) \in id_A$ for all $a \in A$. We do not formalise the execution semantics of a process model, but assume an interpretation of the model following on common process description languages, such as BPMN, EPCs, or UML activity diagrams. We do not assume a certain definition of semantics for the inclusive OR construct, which raises serious issues in cyclic structures. We solely assume the existence of such a definition. Under the assumption of well-defined execution semantics, the set of all valid execution sequences, as well as the notion of a case, are defined as follows.

Definition 2. *(Execution Sequence, Case)*
The set of *execution sequences* $\mathcal{E}_P$ for a process model $P = (A, a_i, a_o, C, F, T)$ is the set of all lists of the form $\sigma = \langle a_i, a_1, \ldots, a_n, a_o \rangle$ with $n > 0$, $n \in \mathbb{N}$, $a_j \in A$ for all $0 < j \leq n$, that can be created following on the execution semantics of P. A *case* c that has been observed for P is a list of the form $c = \langle a_1, \ldots, a_n \rangle$ with $n > 0$, $n \in \mathbb{N}$, and $a_j \in A$ for all $0 < j \leq n$.

Note that we speak of an *execution sequence* of a process model, solely if the sequence is valid regarding the process model, i.e., it can be completely replayed by the model. In contrast, a *case* is a non-empty sequence over the activities of a process model. As a short-hand notation, we use $A_c \subseteq A$ to refer to the subset of activities of a process model that is contained in case c.

3.2. Causal Behavioural Profiles for Process Models

In order to capture the constraints imposed by a process model on the order of activity execution, we rely on the concept of a behavioural profile [18]. Behavioural profile defines relations for all pairs of activities of a process model. These relations, in turn, might be interpreted as the essential behavioural characteristics specified by the models. All behavioural relations of behavioural profile are based on the notion of *weak order*. That is, two activities are in weak order, if and only if there exists an execution sequence in which one activity occurs after the other.

Definition 3 (Weak Order (Process Model)).
Let $P = (A, a_i, a_o, C, F, T)$ be a process model and $\mathcal{E}_P$ its set of execution sequences. The *weak order relation* $\succ_P \subseteq (A \times A)$ contains all pairs (x, y), such that there exists an execution sequence $\sigma = \langle n_1, \ldots, n_m \rangle$ in $\mathcal{E}_P$ and there exist two indices $j, k \in \{1, \ldots, m\}$ with $j < k \leq m$ for which holds $n_j = x$ and $n_k = y$.

Based thereon, we define the relations of the behavioural profile for pairs of activities. Each pair can be related by weak order in different ways.

Definition 4 (Behavioural Profile (Process Model)).
Let $P = (A, a_i, a_o, C, F, T)$ be a process model. A pair $(x, y) \in (A \times A)$ is in at most one of the following relations:

- The *strict order relation* $\rightsquigarrow_P$, iff $x \succ_P y$ and $y \not\succ_P x$.
- The *exclusiveness relation* $+_P$, iff $x \not\succ_P y$ and $y \not\succ_P x$.
- The *interleaving order relation* $||_P$, iff $x \succ_P y$ and $y \succ_P x$.

The set $\mathcal{B}_P = \{\rightsquigarrow_P, +_P, ||_P\}$ is the *behavioural profile* of P.

Note that we say that a pair (x, y) is in *reverse strict order*, denoted by $x \rightsquigarrow_P^{-1} y$, if and only if $y \rightsquigarrow_P x$. Further, the relations of the behavioural profile along with reverse strict order partition the Cartesian product of activities [18]. We illustrate the relations of the behavioural profile by means of our example model in Figure 1. Here, for instance, it holds $I \rightsquigarrow D$. Evidently, strict order does not imply the actual occurrence, i.e., activity D might not be executed. It holds $B + C$ as both activities will never occur in a single valid execution sequence of the model, and $C||F$ as C might occur before F and vice versa. Note that it holds $B||J$ due to the control flow cycle. An activity is either said to be *exclusive to itself* (e.g., $I + I$) or *in interleaving order to itself* (e.g., $B||B$). The former holds, when an activity cannot be repeated, whereas the latter implies that there may be multiple executions of the activity.

The concept of a behavioural profile relates pairs of activities according to their *order* of potential occurrence, whereas further behavioural characteristics are not considered. In particular, *causality* between activities is not covered. Causality involves two orthogonal aspects, i.e., the *order* of activity occurrences and their *causal coupling* (the occurrence of one activity enforces the occurrence of another activity). The former is addressed by the behavioural profile by the (reverse) strict order relation. The latter is not captured. To cope with these aspects, the behavioural profile has been extended by a fourth relation yielding the causal behavioural profile [19].

Definition 5 (Causal Behavioural Profile (Process Model)).
Let $P = (A, a_i, a_o, C, F, T)$ be a process model.

- A pair $(x, y) \in (A \times A)$ is in the *co-occurrence relation* $\gg_P$, iff for all execution sequences $\sigma = \langle n_1, \ldots, n_m \rangle$ in $\mathcal{E}_P$ it holds that $n_i = x$ with $1 \leq i \leq m$ implies that there is an index $j \in \{1, \ldots, m\}$, such that $n_j = y$.
- The set $\mathcal{B}_P^+ = \mathcal{B}_P \cup \{\gg_P\}$ is the *causal behavioural profile* of P.

Although not completely orthogonal, the co-occurrence relation of the behavioural profile shows only minor interrelation with the relations of the behavioural profile. In fact, the only conclusion that can be drawn relates to the co-occurrence of exclusive activities. Two activities that are exclusive never appear together in an execution sequence, so that they cannot be co-occurring.

Using the causal behavioural profile, we speak of causality between two activities $a_1, a_2 \in A$, if they are in strict order, $a_1 \rightsquigarrow_P a_2$, and the occurrence of the first implies the occurrence of the second, $a_1 \gg_P a_2$. Again, we refer to the example in Figure 1 for illustration purposes. In this model, three activities have to be executed in any completed case, namely I,A, and O. Moreover, it holds $C \gg G$ and $G \gg C$. Thus, all complete execution sequences of the process model that contain activity C are required to also contain activity G, and vice versa. In addition, the model specifies a strict order relation between both activities, $C \rightsquigarrow G$, such that we speak of a causal dependency from C to G.

Computation of the behavioural profile is done efficiently for sound process models that do not contain control nodes of type *or*. Soundness is a correctness criteria often used for process models that guarantees the absence of behavioural anomalies, such as deadlocks or livelocks [24]. It has been defined for workflow nets, a structural class of Petri nets. If a process model does not show *or*-type control nodes, it can be translated into a free-choice workflow net following on common formalisations of process modelling languages (see [25] for a survey). Hence, the soundness criterion can directly be applied to the notion of a process model. Then, techniques introduced for the derivation of behavioural profiles of sound free-choice workflow nets are reused. Those allow for the computation of behavioural profiles in cubic time to the size, i.e., number of nodes, of the model [18]. If certain structural assumptions are met, computation is even more efficient when using structural decomposition techniques [19]. This approach is also leveraged in order to compute the co-occurrence relation of the causal behavioural profile. Again, this can be done in cubic time to the size of the model, if unstructured model fragments are acyclic or do not show concurrency, see [19] for further details. While all these approaches leverage structural information and, therefore, are restricted to a certain class of models, a generic computation algorithm has been presented in [26]. This algorithm is based on the notion of a Petri net complete prefix unfolding that represents the behaviour of the net. Hence, it is applicable in the general case at the expense of computational complexity as the construction of the prefix is an NP-complete problem. Nevertheless, this approach can be combined with the aforementioned approaches so that it is to be applied solely for rather small sub-parts of a process model.

4. Compliance Measures based on Behavioural Profiles

This section introduces compliance measures based on behavioural profiles. First, Section 4.1 shows how the concepts introduced in the previous section can be lifted to cases of a process log. Second, we elaborate on a hierarchy between the relations of the behavioural profile in Section 4.2. We introduce measures for different compliance aspects in Section 4.3. Finally, Section 4.4 elaborates on aggregations of these measures to arrive at a single compliance value for a case.

4.1. Causal Behavioural Profiles for Cases of Process Logs

In order to lift the concept of behavioural profiles to cases of process logs, first and foremost, we have to clarify the notion of weak order for cases. Following on the definition given for process models, two activities are in weak order in a case, if the first occurs before the second.

Definition 6 (Weak Order (Case)). Let $c = \langle n_1, \ldots, n_m \rangle$ be a case and $P = (A, a_i, a_o, C, F, T)$ a process model with $A_c \subseteq A$. The *weak order relation* $\succ_c \subseteq (A_c \times A_c)$ contains all pairs (x, y), such that there exists two indices $j, k \in \{1, \ldots, m\}$ with $j < k \leq m$ for which holds $n_j = x$ and $n_k = y$.

Based thereon, we define the behavioural profile of a case.

Definition 7 (Behavioural Profile (Case)).
Let $c = \langle n_1, \ldots, n_m \rangle$ be a case and $P = (A, a_i, a_o, C, F, T)$ a process model with $A_c \subseteq A$. A pair $(x, y) \in (A_c \times A_c)$ is in at most one of the following relations:

- The *strict order relation* $\rightsquigarrow_c$, iff $x \succ_c y$ and $y \not\succ_c x$.
- The *exclusiveness relation* $+_c$, iff $x \not\succ_c y$ and $y \not\succ_c x$.
- The *interleaving order relation* $||_c$, iff $x \succ_c y$ and $y \succ_c x$.

The set $\mathcal{B}_c = \{\rightsquigarrow_c, +_c, ||_c\}$ is the *behavioural profile* of c.

Again, the pair (x, y) is in *reverse strict order*, denoted by $x \rightsquigarrow_c^{-1} y$, if and only if $y \rightsquigarrow_c x$. The relations of the behavioural profile along with reverse strict order partition the Cartesian product of activities of a case. For the example case $c_2 = \langle I, A, C, B, G, F, O \rangle$, e.g., it holds $C \rightsquigarrow_{c_2} F$ and $B \rightsquigarrow_{c_2}^{-1} A$.

There are fundamental differences when interpreting the behavioural profile for a process model and a case. On the one hand, in contrast to the profile of a process model, exclusiveness between two activities can be observed in a case solely as a self-relation. In other words, for all pairs of activities (x, y), for which we observe $x + y$ in a case, it holds $x = y$. On the other hand, activities that might be enabled concurrently in a process model (e.g., C and F in our example) are related by interleaving order in the behavioural profile of the model ($C||F$). However, if both activities are not executed multiple times, they might be related by strict order or reverse strict order in the profile of a corresponding case. For instance, for case $c_2 = \langle I, A, C, B, G, F, O \rangle$, we observe $C \rightsquigarrow_{c_2} F$ as the behavioural relation for activities C and F.

We also lift the definition of a causal behavioural profile to cases. All activities of a case are co-occurring.

Definition 8 (Causal Behavioural Profile (Case)).
Let $c = \langle n_1, \ldots, n_m \rangle$ be a case and $P = (A, a_i, a_o, C, F, T)$ a process model with $A_c \subseteq A$.

- The *co-occurrence relation* $\gg_c = (A_c \times A_c)$ contains all pairs of activities in the case.
- The set $\mathcal{B}_c^+ = \mathcal{B}_c \cup \{\gg_c\}$ is the *causal behavioural profile* of c.

4.2. A Hierarchy of Behavioural Relations

As mentioned above, there is a fundamental difference between behavioural profiles of process models and of cases. The former defines relations based on the set of *all* possible execution sequences, whereas the latter considers only *one* observed execution sequence as defined by the case. We introduce a hierarchy between the relations of behavioural profiles (we neglect the co-occurrence relation at this stage). The idea is to order the behavioural relations based on their *strength*. We consider the exclusiveness relation as the strongest relation, as it completely disallows two activities to occur together in an execution sequence. In contrast, the interleaving order relation can be seen as the weakest relation. It allows two activities to occur in any order in an execution sequence. Consequently, the strict order and reverse strict order relation are intermediate relations, as they disallow solely a certain order of two activities. We formalize this hierarchy between behavioural relations as a *subsumption predicate*. Given two behavioural relations between a pair of two activities, this predicate is satisfied, if and only if the first relation is equal or weaker than the second.

Definition 9 (Subsumption Predicate).
Given two behavioural relations $R, R' \in \{\rightsquigarrow, \rightsquigarrow^{-1}, +, ||\}$ of the same or different behavioural profiles, the *subsumption predicate* $\mathcal{S}(R, R')$ is satisfied, iff $(R \in \{\rightsquigarrow, \rightsquigarrow^{-1}\} \wedge R' = +)$ or $R = R'$ or $R = ||$.

Again, we illustrate this concept using the example model in Figure 1 and case $c_2 = \langle I, A, C, B, G, F, O \rangle$. As mentioned above, for activities C and F, it holds $C||F$ in the profile of the process model and $C \rightsquigarrow_{c_2} F$ in the profile of the case. The former specifies that C and F might occur in any order in an execution sequence, owing to the interleaving semantics of activities that are enabled concurrently. The latter, in turn, captures the fact that the occurrences of C and F in the case are ordered. However, we see that there is a subsumption relation between both relations, as $\mathcal{S}(||, \rightsquigarrow_{c_2})$ is satisfied. This information has to be taken into account when assessing compliance of cases. This stems from the fact that a single case does not hint at the potential interleaving execution of activities.

4.3. Measures for Compliance Aspects

For our measurements of compliance between a process model and a case, we consider two aspects separately, namely the *order* and *causality* of activity execution. These aspects relate to the questions of *what* activities should be contained in the case and *how* these activities should be ordered. This section shows how both aspects are assessed by a separate compliance degree. First, order and exclusiveness of activity execution is assessed by the degree of behavioural profile compliance. Second, causal dependencies between occurrences of activities are measured by the degree of co-occurrence compliance.

Behavioural profile compliance. The order of execution of activities as specified by a case should be in line with the ordering constraints as imposed by a process model. We achieve a quantification of any behavioural deviation based on the notion of behavioural profiles and the hierarchy of behavioural relations. We analyse the Cartesian product of activities in a case and determine, whether the behavioural relation for a pair of activities in the case is subsumed by the relation specified in the process model. Compliance assessment based on the relations of the behavioural profile takes into account distinct activities that are meant to be mutually exclusive. Those activities will be related by the exclusiveness relation of the behavioural profile of the process model. Once these activities occur in a case, they will be related by (reverse) strict order or interleaving order in the behavioural profile of the case. Exclusiveness can be observed in a case solely as a self-relation. Hence, the mutual execution constraint imposed by the process model is counted as being violated.

We define two degrees of behavioural profile compliance that differ with respect to their normalisation. First, the degree of *model-relative* behavioural profile compliance is defined as the ratio of consistent behavioural relations relative to the number of activity pairings in the case. This degree considers all activity pairs that occur in the case. Hence, it directly depends on the number of activities of the process model that have been executed already. Second, it may be argued that activity pairs that show interleaving order in the process model should be neglected. Following on the argumentation given in Section 4.2, interleaving order can be interpreted as the absence of any ordering constraint as it allows for the occurrences in any order. In fact, a constraint of the process model related to interleaving order cannot be violated by any case due to the subsumption predicate. Therefore, we also define the degree of *constraint-relative* behavioural profile compliance that is independent of the number of activities in the case, but depends on the number of exclusiveness and strict order constraints imposed by the process model. Behavioural profile compliance is defined as follows.

Definition 10 (Behavioural Profile Compliance).
Let $c = \langle n_1, \ldots, n_m \rangle$ be a case and $P = (A, a_i, a_o, C, F, T)$ a process model with $A_c \subseteq A$.

- The *set of profile consistent case pairs* $PC_c \subseteq (A_c \times A_c)$ contains all pairs of activities (x, y), for which the behavioural relation in c is subsumed by the relation in P, i.e., $\forall\ R \in (\mathcal{B}_P \cup \{\rightsquigarrow_P^{-1}\}), R' \in (\mathcal{B}_c \cup \{\rightsquigarrow_c^{-1}\})$ it holds $(xRy \wedge xR'y) \Rightarrow \mathcal{S}(R, R')$.
- The degree of *model-relative behavioural profile compliance* of c to P is defined as

$$\mathcal{MBC}_c = \frac{|PC_c|}{|A_c|^2}.$$

- The degree of *constraint-relative behavioural profile compliance* of c to P is defined as

$$\mathcal{CBC}_c = \begin{cases} 1 & \text{if } ||_c = (A_c \times A_c), \\ \frac{|PC_c \setminus ||_c|}{|(A_c \times A_c) \setminus ||_c|} & \text{else.} \end{cases}$$

Both compliance degrees are between zero, i.e., no compliance at all with respect to the behavioural profile, and one indicating full compliance. Computation of both degrees requires iteration over the Cartesian product of activities in the case. Hence, the computation does not add to the complexity needed to derive behavioural profiles for process models, cf., Section 3.2. Given case $c_2 = \langle I, A, C, B, G, F, O \rangle$ and our initial example (cf., Figure 1), we see that an order constraint imposed by the model is not satisfied. That is, $F \rightsquigarrow G$ is specified in the model, whereas we have $F \rightsquigarrow_{c_2}^{-1} G$ in the profile of the case. In addition, the given case violates constraints on mutual exclusion of activity execution. The process model defines $B + C$, $B + F$, and $B + G$, whereas we have $B \rightsquigarrow_{c_2}^{-1} C$, $B \rightsquigarrow_{c_2}^{-1} F$, and $B \rightsquigarrow_{c_2}^{-1} G$ in the case. Quantification of these violations relative to the number of considered activities yields a degree of model-relative behavioural profile compliance of $\mathcal{MBC}_{c_2} = \frac{41}{49} \approx 0.84$ for this particular case. Once constraints on interleaving order are neglected, we derive a degree of constraint-relative behavioural profile compliance of $\mathcal{CBC}_{c_2} = \frac{38}{46} \approx 0.83$. Here, the interleaving order defined by the process model between activities C and F as well as for B in relation to itself are not considered in the compliance assessment.

Co-occurrence Compliance. Beyond execution order and exclusiveness, causal dependencies between activities have to be taken into account. As discussed in Section 3.2, causality of activity execution comprises two orthogonal aspects, the order of activity occurrences and their causal coupling. In the causal behavioural profile, the former is addressed by the relations of the behavioural profile, while the co-occurrence relation captures the latter aspect. For any compliance assessment of a case, therefore, the former aspect is considered in the behavioural profile compliance. Consequently, a second compliance measure is introduced to cope with the co-occurrence constraints that are induced by the process model. Informally, we check whether all activities for which the occurrence is implied by the current state of the case are in the case as well.

In general, the ratio of co-occurrence constraints of the process model that are met in the case and all co-occurrence constraints would be a straight-forward measure for this aspect. However, we want to consider also cases that may not have completed yet, such that activities that are missing according to a co-occurrence constraint might be added later on. Hence, not all co-occurrence constraints of the process model are required to be met in the case. Instead, we consider only those activities that are required to be

in the case by a co-occurrence constraint, for which we can deduce from the case that they should have already been observed. That is, an activity is considered, if it is either in the case, or it is in strict order with one of the activities in the case. Consider case $c_4 = \langle I, C, E \rangle$ of our initial example. The process model in Figure 1 specifies that an occurrence of activity I implies an occurrence of both, activities A and O. Activity A is not in the case although we know that it should have been observed already owing to the strict order relation between A and both activities, C and E. The mandatory activity O, in turn, is not required to occur in the case, as the case does not contain any activity that is in strict order with O.

Besides the question of incomplete cases, the normalisation of a degree of co-occurrence compliance deserves further discussion. Again, the normalisation may be based either on the number of constraints imposed by the process model, or the number of activities in the case. First, the degree is normalised by the number of *potential* co-occurrence constraints. We refer to this degree as *model-relative* co-occurrence compliance. Second, the degree is normalised based on the number of constraints. Then, the *constraint-relative* degree is the ratio of the satisfied co-occurrence constraints and all *actual* co-occurrence constraints for activities that are in the case or can be expected to be in the case. Both degrees of compliance for co-occurrence constraints are formalised as follows.

Definition 11 (Co-Occurrence Compliance).
Let $c = \langle n_1, \ldots, n_m \rangle$ be a case and $P = (A, a_i, a_o, C, F, T)$ a process model with $A_c \subseteq A$.

- The *set of expected case activities* $EA_c \subseteq A$ contains all activities that are in the case or that can be expected to be in the case, i.e., $EA_c = A_c \cup \{a \in A \mid \exists\, b, d \in A_c \,[\, a \rightsquigarrow_P d \;\wedge\; b \gg_P a \;\wedge\; (b = d \;\vee\; b \rightsquigarrow_P d) \,]\}$.
- The *set of expected case activity pairs* $EP_c \subseteq (A \times A)$ is defined as $EP_c = (EA_c \times A_c) \setminus id_{A_c}$.
- The degree of *model-relative co-occurrence compliance* of c to P is defined as

$$\mathcal{MCC}_c = \frac{|(EA_c \times A_c) \setminus id_{A_c} \cap \gg_P| + |EP_c \setminus \gg_P|}{|EP_c|}.$$

- The degree of *constraint-relative co-occurrence compliance* of c to P is defined as

$$\mathcal{CCC}_c = \begin{cases} 1 & \text{if } \gg_P = \emptyset, \\ \frac{|(EA_c \times A_c) \setminus id_{A_c} \cap \gg_P|}{|EP_c \cap \gg_P|} & \text{else.} \end{cases}$$

Computation of both degrees requires to determine the set of expected case activities. As these activities are identified by analysing relations between three activities, this step requires at most iteration over the triples of all activities in the process model. Hence, it requires cubic time with respect to the size of the model. Once this set is determined, the degrees are derived by iterating at most over the Cartesian product of activities of the model. Again, we conclude that the computation of these degrees does not add to the complexity needed to derive behavioural profiles for process models, cf., Section 3.2. We illustrate co-occurrence compliance using our initial example and case $c_2 = \langle I, A, C, B, G, F, O \rangle$. We see that, for instance, $C \gg D$ is not satisfied. This is penalised as the case contains G and it holds $D \rightsquigarrow G$ in the process model. In other words, the occurrence of G in the case provides us with evidence that we should have observed D, too. The same holds true for activity E, which can be expected to be in

the case. Computation of the model-relative degree of co-occurrence compliance yields a value of $\mathcal{MCC}_{c_2} = \frac{64}{72} \approx 0.89$. Here, eight constraint violations are assessed relative to the number of potential co-occurrence constraints. For instance, the activity pair (C, A) is taken into account even though it holds $C \not\gg A$. Computation of the constraint-relative degree of co-occurrence compliance, in turn, yields a value of $\mathcal{CCC}_{c_2} = \frac{36}{44} \approx 0.82$. That is, 36 out of 44 co-occurrence constraints of activities that can be expected to be in the case are satisfied. For case $c_4 = \langle I, C, E \rangle$, the absence of activity A is penalised. From the co-occurrence constraint $I \gg A$ we deduced that activity A is mandatory for completing the process, i.e., it is a mandatory activity. Due to the constraint $A \rightsquigarrow C$, we also know that activity A should have been observed already in the case. For this case, we compute compliance values of $\mathcal{MCC}_{c_4} = \frac{9}{12} = 0.75$ and $\mathcal{CCC}_{c_4} = \frac{5}{8} \approx 0.63$, respectively.

It is worth to mention that the co-occurrence compliance degree might be *overestimated*. An example for this phenomenon would be the case $\langle I, A, J \rangle$ for the model in Figure 1. There is a causal coupling $J \gg B$ in this model. However, the absence of B would not be penalised as there is no activity in the case that is in strict order with B and, therefore, would provide us with sufficient evidence that B should have already been observed.

4.4. Aggregated Compliance Measures

The compliance degrees for the separate compliance aspects introduced in the previous section are the foundation for aggregated measures for the compliance of case. Such an aggregated measure is defined as the sum of the enumerators divided by the sum of the denominators of the respective degrees. Hence, differences in the denominators are taken into account.

The first compliance measure combines the two constraint-relative compliance measures. Here, differences in the denominators stem from the fact that activities that are not contained in the case but are expected to be there are considered in the co-occurrence compliance, but not in the behavioural profile compliance.

Definition 12 (Constraint-Relative Case Compliance).
Let $c = \langle n_1, \ldots, n_m \rangle$ be a case and $P = (A, a_i, a_o, C, F, T)$ a process model with $A_c \subseteq A$. Let PC_c be the set of profile consistent case pairs, EA_c the set of expected case activities, and EP_c the set of expected case activity pairs. The *constraint-relative case compliance* of c to P is defined as

$$\mathcal{CC}_c = \begin{cases} 1 & \text{if } (\gg_P = \emptyset) \wedge (||_P = (A_c \times A_c)), \\ \frac{|PC_c \setminus ||_c| + |(EA_c \times A_c) \setminus id_{A_c} \cap \gg_P|}{|(A_c \times A_c) \setminus ||_c| + |EP_c \cap \gg_P|} & \text{else.} \end{cases}$$

Model-relative case compliance builds upon the model-relative measures for behavioural profile compliance and co-occurrence compliance. Again, differences in the denominators may stem from activities that are not in the case, but are expected to be there.

Definition 13 (Model-Relative Case Compliance).
Let $c = \langle n_1, \ldots, n_m \rangle$ be a case and $P = (A, a_i, a_o, C, F, T)$ a process model with $A_c \subseteq A$. Let PC_c be the set of profile consistent case pairs, EA_c the set of expected case activities, and EP_c the set of expected case activity pairs. The *model-relative case compliance* of c to P is defined as

$$\mathcal{MC}_c = \frac{|PC_c| \ + \ |(EA_c \times A_c) \setminus id_{A_c} \cap \gg_P| + |EP_c \setminus \gg_P|}{|A_c|^2 \ + \ |EP_c|}.$$

Table 1: Compliance results for the cases of the initial example (cf., Section 2)

Case	$\mathcal{CBC}$ Constr.-Rel. Beh. Profile	$\mathcal{MBC}$ Model-Rel. Beh. Profile	$\mathcal{CCC}$ Constr.-Rel. Co-Occurrence	$\mathcal{MCC}$ Model-Rel. Co-Occurrence	$\mathcal{CC}$ Constr.-Rel. Compliance	$\mathcal{MC}$ Model-Rel. Compliance
$c_1 = \langle I, A, E, C, D, F, G, O \rangle$	1.00	1.00	1.00	1.00	1.00	1.00
$c_2 = \langle I, A, C, B, G, F, O \rangle$	0.83	0.84	0.82	0.89	0.82	0.87
$c_3 = \langle I, A, B, J, H, B, O, G \rangle$	0.80	0.84	0.69	0.85	0.74	0.85
$c_4 = \langle I, C, E \rangle$	1.00	1.00	0.63	0.75	0.80	0.86
$c_5 = \langle F, C, D, G \rangle$	1.00	1.00	0.50	0.62	0.64	0.72

Applied to our example process model and the five exemplary cases introduced in Section 2, our compliance measures yield the results illustrated in Table 1. As expected, the first case c_1, which represents a valid execution sequence of the process model satisfies all constraints, such that our measures indicate full overall compliance. In contrast, the constraints induced by the relations of the behavioural profile of the process model are not fully satisfied in the second case c_2 (e.g., the exclusiveness in the model between B and C is broken in the case). In addition, the co-occurrence dependencies are not completely respected in the case either, as discussed above. For case c_3, similar observations can be made leading to overall compliance values between 0.74 and 0.85. Here, the differences in the normalisation of our two aggregated compliance measures become visible. An assessment that is relative to the size of the executed part of the process model leads to a higher compliance value. That is due to the fact that pairs of activities, for which there is no explicit constraint in the process model, lower the impact of activity pairs with violated constraints. Regarding case c_4, we already discussed the absence of activities that are mandatory for completion of the process and, therefore, should be contained in the case. We see that this deviation impacts on the compliance degrees that consider the co-occurrence relation. The compliance degrees based on the behavioural profile equal one, as case c_4 does not violate any ordering or exclusiveness constraints. Similarly, case c_5 shows full behavioural profile compliance as the ordering constraints imposed by the process model are satisfied. Still, case c_5 is incomplete as it represents a valid subtrace of the process model, whereas the first part of the case (comprising activities I, A, and E) is missing. Consequently, various co-occurrence constraints that impact on the compliance values are violated.

The compliance measures based on the relations of the behavioural profile ($\mathcal{CBC}$ and $\mathcal{MBC}$) are largely independent of those that rely on the co-occurrence relation ($\mathcal{CCC}$ and $\mathcal{MCC}$). This follows from the fact that, besides one exception, the respective behavioural

relations are orthogonal, cf., Section 3.2. Therefore, the aggregated measures ($\mathcal{CC}$ and $\mathcal{MC}$) should be used in order to assess compliance by taking the complete spectrum of constraints as imposed by the causal behavioural profile into account.

Regarding the actual compliance values obtained with the proposed measures, conclusions can only be drawn against the background of a concrete process and environment. There is a variety of factors that influence the question whether or not a certain degree of non-compliance is acceptable. The severity of the implications that follow from non-compliance behaviour and the reliability of the logging mechanism would be examples for such factors.

5. Compliance Measurement and Noise

Compliance measurement builds on the assumption that a normative process model exists and that the process log gives an accurate account of how individual cases have been processed. Research on process mining has acknowledged the existence of noise in real-world log and implemented measures to deal with it. Noise stems among others from inaccurate logging mechanisms in information systems or race conditions when writing two log entries. Also, the execution order of activities may not be enforced explicitly or people may deliberately work around the system. A classification of noise patterns has been introduced by Weijters et al. [27], which has been extended by Günther [28]. Based thereon, we distinguish two major categories of noise: *missing parts of traces* and *perturbation*. We use these categories to discuss the impact of noise on the compliance measurement results.

Missing parts of a trace are due to the fact that the logging mechanism of a process-aware information system was not available from a particular period of time. In case of a missing head, the recording started only after the case was already in processing; a missing trail can result when cases are still processed when the analysis period is closed; and a missing episode might stem from a temporarily deactivated logging mechanism (see Figure 2). These three cases have in common that a part of the original trace is missing. Obviously, this category of noise does not change the order between activities in the trace. Therefore, the behavioural profile compliance measures are not affected by this kind of noise. In contrast to that, the co-occurrence measures are penalized by missing parts. This penalty depends on how many co-occurrence relations exist between the activities of the missing part and the rest of the trace.

Perturbation involves a wrong recording of activity order, the wrong recording of an additional event, or the recording of alien events. These perturbations affect the compliance measures to a different degree. Consider the case that two events are recorded in wrong order, for instance due to a race condition in the logging mechanism (see Figure 3, top). If there is an order constraint between these two activities, we now observe a violation in the perturbed trace. If both were in interleaving order anyways, then the compliance degree is not affected. In the same vein, ordering constraints between the two perturbed activities and the activities that occur between them are affected. We observe violation, if an activity was not in interleaving order, but in strict order with one of the perturbed activities. Co-occurrence constraints are not violated by this kind of noise, as the set of activities in the case does not change. Figure 3, middle, shows the case of an additional record of an event. In this case, the perturbed trace shows violations for all activities between the first and the second record if an order constraint

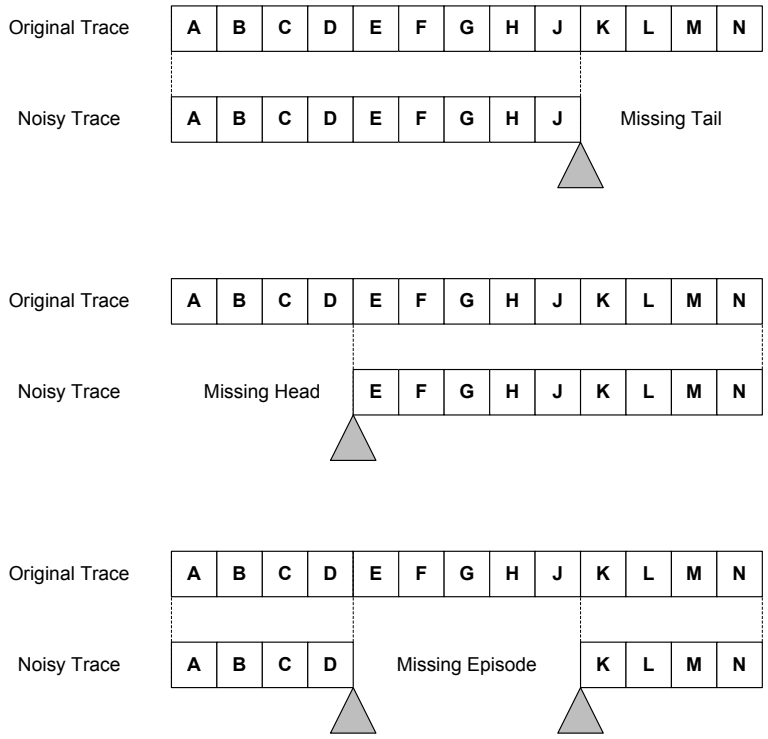

Figure 2: Types of Noise with Missing Parts of the Trace

was defined between them and the repeated activity. Again, co-occurrence relations are not violated. In case an alien event is record (see Figure 3, bottom), neither order relations nor co-occurrence relations are violated.

Altogether, it can be noted that behavioural profile compliance is robust against noise in terms of missing parts of traces and alien events. Co-occurrence compliance is robust against perturbation noise. In case the impact of noise matters, it depends on the size of the missing part and on the number of perturbed events.

6. Diagnostic Information on Non-Compliance

The measures introduced in Section 4 give us an overall insight into the compliance of a single case in its relation to a normative process model. In order to identify reasons for non-compliant processing, diagnostic information on compliance violations has to be derived. In particular, the root cause of a compliance violation should be identified. First, we discuss concepts for such a root cause analysis on the level of a single case in Section 6.1. Then, Section 6.2 turns the focus on diagnostic information for a process log.

6.1. Root Cause Analysis for a Single Case

To identify reasons for non-compliant processing in a single case, first and foremost, it has to be known which kind of violations are observed. Such feedback is given as a set of

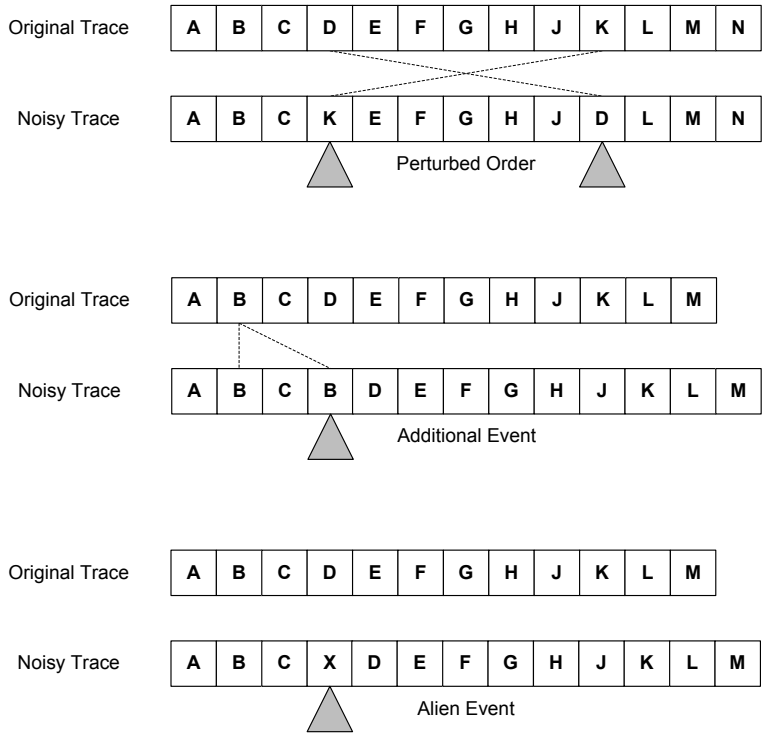

Figure 3: Types of Noise with Perturbation

triples, referred to as *compliance violation triples*, each consisting of a pair of activities along with the violated constraint, i.e., a relation of the causal behavioural profile. Against the background of our pairwise compliance assessment, feedback on non-compliance in a case is also given for single activities. That is, for each activity we count the number of constraint violations in which this activity is involved. As a single non-compliant execution of an activity may result in various constraint violations related to this activity, this kind of feedback highlights activity executions that are most problematic in the sense that they heavily impact on the overall compliance measures. The ratio between the violated constraints that relate to a certain activity and all violated constraints is referred to as the *compliance violation impact* of the activity. This value measures how much of the observed compliance violation is related to the respective activity.

Definition 14 (Compliance Violation Triple and Impact).
Let $c = \langle n_1, \ldots, n_m \rangle$ be a case of a process model $P = (A, a_i, a_o, C, F, T)$ and $\mathcal{B}_P^+$ the causal behavioural profile of P. Let PC_c be the set of profile consistent case pairs and EA_c the set of expected case activities.

- The set of *compliance violation triples* $\mathcal{V}_c \subseteq (A \times A \times \mathcal{B}_P^+)$ for c contains all pairs (x, y, R) such that $(x\ R\ y)$ and either $(x, y) \notin PC_c$ or $(x \in EA_c\ \wedge\ x \gg_c y\ \wedge\ x \neq y\ \wedge\ y \notin A_c)$.

Table 2: Feedback on non-compliance for case c_3 of the example (cf., Section 2)

$c_3 = \langle I, A, B, J, H, B, O, G \rangle$	
Compliance Violation Triples	(F,C,≫),(C,F,≫),(G,D,≫),(F,E,≫), (G,E,≫), (D,F,≫),(F,D,≫),(E,C,≫),(E,D,≫),(C,E,≫), (G,F,≫),(E,F,≫), (D,C,≫),(G,C,≫),(D,E,≫), (C,D,≫), (H,G,+),(J,G,+),(B,G,+),(G,B,+), (G,J,+),(G,H,+),(O,G,$\rightsquigarrow^{-1}$),(G,O,$\rightsquigarrow$)
Compliance Violation Impact	$\mathcal{VI}(G) = 0.5$ $\mathcal{VI}(D) = \mathcal{VI}(E) = \mathcal{VI}(C) = \mathcal{VI}(F) = 0.29$ $\mathcal{VI}(B) = \mathcal{VI}(H) = \mathcal{VI}(J) = \mathcal{VI}(O) = 0.08$

- The *compliance violation impact* of an activity $a \in A$ is defined as

$$\mathcal{VI}(a) = \frac{|\{(x, y, R) \in \mathcal{V}_c \mid x = a \vee y = a\}|}{|\mathcal{V}_c|}.$$

As illustrated in Table 1, four out of five of our example cases show overall compliance values below one. Table 2 depicts the compliance violation triples for case c_3. Those indicate the concrete problems that have been found for the case. In addition, we also illustrate the compliance violation degrees of single activities. Evidently, the execution of activity G is most problematic for case c_3. Half of the violated constraints relate to this activity, so that its execution can be seen as the root cause for the non-compliance of this particular case.

6.2. Root Cause Analysis for a Process Log

By applying our compliance measures to all the cases of a process log and computing their average values, we establish an understanding of overall compliance of our process execution. In this case, feedback on non-compliance should not be limited to single cases. Instead, the frequency with which certain violations are observed has to be known in order to identify the reasons for non-compliant processing in general. In addition, dependencies between compliance violations can also be seen as valuable information on non-compliance. That is due to the fact that a certain violation may simply be caused by another violation that happened before.

In order to address the need for aggregated analytic information on non-compliance for a process log, we focus on violation triples as introduced in the previous section for a single case. Further, we adapt the notions of *support* and *confidence* known from the field of association rules mining [29, 30]. The common formalism used for association rules mining identifies patterns that are built of items given a set of transactions. These transactions, in turn, are built of items. Adapted to our setting, a transaction is represented by a single case, while an item is a certain constraint violation that may be observed in a case. We define the support for a dedicated compliance violation triple as the number of cases in a log in which it can be observed.

Table 3: Compliance violations (support > 1) for the example log

Compliance Violation Pairs	Support
(C,E,≫),(D,E,≫),(G,E,≫),(F,E,≫)	3
(G,D,≫),(B,G,+),(E,D,≫),(C,D,≫),(I,A,≫), (E,A,≫),(F,D,≫),(G,B,+),(C,A,≫)	2

Definition 15 (Support for Compliance Violation).
Let $C = \{c_1, \ldots, c_n\}$ be a log of a process model $P = (A, a_i, a_o, C, F, T)$.

- The *set of logs supporting a compliance violation triple* $v \in (A \times A \times \mathcal{B}_P^+)$ in C is defined as $\mathcal{SU}(v) = \{c_i \in C \mid v \in \mathcal{V}_{c_i}\}$.
- The *support for a compliance violation triple* $v \in (A \times A \times \mathcal{B}_P^+)$ in C is defined as $sup(v) = |\mathcal{SU}(v)|$.

Table 3 shows the compliance violation triples with support more than one for the cases of the example log introduced in Section 2. It illustrates that there are four compliance violation triples that relate to three out of the five cases. Moreover, all of them represent violations of co-occurrence dependencies that imply the execution of activity E. This observation, in turn, provides a starting point for the analysis of the respective process. The reasons for missing executions of activity E have to be determined as root causes for non-compliant processing.

While the support for compliance violation triples helps to separate frequent and rare compliance violations, the analysis of non-compliance is more effective if dependencies between violations are taken into account. A deviation from the processing as specified in the process model might cause several subsequent violations of the defined control flow. Such dependencies stem from data dependencies between activities that are often not explicitly captured in the process model. In order to detect these dependencies and focus on the actual cause of a series of compliance violations, we adapt the notion of confidence of association rules. Confidence relates rules between items to their statistical significance and, therefore, reflects the strength of a rule. In our setting, a rule is an implication between two compliance violation triples, for which we define confidence as follows.

Definition 16 (Confidence for Compliance Violation Rule).
Let $C = \{c_1, \ldots, c_n\}$ be a log of a process model $P = (A, a_i, a_o, C, F, T)$. For two distinct compliance violations triples $v_1, v_2 \in (A \times A \times \mathcal{B}_P^+)$ the *confidence for a violation rule* from v_1 to v_2 is defined as

$$conf(v_1 \Rightarrow v_2) = \begin{cases} 0 & \text{if } \ sup(v_1) = 0, \\ \frac{|\mathcal{SU}(v_1) \cup \mathcal{SU}(v_2)|}{sup(v_1)} & \text{else.} \end{cases}$$

Analysis of compliance violation rules is reasonable solely for compliance violation triples for which the support exceeds a certain threshold in the process log. This threshold has to be defined depending on the number of cases in a log, even though it may be adapted

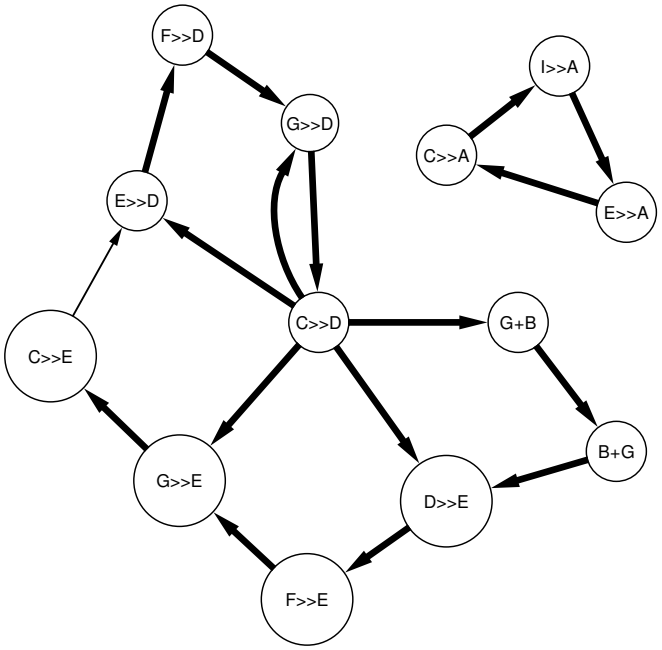

Figure 4: Rules between compliance violation triples (confidence > 0.6) for the example log in Section 2

as a part of the analysis. Similarly, rules that exceed a certain threshold with respect to their confidence should be investigated.

Figure 4 depicts the rules between compliance violation triples for the example log introduced in Section 2. Here, nodes depict compliance violation triples that show a support larger than one (i.e., those that are listed in Table 3), while the node size reflects the different support values. Edges represent rules between compliance violation triples for which the confidence value is above the threshold of 0.6. Again, the edge strength depends on the confidence value. In our example, all except for one rule show a confidence value of one. That is, the occurrence of the source compliance violation triple always implies the occurrence of the target compliance violation triple. Note that we did a transitive reduction for the edges in the graph. A transitive reduction is not unique in case of a cyclic graph, while identification of the minimal transitive reduction is an NP-complete problem [31]. In order to provide an overview of the interplay of compliance violation rules, however, one non-minimal reduction is sufficient. When interpreting the results, we see that there are two independent clusters of compliance violation triples that manifest in disconnected subgraphs. These cluster represent compliance violations that occur independent from each other and, therefore, have to be analysed separately. Focussing on the bigger subgraph, we see that the compliance violation related to the co-occurrence between activities C and D implies various other compliance violations. Although this violation cannot be seen as the only root cause (it is part of a cycle of compliance violation rules), there is some evidence that this violation is fundamental and many other violations are causally dependent. Hence, the implementation of the process should be investigated for reasons that break the causality between activities C and D.

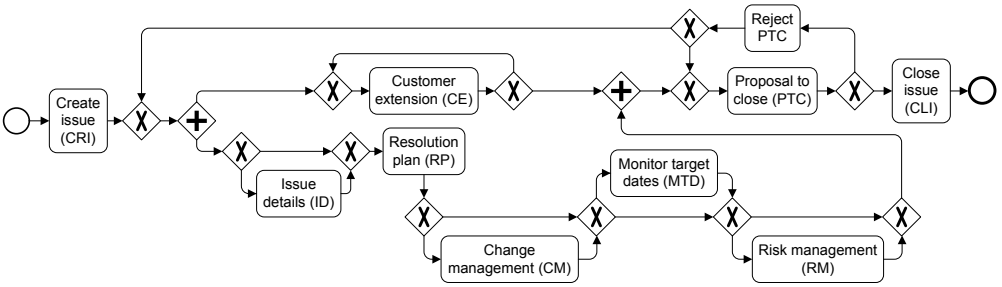

Figure 5: BPMN model of the Security Incident Management Process (SIMP)

We restricted our discussion on rules between two compliance violation triples. However, the introduced concepts may be lifted to rules between more than two compliance violation triples in a straight-forward manner.

7. Case Study: Security Incident Management Process

To demonstrate and evaluate our approach, we implemented all introduced concepts in a prototypical tool and applied it in a case study on the Security Incident Management Process (SIMP). In this section, first, we give background information on this process. Second, we present the results for a process log using the compliance measures introduced in this paper and discuss related measures. Third, the concepts introduced for the root cause analysis of non-compliance are applied to the log.

Background. SIMP is an issue management process used in global service delivery centres. The process and the log have been minimally modified to remove confidential information. Figure 5 shows the BPMN model of SIMP solicited from domain experts.

SIMP is used in one of IBM's global service delivery centres that provides infrastructure management and technical support to customers. When a customer reports a problem or requests a change, an *issue* is created, spawning a new instance of the process. Details about the issue *may* be updated, a plan to resolve the issue *must* be created, and change management related activities *may* be performed if required. Then, target dates for issue resolution *may* be monitored and relevant risks *may* be documented. A *Customer Extension* of target dates may be processed during any of the above activities (parallel path). Once the steps for resolution are taken and verified, the resolver *must* propose to close the issue. Based on the evidence that the issue is indeed resolved, the issue creator *may* close the issue. Otherwise, the proposal *must* be rejected.

For the SIMP, we analysed 852 cases, each consisting of a set of log entries. Such a log entry has an activity name, activity description, and the time-stamp marking the time of execution of the activity. Although the process is standardized and documented, it is not orchestrated via workflow tools in the IBM's global service delivery centre under investigation. Instead, it is manually carried out. Hence, the employees are free to deviate from the process. As a result, the cases may or may not specify valid execution sequences of the process model. The process log has been created using a proprietary tool, in which an employee submits the execution of a certain activity. Correlation of log entries to cases has been managed explicitly by the logging tool.

Table 4: Compliance results for the SIMP derived from 852 cases

	$\mathcal{CBC}$ Constr.-Rel. Beh. Profile	$\mathcal{MBC}$ Model-Rel. Beh. Profile	$\mathcal{CCC}$ Constr.-Rel. Co-Occurrence	$\mathcal{MCC}$ Model-Rel. Co-Occurrence	$\mathcal{CC}$ Constr.-Rel. Compliance	$\mathcal{MC}$ Model-Rel. Compliance
Avg	0.98	0.99	0.96	0.96	0.97	0.97
StDev	0.06	0.04	0.11	0.09	0.08	0.069
Min	0.08	0.31	0.60	0.60	0.53	0.58
Max	1.00	1.00	1.00	1.00	1.00	1.00
Share of Compliant Cases						
	78.64%	78.64%	84.39%	84.39%	76.29%	76.29%

Compliance Measures. For each case, we analysed its compliance using the measures proposed in this paper. Table 4 gives a summary of this analysis in terms of the average compliance value of all cases (using the arithmetic mean) along with the standard deviation, the observed minimal and maximal compliance values, and the share of fully compliant cases. The latter show a value of one for all compliance degrees. The compliance values were discussed with the manager of the process. The average values reflect the manager's perception that SIMP is running satisfactory and most cases are handled in a compliant way. While up to a quarter of the cases does represent fully compliant behaviour, the high average values and the low standard deviation for our compliance measures indicate that there are solely marginal deviations in most cases. Still, as the minimum values show, it was also possible to identify cases of very low compliance. Here, behavioural compliance values of 0.08 or 0.31 represent outliers that have been caused by a case that mixed the log entries for two separate process instances. Moreover, it is interesting to see that the different normalisations, either based on the number of constraints or the considered process model fragment, does not impact on the compliance values for our case study significantly.

We are not able to directly compare our results with the fitness measure proposed in [10] and discussed in detail in Section 2. This is mainly due to the inherent complexity of the state space exploration, which is exponential in the general case. Even a maximally reduced Petri net of our SIMP contains a lot of silent steps owing to several activities, for which execution is optional. That, in turn, leads to a significant increase of the state space to investigate when trying to replay a case. While compliance values might still be derived following the most greedy strategy, these results are of a limited validity as they highly underestimate the degree of compliance. However, it is worth to mention that even with the most greedy strategy, computation of the compliance values for the process log took around 15 seconds. In contrast, computing the compliance measures proposed in this paper for the process log, in turn, was done within milliseconds. Moreover, an isolated analysis of a sample of 30 cases for which computation of the fitness measure is possible with a 5-step-ahead strategy revealed that the fitness compliance values are all lower than the values derived by our measures. This is in line with the criticism of [17] that the fitness concept appears to be rather strict.

The differences between the fitness values and our compliance values stem from the

Table 5: Feedback on non-compliance for a dedicated case

Compliance	(CLI,CLI,+),(CM,CE,≫),
Violation	(CRI,CE,≫),(RPTC,CE,≫),
Triples	(CLI,CE,≫),(PTC,CE,≫),(RP,CE,≫)
Compliance	$\mathcal{VI}(CE) = 0.86$
Violation	$\mathcal{VI}(CLI) = 0.29$
Impact	$\mathcal{VI}(RP) = \mathcal{VI}(CM) = 0.14$
	$\mathcal{VI}(CRI) = \mathcal{VI}(RPTC) = \mathcal{VI}(PTC) = 0.14$

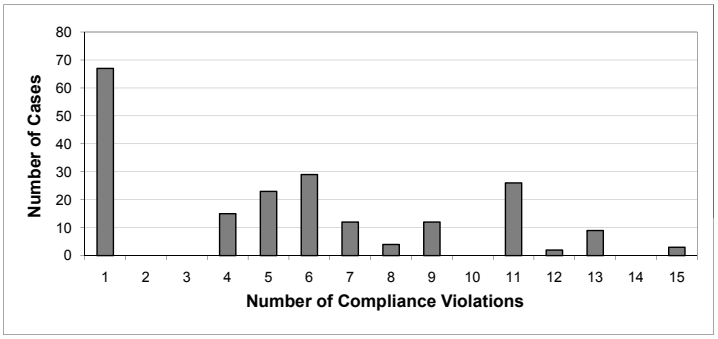

Figure 6: Number of cases (out of 852) for which we observed a certain number of compliance violation triples

particular grounding of the compliance measures. We illustrate this aspect using case $c_4 = \langle I, C, E \rangle$ of our initial example. In Section 2, we discussed that the fitness for this case is 0.33, whereas Section 4.4 shows that we derive aggregated compliance values of 0.80 or 0.86, respectively. In this case, the absence of a single activity A is the source of compliance violation. In the computation of the fitness measure, such a single violation may impact on a large number of activities in the case (or even all activities), i.e., it may lead to a large number of non-compliant activity executions. In contrast, such a violation impacts solely on the relations between the respective activities in the computation of our compliance measures, while the relations between the remaining activities are assessed independently. Hence, our compliance measure tend to yield higher compliance values.

Root Cause Analysis. We collected the compliance violation triples for all cases of the log for which we observed an overall compliance value (using constraint-relative or model-relative normalisation) below one. Figure 6 gives an impression on the amount of the collected violation triples and how they are distributed over the cases. For each number of compliance violation triples, the chart depicts the number of cases in the collection that showed an according number of violations. For nearly 70 cases we observe solely a single constraint violation. Further, at most 15 compliance violations are detected in a single case. Hence, we conclude that in our setting, the set of violated constraints for single case can still be handled by a process analyst.

Table 6: Compliance violations (support > 20) observed in the set of 852 cases

Compliance Violation Pairs	Support
(CLI,CLI,+)	177
(CRI,RP,≫),(CE,RP,≫),(CLI,RP,≫),(PTC,RP,≫)	109
(CRI,CE,≫),(CLI,CE,≫),(PTC,CE,≫),(RP,CE,≫)	74
(CM,CE,+)	35
(CM,RP,+)	34

In order to demonstrate the application of the concepts to support root cause analysis for a single case, Table 5 depicts the compliance violation triples for a dedicated case along with the compliance violation impact of the contained activities (see Figure 5 for the resolution of the activity abbreviations). The chosen case has an overall compliance value of 0.86 or 0.91, depending on the applied normalisation. Hence, the case is a typical example for a non-compliant case that shows minor behavioural deviations from the normative process model. Evidently, there is a single activity that participates in more than 80% of the violated behavioural constraints. The activity representing the *customer extension* (*CE*) can, therefore, be seen as the root cause of the compliance violation. The compliance violation triples also indicate the type of violation that is related to this activity. The violated co-occurrence constraints all require the occurrence of activity *CE*, so that the absence of this activity causes most of the compliance violation. The fact that the absence of activity *CE* does not explain the whole compliance violation is due to the violated exclusiveness for the activity *close issue* (*CLI*) in relation to itself. That is, the case comprises two log entries that report an execution of this activity even though the process model allows for at most one execution.

We demonstrate the root cause analysis just for one exemplary case. However, a review of our results suggests that similar observations can be made for most of the non-compliant cases.

Turning the focus on the root cause analysis for the whole process log, Table 6 lists the compliance violation triples with the highest support in the set of 852 cases. The deviation from the process model that is observed most frequently relates to the activity *close issue*. While it can be executed at most once in the process model, 177 cases record at least two executions of this activity. Moreover, in a large number of cases, co-occurrence relations that require the presence of the activity representing the *resolution plan* (*RP*) or the *customer extension* (*CE*) are violated. Although both activities are mandatory for the completion of the process according to the process model in Figure 5, they are absent in a large number of cases. Based on these results, the necessity to deviate from the standard processing for closing an issue and the potential to skip the execution of activities *RP* and *CE* under certain circumstances can be evaluated by the management of the Security Incident Management Process. Still, any judgement on whether these deviations are acceptable can only be done once the reasons for repeating or skipping the respective activities have been investigated in the concrete cases. Such investigations have not yet been complete for SIMP as illustrated in this case study.

Finally, the identified dependencies between the identified compliance violations are

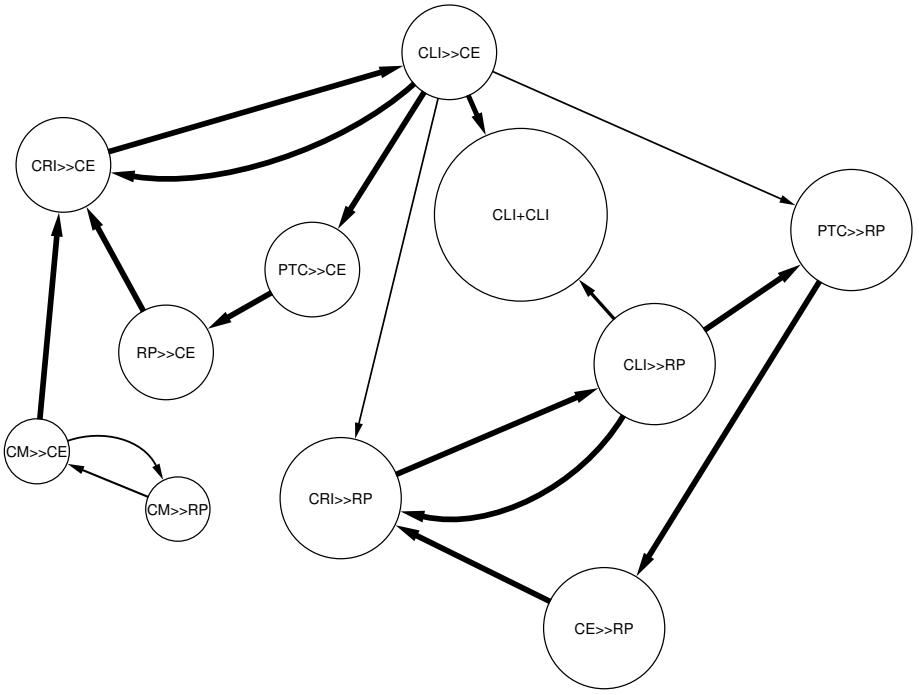

Figure 7: Rules between compliance violation triples (confidence > 0.6) observed in the set of 852 cases

visualised in Figure 7. In this graph, nodes depict compliance violation triples with a support larger than 20 in the collection of 852 cases, while edges represent rules between them that have a confidence above the threshold of 0.6. Again, the node size reflects the support values, while the edge strength correlates with the confidence value. The graph suggests that the violation of the exclusiveness constraint for activity *close issue* (CLI) is independent of the other frequently observed violations. Further, the violations of the co-occurrence constraints related to activities RP and CE build clusters of rules of high confidence. Hence, these violations always occur together in a case. Still, we see that both clusters are rather independent of each other. We conclude that the absence of activities RP and CE, along with the constraint related to activity CLI, have to be seen as independent root causes of the non-compliant behaviour in our process log.

8. Related Work

Compliance measures are at the core of process mining that aims at automatic construction of a process model from a process log [1, 2, 3, 4, 5]. Similar relations, but not exactly those of behavioural profiles, are used in [32] to characterise a process as a pre-processing step for deriving a model. Please refer to [19] for a discussion of the conceptual differences between the relations used in [32] and those of the behavioural profile.

As mentioned before, there are various different approaches for measuring compliance of a log in the sense of its fitness. That is, the share of cases that can be replayed in the

process model can be used as a compliance measure [11, 12]. Beyond these coarse-grained approaches, a case may be replayed step-wise to quantify the number of execution steps that conform to the process model semantics [9, 10, 13]. All of these approaches are based on state concepts and, hence, have to cope with the state explosion problem. In this paper, we demonstrated that our approach benefits from the efficient calculation of the behavioural profiles from free-choice process models as defined in [18, 19], cf., Section 3.2. Therefore, in contrast to the classical fitness calculation, our measures can be computed within milliseconds. Further, our case study provided us with evidence that our measures are close to managers' perception of compliance.

The visualisation of log data to enable effective analysis has been addressed in recent work. In [33], the authors advocate the application of a dotted chart analysis to assess the performance of business operations with a focus on their time dependencies. Such a chart supports the manual analysis of long-running instances, in particular. Another approach leverages multi sequence alignment techniques known from bioinformatics to construct a so called trace alignment [34]. Once such an alignment between cases has been established, patterns of common behaviour and rare deviations may be identified. The drawback of this approach is the inherent complexity as finding an optimal alignment for a set of cases is a computationally hard problem. Nevertheless, [34] already showed the application of the technique in two case studies. As such an alignment assumes a different perspective compared to our feedback on behavioural constraint violations, both approaches can be seen as complementary. Further, the detection of differences between process models, not logs, is also discussed in related work. The approaches presented in [35, 36] provides a systematic framework of diagnosis and resolution of such mismatches.

The concept of behavioural profiles in general relates to different notions of behavioural equivalence such as trace equivalence and bisimulation. These notions build on state concepts, which means that they cannot be decided efficiently in the general case. They also yield only a true or false answer and they are not directly applicable to execution sequences [37, 38]. Behaviour inheritance is closely related to these notions. Basten et al. [39] define protocol inheritance and projection inheritance based on labelled transition systems and branching bisimulation. A model inherits the behaviour of a parent model, if it shows the same external behaviour when all actions that are not part of the parent model are either *blocked* (protocol inheritance) or *hidden* (projection inheritance). Similar ideas have been presented in [40, 41]. The boolean characteristics of these notions have been criticized as inadequate for many process measurement scenarios [10].

The question of process similarity has been addressed from various angles. Focussing on behavioural aspects, [42, 43] introduce similarity measures based on an edit distance between workflows. Such an edit distance might be based on the language of the workflow, the underlying automaton, or based on the n-gram representation of the language. A similar approach is also taken in [44], in which the authors measure similarity based on high-level change operations that are needed to transform one model into another. Close to our behavioural abstraction of a behavioural profile are causal footprints as introduced in [45]. The authors also show how the footprints can be leveraged to determine the similarity between process models. All these similarity notions are either expensive in terms of calculation, whereas behavioural profiles can be calculated efficiently for a broad class of models.

The compliance of workflow executions with normative process models is also an important aspect of role-based access control (RBAC). In essence, role-based access

control deals with the specification and enforcement of constraints that relate to order and exclusiveness of roles or subjects executing particular activities of a workflow. Such constraints include among others separation of duty requirements. Separation of duty implies that either particular activities have to be exclusive altogether, or that those roles or subjects executing a pair of activities have to be exclusive (also referred to as four-eyes principle) [46, 47, 48]. The major share of research in this area has focussed on the specification and verification of RBAC policies [49, 50, 51], among others on consistency and satisfiability of constraint sets [52, 53], as well as on engineering and enforcement by design [54]. Log files have been used for mining roles in this area [55, 56], while an a posteriori compliance control has only been considered recently [57, 58, 59]. The approach reported in this paper informs this stream of research. Once process models are annotated with RBAC constraints as defined in [60] and a log includes role and subject information, the concept of causal behavioural profiles can be extended for checking also separation of duty constraints.

9. Conclusion

In this paper, we have discussed the challenges of providing compliance measurements and feedback on potential deviations in an efficient and effective way. Our contribution is a novel proposal of measures for compliance measurement based on behavioural constraints on pairs of activities. By using behavioural profiles as the underlying equivalence notion, we avoided performance problems of state-based measures. We discussed alternatives for normalising our measures and also elaborated on the impact of common noise patterns on them. In addition, our contribution comprises concepts that enable effective root cause analysis for non-compliant cases. We provide diagnostic information on the level of single cases, as well as for a process log. All these measures and concepts have been implemented and validated in a case study with an international service provider.

Still, we also have to reflect on some limitations of our approach. The approach relies on the assumption that all activities in a process model are unique. Although the behavioural profile can be lifted to labelled process models (one label may be assigned to more than one activity) without computational overhead [26], lifting the co-occurrence relation from activities to activity labels cannot be done efficiently. Besides these technical issues, an analysis based on labels can be expected to be less meaningful. For instance, two identically labelled activities at the start and the end of a process (think of a logging activity) would yield an interleaving order for this label and all other activity labels used in the process model. Hence, there would be no explicit ordering constraint that could be violated in a process log. Further, our approach distinguishes activities that may be executed at most once or multiple times, without considering a concrete number a certain activity gets executed. This abstraction is more severe in the context of labelled activities as there might be a specific number of activity executions that must be observed.

In future work, we aim to study the merits of our novel approach in further industry collaborations. The performance gain of using behavioural profiles is of serious importance for various use cases. Up until now, compliance measurement had to be conducted offline in a batch mode due to being very time consuming. We aim to investigate those scenarios where an instantaneous compliance measurement is valuable. In particular, compliance measurement in the financial industry might eventually benefit from this innovation, e.g., to cancel running transactions that exhibit non-compliant behaviour. Moreover, we aim

at investigating resolution strategies that may be proposed to mitigate non-compliant behaviour. Our root cause analysis proved to isolate compliance violations that are independent of each other. Hence, we assume that a resolution strategy in terms of a change operation for the process model can be derived automatically in many cases.

[1] W. M. P. van der Aalst, T. Weijters, L. Maruster, Workflow mining: Discovering process models from event logs, IEEE Trans. Knowl. Data Eng. 16 (9) (2004) 1128–1142.

[2] W. M. P. van der Aalst, H. A. Reijers, A. J. M. M. Weijters, B. F. van Dongen, A. K. A. de Medeiros, M. Song, H. M. W. E. Verbeek, Business process mining: An industrial application, Inf. Syst. 32 (5) (2007) 713–732.

[3] R. Agrawal, D. Gunopulos, F. Leymann, Mining process models from workflow logs, in: H.-J. Schek, F. Saltor, I. Ramos, G. Alonso (Eds.), EDBT, Vol. 1377 of Lecture Notes in Computer Science, Springer, 1998, pp. 469–483.

[4] A. K. A. de Medeiros, W. M. P. van der Aalst, A. J. M. M. Weijters, Workflow mining: Current status and future directions, in: R. Meersman, Z. Tari, D. C. Schmidt (Eds.), CoopIS/DOA/ODBASE, Vol. 2888 of Lecture Notes in Computer Science, Springer, 2003, pp. 389–406.

[5] A. Datta, Automating the discovery of as-is business process models: Probabilistic and algorithmic approaches, Information Systems Research 9 (3) (1998) 275–301.

[6] A. Rozinat, A. K. A. de Medeiros, C. W. Günther, A. J. M. M. Weijters, W. M. P. van der Aalst, The need for a process mining evaluation framework in research and practice, in: A. H. M. ter Hofstede, B. Benatallah, H.-Y. Paik (Eds.), Business Process Management Workshops, Vol. 4928 of Lecture Notes in Computer Science, Springer, 2007, pp. 84–89.

[7] J. D. Weerdt, M. D. Backer, J. Vanthienen, B. Baesens, A critical evaluation study of model-log metrics in process discovery, in: Proceedings of the 6th International Workshop on Business Process Intelligence (BPI 2010), Hoboken, NJ, USA, 2010.

[8] W. van der Aalst, V. Rubin, B. van Dongen, E. Kindler, C. Günther, Process mining: A two-step approach to balance between underfitting and overfitting, Software and Systems Modeling 9 (2010) 87–111.

[9] A. Rozinat, W. M. P. van der Aalst, Conformance checking of processes based on monitoring real behavior, Inf. Syst. 33 (1) (2008) 64–95.

[10] A. K. A. de Medeiros, W. M. P. van der Aalst, A. J. M. M. Weijters, Quantifying process equivalence based on observed behavior, Data Knowl. Eng. 64 (1) (2008) 55–74.

[11] A. Weijters, W. van der Aalst, A. A. de Medeiros, Process mining with the heuristicsminer algorithm, BETA Working Paper Series WP 166, Eindhoven University of Technology, Eindhoven (2006).

[12] G. Greco, A. Guzzo, L. Pontieri, D. Saccà, Discovering expressive process models by clustering log traces, IEEE Trans. Knowl. Data Eng. 18 (8) (2006) 1010–1027.

[13] S. Goedertier, D. Martens, J. Vanthienen, B. Baesens, Robust process discovery with artificial negative events, Journal of Machine Learning Research 10 (2009) 1305–1340.

[14] R. Glabbeek, Handbook of Process Algebra, Elsevier, 2001, Ch. The Linear Time – Brancing Time Spectrum I. The semantics of concrete, sequential processes.

[15] A. Valmari, The state explosion problem., in: W. Reisig, G. Rozenberg (Eds.), Lectures on Petri Nets I: Basic Models, Advances in Petri Nets, the volumes are based on the Advanced Course on Petri Nets, held in Dagstuhl, September 1996, Vol. 1491 of Lecture Notes in Computer Science, Springer, 1998, pp. 429–528.

[16] A. Adriansyah, B. van Dongen, W. van der Aalst, Towards robust conformance checking, in: Proceedings of the 6th International Workshop on Business Process Intelligence (BPI 2010), 2010.

[17] K. Gerke, J. Cardoso, A. Claus, Measuring the compliance of processes with reference models, in: R. Meersman, T. S. Dillon, P. Herrero (Eds.), OTM Conferences (1), Vol. 5870 of Lecture Notes in Computer Science, Springer, 2009, pp. 76–93.

[18] M. Weidlich, J. Mendling, M. Weske, Efficient consistency measurement based on behavioural profiles of process models, IEEE Transactions on Software Engineering. To appear.

[19] M. Weidlich, A. Polyvyanyy, J. Mendling, M. Weske, Efficient computation of causal behavioural profiles using structural decomposition, in: J. Lilius, W. Penczek (Eds.), Petri Nets, Vol. 6128 of Lecture Notes in Computer Science, Springer, 2010, pp. 63–83.

[20] M. Weidlich, A. Polyvyanyy, N. Desai, J. Mendling, Process compliance measurement based on behavioural profiles, in: B. Pernici (Ed.), CAiSE, Vol. 6051 of Lecture Notes in Computer Science, Springer, 2010, pp. 499–514.

[21] H. Schuldt, G. Alonso, C. Beeri, H.-J. Schek, Atomicity and isolation for transactional processes, ACM Trans. Database Syst. 27 (1) (2002) 63–116.

[22] O. I. Lindland, G. Sindre, A. Sølvberg, Understanding quality in conceptual modeling, IEEE Software 11 (2) (1994) 42–49.
[23] J. Vanhatalo, H. Völzer, F. Leymann, S. Moser, Automatic workflow graph refactoring and completion, in: A. Bouguettaya, I. Krüger, T. Margaria (Eds.), ICSOC, Vol. 5364 of Lecture Notes in Computer Science, 2008, pp. 100–115.
[24] W. M. P. van der Aalst, Workflow verification: Finding control-flow errors using petri-net-based techniques, in: W. M. P. van der Aalst, J. Desel, A. Oberweis (Eds.), Business Process Management, Vol. 1806 of Lecture Notes in Computer Science, Springer, 2000, pp. 161–183.
[25] N. Lohmann, E. Verbeek, R. M. Dijkman, Petri net transformations for business processes - a survey, T. Petri Nets and Other Models of Concurrency 2 (2009) 46–63.
[26] M. Weidlich, F. Elliger, M. Weske, Generalised computation of behavioural profiles based on petri-net unfoldings, in: Proceedings of the 7th Proceedings of the 7th International Workshop on Web Services and Formal Methods (WS-FM'10), Hoboken, NJ, US, 2010, to appear.
[27] A. Weijters, W. van der Aalst, Rediscovering workflow models from event-based data using little thumb, Integrated Computer-Aided Engineering 10 (2) (2003) 151–162.
[28] C. W. Günther, Process mining in flexible environments, Ph.D. thesis, Technische Universiteit Eindhoven (2009).
[29] R. Agrawal, T. Imielinski, A. N. Swami, Mining association rules between sets of items in large databases, in: P. Buneman, S. Jajodia (Eds.), SIGMOD Conference, ACM Press, 1993, pp. 207–216.
[30] R. Agrawal, R. Srikant, Fast algorithms for mining association rules in large databases, in: J. B. Bocca, M. Jarke, C. Zaniolo (Eds.), VLDB, Morgan Kaufmann, 1994, pp. 487–499.
[31] D. J. Rose, R. E. Tarjan, Algorithmic aspects of vertex elimination, in: Proceedings of 7th annual ACM symposium on Theory of computing, Albuquerque, New Mexico, United States, 1975, pp. 245 – 254.
[32] W. Aalst, A. Weijters, L. Maruster, Workflow mining: Discovering process models from event logs, IEEE Trans. Knowl. Data Eng. 16 (9) (2004) 1128–1142.
[33] M. Song, W. van der Aalst, Supporting process mining by showing events at a glance, in: K. Chari, A. Kumar (Eds.), Proceedings of 17th Annual Workshop on Information Technologies and Systems (WITS 2007), Montreal, Canada, 2007, pp. 139–145.
[34] R. P. J. C. Bose, W. M. P. van der Aalst, Trace alignment in process mining: Opportunities for process diagnostics, in: R. Hull, J. Mendling, S. Tai (Eds.), BPM, Vol. 6336 of Lecture Notes in Computer Science, Springer, 2010, pp. 227–242.
[35] J. M. Küster, C. Gerth, A. Förster, G. Engels, Detecting and resolving process model differences in the absence of a change log, in: Dumas et al. [61], pp. 244–260.
[36] R. M. Dijkman, Diagnosing differences between business process models, in: Dumas et al. [61], pp. 261–277.
[37] R. Glabbeek, U. Goltz, Refinement of actions and equivalence notions for concurrent systems, Acta Inf. 37 (4/5) (2001) 229–327.
[38] J. Hidders, M. Dumas, W. Aalst, A. Hofstede, J. Verelst, When are two workflows the same?, in: M. D. Atkinson, F. K. H. A. Dehne (Eds.), CATS, Vol. 41 of CRPIT, Australian Computer Society, 2005, pp. 3–11.
[39] T. Basten, W. Aalst, Inheritance of behavior, JLAP 47 (2) (2001) 47–145.
[40] J. Ebert, G. Engels, Observable or Invocable Behaviour - You Have to Choose, Technical Report 94-38, Department of Computer Science, Leiden University (December 1994).
[41] M. Schrefl, M. Stumptner, Behavior-consistent specialization of object life cycles, ACM Trans. Softw. Eng. Methodol. 11 (1) (2002) 92–148.
[42] A. Wombacher, Evaluation of technical measures for workflow similarity based on a pilot study, in: R. Meersman, Z. Tari (Eds.), OTM Conferences (1), Vol. 4275 of Lecture Notes in Computer Science, Springer, 2006, pp. 255–272.
[43] A. Wombacher, M. Rozie, Evaluation of workflow similarity measures in service discovery, in: M. Schoop, C. Huemer, M. Rebstock, M. Bichler (Eds.), Service Oriented Electronic Commerce, Vol. 80 of LNI, GI, 2006, pp. 51–71.
[44] C. Li, M. Reichert, A. Wombacher, On measuring process model similarity based on high-level change operations, in: Q. Li, S. Spaccapietra, E. S. K. Yu, A. Olivé (Eds.), ER, Vol. 5231 of Lecture Notes in Computer Science, Springer, 2008, pp. 248–264.
[45] B. Dongen, R. M. Dijkman, J. Mendling, Measuring similarity between business process models, in: Z. Bellahsene, M. Léonard (Eds.), CAiSE, Vol. 5074 of LNCS, Springer, 2008, pp. 450–464.
[46] N. Li, M. Tripunitara, Z. Bizri, On Mutually Exclusive Roles and Separation-of-Duty, ACM Transactions on Information and System Security (TISSEC) 10 (2).

[47] G. Ahn, R. Sandhu, Role-based Authorization Constraints Specification, ACM Transactions on Information and System Security (TISSEC) 3 (4).
[48] D. Ferraiolo, R. Sandhu, S. Gavrila, D. Kuhn, R. Chandramouli, Proposed NIST Standard for Role-Based Access Control, ACM Transactions on Information and System Security (TISSEC) 4 (3).
[49] E. Bertino, E. Ferrari, V. Atluri, The Specification and Enforcement of Authorization Constraints in Workflow Management Systems, ACM Transactions on Information and System Security (TISSEC) 2 (1).
[50] D. Ferraiolo, J. Barkley, D. Kuhn, A Role-Based Access Control Model and Reference Implementation within a Corporate Intranet, ACM Transactions on Information and System Security (TISSEC) 2 (1).
[51] S. Oh, R. S. Sandhu, X. Zhang, An effective role administration model using organization structure, ACM Trans. Inf. Syst. Secur. 9 (2) (2006) 113–137.
[52] K. Tan, J. Crampton, C. Gunter, The Consistency of Task-Based Authorization Constraints in Workflow Systems, in: Proc. of the 17th IEEE Workshop on Computer Security Foundations (CSFW), 2004.
[53] J. Crampton, H. Khambhammettu, Delegation and satisfiability in workflow systems, in: I. Ray, N. Li (Eds.), SACMAT 2008, 13th ACM Symposium on Access Control Models and Technologies, Estes Park, CO, USA, June 11-13, 2008, Proceedings, ACM, 2008, pp. 31–40.
[54] M. Strembeck, G. Neumann, An Integrated Approach to Engineer and Enforce Context Constraints in RBAC Environments, ACM Transactions on Information and System Security (TISSEC) 7 (3).
[55] M. Kuhlmann, D. Shohat, G. Schimpf, Role mining-revealing business roles for security administration using data mining technology, in: Proceedings of the eighth ACM symposium on Access control models and technologies, ACM, 2003, p. 186.
[56] I. Molloy, N. Li, T. Li, Z. Mao, Q. Wang, J. Lobo, Evaluating role mining algorithms, in: Proceedings of the 14th ACM symposium on Access control models and technologies, ACM, 2009, pp. 95–104.
[57] S. Etalle, W. Winsborough, A posteriori compliance control, in: Proceedings of the 12th ACM symposium on Access control models and technologies, ACM, 2007, p. 20.
[58] M. Gelfond, J. Lobo, Authorization and obligation policies in dynamic systems, Logic Programming (2009) 22–36.
[59] R. Accorsi, C. Wonnemann, Auditing workflow executions against dataflow policies, in: Business Information Systems, Springer, 2010, pp. 207–217.
[60] C. Wolter, A. Schaad, Modeling of task-based authorization constraints in bpmn, in: G. Alonso, P. Dadam, M. Rosemann (Eds.), Business Process Management, 5th International Conference, BPM 2007, Brisbane, Australia, September 24-28, 2007, Proceedings, Vol. 4714 of Lecture Notes in Computer Science, Springer, 2007, pp. 64–79.
[61] M. Dumas, M. Reichert, M.-C. Shan (Eds.), Business Process Management, 6th International Conference, BPM 2008, Milan, Italy, September 2-4, 2008. Proceedings, Vol. 5240 of Lecture Notes in Computer Science, Springer, 2008.