Towards Multi-perspective Conformance Checking with Aggregation Operations

Sicui Zhang[1,2(✉)], Laura Genga[2], Lukas Dekker[3], Hongchao Nie[4], Xudong Lu[1,2], Huilong Duan[1], and Uzay Kaymak[2,1]

[1] School of Biomedical Engineering and Instrumental Science, Zhejiang University, Hangzhou, People's Republic of China
zhangsicui@zju.edu.cn
[2] School of Industrial Engineering, Eindhoven University of Technology, Eindhoven, The Netherlands
[3] Cardiology Department, Catharina Hospital, Eindhoven, The Netherlands
[4] Philips Research, Eindhoven, The Netherlands

Abstract. Conformance checking techniques are widely adopted to validate process executions against a set of constraints describing the expected behavior. However, most approaches adopt a crisp evaluation of deviations, with the result that small violations are considered at the same level of significant ones. Furthermore, in the presence of multiple data constraints the overall deviation severity is assessed by summing up each single deviation. This approach easily leads to misleading diagnostics; furthermore, it does not take into account user's needs, that are likely to differ depending on the context of the analysis. We propose a novel methodology based on the use of aggregation functions, to assess the level of deviation severity for a set of constraints, and to customize the tolerance to deviations of multiple constraints.

Keywords: Conformance checking · Fuzzy aggregation · Data perspective

1 Introduction

Nowadays organizations often define procedures describing how their processes should be performed to satisfy a set of constraints, e.g., to minimize the throughput time or to comply with rules and regulations. A widely used formalism to represent these procedures consists in so-called *process models*, that are graphic or logic formalism representing constraints defined on organization processes, e.g., by the order of execution of the activities. However, it is well documented in literature that real process behavior often deviates from the expected process, which often leads to performance issues or opens the way to costly frauds [12].

M.-J. Lesot et al. (Eds.): IPMU 2020, CCIS 1237, pp. 215–229, 2020.
https://doi.org/10.1007/978-3-030-50146-4_17

In recent years, the increasing use by organizations of information systems (e.g., ERP, SAP, MRP and so on) to support and track the execution of their processes enabled the development of automatic, data-driven techniques to assess the compliance level of the real process behavior. Among them, *Conformance checking* techniques have been gaining increasing attention both from practitioners and academics [1,2,5,6,23]. Given an *event log*, i.e., a log file tracking data related to activities performed during process executions, conformance checking techniques are able to pinpoint discrepancies (aka, deviations) between the log and the corresponding model. While classic conformance checking techniques only deal with the *control-flow* of the process, i.e., the activities execution order, in recent years also some *multi-perspective conformance checking*, aimed to deal also with data constraints, have become more and more relevant [23,25].

Nevertheless, there are still several open challenges to implement multi-perspective conformance checking. Among them, here we focus on the lack of appropriate modeling mechanisms for dealing with the *uncertainty* and *graduality* often characterizing human-decisions in real-world processes. State of the art techniques implement a *crisp* approach: every execution of an activity is considered as either *completely wrong* or *completely correct*. [13,23,25].

While this assumption is well grounded to deal with the control-flow (indeed, each activity is either executed at the right moment, or it is not), when addressing data constraints it can easily lead to misleading results. A well-known example of this issue can be found in the healthcare domain. Let us assume that a surgery department implements a guideline stating that the systolic blood pressure (SBP) of a patient has to be lower than 140 to proceed with a surgery. It is reasonable to expect that sometimes clinicians will not refuse to operate patients whose SBP is 141, since this is quite a small deviation and delaying the surgery could be more dangerous for the patient. Clearly, surgeries performed with this value of SBP are likely to be much less problematic than surgeries performed with a SBP equal to, e.g., 160. However, conformance checking techniques would simply mark both these cases as 'not compliant ', without allowing for any distinction. This behavior is undesirable, since it is likely to return in output a plethora of not-interesting deviations, at the same time hiding those which could deserve further investigation. We investigated this issue in our previous work [29], where we proposed to use fuzzy sets, which are used to present the flexibility in the constraints and the goals in fuzzy optimization [20], to determine the severity of violations of a single soft constraint per activity.

However, the previous work used basic strategy of standard conformance checking techniques for dealing with *multiple constraints deviations*; namely, the total degree of data deviations of that activity is computed by summing up the costs for all the violated constraints. This strategy poses some important limitations when investigating the data compliance. First, it introduces an asymmetry in the assessment of control-flow and data deviations. While control-flow deviations for each activity express the level of compliance of the activity to control-flow constraints (either fully compliant or wrong), in the presence of multiple data constraints the obtained value does not give an indication of the

overall level of compliance to the constraints set. Furthermore, no customization to the user's needs is provided. First, in this setting data violations tend to be considered more severe than control-flow ones, even if this might not fit with user's intention. Furthermore, different contexts might require tailored functions to assess multiple data deviations severity.

In this paper, we address this issue by proposing a novel fuzzy conformance checking methodology based on the use of aggregation functions, which have been proved feasible for modeling simultaneous satisfaction of aggregated criteria [20]. With respect to previous work, the approach brings two main contributions: a) it applies fuzzy aggregation operators to assess the level of deviation severity for a set of constraints, and b) it allows to customize the tolerance to deviations of multiple constraints. As a proof-of-concept, we tested the approach over synthetic data.

The remainder of this paper is organized as follows. Section 2 introduces a running example to discuss the motivation of this work. Section 3 introduces basic formal notions. Section 4 illustrates our approach, and Sect. 5 presents results obtained by a set of synthetic experiments. Section 6 discusses related work. Finally, Sect. 7 draws some conclusions and presents future work.

2 Motivation

Consider, as a running example, a loan management process derived from previous work on the event log of a financial institute made available for the BPI2012 challenge [3,15]. Figure 1 shows the process in BPMN notation. The process starts with the submission of a loan application. Then, the application passes through a first assessment of the applicant's requirements and, if the requested amount is greater than 10000 euros, also through a more thorough fraud detection analysis. If the application is not eligible, the process ends. Otherwise, the application is accepted, an offer to be sent to the customer is selected and the details of the application are finalized. After the offer has been created and sent to the customer, the latter is contacted to discuss the offer with her. At the end of the negotiation, the agreed application is registered on the system. At this point, further checks can be performed on the application, if the overall duration is still below 30 days and the Amount is larger than 10000, before approving it.

Let us assume that this process is supported by some system able to track the execution of its activities in a so-called event log. In practice, this is a collection of *traces*, i.e., sequences of activities performed within the same process execution, each storing information like the execution timestamp of the execution, or other data element [1]. As an example, let us consider the following traces[1] showing two executions of the process in Fig. 1 (note that we use acronyms rather than complete activity names) : $\sigma_1 = \langle (A\_S, \{Amount =$

[1] We use the notation $(act, \{att_1 = v_1, \ldots, att_n = v_n\})$ to denote the occurrence of activity act in which variables $att_1 \ldots att_n$ are assigned to values $v_1, \ldots v_n$. The symbol $\perp$ means that no variable values are changed when executing the activity.

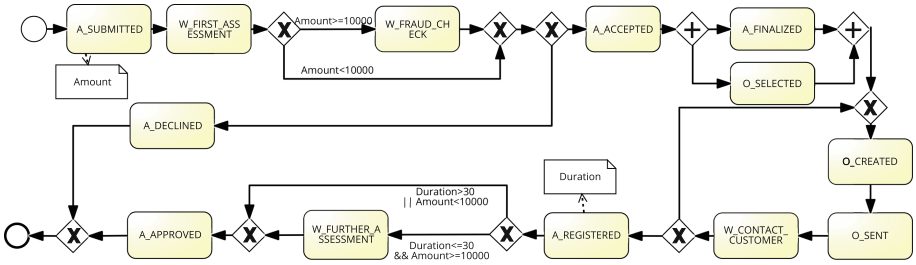

Fig. 1. The loan management process.

8400}), $(W\_FIRST\_A, \perp)$, $(W\_F\_C, \perp)$, $(A\_A, \perp)$, $(A\_F, \perp)$, $(O\_S, \perp)$, $(O\_C, \perp)$, $(O\_S, \perp)$, $(W\_C, \perp)$, $(A\_R, \{Duration = 34\})$, $(W\_F\_A, \perp)$, $(A\_AP, \perp), \rangle$; $\sigma_2 = \langle (A\_S, \{Amount = 1400\}), (W\_FIRST\_A, \perp), (W\_F\_C, \perp), (A\_A, \perp), (A\_F, \perp), (O\_S, \perp), (O\_C, \perp), (O\_S, \perp), (W\_C, \perp), (A\_R, \{Duration = 24\}), (W\_F\_A, \perp), (A\_AP, \perp), \rangle$. Both executions violate the constraints defined on the duration and the amount of the loan, according to which the activity W_F_A should have been anyway skipped.

Conformance checking techniques also attempt to support the user in investigating the *interpretations* of a deviation. In our case, the occurrence of the activity W_F_A could be considered either as a 1) control-flow deviation (i.e., data are corrected but the activity should not have been executed) or as a 2) data-flow deviation (i.e., the execution of the activity is correct but data have not been properly recorded on the system). In absence of domain knowledge in determining what is the real explanation, conformance checking techniques assess the severity (aka, cost) of the possible interpretations and select the least severe one, assuming that this is the one closest to the reality. In our example, conformance checking would consider σ_1 as a control-flow deviation, since the cost would be equal to 1, while data-flow deviation would correspond to 2, having two violated constraints; for σ_2, instead, the two interpretations would be equivalent, since only one data constraint is violated. In previous work [29] we investigated how to use fuzzy membership function to assess severity of data deviations taking into account the magnitude of the deviations. However, the approach still comes with some limitations when considering multiple constraints. Indeed, with this approach the overall severity of the data deviation for an activity is assessed by a simple sum operation. For example, let us suppose that with the method in [29] we obtained a cost of 0.3, 0.8 for the violations of *Amount* and *Duration* in W_F_A in σ_1, thus obtaining a total cost of 1.1, and 0.8 and 0 in σ_2, thus obtaining, a total cost of 0.8. In this setting, activities involving multiple constraints will tend to have an interpretation biased towards control-flow deviations, since the higher the number of constraints, the higher the the data-deviation cost. Furthermore, it is worth noting that the comparison between the two traces can be misleading; in one case, constraints are violated, even if one only slightly deviated; while in the second case only one constraint is violated, even if with quite a strong deviation. However, the final numerical results are quite similar,

thus hiding the differences. This example shows how the use of the simple sum function can impact the results significantly, without the user realizing it and, above all, without providing the user with any customization mechanism. For example, the user might want to assess the data-compliance level in terms of the percentage of satisfied constraints, or by considering only the maximum cost, and so on. However, current techniques do not allow for this kind of customization.

3 Preliminaries

This section introduces a set of concepts that will be used through the paper.

3.1 Conformance Checking: Aligning Event Logs and Models

Conformance checking techniques detect discrepancies between a process model and the real process execution. Here we define the notion of process model using the notation from [2], enriched with data-related notions explained in [13].

Definition 1 (Process model). *A process model* $M = (P, P_I, P_F, A_M, V, W, U, T, G, Values)$ *is a transition system defined over a set of activities* A_M *and a set of variables* V*, with states* P*, initial states* $P_I \subseteq P$*, final states* $P_F \subseteq P$ *and transitions* $T \subseteq P \times (A_M \times 2^V) \times P$*.* $U(V_i)$ *represents the domain of* V_i *for each* $V_i \in V$*. The function* $G : A_M \rightarrow Formulas(V \cup \{V_i' \mid V_i \in V\})$ *is a guard function, i.e., a boolean formula expressing a condition on the values of the data variables.* $W : A_M \rightarrow 2^V$ *is a write function, that associates an activity with the set of variables which are written by the activity. Finally,* $Values : P \rightarrow \{V_i = v_i, i = 1..|V| \mid v_i \in U(V_i) \cup \{\perp\}\}$ *is a function that associates each state with the corresponding pairs variable=value.*

The firing of an activity $s = (a, w) \in A_M \times (V \not\rightarrow U)$ in a state p' is *valid* if: 1) a is enabled in p'; 2) a writes all and only the variables in $W(a)$; 3) $G(a)$ is *true* when evaluate over $Values(p')$. To access the components of s we introduce the following notation: $vars(s) = w$, $act(s) = a$. Function $vars$ is also overloaded such that $vars(V_i) = w(V_i)$ if $V_i \in dom(vars(s))$ and $vars(s, V_i) = \perp$ if $V_i \notin dom(vars(s))$. The set of valid process traces of a model M is denoted with $\rho(M)$ and consists of all the valid firing sequences $\sigma \in (A_M \times (V \not\rightarrow U))^*$ that, from an initial state P_I lead to a final state P_F. Figure 1 provides an example of a process model in BPMN notation.

Process executions are often recorded by means of an information system in event logs. Formally, let S_N be the set of (valid and invalid) firing of activities of a process model M; an **event log** is a multiset of traces $\mathbb{L} \in \mathbb{B}(S_N^*)$. Given an event log L, conformance checking builds an *alignment* between L and M, mapping "moves" occurring in the event log to possible "moves" in the model. A "no move" symbol "$\gg$" is used to represent moves which cannot be mimicked. For convenience, we introduce the set $S_N^{\gg} = S_N \cup \{\gg\}$. Formally, we set s_L to be a transition of the events in the log, s_M to be a transition of the activities in the model. A move is represented by a pair $(s_L, s_M) \in S_N^{\gg} \times S_N^{\gg}$ such that:

- (s_L, s_M) is a *move in log* if $s_L \in S_N$ and $s_M = \gg$
- (s_L, s_M) is a *move in model* if $s_M \in S_N$ and $s_L = \gg$
- (s_L, s_M) is a *move in both without incorrect data* if $s_L \in S_N$, $s_M \in S_N$ and $act(s_L) = act(s_M)$ and $\forall V_i \in V(vars(s_L, V_i) = vars(s_M, V_i)))$
- (s_L, s_M) is a *move in both with incorrect data* if $s_L \in S_N$, $s_M \in S_N$ and $act(s_L) = act(s_M)$ and $\exists V_i \in V \mid vars(s_L, V_i) \neq vars(s_M, V_i))$.

Let $A_{LM} = \{(s_L, s_M) \in S_N^{\gg} \times S_N^{\gg} \mid s_L \in S_N \vee s_M \in S_N\}$ be the set of all legal moves. The *alignment* between two process executions σ_L, $\sigma_M \in S_N^*$ is $\gamma \in A_{LM}^*$ such that the projection of the first element (ignoring $\gg$) yields σ_L, and the projection on the second element (ignoring $\gg$) yields σ_M.

Example 1. Let us consider the model in Fig. 1 and the trace σ_1 in Sect. 2.

Table 1 shows two possible alignments γ_1 and γ_2 for activity W_F_A. For Alignment γ_1, the pair (W_F_A, W_F_A) is a move in both with incorrect data, while in γ_2 the move $(W\_F\_A, \perp)$ is matched with a $\gg$, i.e., it is a move on log. (In remaining part, *Amount* and *Duration* are abbreviated to *A* and *D*).

Table 1. Two possible alignments between σ_M and σ_L

Alignment γ_1		Alignment γ_2	
Log	Model	Log	Model
...	...	...	...
$(W\_F\_A, \{8000, 34\})$	(W_F_A)	$(W\_F\_A, \{8000, 34\})$	$(\gg)$
...	...	...	...

As shown in Example 1, there can be multiple possible alignments for a given log trace and process model. Our goal is to find the *optimal alignment*, i.e., the alignment with minimum cost. To this end, the severity of deviations is assessed by means of a *cost function*.

Definition 2 (Cost function, Optimal Alignment). *Let σ_L, σ_M be a log trace and a trace, respectively. Given the set of all legal moves A_N, a cost function k assigns a non-negative cost to each legal move: $A_N \rightarrow \mathbb{R}_0^+$. The cost of an alignment γ between σ_L and σ_M is computed as the sum of the cost of all the related moves: $K(\gamma) = \sum_{(S_L, S_M) \in \gamma} k(S_L, S_M)$. An **optimal** alignment of a log trace and a process trace is one of the alignments with the lowest cost according to the provided cost function.*

3.2 Fuzzy Set Aggregation Operators

Aggregation operations (AOs) are mathematical functions that satisfy minimal boundary and monotonicity conditions, and are often used for modeling decision

making processes, since they allow to specify how to combine the different criteria that are relevant when making a decision [17,27].

In literature, many AOs have been defined (see [18,19,22] for an overview), with different level of complexity and different interpretations. A commonly used class of aggregation operators are the t-norms, which are used to model conjunction of fuzzy sets. In compliance analysis, one often tries to satisfy all constraints on the data, and so t-norms are suitable operators for modeling soft constraints in compliance analysis. Widely used t-norms are the minimum, product and the Yager operators [21].

In addition to the t-norms, other aggregation operators could also be used, depending on the goals of the compliance analysis. We do not consider other types of aggregation operators in this paper, but, in general, one could use the full flexibility of different classes of fuzzy set aggregation operators that have been used in decision making (see, e.g. [11]).

4 Proposed Compliance Analysis Method

We introduce a compliance checking approach tailored to dealing with decision tasks under multiple guards, to enhance the flexibility of the compliance assessing procedure. To this end, we investigate the use of AOs.

4.1 Aggregated Cost Function

Compliance checking in process analysis is based on the concept of alignment between a process model and a process trace that minimizes a cost of misalignment. The computation of an optimal alignment relies on the definition of a proper cost function for the possible kind of moves (see Sect. 3). Most of state-of-the art approaches adopt (variants of) the standard distance function defined in [2], which sets a cost of 1 for every move on log/model (excluding invisible transitions), and a cost of 0 for synchronous moves. Multi-perspective approaches extend the standard cost function to include data costs. Elaborating upon these approaches, in previous work [29] we defined our fuzzy cost function as follows.

Definition 3 (Data-aware fuzzy cost function). *Let (S_L, S_M) be a move between a process trace and a model execution, $W(S_M)$ be the set of variables written by the activity related to S_M, and let $\mu_i(var(S_L, V_i))$ be a fuzzy membership function returning the compliance degree of single variable $var(S_L, V_i)$. For the sake of simplicity, we write it as μ_i in the following. Then we define $(1 - \mu_i)$ as the data cost of this deviation. The cost $k(S_L, S_M)$ is defined as:*

$$k(S_L, S_M) = \begin{cases} 1 & \text{if } (S_L, S_M) \text{ is a move in log} \\ 1 + |W(S_M)| & \text{if } (S_L, S_M) \text{ is a move in model} \\ \sum_{\forall V_i \in V} (1 - \mu_i) & \text{if } (S_L, S_M) \text{ is a move in both} \end{cases} \quad (1)$$

This cost function assigns a cost equal to 1 for a move in log; 1 plus the number of variables that should have been written by the activity for a move in model; finally, the sum of the cost of the deviations (1-μ_i) for the data variables if it's a move in both. Note that the latter consider both the case of move with incorrect and incorrect data. As discussed in Sect. 2, summing up all the data cost presents important limitations to assess the conformance of multiple constraints. Therefore, in the present work, we propose a new version of our fuzzy cost function with the goal of standardize every move within the range (0,1) and allow the user to customize the cost function to her needs.

Definition 4 (AOs based cost function). *Let $\pi(\mu_1, \mu_2, ..., \mu_n)$ be an user-defined aggregated membership function of multiple variables. Then $(1 - \pi)$ is the overall deviation cost of a set of variables. The cost $k(S_L, S_M)$ is defined as:*

$$k(S_L, S_M) = \begin{cases} 1 & if (S_L, S_M) \text{ is a move in log} \\ 1 + |W(S_M)| & if (S_L, S_M) \text{ is a move in model} \\ 1 - \pi(\mu_1, \mu_2, ..., \mu_n) & if (S_L, S_M) \text{ is a move in both.} \end{cases} \quad (2)$$

4.2 Using A* to Find the Optimal Alignment

The problem of finding an optimal alignment is usually formulated as a search problem in a directed graph [14]. Let $Z = (Z_V, Z_E)$ be a directed graph with edges weighted according to some cost structure. The A* algorithm finds the path with the lowest cost from a given source node $v_0 \in Z_v$ to a node of a given goals set $Z_G \subseteq Z_V$. The cost from each node is determined by an evaluation function $f(v) = g(v) + h(v)$, where:

- $g : Z_V \rightarrow \mathbb{R}^+$ gives the smallest path cost from v_0 to v;
- $h : Z_V \rightarrow \mathbb{R}_0^+$ gives an estimate of the smallest path cost from v to any of the target nodes.

If h is *admissible*,i.e. it underestimates the real distance of a path to any target node v_g, then A* finds a path that is guaranteed to have the overall lowest cost.

The algorithm works iteratively: at each step, the node v with lowest cost is taken from a priority queue. If v belongs to the target set, the algorithm ends returning node v. Otherwise, v is expanded: every successor v_0 is added to the priority queue with a cost $f(v_0)$.

Given a log trace and a process model, to employ A* to determine an optimal alignment we associate every node of the search space with a prefix of some complete alignments. The source node is an empty alignment $\gamma_0 = \langle\rangle$, while the set of target nodes includes every complete alignment of σ_L and M. For every pair of nodes (γ_1, γ_2), γ_2 is obtained by adding one move to γ_1.

The cost associated with a path leading to a graph node γ is then defined as $g(\gamma) = K(\gamma) + \epsilon|\gamma|$, where $K(\gamma) = \sum_{s_L, s_M \in \gamma} k(s_L, s_M)$, with $k(s_L, s_M)$ defined as in (1), $|\gamma|$ is the number of moves in the alignment, and ϵ is a negligible cost, added to guarantee termination. Note that the cost g has to be strictly

increasing. While we do not give a formal proof for the sake of space, it is straight to see that g is obtained in our approach by the sum of all non negative elements. Therefore, while moving from an alignment prefix to a longer one, the cost can never decrease. For the definition of the heuristic cost function $h(v)$ different strategies can be adopted. Informally, the idea is computing, from a given alignment, the minimum number of moves (i.e., the minimum cost) that would lead to a complete alignment. Different strategies have been defined in literature, e.g., the one in [2], which exploits Petri-net marking equations, or the one in [28], which generates possible states space of a BPMN model.

Example 2. Let us analyze possible moves to assign to the activity W_F_A in σ_1. Let us assume that the memberships of the variables are $\mu_A = 0.4$ and $\mu_D = 0.2$. According to (2) and *Product* t-norm we get the fuzzy cost function $k(S_L, S_M)$.

$$k(S_L, S_M) = \begin{cases} 1 & , \textit{move in log} \\ 1 & , \textit{move in model} \\ 1 - \mu_A \cdot \mu_D & , \textit{move in both} \end{cases} \tag{3}$$

Figure 2 shows the portion of the space states for the alignment building of σ_1. At node #11, $f = 0$, since no deviations occurred so far. From here, there are two possible moves that could be selected, one representing a move on log (on the left), one a move on model (on the right) and finally a move in both (in the middle). Since using the *Product* aggregation the data cost is equal to 0.92, the algorithm selects the move in both, being the one with the lowest cost.

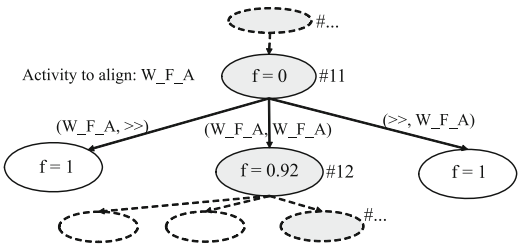

Fig. 2. The alignment with the new aggregated function.

5 Experiment and Result

This section describes a set of experiments we performed to obtain a proof-of-concept of the approach. We compared the diagnostics returned by an existing approach [29] and our new cost functions with three $t - norm$ aggregations. More precisely, we aimed to get the answer to the question: *what is the impact of different aggregation operations on the obtained alignments?* In particular,

we assess the impact of the aggregation function in terms of a) differences in the overall deviation cost, and b) difference in terms of the interpretation, i.e., the moves selected by the alignment algorithm as the best explanation for the deviation.

5.1 Settings

In order to get meaningful insights on the behavior we can reasonably expect by applying the approach in the real world, we employ a realistic synthetic event log, consisting of 50000, introduced in a former paper [16], obtained starting from one real-life logs, i.e., the event log of the BPI2012 challenge [2]. We evaluated the compliance of this log against a simplified version of the process model in [16], to which we added few data constraints (see Fig. 1). The approach has been implemented as an extension to the tool developed by [28], designed to deal with BPMN models. Our process model involves two constraints for the activity W_F_A, i.e., $Amount \geq 10000$ and $Duration \leq 30$.

Here we assume that $Amount \in (3050, 10000)$ and $Duration \in (30, 70)$ represent a tolerable violation range for the variables. Since we cannot refer to experts' knowledge, we derived these values from simple descriptive statistics. In particular, we considered values falling within the third quartile as acceptable. The underlying logic is that values which tend to occur repeatedly are likely to indicate acceptable situations. Regarding the shape of the membership functions for the variables, here we apply the linear function μ, as reported below.

$$\mu_1(A) = \begin{cases} 1 & \text{, if } A \geq 10000 \\ 0 & \text{, if } A \leq 2650 \\ \frac{A-2650}{7350} & \text{, if } 2650 < A < 10000; \end{cases} \qquad \mu_2(D) = \begin{cases} 1 & \text{, if } D \leq 30 \\ 0 & \text{, if } D \geq 69 \\ \frac{69-D}{39} & \text{, if } 30 < D < 69 \end{cases} \tag{4}$$

For the classic sum function, we use the cost function provided by (1); while for the new approach with AOs, we apply the cost function in (2). We tested the $t-norms$: $Minimum$, $Product$, and $Yager$.

When data deviations and control-flow deviations show the same cost, we picked the control-flow move. This assumption simulates what we would do in a real-world context. Indeed, without a-priori knowledge on the right explanation, it is reasonable to assume that it is more likely that the error was executing the activity, rather than accepting out-of-range data deviations.

5.2 Results

Note that here we focus on the activity W_F_A, since, in our log, is the only one involving multiple data constraints. Table 2 shows differences in terms of number and type of moves, as well as in terms of costs. The columns *#move in* log, *#move in data* show the number of traces in which the alignment has selected

[2] https://www.win.tue.nl/bpi/doku.php?id=2012:challenge.

for the activity W_F_A a move in log or a move in data, respectively. The column "Average costs" shows the average alignment cost. The conformance checking algorithms selects for each activity the move corresponding to the minimum cost. Therefore, the differences among the chosen move depend on the different costs obtained on W_F_A when applying different operators. To provide a practical example of the impact of the aggregated cost on the obtained diagnostics, below we discuss the results obtained for one trace.

Table 2. Number of different moves of the activity W_F_A.

	#move in log	*#move in data*	Average cost
Sum	707	350	0.823
Min	660	397	0.804
Product	660	397	0.814
Yager	678	379	0.811

Table 3. The cost of possible moves

	#move in log	*#move in data*
Sum	1	1.003
Min	1	0.513
Product	1	0.751
Yager	1	0.709

Table 4. The optimal alignments

	Log	Model	Move type	Cost
S	W_F_A	$\gg$	*#move in log*	1
M	W_F_A	W_F_A	*#move in data*	0.513
P	W_F_A	W_F_A	*#move in data*	0.751
Y	W_F_A	W_F_A	*#move in data*	0.709

Example 3. Let us consider the *trace* $\sigma_{\#2859} = \langle (A\_S, \{Amount = 6400\}), (W\_FIRST\_A, \bot), (A\_A, \bot), (A\_F, \bot), (O\_S, \bot), (O\_C, \bot), (O\_S, \bot), (W\_C, \bot), (W\_F, \bot), (O\_C, \bot), (O\_S, \bot), (W\_C, \bot), (A\_R, \{Duration = 50\}), (W\_F\_A, \bot), (A\_AP, \bot), \rangle$. According to their membership functions (4), $\mu_1(A = 6400) = 0.5102$ and $\mu_2(D = 50) = 0.4872$. Therefore, the corresponding costs are 0.4898 and 0.5128. Table 3 shows the cost of possible moves for W_F_A according to the aggregation functions. Table 4 shows the move picked by each function to build the alignment. Using the Sum function, the data cost is 1.003, so that a move-in-log is chosen as an optimal alignment. In the other cases, instead, the move in data is the one with the lowest cost. Since both the deviations fall in the acceptable range, this interpretation is likely to be more in line with the user's expectations.

The observations made for the example can be generalized to the overall results of Table 2, which shows a set of traces whose interpretation is heavily

affected by the chosen cost function. As expected, the Sum function is the most biased towards the choice of move in log interpretation. It selects 40 moves in log more that Product and Min, and 29 more than Yager. One can argue that this choice is likely not one the human analyst would have expected. Indeed, we are using Yager with $\omega = 2$ [11], that means that when both the variables show severe deviations, we expect the data cost to be 1 and move-in-log to be picked. This means that at least 29 of the aligned traces were marked as move-in-log also if both the constraints did not show severe deviations. We argue that this behavior can be misleading for the analyst or, anyway, not being in line with her needs. The Product function marks other 18 traces as move-in-data, in addition to the ones marked by the Yager. This was expected, since the Product function relaxes the requirements on the full satisfaction of the set of constraints. Nevertheless, this implies that in all these 18 traces the deviations always fell in the tolerance range. Therefore, also these situations might have been better represented as data deviations, depending on the analysts' needs. As regards the Min function, it returns a full data deviation in the presence of at least one deviation outside the deviation range, which explains why it returned the same alignments of the Product function. The overall alignments costs are in line with the expectations. The Sum function returns the highest average cost, as expected, the Min the lowest, while the Yager and the Product behave similarly, and the difference can likely be explained with the 18 traces of difference discussed above. While the absolute difference among the costs is not very relevant, these results show that both the alignments and the assessment of the deviations are impacted by the choice of the cost function, thus highlighting once again the need for a more flexible approach to compliance assessment allowing the user to tailor the cost function to her context.

6 Related Work

During the last decades, several conformance checking techniques have been proposed. Some approaches [9,10,26] propose to check whether event traces satisfy a set of compliance rules, typically represented using declarative modeling. Rozinat and van der Aalst [24] propose a token-based technique to replay event traces over a process model to detect deviations, which, however, has been shown to provide misleading diagnostics in some contexts [4]. Recently, alignments have been proposed as a robust approach to conformance checking based on the use of a cost function [2]. While most of alignment-based approaches use the standard distance cost function as defined by [2], some variants have been proposed to enhance the provided diagnostics, e.g., the work of Alizadeh et al. [8], which computes the cost function by analyzing historical logging data. Besides the control flow, there are also other perspectives like data, or resources, that are often crucial for compliance checking analysis. Few approaches have investigated how to include these perspectives in the analysis: [7] extends the approach in [8] by taking into account data describing the contexts in which the activities occurred. Some approaches proposed to compute the control-flow first then assessing the

compliance with respect to the data perspective, e.g. [13]. These methods gives priority to check the control flow, with the result that some important deviations can be missed. [23] introduces a cost function balancing different perspectives, thus obtaining more precise diagnostics. The approaches mentioned so far assume a crisp evaluation of deviations. To the best of our knowledge, the only work which explored the use of a fuzzy cost function is our previous work [29] which, however, did not consider multiple constraints violation.

7 Conclusion

In this work, we investigated the use of fuzzy aggregation operations in conformance checking of process executions to deal with multiple data constraints for an activity. The proposed approach enhances significantly the flexibility of compliance checking, allowing the human analyst to customize the compliance diagnostic according to her needs. We elaborated upon the relevance of this aspect both theoretically and with some examples.

As a proof of concept, we implemented the approach and tested it over a synthetic dataset, comparing results obtained by cost functions with classic sum function and three different aggregations. The experiments confirmed that the approach generates more "balanced" diagnostics, and introduces the capability of personalizing the acceptance of deviations for multiple guards.

Nevertheless, there are several research directions still to be explored. In future work, first we plan to test our approach with real-world data. Furthermore, we intend to investigate the usage of different aggregation functions, as well as the possibility of extending the notion of aggregation to take into account also other kinds of deviations. Finally, we intend to investigate potential applications, for example in terms of on-line process monitoring and support, with the aim of enhancing the system resilience to exceptions and unforeseen events.

Acknowledgements. The research leading to these results has received funding from the Brain Bridge Project sponsored by Philips Research.

References

1. van der Aalst, W., et al.: Process mining manifesto. In: Daniel, F., Barkaoui, K., Dustdar, S. (eds.) BPM 2011. LNBIP, vol. 99, pp. 169–194. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-28108-2_19
2. Van der Aalst, W., Adriansyah, A., van Dongen, B.: Replaying history on process models for conformance checking and performance analysis. Wiley Interdisc. Rev.: Data Min. Knowl. Discovery **2**(2), 182–192 (2012)
3. Adriansyah, A., Buijs, J.C.: Mining process performance from event logs. In: La Rosa, M., Soffer, P. (eds.) BPM 2012. LNBIP, vol. 132, pp. 217–218. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-36285-9_23
4. Adriansyah, A., van Dongen, B.F., van der Aalst, W.M.P.: Towards robust conformance checking. In: zur Muehlen, M., Su, J. (eds.) BPM 2010. LNBIP, vol. 66, pp. 122–133. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-20511-8_11

5. Adriansyah, A., van Dongen, B.F., van der Aalst, W.M.: Memory-efficient alignment of observed and modeled behavior. BPM Center Report **3**, 1–44 (2013)
6. Adriansyah, A., Munoz-Gama, J., Carmona, J., van Dongen, B.F., van der Aalst, W.M.P.: Alignment based precision checking. In: La Rosa, M., Soffer, P. (eds.) BPM 2012. LNBIP, vol. 132, pp. 137–149. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-36285-9_15
7. Alizadeh, M., De Leoni, M., Zannone, N.: Constructing probable explanations of nonconformity: a data-aware and history-based approach. In: 2015 IEEE Symposium Series on Computational Intelligence, pp. 1358–1365. IEEE (2015)
8. Alizadeh, M., de Leoni, M., Zannone, N.: History-based construction of alignments for conformance checking: formalization and implementation. In: Ceravolo, P., Russo, B., Accorsi, R. (eds.) SIMPDA 2014. LNBIP, vol. 237, pp. 58–78. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-27243-6_3
9. Borrego, D., Barba, I.: Conformance checking and diagnosis for declarative business process models in data-aware scenarios. Expert Syst. Appl. **41**(11), 5340–5352 (2014)
10. Caron, F., Vanthienen, J., Baesens, B.: Comprehensive rule-based compliance checking and risk management with process mining. Decis. Support Syst. **54**(3), 1357–1369 (2013)
11. da Costa Sousa, J.M., Kaymak, U.: Model predictive control using fuzzy decision functions. IEEE Trans. Syst. Man. Cybern. Part B (Cybern.) **31**(1), 54–65 (2001)
12. de Leoni, M., van der Aalst, W.M.P., van Dongen, B.F.: Data- and resource-aware conformance checking of business processes. In: Abramowicz, W., Kriksciuniene, D., Sakalauskas, V. (eds.) BIS 2012. LNBIP, vol. 117, pp. 48–59. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-30359-3_5
13. de Leoni, M., van der Aalst, W.M.P.: Aligning event logs and process models for multi-perspective conformance checking: an approach based on integer linear programming. In: Daniel, F., Wang, J., Weber, B. (eds.) BPM 2013. LNCS, vol. 8094, pp. 113–129. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40176-3_10
14. Dechter, R., Pearl, J.: Generalized best-first search strategies and the optimality of A. J. ACM (JACM) **32**(3), 505–536 (1985)
15. Genga, L., Alizadeh, M., Potena, D., Diamantini, C., Zannone, N.: Discovering anomalous frequent patterns from partially ordered event logs. J. Intell. Inf. Syst. **51**(2), 257–300 (2018). https://doi.org/10.1007/s10844-018-0501-z
16. Genga, L., Di Francescomarino, C., Ghidini, C., Zannone, N.: Predicting critical behaviors in business process executions: when evidence counts. In: Hildebrandt, T., van Dongen, B.F., Röglinger, M., Mendling, J. (eds.) BPM 2019. LNBIP, vol. 360, pp. 72–90. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26643-1_5
17. Grabisch, M., Labreuche, C.: Fuzzy measures and integrals in MCDA. In: Greco, S., Ehrgott, M., Figueira, J.R. (eds.) Multiple Criteria Decision Analysis. ISORMS, vol. 233, pp. 553–603. Springer, New York (2016). https://doi.org/10.1007/978-1-4939-3094-4_14
18. Grabisch, M., Marichal, J.L., Mesiar, R., Pap, E.: Aggregation functions: construction methods, conjunctive, disjunctive and mixed classes. Inf. Sci. **181**(1), 23–43 (2011)
19. Grabisch, M., Marichal, J.L., Mesiar, R., Pap, E.: Aggregation functions: means. Inf. Sci. **181**(1), 1–22 (2011)
20. Kaymak, U., Sousa, C., João, M.: Weighted constraints in fuzzy optimization (2001)

21. Keresztfalvi, T.: Operations on fuzzy numbers extended by yager's family of t-norms. Math. Res. **68**, 163 (1993)
22. Klir, G.J., Yuan, B.: Fuzzy Sets and Fuzzy Logic: Theory and Applications. Prentice-Hall, Inc., Upper Saddle River (1995)
23. Mannhardt, F., de Leoni, M., Reijers, H.A., van der Aalst, W.M.P.: Balanced multi-perspective checking of process conformance. Computing **98**(4), 407–437 (2015). https://doi.org/10.1007/s00607-015-0441-1
24. Rozinat, A., Van der Aalst, W.M.: Conformance checking of processes based on monitoring real behavior. Inf. Syst. **33**(1), 64–95 (2008)
25. Song, W., Jacobsen, H.A., Zhang, C., Ma, X.: Dependence-based data-aware process conformance checking. IEEE Trans. Serv. Comput. (2018). https://doi.org/10.1109/TSC.2018.2821685
26. Taghiabadi, E.R., Gromov, V., Fahland, D., van der Aalst, W.M.P.: Compliance checking of data-aware and resource-aware compliance requirements. In: Meersman, R., et al. (eds.) OTM 2014. LNCS, vol. 8841, pp. 237–257. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-45563-0_14
27. Torra, V., Narukawa, Y.: Modeling decisions: Information Fusion and Aggregation Operators. Springer Science & Business Media, Berlin (2007). https://doi.org/10.1007/978-3-540-68791-7
28. Yan, H., et al.: Aligning event logs to task-time matrix clinical pathways in BPMN for variance analysis. IEEE J. Biomed. Health Inform. **22**(2), 311–317 (2017)
29. Zhang, S., Genga, L., Yan, H., Lu, X., Duan, H., Kaymak, U.: Towards multi-perspective conformance checking with fuzzy sets. arXiv:2001.10730 (2020)