Fuzzy multi-perspective conformance checking for business processes

Applied Soft Computing 130 (2022) 109710

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Fuzzy multi-perspective conformance checking for business processes

Sicui Zhang [a,e], Laura Genga [b], Lukas Dekker [c], Hongchao Nie [d], Xudong Lu [a,*], Huilong Duan [a], Uzay Kaymak [e]

[a] School of Biomedical Engineering and Instrumental Science, Zhejiang University, Hangzhou, PR China
[b] School of Industrial Engineering, Eindhoven University of Technology, Eindhoven, The Netherlands
[c] Cardiology Department, Catharina Hospital, Eindhoven, The Netherlands
[d] Philips Research, Eindhoven, The Netherlands
[e] Jheronimus Academy of Data Science, Eindhoven University of Technology, Eindhoven, The Netherlands

ARTICLE INFO

Article history:
Received 7 April 2021
Received in revised form 30 April 2022
Accepted 5 October 2022
Available online 13 October 2022

Keywords:
Process mining
Conformance checking
Fuzzy sets
Data perspective
Business processes

ABSTRACT

Conformance checking techniques are widely used to monitor the execution of organization processes and to pinpoint possible violations of the prescribed behavior. State-of-the-art approaches adopt a crisp evaluation of deviations: namely, every step in the execution which is not perfectly compliant with the procedural rules is marked as deviant. However, many real-world processes are driven by decisions taken by human actors, which are often characterized by uncertainty. As a consequence, deviations are often tolerated, within some boundaries. In these contexts, assessing small violations at the same level as significant ones hampers the accuracy of the provided diagnostics. In this work, we propose a novel conformance checking approach which allows to consider actors' tolerance to violations when assessing the magnitude of detected deviations, taking into account different kinds of deviating behaviors. Experiments conducted on two real-life clinical data sets have shown that taking the extent of deviations into account leads to more fine-grained diagnostics, thus illustrating the value of the approach.

1. Introduction

Organizations often define *process models* to describe how their processes should be performed. These are graphic or logic formalisms representing constraints defined on organization's processes, e.g., the order of execution of the activities, the authorization to perform a given set of tasks, and so on [1]. However, in many contexts these procedures are used mainly as guidelines for the process actors, rather than being enforced. While this flexibility turns out to be valuable in many situations (e.g., to allow the actors to react to unforeseen or exceptional situations and guarantee business continuity), or even necessary (e.g., doctors in hospitals must have the freedom to decide against the guidelines for the sake of their patients' well-being), at the same time it is well documented that it also paves the way to performance issues, costly frauds, and so on [2]. It is hence crucial for organizations to be able to assess the compliance level of their process executions and to detect the occurred deviations, in order to identify undesirable and/or risky ones.

In recent years, the increasing use of information systems by organizations (e.g., ERP, SAP, MRP and so on) to support and track the execution of their processes enabled the development of automatic *conformance checking* techniques, able to assess the overall level of compliance of process executions and to pinpoint where deviations occurred, thus providing the analyst with valuable diagnostics [3–7]. Nevertheless, these techniques still suffer from some important limitations. First, in many domains experts' decisions are often characterized by some level of *uncertainty*. For instance, in the clinical medicine often exists some tolerance to deviations from clinical protocols [8–10], while the risk versus benefit are balanced in the bounded rational or deviated decision making problems [11]. To give an example, in the clinical process for atrial fibrillation treatment, there is a guideline stating that a patient who is potential to have blood clots, i.e. International Normalized Ratio (INR) less than 2.0, is not recommended for an ablation cure due to the high risk of stroke. Adopting a crisp evaluation, the INR value 2.0 would be considered fully compliant to this pre-operative requirement, while 1.9 would be fully not compliant. This kind of sharp estimate is intuitively unreasonable and discrepant to the clinical experts' decisions, where multiple factors are considered with complementary, contradictory, or competitive relations with a different degree on the overall clinical decision [12]. Not accounting for this uncertainty does not allow to consider the magnitude of the deviation risks. Besides, small and large deviations are considered at the same level of

* Corresponding author.
E-mail address: lvxd@zju.edu.cn (X. Lu).

https://doi.org/10.1016/j.asoc.2022.109710
1568-4946/

compliance, which hampers both the accuracy and the flexibility of the provided diagnostics.

Another under-investigated topic within state-of-the-art conformance checking techniques regards the possibility of tailoring the generated diagnostics to the needs and preferences of the analyst. Indeed, in the presence of multiple non-compliant behaviors detected in a process execution, the approaches in [7,13,14] assess the overall compliance level by summing up the costs for all the violated constraints. However, 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 set of constraints. Furthermore, the method is not adaptive to the user's needs. For instance, 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.

To address the aforementioned challenges, we argue that data-driven approaches to multi-perspective compliance checking should take into account the tacit knowledge and preferences of the user, as well as the knowledge about the specific properties of the process. The way humans combine information can also be quite rich, depending on the context of the application (see, *e.g.*, [15]). This implies that the conformance checking algorithm must rely on a versatile formalism in order to obtain alignment results that match the goals of the conformance analysis. In this work, we use *fuzzy sets theory* [16] as the formalism upon which our novel approach to conformance checking is based in order to deal with the challenges mentioned above. In particular, fuzzy sets are used to represent expert knowledge about the flexible boundaries in process constraints, to account for contexts where there is usually some tolerance to imprecision (*e.g.* the INR condition in atrial fibrillation, as explained above). Furthermore, fuzzy set aggregation operators [17] are used for complex information fusion, similar to the way humans combine information in order to assess process compliance. The extensive set of fuzzy set aggregation operators provide a versatile formalism to capture this combination.

In other works in [18,19], we performed an exploratory study on the use of fuzzy sets in conformance checking. While the obtained results provided promising evidence that fuzzy sets represent a valuable asset to represent uncertainty in human decision making process, the work still has some limitations. In particular, the approach is not able to detect deviations involving both ordering and data constraints, thus decreasing the accuracy of the results. In this work, we extend the approach in [18,19] in order to overcome these drawbacks. Furthermore, we extend the formalization of the approach and we discuss its application in two real-world case studies in the healthcare domain. The main contributions of the present work are then the following.

- We introduce a novel multi-perspective conformance checking approach, based on fuzzy sets theory, able to detect data deviations also for skipped activities and to account for uncertainty when assessing (multiple) data constraints violations.
- We carry out an experimental validation on two real-world case studies, to validate the approach and to prove its feasibility for real-world problems.

The remainder of this paper is organized as follows. Section 2 introduces basic notions used throughout the paper. Section 3 illustrates our approach. Section 4 discusses the results obtained in the case studies. Section 5 discusses related work. Finally, Section 6 draws some conclusions and discusses future work.

2. Background

This section introduces concepts that are used through the paper. We first recall core concepts related to *event logs* and *process models*. Then, we introduce basic elements of *fuzzy sets theory*.

2.1. Process models, events, logs

Conformance checking techniques aim at detecting discrepancies between a process model and the real process executions. Our approach aims at multi-perspective conformance checking, which takes into account not only *control-flow* constraints, related to the activity execution order, but also *data guards*, i.e., constraints on the admissible values for the process variables in different parts of the process. A guard can be any formula over the process variables using relational operators ($<, >, =$) as well as logical operators such as conjunction ($\wedge$), disjunction ($\vee$), and negation ($\neg$). We denote with $\mathsf{Formulas}(X)$ the universe of such formulas defined over a set X of variables. For each $x_i \in X$, $x_i \in \mathbb{R}$.

A plethora of formalisms have been proposed in the literature to represent process models, e.g., Petri net [20], BPMN [21] and so on. Since our approach is not constrained to the use of a specific formalism, here we define the notion of process model using the general notion of *transition systems*, employing the notation from [4] enriched with data-related notions, similarly to what has been done in [13]. This allows for more simple descriptions of our algorithms.

Definition 1 (*Process Model*)**.** A process model $M = (A_M, V, U, \mathsf{Val}, W, T, \Lambda, \mathcal{P}, \mathcal{P}_\mathcal{I}, \mathcal{P}_\mathcal{F})$ is a transition system involving [7]:

- a set of process activities A_M;
- a set of transition identifiers Λ;
- a set of process variables V;
- a set of variable values U; note that U also includes the special symbol $\perp$, that stands for "undefined value";
- a function $\mathsf{Val} : V \rightarrow 2^U$ that defines the values admissible for each variable v, i.e., $\mathsf{Val}(v)$ is the (potentially infinite) domain of v;
- a set of locations $\mathcal{P}$, with initial location(s) $\mathcal{P}_I$ and final location(s) $\mathcal{P}_F$;
- a set of transitions $T : \Lambda \rightarrow (\mathcal{P} \times A_M \times \mathcal{P})$;
- a write function $W : \Lambda \rightarrow 2^V$ that labels each transition with the set of variables written/updated by the transition;
- a guard function $G : \Lambda \rightarrow \mathsf{Formulas}(V \times V)$, which associates each transition with a guard.[1]

Definition 2 (*State of A Process Model*)**.** Let M be a process model. We define a *state* of M as any pair $p = (P, \Phi)$ of location P and variable assignment $\Phi : V \rightarrow U$, and indicates the set of process states with $\Gamma = \mathcal{P} \times (V \rightarrow U)$ [7].

Example 1. Let us consider the process model M in Fig. 1. Here we have the activity set $A_M = \{a, b, c\}$, the variable set $V = \{v_1\}$, the variable the value set $U = \{x_i \in \mathbb{R}\}$. Circles correspond to locations; we have the location set $\mathcal{P} = \{P_1, P_2, P_3, P_4\}$, with initial location $P_I = \{P_1\}$ and final location $P_F = \{P_4\}$. Edges between circles correspond to the transition set $T = \{(t_1, (P_1, a, P_2)), (t_2, (P_2, b, P_3)), (t_3, (P_2, c, P_4)), (t_4, (P_3, c, P_4))\}$. For the sake of simplicity, the figure only shows the process activities as edge labels, together with the guards if defined. In this model, the variable v_1 is updated only once, by the activity a; so, the

[1] If no guard is defined for a transition λ, $G(\lambda) = \mathsf{True}$.

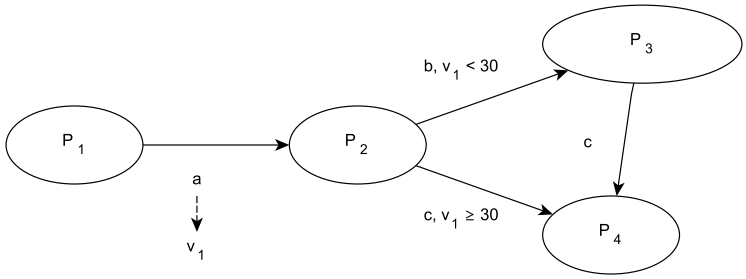

Fig. 1. An example process model by transition systems.

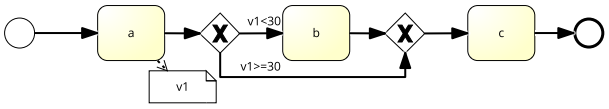

Fig. 2. A process model in BPMN notation equivalent to the model in Fig. 1.

write function $W(a) = v_1$. We represent this by linking a to v_1 by means of a dashed arrow. Furthermore, two guard functions G are defined on the admissible values for v_1 in two transitions, i.e. $(G(t_2) = \{v_1 < 30\})$, $(G(t_3) = \{v_1 \geq 30\})$; both are represented on the corresponding edges. It is straight to see that the set of states for this example is infinite, since v_1 can assume any value within the reals domain and we have for each collection as many states as the possible assignment for each variable. Therefore, we will have a state with $v_1 = 0$, another with $v_1 = 1$ and so on.

The behaviors expressed by the process model in Fig. 1 can be expressed also by other commonly used process model formalisms. As an example, Fig. 2 shows a process model in BPMN notation which is semantically equivalent to the one discussed in Example 1. The rectangles with rounded corners represent process activities, while arrows represent ordering relations among activities. The white rectangles with the folded corner represent data variables, linked by means of a dotted arrow with the activity which writes/updates them. Diamond-shaped elements are called *gateways* and allow to model activity routing. Gateways with the plus symbol are called "AND split-join" (the starting and the closing gateway, respectively) and they represent concurrent activities. The cross gateways are "XOR split-join" and they model alternative choices. Data guards impacting activity routing are expressed as textual labels over the branches of the XOR gateway they refer to.[2]

For the sake of simplicity, for the remainder of the paper we use process models in BPMN, since it is a widely used formalism in industry. However, note that our approach can be used for any language for which it is possible to build a transition system like the one described in Definition 1.

A process model describes all the admissible process executions, hereafter referred to as *process traces*. Generating a process trace means executing every activity from an initial state to a final state, i.e. to generate an execution path of the graph from one of the states corresponding to a location in $\mathcal{P}_{\mathcal{I}}$ to one of the states corresponding to a location in $\mathcal{P}_{\mathcal{F}}$, updating the process variables when needed. Note that every time an activity is executed, only one of the states in each states collection is chosen, i.e., the one consistent with the current variables assignment. We refer to the execution of an activity as a *transition firing* [7].

[2] For a more detailed overview of the BPMN notation, the reader can refer to http://www.bpmn.org/.

Definition 3 (*Transition Firing*)**.** Let M be a process model. A *transition firing* is a tuple[3] $s = (\lambda, w) \in \Lambda \times (V \nrightarrow U)$.

A transition firing $s = (\lambda, w)$ is *valid* in a state (P, Φ) if the following conditions are satisfied [7]:

- $\exists\ P' \in \mathcal{P} \mid T(\lambda) = (P, a, P')$;
- λ writes new values to the defined set of variables, i.e. $\mathrm{dom}(w) = W(\lambda)$;
- $\forall v \in \mathrm{dom}(w),\quad w(v) \in \mathrm{Val}(v)$;
- $G(\lambda)$ evaluates to true with respect to Φ and Φ', where Φ' is defined as follows:

$$\forall v \in V \quad \Phi'(v) = \begin{cases} w(v) & \text{if} \\ \Phi(v) & \text{otherwise.} \end{cases}$$

If any of the previous conditions does not hold, the firing is *invalid*.

In the remainder, the set of valid and invalid transition firings of a process model M is denoted as S_M. On firing of an activity, the process moves from the current state to the next one. This is denoted as $(P, \Phi) \xrightarrow{s} (P', \Phi')$. The concept of single transition firings can easily be extended to firing sequences $\sigma = \langle s_1, \ldots, s_n \rangle$, which can be expressed as $(P_0, \Phi_0) \xrightarrow{\sigma} (P_n, \Phi_n)$ and it is equivalent to writing $(P_0, \Phi_0) \xrightarrow{s_1} (P_1, \Phi_1) \xrightarrow{s_2} (P_2, \Phi_2) \ldots \xrightarrow{s_n} (P_n, \Phi_n)$. A firing sequence is valid if any final state of a transition firing is equal to the initial state of the subsequent one. The set of all valid firing sequences from the initial state(s) to the final state(s) represents the set of *process traces*, i.e., the set of all possible behaviors allowed by the model. In the remainder, given an initial set of locations P_I and a final set of locations P_F we indicate with $\mathcal{B}_{P_I,P_F}$ the set of all valid process traces leading from the initial state to the final state, i.e., $\mathcal{B}_{P_I,P_F} = \{\sigma \in S_M^*$[4] $\mid \exists P_F \ \ (P_I, \Phi_I) \xrightarrow{\sigma} (P_F, \Phi_F)\}$. The technique described in this paper requires a process model to be *relaxed data sound*, which means that there exists at least one valid firing sequence that leads from the initial to the final state(s).

Process executions are often recorded by means of an information system in event logs. An event log consists of *traces*, each collecting the sequence of *events* generated during a process execution.

Definition 4 (*Event, Event Trace, Event Log*)**.** An event $e = (a, D, c, i) \in A_M \times (V \nrightarrow U) \times \mathcal{C} \times \mathbb{N}^+$ is a tuple consisting of an executed activity $a \in A_M$ [7], a function $D : V \to \mathrm{dom}(V)$, which assigns a value to some process variables (possibly all of them), a case identifier c belonging to the case set $\mathcal{C}$ and a number $i \in \mathbb{N}^+$. A case corresponds to a single process execution; the number i identifies the position of the event within the sequence of events occurred within a case. The set of events is denoted by $\mathcal{E}$. An event trace $\sigma_L \in \mathcal{E}^*$ is a sequence of events. An **event log** is a multi-set of event traces L.

[3] We use $2^V \nrightarrow U$ to denote partial functions, i.e., functions whose domain is a subset of V.

[4] S_M^* is the set of all the sequences built with the elements in S_M.

2.2. Fuzzy set theory

Classic set theory define crisp, dichotomous functions to determine membership of an object to a set. Although crisp sets have proven to be useful in various applications, human thoughts and decisions are often characterized by some degree of uncertainty and flexibility, which are hard to represent in a crisp setting [22]. Fuzzy sets extend the classic set theory to represent the uncertainty and flexibility often involved in human decision-making [22,23], providing a meaningful representation of vague concepts expressed in natural language and close to human thinking [23].

While the characteristic function of a crisp set assigns a value of either 0 or 1 to each element in the universal set, the fuzzy sets generalize the values to fall within a specified range and indicate the membership grade of these elements [23]. This grade corresponds to the degree to which that individual is similar or compatible with the concept represented by the fuzzy set [24]. Thus, individuals may belong in the fuzzy set to a greater or lesser degree as indicated by a larger or smaller membership grade. The extreme values in this interval, 0 and 1, then represent, respectively, the total denial and affirmation of the membership in a given fuzzy set as well as the falsity and truth of the associated proposition [23]. Formally, a *fuzzy set* is defined as follows.

Definition 5 (*Fuzzy Sets*)**.** Let N be a collection of objects. A *fuzzy set* [23] F over N is defined as the set of ordered pairs $F = \{n \in N, \mu_F\}$, where μ_F is called the membership function, defined as $\mu_F : N \rightarrow [0, 1]$.

Fuzzy sets are extensions of classical sets, with the characteristic function allowed to take any value between 0 and 1. In the literature several standard membership functions have been defined for practical applications (see, e.g., [23] for an overview of commonly used functions).

In addition to the assessment of single membership functions, the fuzzy set theory also provides various functions to combine multiple variables. Indeed, many decision making processes need to evaluate multiple criteria before taking a final decision. To model these contexts, within the fuzzy set theory the notion of *aggregation operations* (AOs) is introduced. These are mathematical functions that satisfy several set boundary and monotonic conditions, which allow to specify how to combine the different criteria that are relevant when making a decision [25,26].

3. From crisp to fuzzy multi-perspective conformance checking

This work introduces a compliance checking approach aimed to take into account possible uncertainty affecting the definition of data-guards by quantifying the *degree* of deviations. We argue that such a choice allows to obtain more accurate diagnostics, possibly closer to human interpretation. To this end, we investigate the use of fuzzy sets theory.

A conformance checking framework requires to define three main components; a *set of moves*, which determine the kind of deviations that can be detected and from which an *alignment* can be obtained; a *cost function*, which determines the severity of each deviation; and an *alignment building* function, which computes the alignment at minimum cost.

Each of these components is detailed in the following subsections.

3.1. Defining the moves set

Given an event log L and a process model M with the set S_M of transition firings $s = (\lambda, w)$, conformance checking builds an alignment between L and M, whose goal consists in relating each event $e = (a_L, D, c, i)$ (we use a_L to emphasize the activity a in the log) occurring in the event trace to the activity in the process trace and vice versa. To this end, we need to map "moves" occurring in the event log (i.e., events) to possible "moves" in the model (i.e., transition firings). However, since the executions may deviate from the model [4], we might have log/model moves which cannot be mimicked by model/log moves respectively. A "no move" symbol "$\gg$" is used to represent moves which cannot be mimicked. For convenience, we introduce the set $S_M^{\gg} = S_M \cup \{\gg\}$, and the set $\mathcal{E}^{\gg} = \mathcal{E} \cup \{\gg\}$. Hereafter, we refer to a move in a event trace as $s_L = (a_L, D, c, i)$ and to a move in a process trace as $s_M = ((\lambda, (p, a_M, p')), w)$, with the previous position $p = (P, \Phi)$, the next position $p' = (P', \Phi')$, and the activity a_M. In the following, we introduce the notion of *Legal Move set*.

Definition 6 (*Legal Move Set*)**.** A *legal move* is represented by a pair $(s_L, s_M) \in \mathcal{E}^{\gg} \times S_M^{\gg}$, where $s_L = (a_L, D, c, i)$ and $s_M = ((\lambda, (p, a_M, p')), w)$, with $p = (P, \Phi)$ and $p' = (P', \Phi')$, such that:

- (s_L, s_M) is a *move in log* if $s_L \in \mathcal{E}$ and $s_M = \gg$;
- (s_L, s_M) is a *move in model with fulfilled guards* if $s_M \in S_M$ and $s_L = \gg$ and the data guard $G(s_M)$ evaluates True over Φ and Φ';
- (s_L, s_M) is a *move in model with violated guards* if $s_M \in S_M$ and $s_L = \gg$ and $G(s_M)$ evaluates False over Φ and Φ';
- (s_L, s_M) is a *move in both with fulfilled guards* if $s_L \in \mathcal{E}$, $s_M \in S_M$ and $a_L = a_M$ and $G(s_M)$ evaluates True over Φ and Φ';
- (s_L, s_M) is a *move in both with violated guards* if $s_L \in \mathcal{E}$, $s_M \in S_M$ and $a_L = a_M$ and $G(s_M)$ evaluates False over Φ and Φ'.

For the remaining of this manuscript, we refer to these set of moves as "extended moves", since it extends the move set used in previous approaches [18,19]. In particular, there the data costs were considered only for synchronous moves, with the result that data violations occurring for moves on model go undetected. To solve these issues, we introduce here a distinction between *move in model with fulfilled guards* and *move in model with violated guards*. By doing so, we can assess data constraints from the point of the process where they are defined, regardless of whether or not we find a matching event in the log. Once we defined the move set, we are able to formally introduce the notion of *alignment*, defined below.

Definition 7 (*Data-relaxed Alignment*)**.** Let $M = (A_M, V, U, \mathsf{Val}, W, T, \Lambda, \mathcal{P}, \mathcal{P}_I, \mathcal{P}_F, G)$ be a process model and let $M' = (A_M, \emptyset, U', \mathsf{Val}', W', T, \Lambda, \mathcal{P}, \mathcal{P}_I, \mathcal{P}_F, G')$ be a model with the same transitions and activities as M but such that $\mathsf{dom}(\mathsf{Val}') = \mathsf{dom}(U') = \mathsf{dom}(W') = \mathsf{dom}(G') = \emptyset$. Namely, M' represents a model with the same control-flow of M, but with no variables, guards, or writing operations. Let $\mathcal{B}_{M,P_I,P_F}$ be the set of valid traces for M. Let A_{LM} be the set of all legal moves. The *alignment* between a log trace $\sigma_L \in \mathcal{E}^*$ and a process trace $\sigma_M \in S_M^*$ 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 and σ_M is a valid process trace in $\mathcal{B}_{M,P_I,P_F}$.

We would like to point out that the notion of alignment used in this work differs from the definition used in state-of-the-art approaches [7]. According to the latter, given the set of all legal moves A_{LM}, the *alignment* between a log trace $\sigma_L \in \mathcal{E}^*$ and a

Table 1
Two possible alignments built by our approach for the model in Fig. 2 and σ_{L_1}.

Alignment γ_1			Alignment γ_2		
Log	Model	Guard	Log	Model	Guard
a $\{v_1 = 35\}$	a $\{v_1 = 35\}$	N/A	a $\{v_1 = 35\}$	a $\{v_1 = 35\}$	N/A
b	b	N	b	$\gg$	N/A
c	c	N/A	c	c	N/A

process trace $\sigma_M \in S_M^*$ 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 [27]. However, it is worth noting that the set of extended moves can result in process traces that are *invalid* with respect to data constraints; therefore, we cannot refer to the standard notion of alignment. We illustrate this behavior with the example below.

Example 2. Let us consider the model in Fig. 1 and the trace $\sigma_{L_1} = \langle (a, \{v_1 = 35\}), b, c \rangle$. Table 1 shows two possible alignments γ_1 and γ_2 for σ_{L_1} returned by our approach. The columns *Log* and *Model* represent moves in log and in model, respectively, while the *Guard* column provides information on whether the values of the process variables comply with the data guards. The symbol N/A is used to indicate where no guards are defined for the transition, and *N* means the guard is violated. It is worth noting that while the process trace in γ_2 is still a valid process trace, since *b* is not performed, the process trace in γ_1 is an invalid process trace with respect to the process model, since data conditions are violated.

While γ_1 does not fulfill the standard notion of alignment, we intend to be able to generate also this kind of diagnostics. In fact, our approach aims at determining not only if a data violation occurs, but also its magnitude; to this end, we need to consider the actual value stored in the event log. Note that this interpretation differentiates our approach from the multi-perspective alignment [7] which, instead, strives for determining the closest data value that would make the trace compliant with the model, without considering the actual data variable value when generating the diagnostics. We elaborate more upon these differences in Section 5.

3.2. The cost function with fuzzy sets

As shown in Example 2, 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, we introduce a tailored *cost function*.

Definition 8 (*Cost Function, Optimal Alignment*)**.** Let (s_L, s_M) be a move between a log trace and a process trace, and σ_L, σ_M be a log trace and a process 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 overall *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 is (one of) the alignment(s) with the lowest cost according to the provided cost function.

Typical cost functions that accounts explicitly for the data perspective [7] are affected by two main limitations. First, they consider every move either as *completely wrong* or *completely correct*. Second, they support only the interpretation of data violation in terms of the number of incorrect variables, which is not always suitable for the analyst's needs. To differentiate between different magnitude of deviations and to allow the process analyst to customize the interpretation of multiple data violations, we use the fuzzy sets concepts introduced in Section 2. In particular, we propose to define for each data variable (a) a range of values admissible according to the domain experts, and (b) a membership function quantifying to which extent a given value is compliant according to the experts' knowledge. The membership function is hence used to define the "*violation cost*" of the data variable. Aggregation functions are used to assess conditions involving multiple variables. Note that here we aim at fuzzifying only the computation of data moves, allowing some flexibility as to the acceptable violations. However, an activity is either executed or skipped, and so, there is not a notion of "partial" execution. Nevertheless, our approach can in principle be extended to account also for the control-flow perspective, for instance, to deal with cases in which we can distinguish between the start and the end of activities, or when an activity has been replaced by another which performs similar operations or, again, to differentiate between deviations occurring under different conditions. We plan to explore these directions regarding the uncertainty of control flow in future work.

Definition 9 (*Aggregated Cost Function With Fuzzy Sets*)**.** Let (s_L, s_M) be a move between a log trace and a process trace. We define a membership function $\mu(s_M, v_i)$ indicating the compliance degree of the data variable v_i with respect to the data guard in the model corresponding to the transition fired in s_M. Let $\pi(\mu_1, \mu_2, \ldots, \mu_n)$ be a user-defined aggregated membership function of multiple variables $V_1, \ldots, V_n$. Then $(1 - \pi)$ is the overall deviation cost of a set of variables. The cost $k(s_L, s_M)$ is defined as:

$$k(s_L, s_M) = \begin{cases} 1 & \text{if move in log} \\ 1 & \text{if move in model with fulfilled guards} \\ 2 - \pi(\mu_1, \ldots, \mu_n) & \text{if move in model with violated guards} \\ 1 - \pi(\mu_1, \ldots, \mu_n) & \text{if move in both with violated guards} \\ 0 & \text{otherwise.} \end{cases}, \quad (1)$$

Note that the membership function defined on a data variable characterizes a soft constraint on that variable. The aggregation of multiple soft constraints characterizes the combined violations over multiple variables. Further, since a move on model with violated guards involves two kinds of deviations, we consider both of them in computing its cost. In particular, we compute the cost of the moves as the sum between the cost related to the control-flow deviation and the cost related to the violated data guards, i.e., 1- $\pi(\mu_1, \ldots, \mu_n)$.

For calculating the cost function in (1), we first need to define a membership function for every data constraint we are interested in.[5] For the sake of simplicity, in the remainder of the manuscript we refer to the cost function of Definition 9 as "fuzzy cost function".

Example 3. Let us consider again the alignment γ_1 in Table 1 and the model in Fig. 1. According to a crisp cost function, the cost for the second move would be 1, since the value of the variable v_1 does not fulfill the corresponding guard. Now, let us assume interviewing an expert of this process, who tells us that values of v_1 up to 40 are still acceptable, even though not optimal. Since everything above a given value is not acceptable, let us assume

[5] Note that multiple μ functions can be defined for the same data variable, if it is used in multiple guards.

we represent this knowledge using a so-called L-function, defined as follows.

$$\mu(b, V_1) = \begin{cases} 0 & \text{if } V_1 \geq V_{ub} \\ \dfrac{V_{ub} - V_1}{V_{ub} - V_o} & \text{if } V_o < V_1 < V_{ub} \\ 1 & \text{if } V_1 < V_o \end{cases}, \quad (2)$$

where V_{ub} represents the upper bound the analyst is willing to accept, while V_o represents the ideal value represented by the constraint. Therefore, with $V_1 = 35$, $V_o = 30$ and $V_{ub} = 40$, we would obtain a cost equal to 0.5 for this move.

Example 4. Consider now a trace similar to the one considered in Example 3, but with a different data value. In particular, let $\sigma_{L_2} = \langle (a, \{v_1 = 25\}), (c, \perp) \rangle$. This execution is not compliant with respect to the model in Fig. 1 since v_1 being below 30, the activity b should have been performed. According to our set of moves, and assuming to assign the process variables the same values observed in the log trace, the corresponding move $(\gg, (b)$ would be a move on model with violated guards. Using the same membership function from Example 3, the cost of this move is computed as $1 + 1 - 0$, i.e., 2. It is worth noting that, using the move set defined in the approaches [18], the data cost for this move would go undetected, since b is a move on model.

While the total cost of an alignment provides an assessment of the severity of the non-compliant behaviors shown in the trace, conformance checking techniques usually use the so-called Fitness metric when building the diagnostics for the analyst. This value ranges between 0 and 1 and measures the extent to which the log traces can be associated with valid execution paths specified by the process model [28]. Here we refer to the definition of the fitness function for trace σ_L and Model M, $F(\sigma_L, M) \rightarrow [0, 1]$ used by Mannhardt et al. [7], according to which $F(\sigma_L, M) = 1$ if the trace σ_L can be replayed by the model from the beginning to the end without any discrepancy. $F(\sigma_L, M) = 0$ corresponds to the poorest level of the alignment. Let γ_O be the optimal alignment of σ_L and M, and γ_R be the "reference alignment" that represents the "worst case scenario". In particular, the reference alignment is obtained by concatenating the alignment of the process model with the empty trace with a sequence involving as many moves on the log as events in the trace. It represents a case in which none of the events in the log are matched with any of the activities in the model.

Definition 10 (*Fitness Function*)**.** Let $J(\gamma)$ to be the cost of a complete alignment γ. Then the fitness level is defined as follows [13]:

$$F(\sigma_L, M) = 1 - \frac{J(\gamma_O)}{J(\gamma_R)}. \quad (3)$$

3.3. Membership function and aggregation

The application of our method requires the definition of a membership function for a fuzzy set and the selection of an aggregation function for combining different sets. In this section, we provide a brief discussion on how these choices could be made, which can be used as guidelines.

The definition of a membership function depends on the application, the problem under analysis and the preferences of the domain experts. In general, the more information is available for a problem, the better the membership function can be tailored to the specific requirements of the domain. A data constraint implies that there are some admissible values for the constraint and others that are not allowed. This corresponds to a classical (crisp) definition of a data constraint. In many problems, some tolerance is allowed, which induces a preference structure over the data values. Fuzzy constraints are used to model this case. In this case, we define, for each data constraint, a membership function whose core (i.e., the set of values for which the membership value is 1) denotes the set of admissible data values as defined by the domain expert. Outside the core, we define a tolerance interval in which the set of values we consider belong partially to the set of admissible values (see also Section 2). From a conformance point of view, this choice corresponds to determining, for a given data constraint, which values should be considered acceptable, although they are not optimal or preferred.

The definition of the core and the tolerance interval requires knowledge from a process analyst or domain expert. If explicit knowledge is not available, simple data exploration tools (e.g., box plots) on past process executions can be used to support this choice. For example, if we observe multiple deviations of a data value in a given range, it might be reasonable to assume that these deviations could be considered, to a given extent, as acceptable. More complex methods to derive the membership functions from sample data have also been proposed (see e.g., [29]).

After defining the core of the fuzzy constraint and the tolerance interval, we need to select the *shape* (type) of the membership function, which induces the preference structure over (partially) admissible data values. In the literature, different types of parameterized membership functions have been defined (see, e.g., [23] for an overview), with different levels of complexity and different interpretations. When there is no information regarding a preference within the tolerance interval, a regular characteristic function for an interval can be used for the membership. Often, however, there is some information regarding the preference (e.g., the smaller the deviation the better it is). In that case, the *triangular* or *trapezoidal* functions are commonly used when one wants the degree of membership to increase/decrease linearly according to the distance to the core of the set. In the absence of specific needs expressed by the process experts, it is a common practice to start with these functions, since they are the most intuitive ones, and discuss the intervals with the experts to determine whether they are in line with their expectations. Furthermore, the triangular membership functions have the advantage that under some assumptions regarding the underlying probability density function of the data, they reduce the reconstruction error [30]. Similarly, if a normal distribution of the degree of memberships is more appropriate, the *Gaussian* membership functions can be used.

Aggregation of fuzzy sets is a topic studied extensively in the fuzzy systems literature (see [23,31] for an overview). Different functions have different level of complexity and different classes of aggregation functions have different interpretations. For example, t-norms are used for conjunctive aggregation of constraints (aim to satisfy all the constraints at the same time), t-conorms are used for disjunctive aggregation of constraints (aim to satisfy at least one constraint) and averaging operators are used to allow compensation between different constraints [32].

A suitable class of aggregation operators for compliance analysis are the t-norms, since they are used to model conjunction of fuzzy sets, thus resulting in a convenient choice to model the need of satisfying multiple constraints at the same time during a process execution. Widely used t-norms are the minimum operator, which returns the minimum membership function value; the product operator, which returns the product of the membership functions; and the Yager t-norm, which is a parametric t-norm and generalizes various well-known t-norms [33]. In this paper, we focus on the product operator, since it is widely used and allows some interaction between the values being aggregated.

3.4. Finding the optimal alignment

The problem of finding an optimal alignment is usually formulated as a search problem in a directed graph and solved by using a search algorithm such as the A* algorithm to find a least costly path in the graph [34]. Let $Z = (Z_N, Z_E)$ be a directed graph with Z_N being the set of nodes and Z_E the set of edges, weighted according to some cost structure. The A* algorithm finds the path with the lowest cost from a given source node $n_0 \in Z_N$ to a node of a given goal set $Z_G \subseteq Z_N$. The cost for each node n is determined by an evaluation function

$$f(n) = g(n) + h(n), \tag{4}$$

where:

- $g : Z_N \rightarrow \mathbb{R}^+$ gives the smallest path cost from n_0 to n;
- $h : Z_N \rightarrow \mathbb{R}_0^+$ gives an estimate of the smallest path cost from n to any of the goal nodes.

Given a log trace σ_L and a process model M, we associate every node of the search space with a prefix of some complete alignments. Since a different alignment is also associated to every search-space node and vice versa, we use the alignment to refer to the associated state. The source node corresponds to an empty alignment $\gamma_0 = \langle\rangle$, while the set of target nodes correspond to every complete alignment of σ_L and M. For every pair of nodes n_1, n_2 in the search space, we have that $(n_1, n_2) \in Z_E$ only if the alignment γ_2 corresponding to n_2 is obtained by adding one (legal) move to the alignment γ_1 corresponding to n_1. The cost associated with a path leading to a graph node corresponding to an alignment γ is then defined in [7] as

$$g(\gamma) = K(\gamma) + \epsilon|\gamma|, \tag{5}$$

where $K(\gamma) = \sum_{s_L, s_M \in \gamma} k(s_L, s_M)$, with $k(s_L, s_M)$ defined as in Definition 9, and $|\gamma|$ is the number of moves in the alignment, while ϵ is a negligible cost, usually added in literature to guarantee termination. In order to be employed in the A* start algorithm, the cost g has to be strictly increasing. The work in [7] has formally proved that the A* algorithm is always satisfied for cost functions $k(s_L, s_M)$ with non negative elements, which is the case for the cost function in Definition 9.

For the definition of the heuristic cost function $h(\gamma)$, 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, on the basis of the model adopted to represent the process model. For our implementation, we adopt the heuristic cost function h in [5], defined as the difference between the number of remaining events in the log and the number of remaining transitions in the process model, regardless whether or not the match is perfect. For example, if there are enough events to match all the remaining activities in the model, $h = 0$. While a formal introduction on the heuristic cost function is out of the scope of this paper, we would like to point out that previous work has shown that this heuristic cost function still satisfies the admissible rule when considering other perspectives [7], thus resulting suitable for our purposes. It is noted that there is no data cost associated with h since it only estimates the control flow cost with the number of the remaining events. We would like to point out, however, that our approach is not constrained to the use of a specific heuristic.

Algorithm 1 provides a high-level overview of the overall alignment algorithm. The algorithm takes as input a process model $\mathcal{M}$, a log trace σ_L, a cost function $\mathcal{K}$, and a *policy* defining how to deal with missing variables. $\mathcal{B}$ is the set of valid traces generated from the model. Different alternative strategies can be used for determining how to consider these missing variables. We argue that this is a choice depending on the analysts' preference and, consequently, we allow her to customize this choice. Examples of policies are, for instance, considering all these variables as invalid, i.e., assigning them a value outside of their domain, so that every guard using the variables are assigned the maximum cost. Alternatively, the user might decide to assign them to a default value or, for numeric variables, to the average of the values observed in the other traces, and so on. After the missing variables have been pre-processed according the policy, e.g., set to be median, mean, etc. (line 1), and the data conditions are abstracted from the process model (line 2), an initial alignment γ_c and two sets, $\mathcal{OPEN}$ and $\mathcal{CLOSED}$ are initialized (lines 3–5). The $\mathcal{OPEN}$ set includes all successors of the current optimal alignment, and the alignments that have not been chosen in the past. It thus represents the set of all possible alignments that can be chosen as the next node. The $\mathcal{CLOSED}$ set, instead, keeps track of all the alignments that have been selected up to a given point in the search process. The while loop (lines 6–13) involves the key element of the search process. The condition states that the loop continues until we find a "final" alignment, i.e., an alignment γ_c such that its projection on the moves on model $\gamma_c \mid_M$ returns a process trace belonging to the process trace set $\mathcal{B}_{M,P_I,P_F}$ from the initial position P_I to the final position P_F of model M, while its projection on the moves on log $\gamma_c \mid_{\sigma_L}$ returns the log trace σ_L. First, the algorithm invokes the function *expand_alignment* (line 7), which generates the set of legal moves (according to Definition 6) that can be obtained from γ_c, thus obtaining a set of new candidate alignments Γ.

For each alignment $\gamma' \in \Gamma$ the cost is computed, according to Definition 9. Then, the alignment is put in the $\mathcal{OPEN}$ set, together with its corresponding cost (lines 9–10). Once the cost of each expansion has been computed, the function *pick_min_cost* returns the alignment corresponding to the minimum cost (line 11), and the selected alignment becomes the new γ_c, and is removed from the $\mathcal{OPEN}$ set. Then the current alignment is put in the $\mathcal{CLOSED}$ set, to avoid this node to be expanded in further iterations. If more than one alignment correspond to the optimal one, the function picks one arbitrarily.

The core element of Algorithm 1 is represented by the function *expand_alignment*, described in Algorithm 2. On line 2, we compute the *control flow successors* of an alignment γ. The set of

Algorithm 1: Multi-perspective alignment searching process

Input : Process Model ($\mathcal{M}$), log trace σ_L, cost function $\mathcal{K}$, missing variables policy *POLICY*, the set of valid traces $\mathcal{B}$.
Output: Optimal alignment γ

1 $\sigma_L = preprocessMissingVariables(\sigma_L, POLICY)$; // assign values
2 $abstract\_data(M)$; // abstract the data conditions
3 γ_c=$\langle\rangle$;
4 $\mathcal{OPEN} = \emptyset$;
5 $\mathcal{CLOSED} = \emptyset$;
6 **while** $\gamma_c \mid_M \notin \mathcal{B}_{M,P_I,P_F} \wedge \gamma_c \mid_{\sigma_L} \neq \sigma_L$ **do**
7 $\quad \Gamma_n = expand\_alignment(\gamma_c, M, \sigma_L)$;
8 $\quad$ **foreach** $\gamma' \in \Gamma_n$ **do**
9 $\quad\quad k_{\gamma'} = compute\_cost(\gamma', K)$;
10 $\quad\quad \mathcal{OPEN}.add(\gamma', k_{\gamma'})$;
11 $\quad \gamma_c = pick\_min\_cost(\mathcal{OPEN})$;
12 $\quad \mathcal{OPEN}.remove(\gamma_c)$;
13 $\quad \mathcal{CLOSED}.add(\gamma_c)$;
14 **return** γ_c

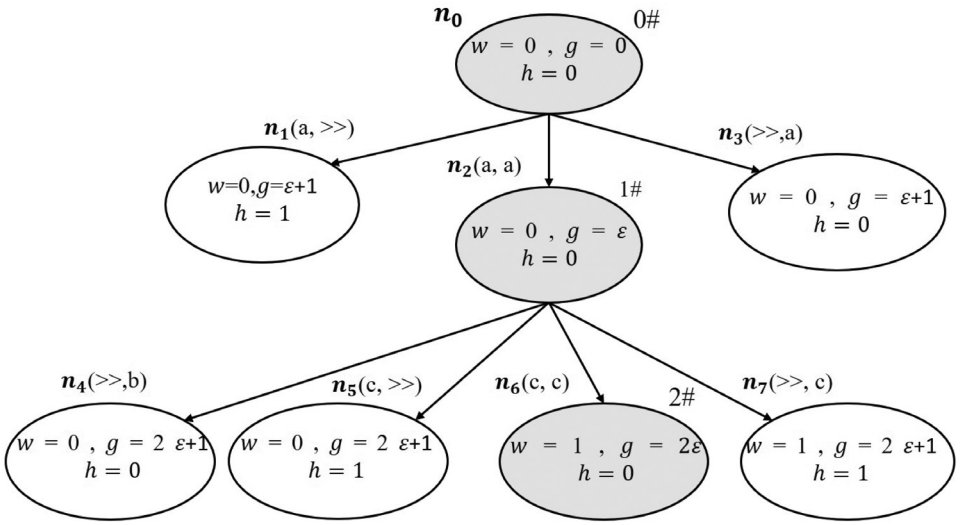

Fig. 3. The search process constructed to find an optimal alignment with 1.

control-flow successors γ_{ctr_γ} of a trace σ_L and a model M with initial and final locations P_I, P_F is defined as [7]:

$$\gamma_{ctr_\gamma} = \{\gamma_c \in A^*_{LM} \mid \gamma_c = \gamma \oplus \langle (s_L, s_M) \rangle \wedge \gamma_c|_{\sigma_L} \in prefix(\sigma_L) \wedge \exists \sigma_M \in \mathcal{B}_{M,P_I,P_F} \mid \gamma_c|_{\sigma_M} \in prefix(\sigma_M)\}.$$

The control-flow alignment γ_c belongs to the set of all legal moves A^*_{LM}. The symbol $\oplus$ stands for sequence concatenations, while $|_{\sigma_L}, |_{\sigma_M}$ represent the projection over the log/process trace σ_L and σ_M respectively. The mark *prefix* presents a prefix of complete alignments. Hence, we only consider moves on log/model.

Informally, γ_{ctr_γ} corresponds to all the possible log/model moves that can be executed at the current point of the log/process trace, neglecting the data, where σ_{M_c} and σ_{L_c} stands for the log/model moves respectively (line 3–4). While the concrete implementation depends on the adopted process model formalism, the high level idea consists of determining (a) the successor event $s_{L_{next}}$ in the log trace that comes after the last event $s_{L_{prev}}$ (line 5–6), and (b) the set of process activities $S_{M_{next}}$ that are successors of the activities in the current model moves (line 8) after the data conditions are abstracted (line 7). These elements are then combined to generate the set of all possible moves with a for-each loop (lines 9–18). First, we need to select a variable assignment Φ for each move in model. It should be noted that, in theory, this set of moves becomes infinite, since variables in process traces can have potentially infinite domains. However, in this work we exploit the variable assignment D of the last event $s_{L_{prev}}$ in the event log to generate a finite number of moves with the assignment Φ, and initialize the next Φ' with the same assignment (line 10–11). As explained above, we are interested in determining if the variables written in the event log enable compliant process executions with respect to the defined set of guards. To this end, for each move on model from the set of possible model successors $S_{M_{next}}$ we assign variables in the state (P, Φ) before and after the corresponding transition λ with the process activity a_M at line 12. Note that in this way we obtain a move with fulfilled/violated guards according to Definition 6, depending on the values observed in the event log and the data guards. A move in model is then added to Γ_n (line 13). Then, if the next log successor has the same activity label as the next move successor, we update the variables assignment Φ' of the final state using the variable assignments D of the successor event $s_{L_{next}}$ (line 14–16). In this case, a move in both is also added to Γ_n (line 17). Once all the model successors have been explored, we finally add a move in log (line 18).

Algorithm 2: The *expand_alignment*$(\gamma, \mathcal{M}, \sigma_L)$ function

Input : The current partial alignment γ, a process model $\mathcal{M}$, a log trace σ_L
Output: A list of possible alignment successors Γ_n

1 Γ_n=[] ;
2 $\gamma_{ctr_\gamma} = control\_flow(\gamma)$;
3 $\sigma_{M_c} = \gamma_{ctr_\gamma}|_M$;
4 $\sigma_{L_c} = \gamma_{ctr_\gamma}|_{\sigma_L}$;
5 $s_{L_{next}} = log\_successor(\sigma_{L_c}, \sigma_L)$;
6 $s_{L_{prev}} = last\_event(\sigma_{L_c})$;
7 $abstract\_data(M)$;
8 $S_{M_{next}} = model\_successors(\sigma_{M_c}, M)$;
9 **foreach** $s_{M_{next}} \in S_{M_{next}}$ **do**
10 $\Phi = D_{s_{L_{prev}}}$;
11 $\Phi' = \Phi$;
12 $s_{M_{next}} = (\lambda, ((P, \Phi), a_M, (P', \Phi')))$;
13 $\Gamma_n.add(s_{M_{next}}, \gg)$; // move in model
14 **if** $a_L = a_M$ **then**
15 $\Phi' = D_{s_{L_{next}}}$;
16 $s_{M_{next}} = (\lambda, ((P, \Phi), a_M, (P', \Phi')))$;
17 $\Gamma_n.add(s_{L_{next}}, s_{M_{next}})$; // move in both
18 $\Gamma_n.add(\gg, s_{L_{next}})$; // move in log
19 **return** Γ_n

Below we present a toy example to illustrate the application of Algorithm 1.

Example 5. Let us consider the process model in Fig. 1 and the example trace $\sigma_{L_2} = \langle (a, \{v_1 = 10\}), (c) \rangle$. Fig. 3 shows part of the search space explored while looking for an optimal alignment. Each node n together with a move symbol corresponds to a (partial) alignment. We highlighted in gray the nodes chosen at each step (i.e., the ones corresponding to the minimum cost), and marked the order of the picked nodes with a "#" and the number. Every edge corresponds to a legal move. We report the costs values computed for the displayed state. For the sake of clarity, we indicate separately the data cost and the control-flow

components respectively with variables w and g, and h[6] stands for the estimated cost value in Fig. 3.

The first node corresponds to the empty alignment, with all the costs equal to 0. At this stage, the current event $s_{L_1} = (a, \{v_1 = 10\})$, and the possible model successor is only $S_{M_1} = \{a\}$. There are three possible legal moves[7] (and, hence, three successor nodes): $n_1(a, \gg)$, $n_2(a, a)$ and $n_3(\gg, a)$. For each of them, the variable v_1 is assigned to 10. The data cost is $w = 0$ for all of the successor nodes, since no data guards are defined on a. The node $n_2(a, a)$ is a synchronous move with no deviation on control flow, then $g(n_2) = \epsilon$ (ϵ is the negligible cost for a final termination, see the cost function in (5)) and $h(n_2) = 0$, thereby $f(n_2) = \epsilon$. For other two alignment nodes $n_1(a, \gg)$ and $n_3(\gg, a)$ with control flow deviation, $g(n_1) = g(n_3) = 1 + \epsilon$ and $h(n_1) = 1$ and $h(n_3) = 0$, respectively (correspondingly, $f(n_1) = 2 + \epsilon$ and $f(n_3) = 1 + \epsilon$). All three successors are added into the $\mathcal{OPEN}$ set. $n_2(a, a)$ is clearly the one corresponding to the minimum cost. Therefore, it becomes the current alignment, and then be moved to the $\mathcal{CLOSED}$ set. Then, the algorithm continues to search for the successors of node $n_2(a, a)$. The set of trace successors is $s_{L_2} = \{(c)\}$, and the set of transition successors is $S_{M_2} = \{b, c\}$. Therefore, the set of possible moves is $\{n_4(\gg, b),\ n_5(\gg, c),\ n_6(c, c),\ n_7(c, \gg)\}$. Note that no data were updated in the event log, and so, each move has the same variable assignment as the first move, i.e., $v_1 = 10$. For node $n_4(\gg, b)$, control-flow cost $g(n_4) = 1 + 2\epsilon$ and estimated cost $h(n_4) = 0$, as the variable v_1 is compliant to the guard $G(b) : v_1 < 30$, data cost $w(n_4) = 0$, so overall cost $f(n_4) = 1 + 2\epsilon$. However, for the firing of a to c, the guard $G(c) : v_1 >= 30$ is violated. Suppose we assume $v_1 = 10$ totally deviates the fuzzy tolerance margin, and so for both $n_5(\gg, c)$ and $n_6(c, c)$ the data cost $w = 1$. For node $n_5(\gg, c)$, $h(n_5) = 1$ and $g(n_5) = 1 + 2\epsilon$, costs are $f(n_5) = 3 + 2\epsilon$. For node $n_6(c, c)$, $h(n_6) = 0$ and $g(n_6) = 2\epsilon$ costs are $f(n_6) = 1 + 2\epsilon$. For the last node $n_7(c, \gg)$, there are no corresponding data constraints, being a move in log, so $w(n_7) = 0$, $h(n_7) = 1$, $g(n_7) = 1 + 2\epsilon$ and the overall cost $f(n_7) = 2 + 2\epsilon$. Then the four nodes are added into the $\mathcal{OPEN}$ set. For all the remaining nodes in the $\mathcal{OPEN}$ set, even though both $n_6(c, c)$ and $n_4(\gg, b)$ have the same minimum cost $f = 1 + 2\epsilon$, node $n_6(c, c)$ has firstly reached the end of both the trace and the process model. Therefore, $n_6(c, c)$ is moved from the $\mathcal{OPEN}$ set to the $\mathcal{CLOSED}$ set, and marked as "2#" selected node.

4. Experiments

This section describes the results obtained by a set of experiments we carried out on two real-life event logs to evaluate the usefulness of the proposed approach. In particular, we are interested in assessing the impact on the final diagnostics of (a) the introduction of an additional data move, and (b) the fuzzification of the data costs. More precisely, we expect that the introduction of the extended set allows us to pinpoint deviations that would go undetected with previous frameworks [19]; while the fuzzification of the cost function is expected to increase the fitness values of traces involving data deviations within the established tolerance values. Accordingly, we consider different metrics to evaluate the results; namely, we consider the number of detected violations to assess the impact of factor (a), and the fitness value for factor (b). In both cases, an increasing of the metric corresponds to an improvement of the diagnostic, i.e., an increased accuracy, intended as capability of detecting different kinds of deviations (a), or as the capability of providing a more fine-grained analysis tailored on domain experts' expectations (b).

For the sake of space, we only presented the results from cost function with one membership function and one aggregation function. We carried out the experiments with other membership functions, e.g., z-shaped, and s-shaped functions, and obtained the similar results as the trapezoidal function presented in Section 4; therefore, we chose to only present the results of this one, being the simplest and most commonly used. As regards the aggregation functions, here we presented only the results related to the t-norm, since it was the one that fit best according to experts from the Catharina case. Interested readers can refer to [18] for a more in-depth comparison of alternative aggregation functions.

4.1. Catharina Hospital cryo-ablation procedure

We conducted the study on the peri-operative process of cryo-ablation procedure in Heart and Vascular Center of Catharina Hospital Eindhoven, the largest cardiovascular center in the Netherlands. We derived the process model by analyzing local protocols, performing shadowing studies and conducting interviews with cardiology experts.

4.1.1. Experiment setting

Model description. Fig. 4 shows the process model describing the cryo-ablation procedure in BPMN language. Different colors are used to highlight different execution paths. The process starts when a patient is added into the waiting list of the cryo-ablation procedure (Waitfor_Schedule) and the procedure is scheduled to a specific date by the planner (Scheduled). On the procedure day, the patient arrives to the heart launch room and report their recent medication list (Admission). In particular, variables related to anticoagulants, i.e. type (NOAC: non-oral anticoagulants, or Vitamin − K), status (stop or continue), are recorded. Then the patient takes measurement on blood pressure (Measure_Measure), which results in an update of the variable BP (blood pressure). If the patient is taking vitamin-k (V-K) type anticoagulants, e.g. acenocoumarol, or marcoumar, he also needs to test the INR (international normalized ratio), a laboratory measurement of how long it takes for blood to form a clot (Test_Test). Accordingly, the variable INR is updated. Note that the execution of this activity depends on the data guard reported on the corresponding gateway. After these first measurement steps, the clinicians evaluates whether the patient's condition satisfies the requirements for starting the cryo-ablation procedure. This means checking whether the variables BP and INR fulfill the data guards on the gateway. If not, the procedure is canceled (Cancellation). Otherwise, the patient goes through a standard set of activities, from the preparation for the surgery (Prepare), to the transfer of the patient out of the Cathlab, where the surgery takes place (Leave_Leave). In the post-operative care, the anticoagulant medication should be restarted (Restart_Restart) if it was stopped before the procedure. Finally, the patient is discharged after several hours or after an overnight rest (Discharge).

Data preparation. We received an anonymous data-set containing all the in-hospital records of the cryo-ablation process from 2017 to 2019. We selected the completed cases in the procedure list and filtered the ones including less than four events, since they likely represent cases where some issues occurred in the logging. Events corresponding to the same activity and logged too close in time to each other (i.e., within a minute) are considered as logging errors, and removed. Similarly, events without timestamps or mandatory variables were removed. For related clinical

[6] For this example we use the heuristic function introduced in [5] to calculate h of all the nodes.

[7] For the sake of simplicity, here we use only the activity labels and the no-move symbols to refer to moves.

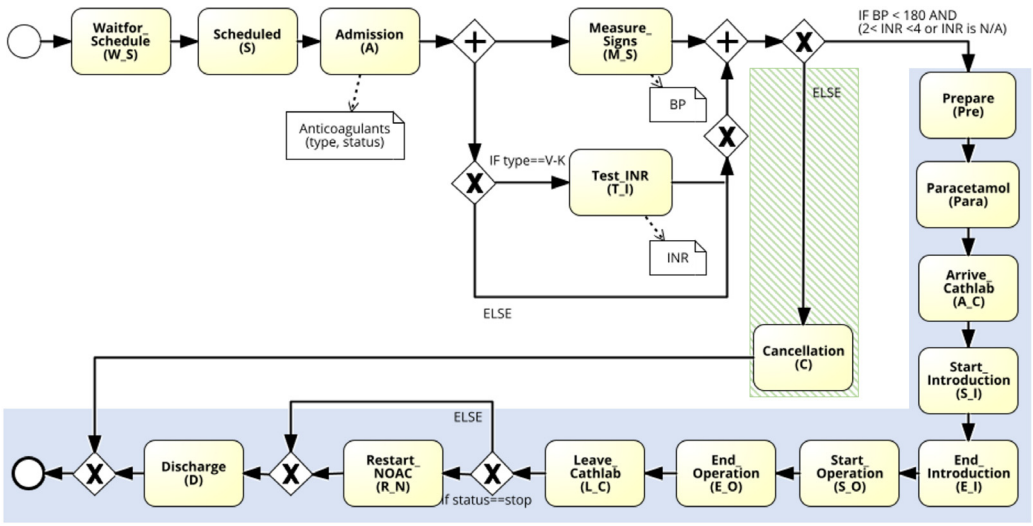

Fig. 4. The peri-operative process Model with different path choices.

Table 2
Basic statistics for two real-life event logs.

#Cases	#Traces	#Events	Avg. #Events	Min #Events	Max #Events
Cryo-ablation	1206	15068	11	4	32
Sepsis	951	7666	8	4	14

indicators like anticoagulants, since the peri-operative medication list should be collected before the procedure, we filtered the medication records which ended 30 days before the procedure, or started after the patient discharge. Table 2 shows some basic statistics for the two event log of the case studies, i.e., the total number of traces/events and the average, max and min number of events per trace. The cryo-ablation log involves 1206 traces and 15,068 events in total. On average, a trace consists of 11 events. The shortest traces involve 4 events, and usually correspond to patients for which the procedure was canceled. We also observed several traces longer than expected, with a maximum of 32 events. Digging into the log, we found out that some process activities have been executed multiple times for some patients. These often correspond to administrative activities (e.g., the admission or the canceling of the procedure). Discussing these findings with the process experts, we found out that when a procedure had to be rescheduled for a patient, this is logged in the system as repeated administrative activities for that patient, rather than opening a new case. We also observed that for some patients the administration of analgesic drugs (paracetamol) was performed multiple times, possibly depending on the patients' requests. Among the 1206 traces, 1167 have records on arriving and leaving Cathlab, which can be considered to indicate the proper completion of a cryo-ablation procedure. For the data perspective, we consider the four variables shown in the process model. For anticoagulant information, 878 traces have the medication records during patient admission, which means variables *type* and *status* (for anticoagulants) are missing in 289 traces. Therefore, the related rules to these variables are also not applicable for these cases. For other two value-type variables, blood pressure BP is missing in 97 cases, and only one trace missed the necessary INR information. Since these variables are critical for the verification of the ablation procedure, we discussed with clinical experts and agreed to replace the missing value with the median of the existing variables.

The cost function setting. Dichotomous functions are used to obtain binary memberships for string variables, i.e., *type* and *status*, to the related rule compliance. For numeric variables BP and INR, we defined membership functions following Definition Definition 9 and discussing with the clinical experts. For the blood pressure, clinicians indicated that deviations up to 200 are still considered acceptable, and they would consider these values as less compliant to the protocol the closer to 200 they are. For the INR, we observed that all the values are distributed within the range (1,3.9), so that the higher boundary of the data guard is never violated. Furthermore, clinicians considered the possibility of violating this high boundary very unlikely. Therefore, we decided to implement a simplified membership function, stating that INR should be higher than 2. Also in this case, clinicians are willing to accept lower values which are reasonably close to 2, and 1 was indicated as the lowest acceptable value. Therefore, for the variables we derived the following fuzzy boundaries: (180, 200) for BP, and (1,2) for INR. To model the intuition of a linearly decreasing level of compliance within the fuzzy boundaries, we chose to use the trapezoidal membership function. In particular, we chose a right-shouldered trapezoid for the membership function of the BP variable, and a left-shouldered trapezoid for the value of INR, since we want to model that above/below the highest/lowest boundary, the compliance degree drops to zero. The membership functions are specified below.

$$\mu_{\mathsf{BP}} = \begin{cases} 1 & , \text{if } \mathsf{BP} \leq 180 \\ 0 & , \text{if } \mathsf{BP} \geq 200 \\ \frac{200-\mathsf{BP}}{20} & , \text{if } 180 < \mathsf{BP} < 200 \end{cases}$$

$$\mu_{\mathsf{INR}} = \begin{cases} 1 & , \text{if } \mathsf{INR} \geq 2 \\ 0 & , \text{if } \mathsf{INR} \leq 1 \\ \mathsf{INR} - 1 & , \text{if } 1 < \mathsf{INR} < 2 \end{cases}$$

Regarding the choice of an aggregation function, we choose the Product $\mathsf{t-norm}$ operator, i.e., we determine the overall compliance level as the product of the compliance level of each single variable.

The overall cost function is thus formulated as follows, with n standing for the number of involved variables on model firing s_M,

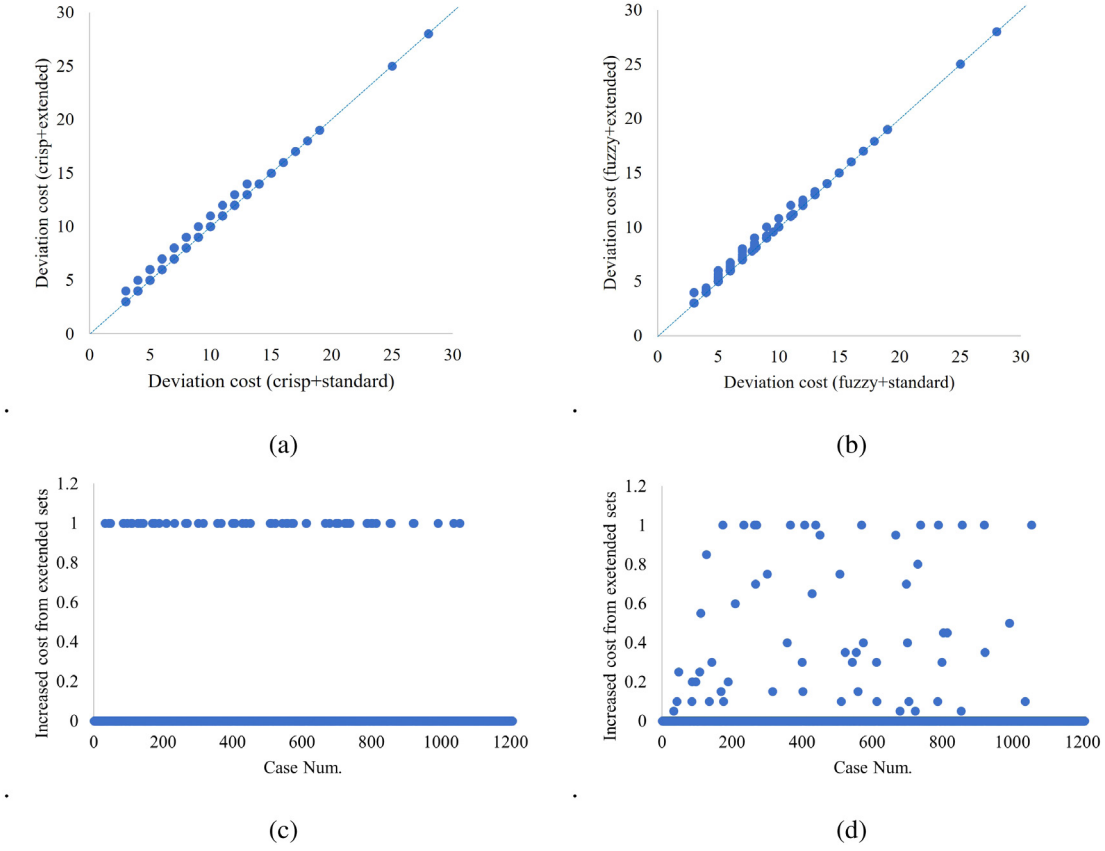

Fig. 5. Comparisons of the deviation costs obtained by: the crisp method with standard/extended moves set (a); the fuzzy method with standard/extended moves set (b). Variance of the deviation costs for additional deviations detected by the crisp method (c) and the fuzzy method (d).

and μ_i for the membership of each variable:

$$k(s_L, s_M) = \begin{cases} 1 & \text{if move in log} \\ 1 & \text{if move in model without fulfilled guards} \\ 2 - \prod_{i=1}^{n} \mu_i & \text{if move in model with deviated guards} \\ 1 - \prod_{i=1}^{n} \mu_i & \text{if move in both with deviated guards} \\ 0 & \text{otherwise.} \end{cases} \quad (6)$$

As discussed in Section 2, the conformance checking algorithm always returns one optimal alignment. It should be noted that when multiple alignments with the same cost exist, our algorithm returns the alignment with the higher amount of data deviations. This is a design choice, decided together with the domain experts, which considers explanations based on data guards violation closer to reality than explanations based on control-flow deviations. However, different settings can be chosen.

Competitor cost functions. We compare the diagnostics obtained by applying our cost function with (a) the cost function with fuzzy sets with standard moves introduced by Zhang et al. [19] and (b) two data-aware cost functions with standard and extended moves respectively, both using a crisp cost assessment, as done by state-of-the-art techniques [7]. More precisely, to assess the impact of the introduction of the extend move set we compare the deviation costs obtained by the crisp (fuzzy) cost functions with the standard and the extended move set. To assess the impact of the use of fuzzy sets, we compare the fitness results obtained by the crisp cost function and the fuzzy cost function, both with the standard and the extended move set. Finally, we discuss differences in the alignment detected in these comparison groups. We implemented all cost functions within our framework, described by Algorithm 1, to ensure that differences in diagnostics are due only to differences in cost function definition. In order to have a meaningful comparison we need to ensure that the difference in terms of fitness is due to difference in computing the alignment cost. To this end, we set the data guards always equal to *True* when computing the reference alignment γ_k (see Definition 10). This ensures that the reference alignment used to compute the fitness is the same for all the approaches. We also validate the obtained results with domain experts from Catharina hospital.

4.1.2. Results

Deviation cost comparison. Fig. 5 shows the deviation costs computed for each trace by the tested approaches. In Figs. 5(a) and 5(b), the *x*-axis represents the cost returned by the crisp (fuzzy) approach with standard moves, while the *y*-axis shows the results obtained by the crisp (fuzzy) approach with extended moves. For all traces on the main diagonal, the deviation cost remains unchanged. Dots above (below) the diagonal represent cases in which the approach on the *y*-axis obtained a cost value higher (lower) than the approach on the *x*-axis. As expected, the use of the extended approach always results in the same or higher costs than the one with standard sets, both for the crisp and the fuzzy case. In particular, we found 60 cases involving deviations detected only by using the extended move set. Note that the differences are higher for the crisp case; this is expected, since with a crisp cost function each additional detected deviation will be assigned a cost of one, while in the fuzzy case it will be assigned to a cost between 0 and 1. This behavior is evident in

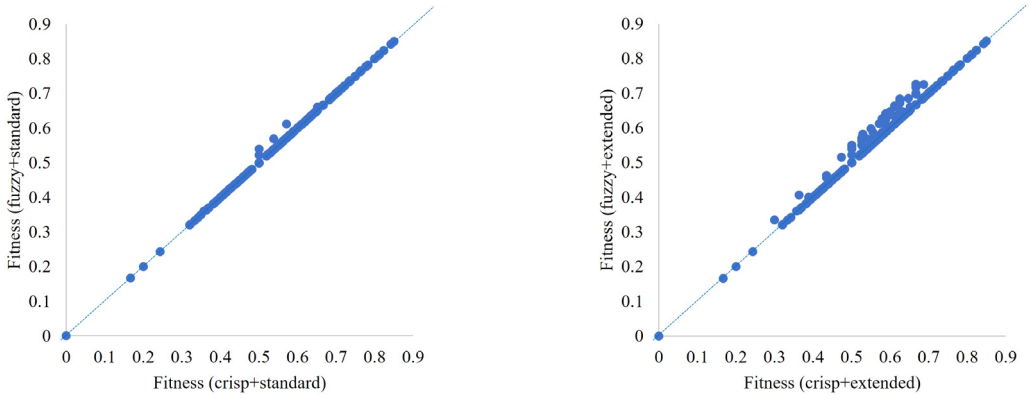

Fig. 6. Comparison of the fitness levels returned by (a) the crisp method and the fuzzy method with standard moves; (b) a crisp approach with the extended moves set and a fuzzy approach with the extended moves set.

Figs. 5(c) and 5(d), which show the cost variance for each of the 60 cases for the crisp and fuzzy cost function, respectively. Note that the average increased cost using the fuzzy extended move set is 0.487.

Fitness result comparison. In Fig. 6(a), the *x*-axis (y-axis) represents the fitness level of the crisp (fuzzy) approach with standard moves; similarly,in Fig. 6(b) shows the fitness level of the crisp/fuzzy cost function with extended moves. For all traces on the main diagonal, the fitness level remains unchanged. Dots above (below) the diagonal represent cases in which the fuzzy approach obtained a fitness level higher (lower) than the benchmark. We see that the fuzzy approach obtains always the same or higher fitness levels than the crisp cases. In particular, when the standard move set is applied, 6 traces are higher than the main diagonal, with an average increase of 4.62% in fitness compared to the crisp approach. This number increases to 53 traces with higher fitness for the fuzzy approach when the extended move set is applied, with an average increase of 6.33% in fitness compared to the crisp approach. This difference is due to the use of the extended move set, which enables the detection of data violations neglected in the standard move set.

To provide some insights on the general level of compliance of the event log, we also report in Fig. 7 the fitness level distribution obtained by the proposed approach. The *x* axis shows nine bins of fitness values, each corresponding to a range of 0.1. The number above each bin corresponds to the number of traces with fitness level falling in this bin. The log is characterized by a relatively high level of fitness. Indeed, 877 traces (72.7% of the entire log) show a level of compliance higher than 0.6, with a peak in the interval (0.6, 07]. The overall average fitness level is 0.616. Nevertheless, there is a significant portion of non-compliant traces, some of which with a level of fitness below 0.5 (86 traces, i.e., 12% of the log). Only a few dozen cases deviate strongly or, instead, comply almost perfectly. In particular, only 4 out of 1206 traces have the fitness level lower than 0.3; while only 43 traces have a fitness level higher than 0.8 Note that this implies that even the most compliant executions still have slight violations with respect to the protocols.

The alignment interpretation. In addition to the detected deviations and overall fitness levels, there are also differences in the alignments obtained (and, hence, in the interpretation of the diagnostics). These differences are mostly visible on the activities subjected to data-guards, i.e., whether to cancel or continue the procedure, which relies on the value of BP and INR respectively. Fig. 4 shows the data-aware paths of the alignment model, where one is activity Cancellation in green striated block, and the other is represented by activity Prepare in pure blue block, respectively.

Table 3
Number of cases that are aligned to different activities: *Prepare* and *Cancellation*.

	Prepare	*Cancellation*
(crisp, standard)	1139	67
(crisp, extended)	1128	78

We analyzed alignment differences grouping the approaches pairwise. Namely, we first compare the (crisp, standard) approach against the (crisp, extended) one, to illustrate the effect of the extended move set; then, we compare the (crisp, extended) approach against the (fuzzy, extended) one to highlight differences due to the introduction of the fuzzy sets.

Table 3 reports the number of traces representing the patients aligned to the preparation path or to the cancellation path with the (crisp, standard) and (crisp, extended) approaches.

According to Fig. 5(c), there are 60 cases were additional data deviations are detected thanks to the use of extended sets. The results in Table 3 show that the detection of these deviations has a strong impact on the alignment for 11 cases, which are assigned to the *Cancellation* path rather than to the *Prepare*. As an example, we discuss one of these cases in the following.

Example 6. Let us consider the *trace* $\sigma_{id:682} = \langle \ldots, (M\_S, \mathsf{BP} = 201), (T\_I, \mathsf{INR} = 3.8), (A, \{type = V - K, status = stop\}), (A\_C, \perp), (S\_I, \perp), (Para, \perp), (S\_O, \perp), (L\_C, \perp), (D, \perp) \rangle$.[8]

Table 4 presents the alignments determined by the (crisp, standard) approach (Table 4.a) and by the (crisp, extended) one (Table 4.b) . The alignment (a) matched the trace to the path of *Prepare* in the process model (in Fig. 4) with seven control-flow deviations. This alignment hence suggests that the patient has been rightfully undergone for the ablation procedure, but some activities have not been logged properly. However, it should be noted that this alignment does not take into account that the value of the BP = 201 violates the rule of *Prepare* path, thus generating a misleading diagnostics. Instead, the alignment (b) did take the data cost into account, which led to consider the *Cancellation* path as the best fit. The reason for the difference arises from the data violations. According to the clinical rule, BP is fully violated. The standard move set ignored the data perspective

[8] For the sake of simplicity, we report only the portion of trace interested by the data-guard.

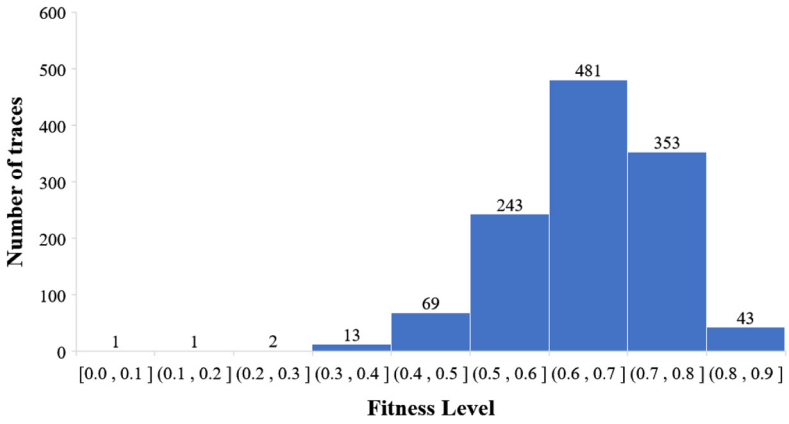

Fig. 7. Distribution of the fitness levels returned by the proposed approach.

Table 4
Optimal alignment returned by the classical crisp approach with standard (a) and extended (b) sets.

(a) Standard sets

Log	Model	*cost*
...	...	...
M_S (BP=201)	*M_S*	0
T_I (INR= 3.8)	*T_I*	0
A	≫	1
≫	*Pre*	1
≫	*Para*	1
A_C	*A_C*	0
S_I	*S_I*	0
Para	≫	1
≫	*E_I*	1
S_O	*S_O*	0
≫	*E_O*	1
L_C	*L_C*	0
≫	*R_N*	1
D	*D*	0

(b) Extended sets

Log	Model	*cost*
...	...	...
M_S (BP=201)	*M_S*	0
T_I (INR= 3.8)	*T_I*	0
A	≫	1
A_C	≫	1
S_I	≫	1
Para	≫	1
S_O	≫	1
L_C	≫	1
D	≫	1
≫	*C*	1

when control-flow is not aligned, so the overall cost is only 1 when *Prepare* is missing. Therefore, the overall cost for alignment (a) with the path starting from *Prepare* is seven. However, with the extended sets where data violation is always considered, the cost of choosing *Prepare* path will be eight (7 control-flow and 1 data cost). At the same time, the overall cost of choosing *Cancellation* path is also eight. The (crisp, extended) approach returns hence two possible optimal alignments for this trace; however, the conformance checking algorithm always picks only one optimal result and assign higher priority to data violation, thus returning the alignment (b). Note that both the optimal alignments returned by the (crisp, extended) approach suggest the same interpretation, i.e., that the procedure should have not been performed because of the violation of the clinical rule. This is more in line with the clinical protocols, completely ignored in alignment a).

We conducted a second comparison on the alignments with the preparation path or to the cancellation path with the (crisp, extended) and (fuzzy, extended) approaches in Table 5.

According to Fig. 6(b), 53 cases return higher fitness after the introduction of fuzzy sets from an general overview. Table 5 shows that these differences impacted the alignment of ten cases. We also pick here a representative trace to inspect the different alignments obtained by both approaches.

Example 7. Let us consider the *trace* $\sigma_{id:717} = \langle \ldots, (A, \{type = vitamin - K, status = stop\}), (T\_I, \mathsf{INR} = 1.9), (M\_S, \mathsf{BP} = 111), (A\_C, \bot), (S\_O, \bot), (E\_O, \bot), (L\_C, \bot), (D, \bot) \rangle$.

Table 5
Number of cases that are aligned to different activities: *Prepare* and *Cancellation*.

	Prepare	*Cancellation*
(crisp, extended)	1128	78
(fuzzy, extended)	1138	68

Table 6
Optimal alignment returned by the cost functions with crisp (a) and fuzzy (b).

(a) Crisp sets

Log	Model	*k*
...	...	...
T_I (INR= 1.9)	*T_I*	0
M_S (BP= 111)	*M_S*	0
A_C	≫	1
S_O	≫	1
E_O	≫	1
L_C	≫	1
D	≫	1
≫	*C*	1

(b) Fuzzy sets

Log	Model	*k*
...	...	...
T_I (INR= 1.9)	*T_I*	0
M_S (BP= 111)	*M_S*	0
≫	*Pre*	1.1
≫	*Para*	1
A_C	*A_C*	0
≫	*S_I*	1
≫	*E_I*	1
S_O	*S_O*	0
E_O	*E_O*	0
L_C	*L_C*	0
≫	*R_N*	1
D	*D*	0

Table 6 presents the alignments returned by the (crisp, extended) and by the (fuzzy, extended) approach (Table 6.a and Table 6.b, respectively). The alignment (a) shows that (crisp, extended) approach mapped the trace to the path of *Cancellation*, assigning all the other activities to moves on log. Instead, in alignment (b) *Prepare* path was picked, where an higher number of activities was matched. In this case, the alignment difference arises from the cost functions. According to the membership functions, $\mu_{\mathsf{BP}} = 1$ and $\mu_{\mathsf{INR}} = 0.9$. Accordingly, with data aware cost function, the product-norm aggregated data cost of activity Prepare is 0.1, i.e., the patient data only slightly violates the clinical protocol. We argue that such a fine-grained diagnostic provides the domain experts with more knowledge than the crisp one. Indeed, the latter does not consider the magnitude of the deviation; therefore, the alignment corresponding to the cancellation of the procedure and to the execution of the procedure with a violation of the constraints are considered as equally valid (same cost). In contrast, the fuzzy approach does take the magnitude into account, thus highlighting that the execution of the procedure is the more likely explanation, since the data deviation stays within the tolerance boundaries.

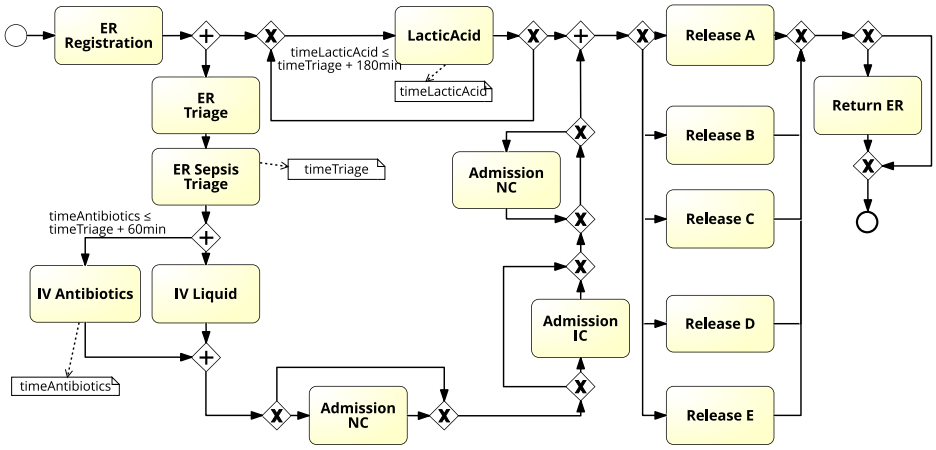

Fig. 8. The sepsis care process model.

4.2. Sepsis care procedure

Our second case study has been carried on a public dataset[9] regarding the care process of sepsis, a life-threatening condition that requires immediate treatment. The scope of this case study was on the patients admitted to the emergency ward of the hospital because suspected cases of sepsis.

4.2.1. Experiment setting

Model description. We refer to the process model obtained by previous studies [14]. However, we had to simplify such model because of limitations of the tools we are currently using for the analysis. Indeed, our tool exploits the process model representation implemented by [35] which, however, results in high computational complexity in the presence of high degree of parallelism and loops, up to the point of making the computation infeasible. To address these issues, we kept only two parallel branches out of four of the original model, also removing from the event log those activities not represented by the simplified process. We would like to point out that this change does not affect the results of our analysis significantly. Indeed, we kept those activities interested by the presence of a data guard; this allows us to analyze and compare the results obtained by the different cost functions we are testing, which is the main goal of these experiments. We plan to develop a more efficient implementation of our approach in future work. Fig. 8 shows the resulting model used in this analysis. The process starts with the *registration* (*ER*) of the patient in the emergency ward. Then, different concurrent activities take place, in particular the filling of general and specific sepsis triage document (*ERTriage*, *ERSepsisTriage*), the administering of antibiotics and of liquid via intravenous (*IVAntibiotics*, *IVLiquid*), the measurement of the lactic acid level (*LacticAcid*), the admission of the patient to a normal or to an intensive care ward (*AdmissionNC*, *AdmissionIC*). Then, the patient will be discharged, with a different code (*ReleaseA*, *B*, *C*, *D*, *E*). In some cases, however, the patient will have to return to the emergency ward (*ReturnER*). There are only two data guard expressions, encoding two rules on the time perspective: from the moment of triage, (1) patients should be given antibiotics within 60 min; (2) the lactic acid measurement should be done within 180 min. The variable *timeTriage* records the time when activity *ER Sepsis Triage* was executed, while variable *timeAntibiotics* and *timeLacticAcid* record the execution time of activity *IV Antibiotics* and *LacticAcid* respectively.

Data preparation. The event-log covers the traces of 951 patients, collected in 1.5 years. General log statistics are reported in Table 2, Since the time perspective is tightly related to the control-flow executions, here we assume that when the activity, e.g. *LacticAcid*, is missing, then the variable execution time, e.g. *timeLacticAcid*, is also considered as a missing deviation. We filtered out the traces with less than three event, which we consider as outliers.

The cost function setting. We define variables $\delta t_A = timeAntibiotics - timeTriage$, and $\delta t_L = timeLacticAcid - timeTriage$, and then the crisp data guards for the variables are (1) $\delta t_A \leq 60$, and (2) $\delta t_L \leq 180$. Since the rules for the time variables are independent on different activities, aggregation is not applicable in the cost functions for this case. Since this is a public dataset, we do not have expert's knowledge available to define the cost function; therefore, we carried out some basic data exploration. More precisely, we used the boxplot distribution of the violated variables, and picked the two-quarter value as the acceptable boundary. The reason underlying this choice is that if one observes a frequently occurring violation of a guard, it is reasonable that such a violation is considered acceptable by the process actors. According to this observation, we derived the fuzzy boundaries (60, 156.96) for δt_A and (180, 350.58) for δt_L. Here we also applied the commonly used right-shouldered trapezoid for both variables. The membership functions are defined as below.

$$\mu_{t_A} = \begin{cases} 1 & , \text{if } \delta t_A \leq 60 \\ 0 & , \text{if } \delta t_A \geq 156.9 \\ \frac{156.96 - \delta t_A}{96.96} & , \text{if } 60 < \delta t_A < 156.96 \end{cases}$$

$$\mu_{t_L} = \begin{cases} 1 & , \text{if } \delta t_L \leq 180 \\ 0 & , \text{if } \delta t_L \geq 350.58 \\ \frac{350.58 - \delta t_L}{170.58} & , \text{if } 180 < \delta t_L < 350.58 \end{cases}$$

Note that the time perspective used to define the guards in this model is highly connected to the control-flow. Namely, if the required activity is not executed, the time constraints will be violated as well, so the cost for the violation of both control-flow and time perspective would be two. Therefore, the overall cost function with time perspective $k_t(S_L, S_M)$ can be defined as

[9] The event log can be obtained from: https://doi.org/10.4121/uuid:915d2bfb-7e84-49ad-a286-dc35f063a460.

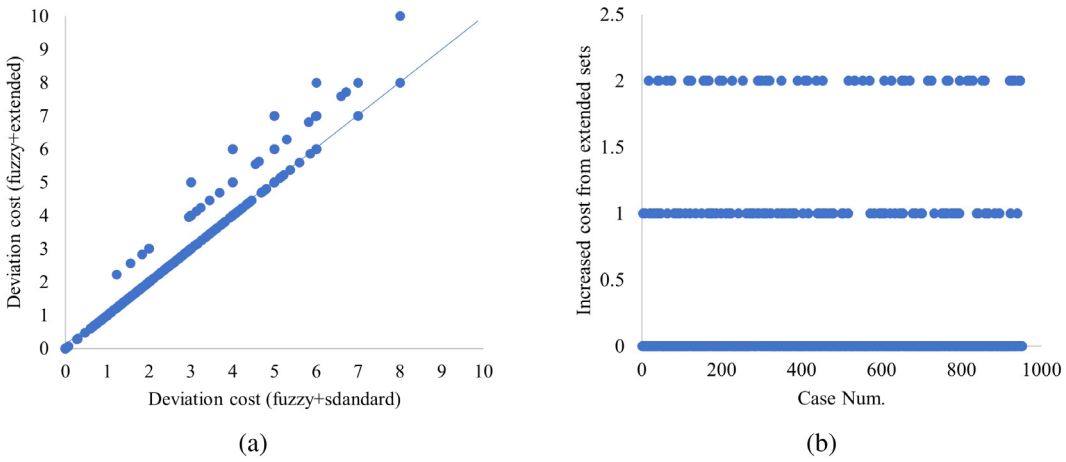

Fig. 9. Deviation costs returned by the fuzzy method with/without the extended moves set (a); Variance of the deviation costs returned by the fuzzy method with/without the extended moves set (b).

follows, with μ_i corresponding to the membership function μ_{t_A} (for activity IV Antibiotics) or μ_{t_L} (for activity LacticAcid).

$$k_t(S_L, S_M) = \begin{cases} 1 & \text{if move in log} \\ 1 & \text{if move in model with fulfilled guards} \\ 2 & \text{if move in model with violated guards} \\ 1-\mu_i & \text{if move in both with violated guards} \\ 0 & \text{otherwise.} \end{cases} \tag{7}$$

Evaluation metrics. We compared the same approaches described in the cryo-ablation case, using the same metrics. However, in these dataset there are no data guards determining different execution paths; therefore, we will not delve into the alignment comparison, since the picked execution paths are expected to be the same with respect to the tested cost functions.

4.2.2. Results

Deviation cost comparison. Fig. 9 shows the deviation costs computed for each case in the log. For the sake of simplicity, here we only focus on the fuzzy method with and without the extended set (y-axis and *x*-axis in Fig. 9(a)), since similar results are obtained for the crisp functions. The deviation cost remains unchanged for all traces on the main diagonal, and the dots above the diagonal represent traces in which the approach with extended sets obtained a cost value higher than the standard sets. We found 162 traces involving deviations detected only by using the extended move set. To delve into these differences, we report the cost variance for these traces in Fig. 9(b) Note that the variance levels are always either 1 or 2. These crisp values are due to the characteristic of the time perspective, which is highly affected by the control-flow executions. In particular, the cost function always returns a fully violation (i.e., a cost equal to 2) when the activity is not executed, i.e. for move in model with violated time guards. When compared to the deviation costs obtained with the standard and the extended move set, we can obtain a cost difference of either one, if the move-on-model is detected by the standard move set but the data violation is not, or two, when the move-on-model is skipped altogether. The deviation variances by extended sets which always returned a crisp cost when control flow is violated and marked as a "move in model".

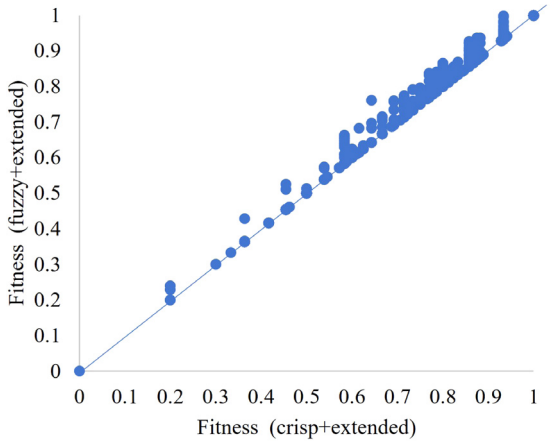

Fig. 10. Comparison of the fitness levels returned by fuzzy/crisp method with the extended sets.

Fitness result comparison. In Fig. 10, the *x*-axis (y-axis) represents the fitness level of the crisp (fuzzy) approach with extended moves of each trace. We see that the fuzzy approach always obtained the same or higher fitness level than the crisp one, and 240 cases returned higher fitness from the introduction of fuzzy boundaries, with an average increase of 4.18%. Also in this case, here we will not present the comparison on crisp/fuzzy approach with standard moves, since it returned the same results on increased fitness, due to the characteristics of time perspective.

As concerns the general level of compliance of the whole event log, we report in Fig. 11 the fitness level distribution obtained by our approach. The *x* axis shows ten bins in which we grouped the fitness values from 0 to 1, and the *y* axis shows the number of traces dropped in each fitness bin. The number above each bin corresponds to the number of cases with fitness level falling upon this bin. The figure shows that the log has a high level of overall average fitness, 0.78, and 82.8% of traces has reached to a fitness higher than 0.6. In particular, only 14 out of 951 traces have the fitness level lower than 0.4, implying that most of the cases clinical activities are compliant to the local protocol.

4.3. Discussion

The experiments show that our fuzzy multi-perspective conformance checking method is capable of providing more accurate

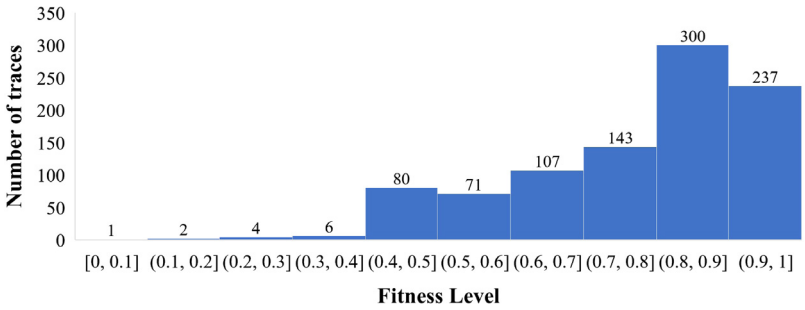

Fig. 11. Distribution of the fitness levels returned by fuzzy multi-perspective conformance checking method with the extended sets.

and fine-grained diagnostics than the tested competitors. The introduction of extended sets allows to highlight data deviations independent to control-flow perspective, where standard sets only count the data violations when control flow is aligned (Figs. 5 and 9). Furthermore, the approach consistently obtained higher level of fitness than the crisp one, due to the fact that possible tolerance to deviations from domain experts are taken into account (Figs. 6 and 10). It should be emphasized that fitness comparisons are not feasible for the extended/standard sets, since they detect different set of deviations. Only when the detected deviations are the same, a higher level of fitness corresponds to more accurate diagnostics.

With regards to the alignment results, for the Catharina case we observe that both the extended sets and fuzzy sets had quite a strong impact on the interpretation of some patient's trace. Example 6 shows that the introduction of extended sets detect data violations which are otherwise neglected when building the alignment. Example 7 shows that the use of fuzzy sets is capable of modeling tolerance on data violations, allowing the analyst to recognize that for same patients the most likely explanation was that the procedure continued, since the data violations were within the boundaries. To better understand the consequences of these differences, we also interviewed the cardiology experts and discussed with them the alignments with the strongest differences. From these interviews, it emerged that the alignment computed by the our method provides a more informative and meaningful diagnostic, closer to what would be the human interpretation of the occurred deviations. It is noteworthy that, while the alignments were not affected by the use of fuzzy sets in the cost function for the sepsis case, the approach allowed anyway to highlight different degrees of guards violations in different process executions. We argue that such an information is very valuable in diagnostic terms, for instance because it allows the analyst to focus on cases with strong violations which are arguably the ones where more attention is needed to determine what caused the violations.

Similarly to any conformance checking approach, our framework is applicable for any process which satisfies three requirements: (1) there is a normative process model represented in a formal notation, e.g., BMPN, or Petri-net; (2) the model involves constraints defined on multiple perspectives; (3) there are data tracking historic executions available. Previous work has shown that conformance checking techniques proved to be valuable in a number of domains, for instance the healthcare [1], the financial domain [7], the web service selection [36] or application management in public administration [6].

Nevertheless, there are some limitations to this work. First, due to the lack of usable public datasets for multi-perspective conformance checking, we considered only two case studies for our experiments. Future research should extend the experimental set, to better investigate consequences of fuzzification of the cost function for process models involving, e.g., more complex control-flow construct, or more elaborated data-guards. Besides, the validity and the usefulness of the compliance results are strongly dependent on the correct design of the membership function; while some guidelines can be provided and some data exploration techniques can be used to aid the decisions, this is a problem specific step and there is not a one-fit-all approach that can be effortless brought from one case to another. A good collaboration with the domain experts is essential in order to derive a meaningful cost function. Finally, as we mentioned in the previous section, the tool we currently use for the implementation of the approach suffers from some computational complexity issues. A more efficient implementation can be designed to overcome these issues.

Table 7
Overview on related work.

	Multiple perspectives	Uncertainty	Optimal alignment
[4,37,38]			X
[39–41]	X		
[1,13,14]	X		X
[42,43]		X	
[44]		X	X

5. Related work

Table 7 provides an overview on related work on conformance checking, highlighting three key aspects of the approaches, namely (a) whether *additional perspectives* than the control-flow are taken into account, (b) whether the approach explicitly represents *uncertainty* related to the process execution and/or its modeled behavior, and (c) whether the approach outputs an *optimal alignment* that can be used to interpret the observed deviations. Note that this does not intend to be an exhaustive review; we mention frequently-cited methods in literature for each category.

Some approaches [39,40] propose to check whether log traces satisfy a set of compliance rules, typically represented by using declarative modeling, without building an alignment between the log traces and the model. Rozinat and van der Aalst [28] propose a token-based technique to replay event traces over a process model to detect deviations. Since it has been shown that token-based techniques can provide misleading diagnostics [45], to overcome these limitations alignments have been proposed as a robust approach to conformance checking [4]. Alignments are able to pinpoint deviations causing nonconformity based on a given cost function. While most of alignment-based approaches use the standard distance cost function [4], some variants have been proposed to enhance the quality of the provided diagnostics, for instance by analyzing historical logging data to take probability into account [37]. Other approaches investigated decomposition strategies to enhance the computational efficiency of the alignment building step [38].

Besides the control flow, there are also other perspectives like data, or resources, that are often crucial for compliance checking analysis. Few approaches in literature have investigated how to include these perspectives in the analysis. Some approaches proposed to compute the control-flow first and then to assess process executions compliance with respect to the data perspective [1,13]. These methods assume that the control flow is more important than other perspectives for an optimal alignment, with the result that some important deviations can be missed. On the contrary, [41] assumes to align the data perspective to obtain a data-aware reference trace, and then replay it with the input trace to detect the mismatch events by classical control flow conformance. Authors use control-flow and data-dependencies to derive a graph representation of the log traces, then computing for each of them a graph-based similarity metric with the most similar execution that can be derived by the process model. The research of Mannhardt et all [7] represents the approach closest to our framework. They introduce a cost function able to account and balance all process perspectives together when building the alignment. However, there are two main differences between their framework and ours. First, they adopt a crisp evaluation of deviations. Second, the interpretation given in their framework to violations of data guards consists in a wrong data writing operation carried out by one of the process activities. Our approach, instead, assumes that data have been properly stored, but the actors are willing to accept violations. This difference in the semantic of the data violation has a direct impact on how the alignments are built. In the approach of Mannhardt et all [7] the goal consists in determining a valid variable assignment which is as close as possible to the values observed in the log trace. To this end, the authors employ MILP for variable assignments to perform a *data alignment*, i.e., to obtain the process trace most similar to the log trace. In our approach, instead, the goal is to build alignments *with data* and highlight the guards that have been violated. Therefore, variables in the process trace are assigned to the same values observed in the log. Our approach hence adopts a more conservative position than the data alignment one. In data alignment approach, all data costs are explained within a single *incorrect write* operation. After that, an alternative, valid value is assigned to the variable to obtain a process trace compliant with the constraints. In our approach, instead, variables are assigned as it is shown in the log, even if this means that no process traces valid according to the data perspective can be generated. Guards violations are then counted when computing the overall cost; and so, variables that are assessed multiple times will weight more to the definition of the final cost. Note that within this framework, the alignments we build are still optimal. However, instead of representing the closest valid execution, they represent the most likely explanation with respect the given variable assignments. We argue that both approaches, data alignment and alignment with data, represent valid alternatives. Our approach is tailored to contexts in which we assume data are mostly logged properly and actors have the freedom to deviate from the guards. In these cases, the analyst is mostly interested in understanding which guards have been violated, and to which extent.

The approaches discussed so far did not explicitly deal with possible uncertainty related to the event log or to the process model. There are few recent studies about the topic. Leemans et al. [44] propose an approach tailored to stochastic Petri net, to take into account the probability of different execution paths when assessing the conformance values. Van der Aa et al. [42] propose a framework to deal with uncertain event-activity mappings. Instead of returning alignments, they return the probability that a given trace is compliant with respect to a set of possible mappings. Pegoraro et al. [43] propose a framework dealing with intervals, rather than values, for event names and timestamps, computing the lower bound and the upper bound alignment cost with respect to the defined intervals. While the framework can in principle be extended to deal with other event attributes, the extension of their conformance checking method to multiple perspectives is not discussed.

We would like to conclude this overview by mentioning some relevant studies that have proven that fuzzy sets can be successfully employed to represent humans' decision making processes. Bosma et al. [46] studied a fuzzy approach to modeling Vietnam farmers' decision process in adopting integrated farming systems; Hao et al. [47] applied a fuzzy dynamic weight determination approach to solve the risk decision making problems for the mine emergency; Xue et al. [48] modeled a fuzzy decision tree to realize the human-like decision-making knowledge in inbound ship analysis. The researchers in [49] created a fuzzy decision-making framework for treatment selection with qualitative flexible multiple criteria, and similarly [50] introduced on health care applications a fuzzy linguistic method on the Multiple Criteria Decision Making (MCDM) problem of Prioritizing the elective surgery patient admission in a Chinese public hospital. Some approaches also investigated the application of fuzzy sets for process mining analysis. For example, the study in [51] exploits an existing (fuzzy) rule-based framework to characterize the conformance problem. In [52] the researchers proposed to use fuzzy analytic hierarchy process to manage vagueness in many linguistic judgement. The study in [53] focuses on a fuzzy process miner on a CT scan data-set to help the hospital administrators in maximizing the throughput and optimal resource utilization, thus reducing patient waiting time. However, to the best of our knowledge, no previous work has investigated the use of fuzzy sets to integrate the deviation degree in multi-perspective conformance checking.

6. Conclusions and future work

In this work, we propose a novel compliance checking approach that allows to assess the compliance of process executions taking into account the degree of deviations, to obtain diagnostics that are more accurate and possibly closer to human interpretation. The proposed approach enhances significantly the flexibility of compliance checking, allowing the human analyst to customize the data cost function for multiple variables according to the analysis needs. To achieve these goals, we define a novel deviation cost function, by introducing a new move to account for data-guards violations for skipped activities and by exploiting fuzzy sets theory to compute the data costs.

We implemented the approach and tested it on two real-life clinical processes, comparing the results obtained to the cost functions defined by state-of-the-art techniques in terms of the deviation cost, the overall trace fitness, and the returned alignment. The introduction of extended sets allows to highlight data deviations independent to control-flow perspective, where standard sets only count the data violations when control flow is aligned. Meanwhile the cost returned from the fuzzy approach also remains more accurate diagnostics on the deviation degrees. When assessing the fitness level, the fuzzy approach consistently obtained higher level of fitness than the crisp one, due to the possible tolerance to deviations from domain experts. With regards to the alignment results, the results from the proposed approach provides a more informative and meaningful diagnostic, closer to what would be the human interpretation of the occurred deviations. as the domain experts recognized.

For future work, there are several directions that we plan to explore. First, as we discussed in Section 4.3, we plan to extend the experimental set, in order to test the approach with different datasets and in different contexts. Furthermore, we plan to

investigate the use of fuzzy sets also for the control-flow aspects as well. For instance, analyzing the results of the real-life experiment, we noticed several control-flow deviations where one or more activity execution has been swapped with others with a relatively similar timestamp. This might indicate, for example, manual logging errors, rather than real control-flow violations. We plan to investigate the use of fuzzy sets to introduce some notion of similarity between events with close timestamps, taking then this similarity into account when assessing the control-flow compliance level.

CRediT authorship contribution statement

Sicui Zhang: Conceptualization, Investigation, Methodology, Software, Data curation, Formal analysis, Visualization, Validation, Writing – original draft. **Laura Genga:** Conceptualization, Methodology, Validation, Software, Writing – review & editing. **Lukas Dekker:** Investigation, Validation, Writing – review & editing. **Hongchao Nie:** Funding acquisition, Validation, Writing – review & editing. **Xudong Lu:** Project administration, Writing – review & editing. **Huilong Duan:** Project administration, Writing – review & editing. **Uzay Kaymak:** Project administration, Conceptualization, Validation, Supervision, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

The research has received funding from the Brain Bridge Project sponsored by Philips Research.

References

[1] E.R. Taghiabadi, V. Gromov, D. Fahland, W.P. van der Aalst, Compliance checking of data-aware and resource-aware compliance requirements, in: OTM Confederated International Conferences" on the Move To Meaningful Internet Systems", Springer, 2014, pp. 237–257.
[2] M. De Leoni, W.M. van der Aalst, B.F. Van Dongen, Data-and resource-aware conformance checking of business processes, in: International Conference on Business Information Systems, Springer, 2012, pp. 48–59.
[3] W. van der Aalst, A. Adriansyah, A.K.A. De Medeiros, F. Arcieri, T. Baier, T. Blickle, J.C. Bose, P. Van Den Brand, R. Brandtjen, J. Buijs, et al., Process mining manifesto, in: International Conference on Business Process Management, Springer, 2011, pp. 169–194.
[4] W. van der Aalst, A. Adriansyah, B. Van Dongen, Replaying history on process models for conformance checking and performance analysis, Wiley Interdiscip. Rev.: Data Min. Knowl. Discov. 2 (2) (2012) 182–192.
[5] A. Adriansyah, B.F. Van Dongen, W.M. van der Aalst, Memory-efficient alignment of observed and modeled behavior, BPM Center Rep. 3 (2013).
[6] A. Adriansyah, J. Munoz-Gama, J. Carmona, B.F. van Dongen, W.M. van der Aalst, Alignment based precision checking, in: International Conference on Business Process Management, Springer, 2012, pp. 137–149.
[7] F. Mannhardt, M. De Leoni, H.A. Reijers, W.M. van der Aalst, Balanced multi-perspective checking of process conformance, Computing 98 (4) (2016) 407–437.
[8] K. Kim, Y.-M. Lee, Understanding uncertainty in medicine: concepts and implications in medical education, Korean J. Med. Educ. 30 (3) (2018) 181.
[9] A.L. Simpkin, R.M. Schwartzstein, Tolerating uncertainty—the next medical revolution? N. Engl. J. Med. 375 (18) (2016).
[10] B. Djulbegovic, I. Hozo, S. Greenland, Uncertainty in clinical medicine, in: Philosophy of Medicine, Elsevier, 2011, pp. 299–356.
[11] E. Oussedik, M.S. Anderson, S.R. Feldman, Risk Versus Benefit or Risk Versus Risk: Risk Aversion in the Medical Decision Making Process, Taylor & Francis, 2017.
[12] E.I. Papageorgiou, A new methodology for decisions in medical informatics using fuzzy cognitive maps based on fuzzy rule-extraction techniques, Appl. Soft Comput. 11 (1) (2011) 500–513.
[13] M. De Leoni, W.M. van der Aalst, Aligning event logs and process models for multi-perspective conformance checking: An approach based on integer linear programming, in: Business Process Management, Springer, 2013, pp. 113–129.
[14] F. Mannhardt, Multi-perspective process mining., in: BPM (Dissertation/Demos/Industry), 2018, pp. 41–45.
[15] H.-J. Zimmermann, P. Zysno, Latent connectives in human decision making, Fuzzy Sets and Systems 4 (1980) 37–51.
[16] L.A. Zadeh, Fuzzy sets, Inf. Control 8 (1965) 338–353.
[17] G. Beliakov, A. Pradera, T. Calvo, Aggregation Functions: A Guide for Practitioners, Springer, Berlin, 2007.
[18] S. Zhang, L. Genga, L. Dekker, H. Nie, X. Lu, H. Duan, U. Kaymak, Towards multi-perspective conformance checking with aggregation operations, in: International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems, Springer, 2020, pp. 215–229.
[19] S. Zhang, L. Genga, H. Yan, H. Nie, X. Lu, U. Kaymak, Towards multi-perspective conformance checking with fuzzy sets, Int. J. Interact. Multimed. Artif. Intell. 6(Special Issue on Artificial Intelligence, Paving the Way to the Future) (2021) 134–141.
[20] C.A. Petri, Kommunikation mit Automaten, (Ph.D. thesis), Universität Hamburg, 1962.
[21] P. Wohed, W.M. van der Aalst, M. Dumas, A.H. ter Hofstede, N. Russell, On the suitability of BPMN for business process modelling, in: International Conference on Business Process Management, Springer, 2006, pp. 161–176.
[22] J.-S.R. Jang, C.-T. Sun, E. Mizutani, Neuro-fuzzy and soft computing-a computational approach to learning and machine intelligence [book review], IEEE Trans. Automat. Control 42 (10) (1997) 1482–1484.
[23] G.J. Klir, B. Yuan, Fuzzy Sets and Fuzzy Logic: Theory and Applications, Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1995.
[24] S. Cheng, J.N. Mordeson, Fuzzy linear operators and fuzzy normed linear spaces, in: First International Conference on Fuzzy Theory and Technology Proceedings, Abstracts and Summaries, 1992, pp. 193–197.
[25] M. Grabisch, C. Labreuche, Fuzzy measures and integrals in MCDA, in: Multiple Criteria Decision Analysis, Springer, 2016, pp. 553–603.
[26] V. Torra, Y. Narukawa, Modeling Decisions: Information Fusion and Aggregation Operators, Springer Science & Business Media, 2007.
[27] A. Adriansyah, B.F. Van Dongen, W.M. van der Aalst, Conformance checking using cost-based fitness analysis, in: 2011 IEEE 15th International Enterprise Distributed Object Computing Conference, IEEE, 2011, pp. 55–64.
[28] A. Rozinat, W.M. van der Aalst, Conformance checking of processes based on monitoring real behavior, Inf. Syst. 33 (1) (2008) 64–95.
[29] H.D. Cheng, J.-R. Chen, Automatically determine the membership function based on the maximum entropy principle, Inform. Sci. 96 (1997) 163–182.
[30] W. Pedrycz, Why triangular membership functions? Fuzzy Sets and Systems 64 (1994) 21–30.
[31] M. Grabisch, J.-L. Marichal, R. Mesiar, E. Pap, Aggregation functions: Construction methods, conjunctive, disjunctive and mixed classes, Inform. Sci. 181 (1) (2011) 23–43.
[32] J.M. da Costa Sousa, U. Kaymak, Model predictive control using fuzzy decision functions, IEEE Trans. Syst. Man Cybern. B 31 (1) (2001) 54–65.
[33] R.R. Yager, On a general class of fuzzy connectives, Fuzzy Sets and Systems 4 (1980) 235–242.
[34] R. Dechter, J. Pearl, Generalized best-first search strategies and the optimality of a, J. ACM 32 (3) (1985) 505–536.
[35] H. Yan, P. Van Gorp, U. Kaymak, X. Lu, L. Ji, C.C. Chiau, H.H. Korsten, H. Duan, Aligning event logs to task-time matrix clinical pathways in bpmn for variance analysis, IEEE J. Biomed. Health Inf. 22 (2) (2017) 311–317.
[36] W.M. van der Aalst, M. Dumas, C. Ouyang, A. Rozinat, E. Verbeek, Conformance checking of service behavior, ACM Trans. Internet Technol. (TOIT) 8 (3) (2008) 1–30.
[37] M. Alizadeh, M. De Leoni, N. Zannone, History-based construction of alignments for conformance checking: Formalization and implementation, in: International Symposium on Data-Driven Process Discovery and Analysis, Springer, 2014, pp. 58–78.
[38] J. Munoz-Gama, J. Carmona, W.M. van der Aalst, Single-entry single-exit decomposed conformance checking, Inf. Syst. 46 (2014) 102–122.
[39] F. Caron, J. Vanthienen, B. Baesens, Comprehensive rule-based compliance checking and risk management with process mining, Decis. Support Syst. 54 (3) (2013) 1357–1369.
[40] D. Borrego, I. Barba, Conformance checking and diagnosis for declarative business process models in data-aware scenarios, Expert Syst. Appl. 41 (11) (2014) 5340–5352.
[41] W. Song, H.-A. Jacobsen, C. Zhang, X. Ma, Dependence-based data-aware process conformance checking, IEEE Trans. Serv. Comput. (2018).
[42] H. Van Der Aa, H. Leopold, H.A. Reijers, Efficient process conformance checking on the basis of uncertain event-to-activity mappings, IEEE Trans. Knowl. Data Eng. 32 (5) (2019) 927–940.
[43] M. Pegoraro, M.S. Uysal, W.M. van der Aalst, Conformance checking over uncertain event data, Inf. Syst. 102 (2021) 101810.

[44] S.J. Leemans, W.M. van der Aalst, T. Brockhoff, A. Polyvyanyy, Stochastic process mining: Earth movers' stochastic conformance, Inf. Syst. 102 (2021) 101724.
[45] A. Adriansyah, B.F. Van Dongen, W.M. van der Aalst, Towards robust conformance checking, in: International Conference on Business Process Management, Springer, 2010, pp. 122–133.
[46] R. Bosma, U. Kaymak, J. Berg, van den, H. Udo, Fuzzy modelling of farmer motivations for integrated farming in the Vietnamese Mekong Delta, in: The 14th IEEE International Conference on Fuzzy Systems, 2005. FUZZ'05, Institute of Electrical and Electronics Engineers, United States, 2005, pp. 827–832.
[47] Z. Hao, Z. Xu, H. Zhao, H. Fujita, A dynamic weight determination approach based on the intuitionistic fuzzy Bayesian network and its application to emergency decision making, IEEE Trans. Fuzzy Syst. 26 (4) (2017) 1893–1907.
[48] J. Xue, C. Wu, Z. Chen, P. Van Gelder, X. Yan, Modeling human-like decision-making for inbound smart ships based on fuzzy decision trees, Expert Syst. Appl. 115 (2019) 172–188.
[49] P. Ji, H.-y. Zhang, J.-q. Wang, Fuzzy decision-making framework for treatment selection based on the combined QUALIFLEX–TODIM method, Internat. J. Systems Sci. 48 (14) (2017) 3072–3086.
[50] J. Li, L. Luo, X. Wu, C. Liao, H. Liao, W. Shen, Prioritizing the elective surgery patient admission in a Chinese public tertiary hospital using the hesitant fuzzy linguistic ORESTE method, Appl. Soft Comput. 78 (2019) 407–419.
[51] S. Bragaglia, F. Chesani, P. Mello, M. Montali, D. Sottara, Fuzzy conformance checking of observed behaviour with expectations, in: Congress of the Italian Association for Artificial Intelligence, Springer, 2011, pp. 80–91.
[52] E.S. Pane, A.D. Wibawa, M.H. Purnomo, Event log-based fraud rating using interval type-2 fuzzy sets in fuzzy AHP, in: 2016 IEEE Region 10 Conference (TENCON), IEEE, 2016, pp. 1965–1968.
[53] K. Ganesha, S. Dhanush, S.S. Raj, An approach to fuzzy process mining to reduce patient waiting time in a hospital, in: 2017 International Conference on Innovations in Information, Embedded and Communication Systems, ICIIECS, IEEE, 2017, pp. 1–6.