Measuring the Interestingness of Temporal Logic Behavioral Specifications in Process Mining

Alessio Cecconi[a,*], Giuseppe De Giacomo[b], Claudio Di Ciccio[b], Fabrizio Maria Maggi[c], Jan Mendling[a]

[a] *WU Vienna, Vienna, Austria*
[b] *Sapienza University of Rome, Rome, Italy*
[c] *Free University of Bozen-Bolzano, Bolzano, Italy*

Abstract

The assessment of behavioral rules with respect to a given dataset is key in several research areas, including declarative process mining, association rule mining, and specification mining. An assessment is required to check how well a set of discovered rules describes the input data, and to determine to what extent data complies with predefined rules. Particularly in declarative process mining, Support and Confidence are used most often, yet they are reportedly unable to provide a sufficiently rich feedback to users and cause rules representing coincidental behavior to be deemed as representative for the event logs. In addition, these measures are designed to work on a predefined set of rules, thus lacking generality and extensibility. In this paper, we address this research gap by developing a measurement framework for temporal rules based on Linear-time Temporal Logic with Past on Finite Traces (LTLp_f). The framework is suitable for any temporal rules expressed in a reactive form and for custom measures based on the probabilistic interpretation of such rules. We show that our framework can seamlessly adapt well-known measures of the association rule mining field to declarative process mining. Also, we test our software prototype implementing the framework on synthetic and real-world data, and investigate

*Corresponding author. Authors in alphabetical order.
Email addresses: alessio.cecconi@wu.ac.at (Alessio Cecconi), giuseppe.degiacomo@uniroma1.it (Giuseppe De Giacomo), claudio.diciccio@uniroma1.it (Claudio Di Ciccio), maggi@inf.unibz.it (Fabrizio Maria Maggi), jan.mendling@wu.ac.at (Jan Mendling)

the properties characterizing those measures in the context of process analysis.

Keywords: Declarative Process Mining, Specification Mining, Association Rule Mining, Interestingness Measures, Temporal Rules

1. Introduction

Measuring the degree to which process traces comply with behavioral rules is key in process analysis branches such as conformance checking [1], compliance assessment [2], and discovery of process constraints [3]. To date, several measures have been defined to this end, yet there are two major problems with their application.

First, measures adopted for process mining are defined inconsistently for specific applications. For example, among the most frequently used measures there are Support and Confidence. However, their definition has been customized to the specification languages in use and even for the specific mining algorithms under analysis. For instance, there is a significant difference in the definition of Support used in [3] (percentage of traces fully compliant to a rule) and [4] (percentage of activations that lead to a fulfillment), in a way that the Support of rule "If a is executed, then b will be executed later" on a set of traces like $\{\langle \mathsf{a}, \mathsf{b}, \mathsf{c}, \mathsf{d}\rangle, \langle \mathsf{a}, \mathsf{b}, \mathsf{c}, \mathsf{a}\rangle, \langle \mathsf{a}, \mathsf{c}\rangle\}$ is equal to 0.33 for [3] and 0.5 according to [4]. Furthermore, the definition of those measures are defined ad hoc for specific sets of rules, like DECLARE [5] templates. Such issues hinder the fair comparison and eventually the advancement of rule-based process mining.

Second, the opportunity to adopt available measures from association rule mining has been largely missed so far. A plethora of measures that are reportedly superior in comparison to Support and Confidence [6] have been proposed in this field. Support measures only the satisfaction frequency of a rule and Confidence its validity. Although those are crucial aspects in the assessment of a rule, they do not suffice to avoid spurious result [7]. Markedly, various directions have been explored in prior research, among others by revising Support and Confidence [8, 9, 10], and by defining complementary measures [11, 12, 13].

However, all such measures do not account for the temporal perspective, which is a first-class citizen dimension in process mining.

In this paper, we address the research challenge of defining a general and comprehensive measurement system. More specifically, we propose a framework based on formal semantics grounded in Linear-time Temporal Logic with Past on Finite Traces (LTLp_f) to express Reactive Constraints (RCons) [14] in a way that abstracts from specific rule-specification languages. Such constraints are rules in the form of "IF A THEN B", thus binding the satisfaction of an antecedent A to the occurrence of a consequent B, wherein both A and B are temporal formulas. We show that a probabilistic interpretation of the fine-grained temporal logic evaluation of any such formulas allows us to employ all available association rule mining measures as-is for temporal rules. Markedly, the framework has linear time and space complexity with respect to the input size.

Our contribution extends concepts from association rule mining to temporal logic specifications. In this way, we define a foundation upon which the fitness between measures and data analysis scenarios can be discussed in future research. We conduct an extensive set of simulation experiments, the results of which demonstrate that, driven by known properties, the measures respond differently to changes in the behavior evidenced by event logs. This is an important finding that highlights the need to select measures according to the application context, confirming previous findings for association rules [15] in the realm of temporal logic specifications.

This paper is an extension of our previous conference paper [16] presented at the 2nd International Conference on Process Mining (ICPM 2020). We extend the contribution in the following aspects:

- We extend the framework to provide measures at the level of the event log, and not only descriptive statistics at the trace level (Section 4);
- We revise the experiments based on the new theoretical extension. In particular, we score the proposed measures and rank them in order to identify the best candidates to be used for rule discovery (Section 6);

- We analyze the memory consumption of the framework along with the time performance (Section 5);
- We extend the discussion about the interestingness measures used and exploit their known properties for a better understanding of log behavior (Sections 3, 6 and 6.3).

Additionally, we prove the linear-time performance of the RCons verification in Appendix A.

The remainder of this paper is structured as follows. Section 2 discusses prior research on measures for declarative process mining and specification mining. Section 3 defines preliminaries upon which we define our framework. Section 4 defines the measurement framework. Section 5 presents a computational study of the framework and Section 6 shows the results of an array of simulation experiments and discusses them. Finally, Section 7 summarizes the contribution of the paper and points to opportunities for future research.

2. Related work

Behavioral rules have been widely used to support application scenarios such as association rule mining in machine learning, process discovery and conformance checking in process mining, and specification mining in software engineering. The assessment of rules with respect to the available data is a key component of all these techniques.

In association rule mining, interestingness measures are used to discriminate candidate pairs of relevant co-occurring events. A common technique is to discover frequent rules above a certain Support threshold (frequency) and to prune the results below a certain Confidence threshold (validity). For example, [17] discovers associations between items through an Apriori algorithm based on the downward-closure property of the Support measure. Nevertheless, the use of Support and Confidence alone is reportedly not sufficient to avoid a large number of spurious results [7], i.e., the discovery of rules which are frequently satisfied by

the data although merely by chance, thus threatening their statistical validity. A plethora of new measures have been proposed in the literature to overcome the limits of using only Support and Confidence [11], yet the employment of Support and Confidence as the main interestingness measures remains widespread. The main goal driving the development of better measures is indeed the exclusion of spurious rules, so as to let the more crucial ones stand out. Several measures are directly improving on or refining the results of Support and Confidence (e.g., Lift [8] scales Confidence with the Support of the consequent of a rule), others combine different measures (e.g., Added Value [12] subtracts the Prevalence to the Confidence of a rule), and further ones show complementary information (e.g., Specificity [11] measures to what extent the absence of the consequent is related to the absence of the antecedent of a rule).

In declarative process discovery, interestingness measures are used to prune candidate rules based on user-defined thresholds. This pruning approach is used for DECLARE discovery in [3, 4] and for DCR graphs discovery in [18]. These techniques are mainly based on Support and Confidence, which lead to the aforementioned limits [7]. In addition, the definitions of these metrics also differ depending on the techniques that use them. For instance, the Support measure presented in [3] does not correspond to the Support measure of [4], although both are expressly defined for the sole DECLARE constraints.

In the area of conformance checking, interestingness measures are used to check the degree of conformance of a rule with respect to an execution trace. In [19], Linear Temporal Logic (LTL) rules are checked against each trace in a given event log. This is highly generic as it supports any custom LTL formula, but it reports only binary results, i.e., whether a rule holds in a trace or not. In [20], Burattin et al. use measures like fulfillment ratio and violation ratio, based on the evaluation of the number of activations of a rule (intuitively, the occurrences of its antecedent) that lead to a fulfillment and the number of activations that lead to a violation in an event log. However, these metrics are specifically bound to the set of DECLARE rules, thus not providing a general measurement framework that can be applied to general type of rule.

In specification mining, interestingness measures are also used to prune candidate temporal specifications based on user-defined thresholds. Interestingly, specification mining and declarative process discovery are two largely overlapping concepts from distinct fields. Yang et al. [21] discover 2-value temporal patterns using a trace measure that quantifies partial satisfactions of a rule. Yet, the technique lacks generality as it is limited only to alternation patterns (similar to AlternateResponse and AlternatePrecedence in Table 1) and the adopted computation heuristics are tailored to the software domain. Le et al. [6] emphasize the limits of using only Support and Confidence measures and investigate properties of other measures reviewed in [11]. Their results demonstrate that there are several measures outperforming Support and Confidence, and that the combination of different measures yields better results. However, they limit their study to 2-value temporal patterns (specifically, Response and Precedence in Table 1). Furthermore, their computation of the probability for a temporal specification is based on a sliding window technique [22]: traces are read in chunks of the size of a given window, then the probability of a rule is the percentage of windows in which it is satisfied. They test the effect of different window sizes, showing that their results depend not only on the input rules and the data, but also on this parameter selection. Lemieux et al. [23] extend specification mining to arbitrary LTL specifications (implicitly on finite traces) beyond 2-value templates. However, they resort to the sole Support and Confidence measures to prune uninteresting results, thus incurring in the already mentioned statistical limits [7].

The aforementioned shortcomings of quality measures are also discussed in the field of sequence mining [24] when dealing with discovering patterns (specifically subsequences) to classify sequential data. Egho et al. [25] highlight how measures like Confidence and Lift alone lead to unstable classification results of subsequences, proposing a probabilistic Bayesian-based measure to overcome such an instability and avoiding the requirement of setting thresholds for measures. It falls under the family of techniques based on the minimum description length principle, like [26], where an encoding scheme is used to discover a minimal set

of subsequences which can reproduce the original data. Notably, subsequence interleaving patterns are only a subset of patterns expressible with LTL formulae. Works adopting behavioral rules for classification like [27], on the other hand, fall back to the sole employment of Support.

In summary, despite the discussion in different fields on measures and the problem of spurious relations, there is no technique that supports at the same time a comprehensive and extensible multi-measurements assessment of rules and its applicability on general temporal logic specifications.

3. Preliminaries

To develop our framework, we build on the sound foundations of LTLp$_f$. In this section, we introduce the fundamentals of LTLp$_f$ formulae (Section 3.1) and interestingness measures for association rules (Section 3.2).

3.1. Linear-time Temporal Logic with Past on Finite Traces (LTLp$_f$)

As the formal foundations of our framework, we consider the rules specified in Linear Temporal Logic on Finite Traces (LTL$_f$) [28] as used in DECLARE [5, 29]. LTL$_f$ has the same syntax as LTL [30]. Its semantics is interpreted on finite traces (here abstracted as finite sequences of symbols), and thus take into account that business processes are assumed to eventually terminate [31]. DECLARE focuses on a set of specific LTL$_f$ formulas. Table 1 illustrates some of the most important rules for business process specifications in DECLARE.

LTLp$_f$ is an extension of LTL$_f$ supporting the expression of properties of the past (hence the "p" suffix) [14]. Well-formed LTLp$_f$ formulae are built from an alphabet $\Sigma \supseteq \{\mathsf{a}\}$ of propositional symbols and are closed under the boolean connectives, the unary temporal operators $\bigcirc$ (next) and $\ominus$ (previous), and the binary temporal operators $\mathbf{U}$ (Until) and $\mathbf{S}$ (Since):

$$\varphi ::= \mathsf{a} | (\neg\varphi) | (\varphi_1 \wedge \varphi_2) | (\bigcirc\varphi) | (\varphi_1 \ \mathbf{U} \ \varphi_2) | (\ominus\varphi) | (\varphi_1 \ \mathbf{S} \ \varphi_2).$$

From these basic operators, the following can be derived: Classical boolean abbreviations $\mathit{True}, \mathit{False}, \vee, \rightarrow$; Constant $t_{\text{End}} \equiv \neg \bigcirc \mathit{True}$, denoting the last

Table 1: Some DECLARE constraints expressed as RCons.

Constraint	**LTL$_f$ expression [28]**	**RCon**
PARTICIPATION(a)	$\Diamond \mathsf{a}$	$t_{\text{Start}} \boxdot\!\!\rightarrow \Diamond \mathsf{a}$
INIT(a)	a	$t_{\text{Start}} \boxdot\!\!\rightarrow \mathsf{a}$
END(a)	$\Box\Diamond \mathsf{a}$	$t_{\text{End}} \boxdot\!\!\rightarrow \mathsf{a}$
ATMOSTONE(a)	$\Box(\mathsf{a} \rightarrow \bigcirc(\neg\Diamond \mathsf{a}))$	$\mathsf{a} \boxdot\!\!\rightarrow \bigcirc(\neg\Diamond \mathsf{a})$
RESPONDEDEXISTENCE(a, b)	$\Diamond \mathsf{a} \rightarrow \Diamond \mathsf{b}$	$\mathsf{a} \boxdot\!\!\rightarrow (\Diamond \mathsf{b} \vee \Diamond\!\!\!\cdot \mathsf{b})$
RESPONSE(a, b)	$\Box(\mathsf{a} \rightarrow \Diamond \mathsf{b})$	$\mathsf{a} \boxdot\!\!\rightarrow \Diamond \mathsf{b}$
ALTERNATERESPONSE(a, b)	$\Box(\mathsf{a} \rightarrow \Diamond \mathsf{b}) \wedge \Box(\mathsf{a} \rightarrow \bigcirc(\neg \mathsf{a}\ \mathbf{W}\ \mathsf{b}))$	$\mathsf{a} \boxdot\!\!\rightarrow \bigcirc(\neg \mathsf{a}\ \mathbf{U}\ \mathsf{b})$
CHAINRESPONSE(a, b)	$\Box(\mathsf{a} \rightarrow \Diamond \mathsf{b}) \wedge \Box(\mathsf{a} \rightarrow \bigcirc \mathsf{b})$	$\mathsf{a} \boxdot\!\!\rightarrow \bigcirc \mathsf{b}$
PRECEDENCE(a, b)	$\neg \mathsf{b}\ \mathbf{W}\ \mathsf{a}$	$\mathsf{b} \boxdot\!\!\rightarrow \Diamond\!\!\!\cdot \mathsf{a}$
ALTERNATEPRECEDENCE(a, b)	$(\neg \mathsf{b}\ \mathbf{W}\ \mathsf{a}) \wedge \Box(\mathsf{b} \rightarrow \bigcirc(\neg \mathsf{b}\ \mathbf{W}\ \mathsf{a}))$	$\mathsf{b} \boxdot\!\!\rightarrow \ominus(\neg \mathsf{b}\ \mathbf{S}\ \mathsf{a})$
CHAINPRECEDENCE(a, b)	$(\neg \mathsf{b}\ \mathbf{W}\ \mathsf{a}) \wedge \Box(\bigcirc \mathsf{b} \rightarrow \mathsf{a})$	$\mathsf{b} \boxdot\!\!\rightarrow \ominus \mathsf{a}$
SUCCESSION(a, b)	$\Box(\mathsf{a} \rightarrow \Diamond \mathsf{b}) \wedge (\neg \mathsf{b}\ \mathbf{W}\ \mathsf{a})$	$(\mathsf{a} \vee \mathsf{b}) \boxdot\!\!\rightarrow (\mathsf{a} \wedge \Diamond \mathsf{b}) \vee (\mathsf{b} \wedge \Diamond\!\!\!\cdot \mathsf{a})$
ALTERNATESUCCESSION(a, b)	$\Box(\mathsf{a} \rightarrow \Diamond \mathsf{b}) \wedge \Box(\mathsf{a} \rightarrow \bigcirc(\neg \mathsf{a}\ \mathbf{W}\ \mathsf{b})) \wedge (\neg \mathsf{b}\ \mathbf{W}\ \mathsf{a}) \wedge \Box(\mathsf{b} \rightarrow \bigcirc(\neg \mathsf{b}\ \mathbf{W}\ \mathsf{a}))$	$(\mathsf{a} \vee \mathsf{b}) \boxdot\!\!\rightarrow (\mathsf{a} \wedge \bigcirc(\neg \mathsf{a}\ \mathbf{U}\ \mathsf{b})) \vee (\mathsf{b} \wedge \ominus(\neg \mathsf{b}\ \mathbf{S}\ \mathsf{a}))$
CHAINSUCCESSION(a, b)	$\Box(\mathsf{a} \rightarrow \Diamond \mathsf{b}) \wedge \Box(\mathsf{a} \rightarrow \bigcirc \mathsf{b}) \wedge (\neg \mathsf{b}\ \mathbf{W}\ \mathsf{a}) \wedge \Box(\bigcirc \mathsf{b} \rightarrow \mathsf{a})$	$(\mathsf{a} \vee \mathsf{b}) \boxdot\!\!\rightarrow (\mathsf{a} \wedge \bigcirc \mathsf{b}) \vee (\mathsf{b} \wedge \ominus \mathsf{a})$
COEXISTENCE(a, b)	$(\Diamond \mathsf{a} \wedge \Diamond \mathsf{b}) \vee (\neg\Diamond \mathsf{a} \wedge \neg\Diamond \mathsf{b})$	$(\mathsf{a} \vee \mathsf{b}) \boxdot\!\!\rightarrow (\mathsf{a} \wedge \Diamond \mathsf{b}) \vee (\mathsf{a} \wedge \Diamond\!\!\!\cdot \mathsf{b}) \vee (\mathsf{b} \wedge \Diamond \mathsf{a}) \vee (\mathsf{b} \wedge \Diamond\!\!\!\cdot \mathsf{a})$
NOTCOEXISTENCE(a, b)	$\Box(\mathsf{a} \rightarrow \neg\Diamond \mathsf{b}) \wedge \Box(\mathsf{b} \rightarrow \neg\Diamond \mathsf{a})$	$(\mathsf{a} \vee \mathsf{b}) \boxdot\!\!\rightarrow (\mathsf{a} \wedge \neg\Diamond\!\!\!\cdot \mathsf{b} \wedge \neg\Diamond \mathsf{b}) \vee (\mathsf{b} \wedge \neg\Diamond\!\!\!\cdot \mathsf{a} \wedge \neg\Diamond \mathsf{a})$

instant of a trace; Constant $t_{\text{Start}} \equiv \neg \ominus \mathit{True}$, denoting the first instant of a trace; $\Diamond\varphi \equiv \mathit{True}\ \mathbf{U}\ \varphi$ indicating that φ holds true eventually before t_{End}; $\varphi_1\ \mathbf{W}\ \varphi_2 \equiv (\varphi_1\ \mathbf{U}\ \varphi_2) \vee \Box\varphi_1$, which relaxes $\mathbf{U}$ as φ_2 may never hold true; $\Diamond\!\!\!\cdot\varphi \equiv \mathit{True}\ \mathbf{S}\ \varphi$ indicating that φ holds true eventually in the past, after t_{Start}; $\Box\varphi \equiv \neg\Diamond\neg\varphi$ indicating that φ holds true from the current instant till t_{End}; $\boxminus\varphi \equiv \neg\Diamond\!\!\!\cdot\neg\varphi$ indicating that φ holds true from t_{Start} to the current instant.

Given a finite trace t of length $n \in \mathbb{N}$, an LTLp$_f$ formula φ is satisfied in a given instant i $(1 \leqslant i \leqslant n)$ by induction of the following:

$t, i \models \mathit{True}$; $t, i \not\models \mathit{False}$;

$t, i \models \mathsf{a}$ iff $t(i)$ is assigned with a;

$t, i \models \neg\varphi$ iff $t, i \not\models \varphi$;

$t, i \models \varphi_1 \wedge \varphi_2$ iff $t, i \models \varphi_1$ and $t, i \models \varphi_2$;
$t, i \models \bigcirc\varphi$ iff $i < n$ and $t, i+1 \models \varphi$;
$t, i \models \ominus\varphi$ iff $i > 1$ and $t, i-1 \models \varphi$;
$t, i \models \varphi_1 \ \mathbf{U} \ \varphi_2$ iff $t, j \models \varphi_2$ with $i \leqslant j \leqslant n$, and $t, k \models \varphi_1$ for all k s.t. $i \leqslant k < j$;
$t, i \models \varphi_1 \ \mathbf{S} \ \varphi_2$ iff $t, j \models \varphi_2$ with $1 \leqslant j \leqslant i$, and $t, k \models \varphi_1$ for all k s.t. $j < k \leqslant i$.
A formula φ is satisfied by a trace t, written $t \models \varphi$ iff $t, 1 \models \varphi$. One of the central properties of LTLp_f and LTL_f is that a deterministic finite state automaton (DFS) A_φ can be computed such that for every trace t we have $t \models \varphi$ iff t is in the language recognized by A_φ, as illustrated in [32, 14, 33].

Without loss of generality, in this paper we abstract traces as finite strings of symbols representing *events*. We assume that every event reports on the execution of exactly one task and LTLp_f formulae use those tasks as their propositional symbols – the so-called DECLARE *assumption* [32]. A trace extracted from the real-world Sepsis event log [34] is, e.g., $t_{\text{Sepsis}} = \langle$ER Registration, ER Triage, ER Sepsis Triage, CRP, Lactic Acid, IV Liquid, IV Antibiotics$\rangle$. Notice that this trace complies with the constraints that Mannhardt et al. identified as normative for the Sepsis treatment process [35], including the following ones: *(i)* INIT(ER Registration), i.e., every trace begins with the registration at the emergency department,; *(ii)* ATMOSTONE(ER Triage), i.e., the triage in the emergency room occurs at most once in a process run; *(iii)* RESPONSE(ER Triage, ER Sepsis Triage), i.e., the ER Triage procedure should be eventually followed by the Sepsis-specific triage, *(iv)* PRECEDENCE(ER Sepsis Triage, IV Antibiotics), i.e., the intravenous injection of antibiotics must be preceded by the ER Sepsis Triage procedure.

Table 17 contains an extended set of DECLARE constraints that a correct execution of the Sepsis treatment process should fulfill. For the sake of succinctness, we may use single-letter identifiers in place of full-length task names whenever suitable in the following examples.

An *event log* is a multi-set of traces, i.e., traces can recur multiple times in an event log. The *cardinality* of the event log is the sum of the multiplicities of its traces. Considering the Sepsis event log, the multiplicity of t_{Sepsis} is

Table 2: Contingency tables to which the "IF A THEN B" rules and their variables ($x \in \{A, B\}$) comply with. On the left-hand side, the contingency table is based on probabilities ($P(x)$); on the right-hand side, the contingency table is based on frequencies (where $|x|$ is the number of occurrences of x and N is the total number of occurrences).

	A	$\neg A$	
B	$P(AB)$	$P(\neg AB)$	$P(B)$
$\neg B$	$P(A\neg B)$	$P(\neg A\neg B)$	$P(\neg B)$
	$P(A)$	$P(\neg A)$	1

	A	$\neg A$	
B	$\|AB\|$	$\|\neg AB\|$	$\|B\|$
$\neg B$	$\|A\neg B\|$	$\|\neg A\neg B\|$	$\|\neg B\|$
	$\|A\|$	$\|\neg A\|$	N

13 (i.e., t_{Sepsis} occurs 13 times in the event log). Other exemplary traces of that log are $t'_{\text{Sepsis}} = \langle$ER Registration, ER Triage, ER Sepsis Triage$\rangle$ (occurring 35 times) and $t''_{\text{Sepsis}} = \langle$ER Registration, ER Triage, ER Sepsis Triage, Leucocytes, CRP$\rangle$ (with a multiplicity of 24). The cardinality of the event sub-log consisting of the traces above is thus 72. The cardinality of the whole Sepsis event log is 1050.

3.2. Interestingness measures for association rules

In this section, we revisit key findings of research on research on quality measures in association rule mining. More specifically, we build on prior research that surveys measures in the area of software engineering and association rule mining, namely [36, 37, 11, 15]. These works are specifically suited as a foundation due to the wide coverage of measures and their comparative study of both formal and user-perceived measure properties. The rules under study are in the form "IF A THEN B", where A is called the *antecedent* of the rule, and B its *consequent*. We refer to A and B as *variables* of the rule.

Specifically, we consider only probability-based *objective* measures, i.e., measures depending only on the data as opposed to those requiring user-provided parameters. Objective measures are based on the probabilities derived from the contingency table of the occurrences of the variables, as depicted in Table 2. Table 3 presents the list of measures covered in this study. In the following, we provide a brief description of each measure.

Support [17] measures the frequency of the co-occurrence of the antecedent and

Table 3: Probabilistic measures for association rules with respective range and properties.

Measure	Formula	Range	P1	P2	P3	P4	P5	P6
Support	$P(AB)$	$[0, 1]$	–	sym	↘	var	var	↘
Confidence/Precision	$P(B\|A)$	$[0, 1]$	✓	asym	→	var	const	↘
Coverage	$P(A)$	$[0, 1]$	–	asym	↘	var	var	↘
Prevalence	$P(B)$	$[0, 1]$	–	asym	↗	var	var	↘
Recall	$P(A\|B)$	$[0, 1]$	✓	asym	↘	var	var	↷
Specificity	$P(\neg B\|\neg A)$	$[0, 1]$	–	asym	↘	const	var	↘
Accuracy	$P(AB) + P(\neg A\neg B)$	$[0, 1]$	–	sym	↘	var	var	↘
Lift/Interest	$\frac{P(B\|A)}{P(B)}$ or $\frac{P(AB)}{P(A)P(B)}$	$[0, +\infty)$	–	sym	↘	const	var	↘
Leverage	$P(B\|A) - P(A)P(B)$	$[-1, 1]$	–	sym	↗	var	var	↘
Added Value/ Change of Support/ Centered Confidence	$P(B\|A) - P(B)$	$[-1, 1]$	–	asym	↘	const	var	↘
Relative risk	$\frac{P(B\|A)}{P(B\|\neg A)}$	$[0, +\infty)$	–	asym	↘	const	var	↘
Jaccard	$\frac{P(AB)}{P(A) + P(B) - P(AB)}$	$[0, 1]$	✓	sym	↘	var	var	↘
Certainty factor	$\frac{P(B\|A) - P(B)}{1 - P(B)}$	$[-1, 1]$	–	asym	↘	const	const	↘
Odds ratio/ Bayes Factor	$\frac{P(AB)P(\neg A\neg B)}{P(A\neg B)P(\neg BA)}$	$[0, +\infty)$	–	sym	↘	const	const	↘
Yule's Q	$\frac{P(AB)P(\neg A\neg B) - P(A\neg B)P(\neg AB)}{P(AB)P(\neg A\neg B) + P(A\neg B)P(\neg AB)}$	$[-1, 1]$	–	sym	↘	const	const	↷
Yule's Y	$\frac{\sqrt{P(AB)P(\neg A\neg B)} - \sqrt{P(A\neg B)P(\neg AB)}}{\sqrt{P(AB)P(\neg A\neg B)} + \sqrt{P(A\neg B)P(\neg AB)}}$	$[-1, 1]$	–	sym	↘	const	const	↷
Klosgen	$\sqrt{P(AB)} \times \max(P(B\|A) - P(B), P(A\|B) - P(A))$	$[-1, 1]$	–	asym	↘	const	var	↘
Conviction	$\frac{P(A)P(\neg B)}{P(A\neg B)}$	$[0, +\infty)$	–	asym	↘	const	const	↘
Interestingness Weighting Dependency	$\left(\left(\frac{P(AB)}{P(A)P(B)}\right)^k - 1\right) \times P(AB)^m$	$[0, +\infty)$	–	sym	↘	const	var	↘
Collective Strength	$\frac{P(AB) + P(\neg B\|\neg A)}{P(A)P(B) + P(\neg A)P(\neg B)} \times \frac{1 - P(A)P(B) - P(\neg A)P(\neg B)}{1 - P(AB) - P(\neg B\|\neg A)}$	$[0, +\infty)$	–	asym	?	var	var	↷
Laplace Correction	$\frac{N(AB) + 1}{N(A) + 2}$	$[0.5, 1]$	✓	asym	→	var	var	↘
Gini Index	$P(A) \times (P(B\|A)^2 + P(\neg B\|A)^2)$ $+P(\neg A) \times (P(B\|\neg A)^2 + P(\neg B\|\neg A)^2)$ $-P(B)^2 - P(\neg B)^2$	$[0, 1]$	–	asym	↘	const	var	↘
J-Measure	$P(AB) \log \frac{P(B\|A)}{P(B)} + P(A\neg B) \log \frac{P(\neg B\|A)}{P(\neg B)}$	$(-\infty, +\infty)$	–	asym	↘	const	const	↘
One-Way Support	$P(B\|A) \log_2 \frac{P(AB)}{P(A)P(B)}$	$(-\infty, +\infty)$	–	asym	↘	const	var	↘
Two-Way Support	$P(AB) \log_2 \frac{P(AB)}{P(A)P(B)}$	$(-\infty, +\infty)$	–	sym	↘	const	var	↘
Two-Way Support Variation	$P(AB) \log_2 \frac{P(AB)}{P(A)P(B)} + P(A\neg B) \log_2 \frac{P(A\neg B)}{P(A)P(\neg B)}$ $+P(\neg AB) \log_2 \frac{P(\neg AB)}{P(\neg A)P(B)} + P(\neg A\neg B) \log_2 \frac{P(\neg A\neg B)}{P(\neg A)P(\neg B)}$	$(-\infty, +\infty)$	–	sym	↘	const	const	↘
Ø–Coefficient (Pearson's Linear Correlation Coefficient)	$\frac{P(AB) - P(A)P(B)}{\sqrt{P(A)P(B)P(\neg A)P(\neg B)}}$	$(-\infty, +\infty)$	–	sym	↘	const	var	↘
Piatetsky-Shapiro	$P(AB) - P(A)P(B)$	$[-1, 1]$	–	sym	↘	const	var	↘
Cosine	$\frac{P(AB)}{\sqrt{P(A)P(B)}}$	$[0, +\infty)$	✓	sym	↘	var	var	↘
Loevinger	$1 - \frac{P(A)P(\neg B)}{P(A\neg B)}$	$(-\infty, 1]$	–	asym	↗	const	const	↘
Information Gain	$\log \frac{P(AB)}{P(A)P(B)}$	$(-\infty, +\infty)$	–	sym	↘	const	var	↷
Sebag-Schoenauer	$\frac{P(AB)}{P(A\neg B)}$	$[0, +\infty)$	✓	asym	→	var	const	↘
Least Contradiction	$\frac{P(AB) - P(A\neg B)}{P(B)}$	$(-\infty, +\infty)$	✓	asym	↘	var	var	↘
Odd Multiplier	$\frac{P(AB)P(\neg B)}{P(B)P(A\neg B)}$	$[0, +\infty)$	–	asym	↘	const	const	↘
Example and Counterexample Rate	$1 - \frac{P(A\neg B)}{P(AB)}$	$(-\infty, 1]$	✓	asym	→	var	const	↷
Zhang	$\frac{P(AB) - P(A)P(B)}{\max(P(AB)P(\neg B), P(B)P(A\neg B))}$	$(-\infty, +\infty)$	–	asym	↘	const	const	↷

the consequent. The Support of the sole antecedent of the rule is also called **Coverage**, while the Support of the consequent is called **Prevalence**.

Confidence [17] measures the co-occurrences of the antecedent and the consequent in the fraction of data containing the antecedent.

Recall [11] measures the co-occurrences of the antecedent and the consequent in the fraction of data containing the consequent.

Specificity [11] measures the co-absences of the antecedent and the consequent in the fraction of data not containing the antecedent.

Accuracy [11] measures the fraction of the data either containing both the consequent and the antecedent or neither of the two.

Lift [8] scales the Confidence by the probability of the consequent, to check if the co-occurrence of the antecedent and consequent is more likely than their independence.

Leverage [9] measures the difference between the Confidence of the rule and the independent occurrence of its variables.

Added Value [12] measures the difference between the Confidence of the rule and the probability of the consequent alone, to check if the conditioned occurrence of the consequent differ from its unconditioned occurrence.

Relative Risk [38] measures the ratio of the conditional probability of the consequent given the antecedent to the conditional probability of the consequent given the negation of the antecedent.

Jaccard's Coefficient [39] measures the similarity between the variables using the ratio of their co-occurrence to the union of all their independent occurrences.

Certainty Factor [10] measures the ratio of the Added Value of the rule to the Added Value of the consequent alone, in order to see the variation of probability in the data containing the antecedent.

Odds Ratio [40] measures the ratio of the probability of having the consequent when the antecedent is present to the probability of having the consequent when the antecedent is not present.

Odds Multiplier [40] measures the ratio of the probability of having the an-

tecedent when the consequent is present to the probability of having the antecedent when the consequent is not present.

Yules's Q and Yules's Y [41, 42] are normalization of the Odds Ratio to have it centered around 0 and ranging between -1 and 1.

Klosgen's Measure [43] weights the Support of the rule using its Added Value.

Conviction [13] measures the occurrences of the antecedent without the consequent in comparison to their independence.

Interestingness Weighting Dependency [44] combines Support and Lift of a rule and explicitly gives weights to each of them to let the user decide their relative importance.

Collective Strength [12] measures the ratio of the agreement ratio (number of non-violations per expected number of non-violations) to the violation ratio (number of violations per expected number of violations).

Laplace Correction [45] is a variation of Confidence to take into account small data.

Gini index [46] measures if the entropy introduced by a rule brings a marked difference.

J-measure [47] is an entropy based measure for the information content of a rule.

One-way Support and Two-way Support [48] combine respectively Confidence and Support of a rule with the degree of independence between the variables.

Two-way Support Variation [48] measures the change in the Two-way-Support.

Linear Correlation Coefficient [49] measures the Pearson's correlation between the variables.

Piatetsky-Shapiro [9] measures the difference between the co-occurrences of antecedent and consequent and their independent frequency.

Cosine [37] measures the geometric mean between Lift and Support of a rule.

Loevinger [50] measures the homogeneity between antecedent and consequent.

Information Gain [51] is the logarithm of the Lift.

Sebag-Schoenauer [52] measures the proportion of positive and negative oc-

currences of the antecedent.

Least Contradiction [53] measures the difference between positive and negative occurrences of the antecedent weighted by its frequency.

Example and Counterexample Rate [11] measures the proportion of the antecedent occurrences with and without consequent.

Zhang [54] measures the positive or negative association between the antecedent and the consequent.

Different studies have been dedicated to the analysis of general properties for measures [11, 9, 37, 15]. Properties show the response of measures under certain conditions. Therefore, properties can be used to group similar measures and decide the proper ones to be employed depending on the context. For example, we will analyze the sensitivity of measures to the increase of noise in the data as an important selection criterion for rule monitoring or discovery. We will delve deeper into this aspect in Section 6. In this paper, we focus specifically on a subset of the properties proposed in [37] and [15], as their meaning and effects are reportedly recognizable in a clear manner by the final user. The selected properties are explained below and associated to each measure $\mathcal{M}$ in Table 3.

P1. Null invariance [37]. The measure is unaffected by traces not containing neither A or B. Therefore, it assesses whether the traces not related to the rule affect the measurement or not. To satisfy this property, the measure should not vary when $|\neg A \neg B|$ increases in the contingency table, while the other values remain fixed. In Table 3, we use the '✓' or '–' symbols to indicate whether the property holds or not, respectively. For example, for Confidence and Recall this property holds, whereas for Support, Leverage and Collective Strength it does not.

P2. Asymmetric processing of variables [15]. The measure is asymmetric under variable swap, i.e., the measure of IF A THEN B differs from that of IF B THEN A. The measure enjoys this property if it does not vary upon the swapping of the values of $|\neg AB|$ and $|A \neg B|$ in the contingency table. In Table 3, every measure is marked with "asym" or "sym" to indicate

whether the property is enjoyed or not, respectively. For instance, Support and Leverage are symmetric under variable swap, whereas Confidence, Recall and Collective Strength provide an asymmetric processing of the variables.

P3. Variation with occurrences of B in the absence of A [15]. The value of the measure varies when the occurrences of B in the absence of A increase. In other words, this property focuses on whether the independent occurrence of B influences the measure. Given an IF A THEN B rule, if B is very likely to occur regardless of A, the influence of A on B may be questioned. To verify this, the value of the measure varies when $|\neg AB|$ increases in the contingency values (and the other values remain fixed). In Table 3, measures are marked with '$\searrow$' if the variation is a decrease (as in the case of Support), '$\nearrow$' if it is an increase (e.g., Leverage), '?' if the variation can be either a decrease or an increase depending on the values of B (e.g., Collective Strength), '$\rightarrow$' if the value does not vary at all (e.g., Confidence).

P4. Reference situations: independence [15]. If the variables are independent, then the measure exhibits a known value. The variables are considered independent when their joint probability is equal to the product of their respective probabilities, i.e., $P(AB) = P(A)P(B)$. The measure should have a constant and known value in that case. In Table 3, measures are labeled as "const" (e.g., Lift) if this property holds, and "var" otherwise (e.g., Support).

P5. Reference situations: logical rule [15]. If the rule is always satisfied, then the measure exhibits a known value. An IF A THEN B rule is always satisfied if $P(A\neg B) = 0$. In other words, if there are no counterexamples in the data, the value of the measure that enjoys this property is a known constant (let it be a number or tendency to infinite). In Table 3, measures are labeled as "const" if this property holds (as in the case of Confidence), and "var" otherwise (see, e.g., Lift).

P6. Trend with $P(A\neg B)$ [15]. If the number of counterexamples to the rule rises, the value of the measure reacts exhibiting a decreasing trend that denotes a higher or lower sensitivity. For an IF A THEN B rule, a higher number of counterexamples translates into an increase of $P(A\neg B)$. Against that increase, the measure may show a fast (convex), linear, or slow (concave) decrease. Measures in Table 3 are labeled either as '$\searrow$' (convex, e.g., Conviction), '$\searrow$' (linear, e.g., Support), or '$\searrow$' (concave, e.g., Recall) accordingly.

These properties, according to [15], can be divided into *normative* (i.e., always desirable: **P2**, **P3**, **P4**, **P5**) and *subjective* (i.e., depending on the user needs: **P1**, **P6**). We will resort to these properties to examine the quality measures in the context of process mining.

In the following section, therefore, we extend the aforementioned measures to temporal process rules.

4. Temporal-extended measurement framework

Our framework addresses the limits of Support and Confidence measurements by building on LTLp$_f$ formal semantics and the spectrum of measures defined in different areas of computer science. Furthermore, it is generic as it allows for the usage of any probabilistic measure (including those of Table 3) on any temporal-logic-based rules specification. To this end, Section 4.1 formalizes the reactive temporal specification of rules, Section 4.2 discusses their probabilistic interpretation, and Section 4.4 defines the overall framework.

4.1. Reactive temporal specification

Our first building block is the concept of Reactive Constraint (RCon), originally introduced in [14], the paper which we extend here. A rule typically expresses that the occurrence of given preconditions (activator) implies certain consequences (target). The reactive nature of this kind of rule lies in the fact that the condition on the target is exerted only if the activator is verified. We codify this intuition in RCons, whose semantics is based on LTLp$_f$.

Definition 4.1 (Reactive Constraint (RCon)). *Given an alphabet of propositional symbols* $\Sigma \cup \{t_{Start}, t_{End}, \mathit{True}, \mathit{False}\}$*, let* φ_α *and* φ_τ *be LTLp*$_f$ *formulae over* Σ*. A* Reactive Constraint (RCon) Ψ *is a pair* $(\varphi_\alpha, \varphi_\tau)$ *hereafter denoted as* $\Psi \triangleq \varphi_\alpha \boxdot\!\!\rightarrow \varphi_\tau$*.*

An RCon is interpreted as follows: Each time the activator is true, the target should be true at that point of the trace. For example, $\mathsf{a} \boxdot\!\!\rightarrow \Diamond \mathsf{c}$ is an RCon stating that every time a (the activator, φ_α) is *True*, then also $\Diamond \mathsf{c}$ (the target, φ_τ) must evaluate to *True*. That RCon corresponds to Response(a, c) in Declare as it requires that if a occurs in a trace, it must be eventually followed by c. $\mathsf{c} \boxdot\!\!\rightarrow \Diamond\!\!\!-\, \mathsf{d}$ corresponds to Precedence(d, c) in Declare because it requires that every time c (the activator) occurs in a trace, then it has to be preceded by d (the target). Table 1 provides a list of standard Declare constraints expressed in the form of RCons. An RCon that goes beyond the standard repertoire of Declare is $(\Diamond\!\!\!-\, \mathsf{b} \wedge \Diamond \mathsf{e}) \boxdot\!\!\rightarrow (\neg \mathsf{c} \vee \Diamond \mathsf{f})$: Its activator is the formula $\varphi_\alpha = \Diamond\!\!\!-\, \mathsf{b} \wedge \Diamond \mathsf{e}$, satisfied between the occurrence of b and the occurrence of e in a trace; its target is the formula $\varphi_\tau = \neg \mathsf{c} \vee \Diamond \mathsf{f}$, which evaluates to *True* if either c is *False*, or c occurs and is eventually followed by f. Because at every event of the trace (i.e., any point in time) both the activator and target can be either *True* or *False*, the possible evaluation of an RCon can result in either of the following four combinations.

Definition 4.2 (RCon evaluation). *Given an RCon* $\Psi \triangleq \varphi_\alpha \boxdot\!\!\rightarrow \varphi_\tau$ *and a trace* t *of length* $n \in \mathbb{N}$*, let* i *denote the* i*-th event in the trace (*$1 \leqslant i \leqslant n$*). For each* $t_i \in t$ *the possible evaluations of* Ψ *are:*

$$\varphi_\alpha = \mathit{False},\ \varphi_\tau = \mathit{False} \quad \textit{if} \quad t, i \not\models \varphi_\alpha \text{ and } t, i \not\models \varphi_\tau;$$
$$\varphi_\alpha = \mathit{False},\ \varphi_\tau = \mathit{True} \quad \textit{if} \quad t, i \not\models \varphi_\alpha \text{ and } t, i \models \varphi_\tau;$$
$$\varphi_\alpha = \mathit{True},\ \varphi_\tau = \mathit{False} \quad \textit{if} \quad t, i \models \varphi_\alpha \text{ and } t, i \not\models \varphi_\tau;$$
$$\varphi_\alpha = \mathit{True},\ \varphi_\tau = \mathit{True} \quad \textit{if} \quad t, i \models \varphi_\alpha \text{ and } t, i \models \varphi_\tau.$$

Table 4: Evaluation (0 is *False* and 1 is *True*) and probabilistic interpretation of RCon $\mathsf{a} \boxdot\!\!\rightarrow \Diamond \mathsf{c}$.

Trace $t_1 =$ $\langle$	a,	b,	c,	d,	f,	c,	e,	c,	h $\rangle$
φ_α: a	1	0	0	0	0	0	0	0	0
φ_τ: $\Diamond$c	1	1	1	1	1	1	1	1	0

$P(\varphi_\alpha, t) = 1/9$	$P(\neg\varphi_\alpha \cap \varphi_\tau, t) = 7/9$	$P(\neg\varphi_\alpha \cap \neg\varphi_\tau, t) = 1/9$
$P(\varphi_\tau, t) = 8/9$	$P(\varphi_\alpha \cap \neg\varphi_\tau, t) = 0/9$	$P(\varphi_\alpha \cap \varphi_\tau, t) = 1/9$

$P(\varphi_\tau \mid \varphi_\alpha, t) = 1/1$	$P(\varphi_\tau \mid \neg\varphi_\alpha, t) = 7/8$
$P(\neg\varphi_\tau \mid \varphi_\alpha, t) = 0/1$	$P(\neg\varphi_\tau \mid \neg\varphi_\alpha, t) = 1/8$

For example, the second and third rows of Tables 4 to 6 show the evaluation of RCons $\mathsf{a} \boxdot\!\!\rightarrow \Diamond \mathsf{c}$ (i.e., Response(a, c) in Table 4), $\mathsf{c} \boxdot\!\!\rightarrow \Diamond\!\!\!-\, \mathsf{d}$ (i.e, Precedence(d, c) in Table 5) and $(\Diamond\!\!\!-\, \mathsf{b} \wedge \Diamond \mathsf{e}) \boxdot\!\!\rightarrow (\neg \mathsf{c} \vee \Diamond \mathsf{f})$ (Table 6) on trace $\langle \mathsf{a}, \mathsf{b}, \mathsf{c}, \mathsf{d}, \mathsf{f}, \mathsf{c}, \mathsf{e}, \mathsf{c}, \mathsf{h} \rangle$. Notice that φ_α and φ_τ are evaluated separately at every event of a trace.

The RCon evaluation can be performed efficiently based on the automaton-based techniques defined in [14], adapting it for offline verification. The full discussion on this aspect can be found in Appendix A, but we briefly outline the rationale here. Intuitively, we resort to [14, Theorem 4]: An RCon can be separated in pure-past, pure-present and pure-future components. The respective sub-formulae contain only past temporal operators, none, or only future ones, respectively. As they are LTLp$_f$ formulae, all components correspond to finite state automata (FSAs). The key point is that, by mirroring pure-past formulae and reversing their automata, a single replay of the sub-trace from the beginning to the activator event keeps track of the truth value of the pure-past formula till that point. As we have knowledge of the whole trace, and thus of the suffix too a fortiori, we can apply the same principle to pure-future formulae too: A single replay from the end of the trace to the activator event keeps track of the truth value of the pure-future formula from that point *onwards*.

Table 5: Evaluation (0 is *False* and 1 is *True*) and probabilistic interpretation of RCon $\mathsf{c} \,\square\!\!\rightarrow\, \Diamond \mathsf{d}$.

Trace $t_1 =$ $\langle$	a,	b,	c,	d,	f,	c,	e,	c,	h $\rangle$
φ_α: c	0	0	1	0	0	1	0	1	0
φ_τ: $\Diamond$d	0	0	0	1	1	1	1	1	1

$P(\varphi_\alpha, t) = 3/9$	$P(\neg\varphi_\alpha \cap \varphi_\tau, t) = 4/9$	$P(\neg\varphi_\alpha \cap \neg\varphi_\tau, t) = 2/9$
$P(\varphi_\tau, t) = 6/9$	$P(\varphi_\alpha \cap \neg\varphi_\tau, t) = 1/9$	$P(\varphi_\alpha \cap \varphi_\tau, t) = 2/9$

$P(\varphi_\tau \mid \varphi_\alpha, t) = 2/3$	$P(\varphi_\tau \mid \neg\varphi_\alpha, t) = 4/6$
$P(\neg\varphi_\tau \mid \varphi_\alpha, t) = 1/3$	$P(\neg\varphi_\tau \mid \neg\varphi_\alpha, t) = 2/6$

From this optimization, it follows that any LTLp$_f$ formula can be evaluated at each event reading the trace only twice (as in [4, 55] though for any RCon and not just DECLARE constraints): Once from t_{Start} to t_{End} (past components) and once from t_{End} to t_{Start} (future components). This result implies that the computational cost depends linearly on the number of events in the event log and in the number of rules to verify. Specifically, given an event log L of cardinality $|L|$, assuming that (i) every trace $t \in L$ has a length of up to n, and (ii) $|R|$ rules are under analysis, the cost to evaluate all rules on L is: $O(|L| \times n \times |R|)$.

4.2. Probabilistic interpretation on a trace

The evaluation of RCons indicates whether a rule holds true or false within a trace. In real life, traces often contain noise or partially deviate from desired process specifications. In those occasions wherein the trace may contain also events that do not satisfy the rule, we are interested in understanding *to what degree* a rule is satisfied. As we have previously defined the notion of satisfaction for φ_α and φ_τ on single events (Def. 4.2), we can devise a probabilistic interpretation for RCons over traces.

Definition 4.3 (Probability of an LTLp$_f$ formula in a trace). *Given an*

Table 6: Evaluation (0 is *False* and 1 is *True*) and probabilistic interpretation of RCon $(\Diamond\!\!-\mathsf{b} \wedge \Diamond\mathsf{e}) \,\square\!\!\rightarrow\, (\neg\mathsf{c} \vee \Diamond\mathsf{f})$.

Trace $t_1 =$ $\langle$	a,	b,	c,	d,	f,	c,	e,	c,	h $\rangle$
φ_α: $(\Diamond\!\!-\mathsf{b} \wedge \Diamond\mathsf{e})$	0	1	1	1	1	1	1	0	0
φ_τ: $(\neg\mathsf{c} \vee \Diamond\mathsf{f})$	1	1	1	1	1	0	1	0	1

$P(\varphi_\alpha, t) = 6/9$	$P(\neg\varphi_\alpha \cap \varphi_\tau, t) = 2/9$	$P(\neg\varphi_\alpha \neg\varphi_\tau, t) = 1/9$
$P(\varphi_\tau, t) = 7/9$	$P(\varphi_\alpha \cap \neg\varphi_\tau, t) = 1/9$	$P(\varphi_\alpha \cap \varphi_\tau, t) = 5/9$

$P(\varphi_\tau \mid \varphi_\alpha, t) = 5/6$	$P(\varphi_\tau \mid \neg\varphi_\alpha, t) = 2/3$
$P(\neg\varphi_\tau \mid \varphi_\alpha, t) = 1/6$	$P(\neg\varphi_\tau \mid \neg\varphi_\alpha, t) = 1/3$

LTLp$_f$ formula φ and a trace t of length $|t| = n$, we define the probability of φ in t[1] *as the proportion of the events in t that satisfy φ:*

$$P(\varphi, t) = \frac{|\{i \in [1, n] : t, i \models \varphi\}|}{n}.$$

Definition 4.4 (Joint probability of LTLp$_f$ formulae in a trace). *Given two LTLp$_f$ formulae φ_1 and φ_2 and a trace t of length n, we define the probability of the intersection of φ_1 and φ_2 in t (*joint probability*) as the proportion of the events in t that satisfy both φ_1 and φ_2:*

$$P(\varphi_1 \cap \varphi_2, t) = \frac{|\{i \in [1, n] : t, i \models \varphi_1 \text{ and } t, i \models \varphi_2\}|}{n}.$$

The probabilities of the evaluations of activator and target of an RCon follow

[1] Notice that we use the comma in $P(\varphi, t)$ and similar following expressions to separate the parameters, namely the formula to be evaluated (here, φ) and the structure on which the formula is analyzed (here, t).

Table 7: Contingency table of the probabilities of an RCon $\varphi_\alpha \boxdot\!\!\rightarrow \varphi_\tau$ in a trace.

	φ_α	$\neg\varphi_\alpha$	
φ_τ	$P(\varphi_\alpha \cap \varphi_\tau, t)$	$P(\neg\varphi_\alpha \cap \varphi_\tau, t)$	$P(\varphi_\tau, t)$
$\neg\varphi_\tau$	$P(\varphi_\alpha \cap \neg\varphi_\tau, t)$	$P(\neg\varphi_\alpha \cap \neg\varphi_\tau, t)$	$P(\neg\varphi_\tau, t)$
	$P(\varphi_\alpha, t)$	$P(\neg\varphi_\alpha, t)$	1

Table 8: Contingency table of the probabilities of RCon $(\Diamond\!\!\!-\mathsf{b} \wedge \Diamond\mathsf{e}) \boxdot\!\!\rightarrow (\neg\mathsf{c} \vee \Diamond\mathsf{f})$ in trace $\langle \mathsf{a}, \mathsf{b}, \mathsf{c}, \mathsf{d}, \mathsf{f}, \mathsf{c}, \mathsf{e}, \mathsf{c}, \mathsf{h} \rangle$ (based on the results illustrated in Table 6).

	$\Diamond\!\!\!-\mathsf{b} \wedge \Diamond\mathsf{e}$	$\neg(\Diamond\!\!\!-\mathsf{b} \wedge \Diamond\mathsf{e})$	
$\neg\mathsf{c} \vee \Diamond\mathsf{f}$	5/9	2/9	7/9
$\neg(\neg\mathsf{c} \vee \Diamond\mathsf{f})$	1/9	1/9	2/9
	2/3	1/3	1

from the above definitions (Table 7 shows the resulting contingency table):

$$P(\neg\varphi_\alpha \cap \neg\varphi_\tau, t) = \frac{|\{i \in [1,n] : t, i \not\models \varphi_\alpha \text{ and } t, i \not\models \varphi_\tau\}|}{n};$$
$$P(\neg\varphi_\alpha \cap \varphi_\tau, t) = \frac{|\{i \in [1,n] : t, i \not\models \varphi_\alpha \text{ and } t, i \models \varphi_\tau\}|}{n};$$
$$P(\varphi_\alpha \cap \neg\varphi_\tau, t) = \frac{|\{i \in [1,n] : t, i \models \varphi_\alpha \text{ and } t, i \not\models \varphi_\tau\}|}{n};$$
$$P(\varphi_\alpha \cap \varphi_\tau, t) = \frac{|\{i \in [1,n] : t, i \models \varphi_\alpha \text{ and } t, i \models \varphi_\tau\}|}{n}.$$

For example, Tables 4 to 6 show the probabilities resulting from the evaluation of RCons $\mathsf{a} \boxdot\!\!\rightarrow \Diamond\mathsf{c}$, $\mathsf{c} \boxdot\!\!\rightarrow \Diamond\!\!\!-\mathsf{d}$, and $(\Diamond\!\!\!-\mathsf{b} \wedge \Diamond\mathsf{e}) \boxdot\!\!\rightarrow (\neg\mathsf{c} \vee \Diamond\mathsf{f})$, respectively, on trace $\langle \mathsf{a}, \mathsf{b}, \mathsf{c}, \mathsf{d}, \mathsf{f}, \mathsf{c}, \mathsf{e}, \mathsf{c}, \mathsf{h} \rangle$. Table 8 summarizes the results of Table 6 in a contingency table.

In association rule mining, rules are in the form "IF A THEN B", given an antecedent A and a consequent B. Probabilities defined as above allow for the application of measures defined for association rule mining [11] to the context of temporal logic specifications over finite traces. To that extent, it suffices to map φ_α to A and φ_τ to B, thus having $P(A)$ as $P(\varphi_\alpha, t)$, $P(B)$ as $P(\varphi_\tau, t)$, and $P(AB)$ as $P(\varphi_\alpha \cap \varphi_\tau, t)$. For example, Table 9 shows some measures com-

Table 9: Trace measures computation and event log statistics of a sample of measures for RCon $(\Diamond\!\!-\mathsf{b} \wedge \Diamond \mathsf{e}) \,\square\!\!\rightarrow\, (\neg \mathsf{c} \vee \Diamond \mathsf{f})$. The statistics are computed skipping divisions of zero by zero (marked with "NaN"), whenever they occur.

	Event log	**Support** $P(\varphi_\alpha \cap \varphi_\tau)$	**Confidence** $P(\varphi_\tau \mid \varphi_\alpha)$	**Specificity** $P(\neg\varphi_\tau \mid \neg\varphi_\alpha)$	**Lift** $\frac{P(\varphi_\alpha \cap \varphi_\tau)}{P(\varphi_\alpha)P(\varphi_\tau)}$
$t_1 =$	$\langle \mathsf{a}, \mathsf{b}, \mathsf{c}, \mathsf{d}, \mathsf{f}, \mathsf{c}, \mathsf{e}, \mathsf{c}, \mathsf{h} \rangle$	0.56	0.83	0.33	1.07
$t_2 =$	$\langle \mathsf{b}, \mathsf{d}, \mathsf{a}, \mathsf{f}, \mathsf{g}, \mathsf{d}, \mathsf{e}, \mathsf{d} \rangle$	0.88	1.00	0.00	1.00
$t_3 =$	$\langle \mathsf{a}, \mathsf{c}, \mathsf{d}, \mathsf{b}, \mathsf{c}, \mathsf{e}, \mathsf{f}, \mathsf{c} \rangle$	0.38	1.00	0.20	1.14
$t_4 =$	$\langle \mathsf{b}, \mathsf{c}, \mathsf{c}, \mathsf{e}, \mathsf{a} \rangle$	0.40	0.50	0.00	0.83
$t_5 =$	$\langle \mathsf{b}, \mathsf{c}, \mathsf{d}, \mathsf{a} \rangle$	0.00	NaN	0.25	NaN
	Mean	0.44	0.83	0.16	1.01
	Standard deviation	0.32	0.24	0.15	0.13
	Variance	0.10	0.06	0.02	0.02

puted from the probabilities associated to the activator and the target of RCon $(\Diamond\!\!-\mathsf{b} \wedge \Diamond \mathsf{e}) \,\square\!\!\rightarrow\, (\neg \mathsf{c} \vee \Diamond \mathsf{f})$. These probabilities pertain to the events in the traces, intuitively answering the question: "How likely is it that an event satisfies the constraint?". It follows that also the measures based on them pertain to events with respect to traces, and that their statistics over the entire event logs will preserve the focus on the singles events.

4.3. Probabilistic interpretation on an event log

Following the probability definition for LTLp_f formulae over traces, it is of interest to define similar probabilities over event logs. Intuitively, if the trace probabilities assess the likelihood of the rule correctness in events within a trace, event log probabilities should question the likelihood of the rule correctness in the traces of an event log. As previously mentioned, the descriptive statistics of trace measures across an event log are suitable for this purpose because they preserve the focus on the events. In order to achieve this goal, we have to first derive the conditional probability of the target given the activator in a trace, i.e., $P(\varphi_\tau|\varphi_\alpha, t)$. Intuitively, this is the probability for the target to hold true

when the activator holds true. Notice that this viewpoint is conceptually closer to the notion of Reactive Constraint than the joint probability of activator and target. Furthermore, the conditional interpretation of rules is also more in line with their human interpretation [56]. This makes the conditional probability a suitable means for the probabilistic analysis of a constraint in a trace as a whole.

Definition 4.5 (Conditional probability of LTLp$_f$ formulae in a trace). *Given two LTLp$_f$ formulae φ_1 and φ_2 and a trace t of length n, we define the conditional probability of φ_2 given φ_1 over t as the proportion of events satisfying φ_2 among those that satisfy φ_1:*

$$P(\varphi_2|\varphi_1, t) = \frac{|\{i \in [1, n] : t, i \models \varphi_1 \text{ and } t, i \models \varphi_2\}|}{|\{i \in [1, n] : t, i \models \varphi_1\}|}.$$

From the above definition, it follows that:

$$P(\varphi_\tau|\varphi_\alpha, t) = \frac{|\{i \in [1, n] : t, i \models \varphi_\alpha \text{ and } t, i \models \varphi_\tau\}|}{|\{i \in [1, n] : t, i \models \varphi_\alpha\}|};$$
$$P(\neg\varphi_\tau|\varphi_\alpha, t) = \frac{|\{i \in [1, n] : t, i \models \varphi_\alpha \text{ and } t, i \not\models \varphi_\tau\}|}{|\{i \in [1, n] : t, i \models \varphi_\alpha\}|};$$
$$P(\varphi_\tau|\neg\varphi_\alpha, t) = \frac{|\{i \in [1, n] : t, i \not\models \varphi_\alpha \text{ and } t, i \models \varphi_\tau\}|}{|\{i \in [1, n] : t, i \not\models \varphi_\alpha\}|};$$
$$P(\neg\varphi_\tau|\neg\varphi_\alpha, t) = \frac{|\{i \in [1, n] : t, i \not\models \varphi_\alpha \text{ and } t, i \not\models \varphi_\tau\}|}{|\{i \in [1, n] : t, i \not\models \varphi_\alpha\}|}.$$

Tables 4 to 6 show the conditional probabilities of RCon $(\diamondsuit\mathsf{b} \wedge \Diamond\mathsf{e}) \boxdot\!\!\rightarrow (\neg\mathsf{c} \vee \Diamond\mathsf{f})$ on trace $\langle \mathsf{a}, \mathsf{b}, \mathsf{c}, \mathsf{d}, \mathsf{f}, \mathsf{c}, \mathsf{e}, \mathsf{c}, \mathsf{h} \rangle$. Notably, the conditional probability is not influenced by the total amount of events in the trace, but only by the events of interest.

To devise the probability of an RCon in an event log L (henceforth, *event log probability*), we have to detect the portion of the event log satisfying an LTLp$_f$ formula. To this end, we split the event log into a sub-log that has only the traces in which the activator occurs at least once (i.e., every $t \in L$ such that $P(\varphi_\alpha, t) > 0$), and the complementary sub-log consisting of the traces in which the activator does *not* occur (i.e., every $t \in L$ such that $P(\varphi_\alpha, t) = 0$). Given the above considerations and the definition of conditional probability for RCons in single traces (Def. 4.5), we devise a probabilistic interpretation for RCons over event logs as follows.

Table 10: Contingency table of conditional event log probabilities.

	$P(\varphi_\alpha, t) > 0$	$P(\varphi_\alpha, t) = 0$	
φ_τ	$\frac{\sum_{t \in L} P(\varphi_\tau \mid \varphi_\alpha, t)}{\lvert L \rvert}$	$\frac{\sum_{t \in L} P(\varphi_\tau \mid \neg\varphi_\alpha, t)}{\lvert L \rvert}$	$\frac{\sum_{t \in L : P(\varphi_\alpha, t) > 0} P(\varphi_\tau \mid \varphi_\alpha, t) + \sum_{t \in L : P(\varphi_\alpha, t) = 0} P(\varphi_\tau \mid \neg\varphi_\alpha, t)}{\lvert L \rvert}$
$\neg\varphi_\tau$	$\frac{\sum_{t \in L} P(\neg\varphi_\tau \mid \varphi_\alpha, t)}{\lvert L \rvert}$	$\frac{\sum_{t \in L} P(\neg\varphi_\tau \mid \neg\varphi_\alpha, t)}{\lvert L \rvert}$	$\frac{\sum_{t \in L : P(\varphi_\alpha, t) > 0} P(\neg\varphi_\tau \mid \varphi_\alpha, t) + \sum_{t \in L : P(\varphi_\alpha, t) = 0} P(\neg\varphi_\tau \mid \neg\varphi_\alpha, t)}{\lvert L \rvert}$
	$\frac{\sum_{t \in L} P(\varphi_\alpha, t)}{\lvert L \rvert}$	$\frac{\sum_{t \in L} P(\neg\varphi_\alpha, t)}{\lvert L \rvert}$	1

Definition 4.6 (Conditional probability of LTLp$_f$ formulae in an event log). *Let φ_1 and φ_2 be two LTLp$_f$ formulae and L an event log of cardinality $|L|$. We say that φ_1 is* non-null *in a trace $t \in L$ if and only if $P(\varphi_1, t) > 0$. If $P(\varphi_1, t) = 0$, we say that φ_1 is* null *in t. The* conditional probability of φ_2 given φ_1 *in L is the portion of the event log that consists of traces for which φ_1 is non-null and satisfies φ_2, given the satisfaction of φ_1:*

$$P(\varphi_2 \mid \varphi_1, L) = \frac{\sum_{t \in L : P(\varphi_1, t) > 0} P(\varphi_2 \mid \varphi_1, t)}{|L|}.$$

The conditional probability of φ_2 given $\neg\varphi_1$ *in L is the portion of the event log that consists of traces for which φ_1 is null and satisfies φ_2, given the satisfaction of $\neg\varphi_1$:*

$$P(\varphi_2 \mid \neg\varphi_1, L) = \frac{\sum_{t \in L : P(\varphi_1, t) = 0} P(\varphi_2 \mid \neg\varphi_1, t)}{|L|}.$$

Table 10 shows the resulting contingency table. In the following, we provide the proof of the correctness of our approach.

Theorem 4.1 (Contingency of event log conditionals). *Given two LTLp$_f$ formulae φ_1 and φ_2 and an event log L of cardinality $|L|$, let $|L|_{P(\varphi_1)>0}$ be the number of traces in which φ_1 is non-null and $|L|_{P(\varphi_1)=0}$ the number of traces in which φ_1 is null. It follows that $P(\varphi_2 \mid \varphi_1, L) + P(\neg\varphi_2 \mid \varphi_1, L) + P(\varphi_2 \mid \neg\varphi_1, L) + P(\neg\varphi_2 \mid \neg\varphi_1, L) = 1$.*

Proof 4.1. *In light of the fact that there cannot be a trace where $P(\varphi_1)$ is both 0 and not 0 at the same time, the proof of Theorem 4.1 proceeds as follows.*

$$P(\varphi_2|\varphi_1, L) + P(\neg\varphi_2|\varphi_1, L) + P(\varphi_2|\neg\varphi_1, L) + P(\neg\varphi_2|\neg\varphi_1, L) = 1 \tag{1}$$

$$\frac{\sum_{t\in L:P(\varphi_1)>0} P(\varphi_2|\varphi_1, t)}{|L|} + \frac{\sum_{t\in L:P(\varphi_1)>0} P(\neg\varphi_2|\varphi_1, t)}{|L|}$$
$$+\frac{\sum_{t\in L:P(\varphi_1)=0} P(\varphi_2|\neg\varphi_1, t)}{|L|} + \frac{\sum_{t\in L:P(\varphi_1)=0} P(\neg\varphi_2|\neg\varphi_1, t)}{|L|} = 1 \tag{2}$$

$$\sum_{t\in L:P(\varphi_1)>0} P(\varphi_2|\varphi_1, t) + \sum_{t\in L:P(\varphi_1)>0} P(\neg\varphi_2|\varphi_1, t)$$
$$+ \sum_{t\in L:P(\varphi_1)=0} P(\varphi_2|\neg\varphi_1, t) + \sum_{t\in L:P(\varphi_1)=0} P(\neg\varphi_2|\neg\varphi_1, t) = |L| \tag{3}$$

$$\sum_{t\in L:P(\varphi_1)>0} (P(\varphi_2|\varphi_1, t) + P(\neg\varphi_2|\varphi_1, t))$$
$$+ \sum_{t\in L:P(\varphi_1)=0} (P(\varphi_2|\neg\varphi_1, t) + P(\neg\varphi_2|\neg\varphi_1, t)) = |L| \tag{4}$$

$$\sum_{t\in L:P(\varphi_1)>0} \left(\frac{P(\varphi_1 \cap \varphi_2, t)}{P(\varphi_1, t)} + \frac{P(\varphi_1 \cap \neg\varphi_2, t)}{P(\varphi_1, t)}\right)$$
$$+ \sum_{t\in L:P(\varphi_1)=0} \left(\frac{P(\neg\varphi_1 \cap \varphi_2, t)}{P(\neg\varphi_1, t)} + \frac{P(\neg\varphi_1 \cap \neg\varphi_2, t)}{P(\neg\varphi_1, t)}\right) = |L| \tag{5}$$

$$\sum_{t\in L:P(\varphi_1)>0} \left(\frac{P(\varphi_1 \cap \varphi_2, t) + P(\varphi_1 \cap \neg\varphi_2, t)}{P(\varphi_1, t)}\right)$$
$$+ \sum_{t\in L:P(\varphi_1)=0} \left(\frac{P(\neg\varphi_1 \cap \varphi_2, t) + P(\neg\varphi_1 \cap \neg\varphi_2, t)}{P(\neg\varphi_1, t)}\right) = |L| \tag{6}$$

$$\sum_{t\in L:P(\varphi_1)>0} \left(\frac{P(\varphi_1, t)}{P(\varphi_1, t)}\right) + \sum_{t\in L:P(\varphi_1)=0} \left(\frac{P(\neg\varphi_1, t)}{P(\neg\varphi_1, t)}\right) = |L| \tag{7}$$

$$\sum_{t\in L:P(\varphi_1)>0} 1 + \sum_{t\in L:P(\varphi_1)=0} 1 = |L| \tag{8}$$

$$|L|_{P(\varphi_1)>0} + |L|_{P(\varphi_1)=0} = |L| \tag{9}$$

$$|L| = |L| \blacksquare \tag{10}$$

Probabilities defined as above permit the application of the association rule mining measures presented in Section 3.2 over an entire event log. In the light

Table 11: Event log probabilities and measures of a sample of measures for the RCon $(\Diamond\!\!-\mathsf{b} \wedge \Diamond\mathsf{e}) \boxrightarrow (\neg\mathsf{c} \vee \Diamond\mathsf{f})$.

	Event log	$P(\varphi_\alpha, t)$	$P(\varphi_\tau \mid \varphi_\alpha, t)$	$P(\neg\varphi_\tau \mid \varphi_\alpha, t)$	$P(\varphi_\tau \mid \neg\varphi_\alpha, t)$	$P(\neg\varphi_\tau \mid \neg\varphi_\alpha, t)$
$t_1 =$	$\langle$a, b, c, d, f, c, e, c, h$\rangle$	>0	0.83	0.17	0.67	0.33
$t_2 =$	$\langle$b, d, a, f, g, d, e, d$\rangle$	>0	1.00	0.00	1.00	0.00
$t_3 =$	$\langle$a, c, d, b, c, e, f, c$\rangle$	>0	1.00	0.00	0.80	0.20
$t_4 =$	$\langle$b, c, c, e, a$\rangle$	>0	0.50	0.50	1.00	0.00
$t_5 =$	$\langle$b, c, d, e$\rangle$	=0	NaN	NaN	0.75	0.25
	$P(\varphi_\alpha, L)$= 0.80		$P(\varphi_\tau \mid \varphi_\alpha, L)$= 0.67		$P(\varphi_\tau \mid \neg\varphi_\alpha, L)$= 0.13	
	$P(\varphi_\tau, L)$= 0.82		$P(\neg\varphi_\tau \mid \varphi_\alpha, L)$= 0.15		$P(\neg\varphi_\tau \mid \neg\varphi_\alpha, L)$= 0.05	
	Support: 0.67		Confidence: 0.83		Specificity: 0.25	Lift: 1.02

of the contingency table in Table 10, it suffices to map

$$P(A) \text{ to } P(\varphi_\alpha, L) = \frac{\sum_{t \in L : P(\varphi_\alpha, t) > 0} P(\varphi_\alpha, t)}{|L|},$$

$$P(B) \text{ to } P(\varphi_\tau, L) = \frac{\sum_{t \in L : P(\varphi_\alpha, t) > 0} P(\varphi_\tau \mid \varphi_\alpha, t) + \sum_{t \in L : P(\varphi_\alpha, t) = 0} P(\varphi_\tau \mid \neg\varphi_\alpha, t)}{|L|}, \text{ and}$$

$$P(AB) \text{ to } P(\varphi_\tau \mid \varphi_\alpha, L) = \frac{\sum_{t \in L : P(\varphi_\alpha, t) > 0} P(\varphi_\tau \mid \varphi_\alpha, t)}{|L|}.$$

We remark that the non-trivial mapping from $P(AB)$ to $P(\varphi_\tau \mid \varphi_\alpha, L)$ is intuitively rooted into the inherent nature of "IF A THEN B" rules such as the RCons, as evidenced in [56], and its soundness is evidenced by Theorem 4.1. For example, Table 11 shows a few measures computed from the probabilities of the RCon $(\Diamond\!\!-\mathsf{b} \wedge \Diamond\mathsf{e}) \boxrightarrow (\neg\mathsf{c} \vee \Diamond\mathsf{f})$ over a log composed of five traces. We remark that this result is distinct from the mere aggregation of trace measures. For example, comparing the average of the Support values in Table 9 (0.44) and the Support value presented in Table 11 (0.67), we observe that the former is the average proportion of events in a trace satisfying both the target and the activator, while the latter represents the proportion of traces of the event log satisfying the RCon.

4.4. Measurement system

Given an event log L, a set of RCons R, and a set of probabilistic measures M as input, our framework returns the measurement of every measure in M for each constraint in R both over every single trace $t \in L$ and over the entire event log L. More specifically, the output can be reported at three different levels of detail:

Event level: distinct evaluation of φ_α and φ_τ of each constraint in R on every event of every trace in L;

Trace level: measurement of each measure in M for each constraint in R for every trace in L;

Aggregated view: descriptive statistics over the event log of all the trace-level measures.

Event log level: measurement of every measure in M for every constraint in R for the entire event log L.

For example, Table 9 shows some trace level measures together with their descriptive statistics and Table 11 shows the corresponding event log level measures for the RCon $(\Diamond\!\!\!-\mathsf{b} \wedge \Diamond \mathsf{e}) \,\square\!\!\!\rightarrow\, (\neg \mathsf{c} \vee \Diamond \mathsf{f})$. Since being able to perceive the overall status of a constraint is as important as the possibility to analyze its details in single traces, we report the entire statistical distribution of a measure across the event log to provide a complete information spectrum.

Figure 1 depicts the pipeline of the framework from the input to the output. In the first stage, an RCon is evaluated on each trace of the event log. Then, the evaluation result is used to compute the probabilities of the rule. On top of them, the measures of the rule in each trace and in the entire event log are computed. Also, descriptive statistics over the event log are reported for each trace measure.

We remark that the design of the RCons is crucial for the evaluation and the computation of the measures especially in terms of definition of their activator. Let us take as an example the RespondedExistence(a, b) constraint from the

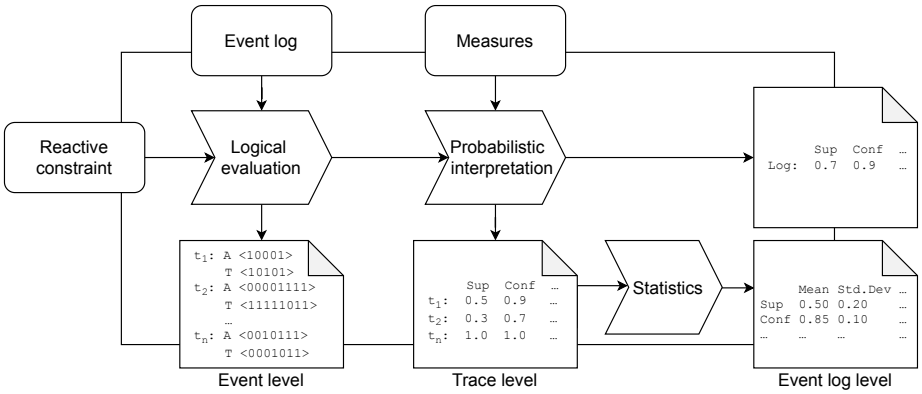

Figure 1: Measurement framework architecture.

repertoire of DECLARE (see Table 1). The classical LTL$_f$ formula underlying RESPONDEDEXISTENCE(a, b) for whole-trace evaluations is $\neg\Diamond \mathsf{a} \vee \Diamond \mathsf{b}$ [28]. However, the formulation of the rule as an RCon can lead to different interpretations:

$\mathsf{a} \boxdot\!\!\rightarrow (\Diamond \mathsf{b} \vee \Diamond\!\!\!-\, \mathsf{b})$: If a occurs, b is expected to occur somewhere in the trace;

$(\Diamond \mathsf{a} \vee \Diamond\!\!\!-\, \mathsf{a}) \boxdot\!\!\rightarrow (\Diamond \mathsf{b} \vee \Diamond\!\!\!-\, \mathsf{b})$: For every event in the trace such that a occurs either in the past or in the future, also b should occur somewhere in the trace;

$\mathit{True} \boxdot\!\!\rightarrow \neg(\Diamond \mathsf{a} \vee \Diamond\!\!\!-\, \mathsf{a}) \vee (\Diamond \mathsf{b} \vee \Diamond\!\!\!-\, \mathsf{b})$: For every event in the trace, if a occurs in the trace, also b is expected to occur;

$t_{\text{Start}} \boxdot\!\!\rightarrow (\neg\Diamond \mathsf{a} \vee \Diamond \mathsf{b})$: At the beginning of the trace, if a occurs in the trace, also b should occur.

All the formulations above are legitimate as they entail that the occurrence of a in the trace demands the occurrence of b. However, the difference in the way the activator is represented turns out to be crucial. The activator, indeed, encodes when the rule is of interest. For example: Are we interested in each occurrence of task a or only in its eventual occurrence in the trace? Do we want the rule to be satisfied in every point of the trace or just at the beginning of the trace? These choices have a clear impact on the measures. Table 12 presents the evaluation of

Table 12: Measurements of a constraint expressed with different formulations on trace $\langle \mathsf{d}, \mathsf{a}, \mathsf{b}, \mathsf{c}, \mathsf{a} \rangle$.

RCon formulation	**Evaluation**	**Support** $P(\varphi_\alpha \varphi_\tau, t)$	**Confidence** $P(\varphi_\tau \mid \varphi_\alpha, t)$
$\mathsf{a} \boxdot\!\!\rightarrow (\Diamond \mathsf{b} \vee \Diamond\!\!\!-\, \mathsf{b})$	φ_α: $\langle 0, 1, 0, 0, 1 \rangle$ φ_τ: $\langle 1, 1, 1, 1, 1 \rangle$	$2/5 = 0.4$	$2/2 = 1$
$(\Diamond \mathsf{a} \vee \Diamond\!\!\!-\, \mathsf{a}) \boxdot\!\!\rightarrow (\Diamond \mathsf{b} \vee \Diamond\!\!\!-\, \mathsf{b})$	φ_α: $\langle 1, 1, 1, 1, 1 \rangle$ φ_τ: $\langle 1, 1, 1, 1, 1 \rangle$	$5/5 = 1$	$5/5 = 1$
$\mathit{True} \boxdot\!\!\rightarrow \neg(\Diamond \mathsf{a} \vee \Diamond\!\!\!-\, \mathsf{a}) \vee (\Diamond \mathsf{b} \vee \Diamond\!\!\!-\, \mathsf{b})$	φ_α: $\langle 1, 1, 1, 1, 1 \rangle$ φ_τ: $\langle 1, 1, 1, 1, 1 \rangle$	$5/5 = 1$	$5/5 = 1$
$t_{\text{Start}} \boxdot\!\!\rightarrow (\neg\Diamond \mathsf{a} \vee \Diamond \mathsf{b})$	φ_α: $\langle 1, 0, 0, 0, 0 \rangle$ φ_τ: $\langle 1, 1, 1, 0, 0 \rangle$	$1/5 = 0.2$	$1/1 = 1$

a trace with the different formulations seen above and their trace measurements for Confidence and Support. Confidence is equal to 1 as each time the activator holds true, also the target holds true. Support (i.e., the frequency of $\varphi_\alpha \cap \varphi_\tau$) varies considerably instead. Notice that this phenomenon comes with neither a good nor with a bad connotation, but stresses the idea that a full control over the formula implies a mindful decision about its design and subsequently on picking the right measures for it. In Table 1, we devised the RCon formulae based on the activators of DECLARE templates described in [3, 20] – hence, e.g., the choice for RESPONDEDEXISTENCE of the first option presented in Table 12. While DECLARE templates are reasonably simple and well-known standard cases, encoding the right activator is crucial for the design of custom RCons and their measures. Lastly, we would like to remark that all the variants in Table 12 take exactly the same amount of computational time to be checked as any formula requires a trace to be read only twice, as described in Section 4.1.

In summary, we have described in this section a novel measurement framework for reactive temporal specifications based on LTLp$_f$, supporting probabilistic interestingness measures at both trace and event log levels. The framework is designed to be suitable for any custom formula in the form of a Reactive Constraint, and any measure that is based on the probability of the activator

and target of the constraints. Therefore, it supports template sets like DECLARE and all the interestingness measures from association rule mining seen in Table 3, though not being limited to them. Next, we evaluate our approach through tests conducted with our implemented prototype.

5. Implementation and performance analysis

We have implemented our measurement framework as a proof-of-concept software prototype built upon the existing declarative process specification processor tool Janus [14, 16]. The Java source-code can be found at `github.com/Oneiroe/Janus`. The core component of the software is the RCons verification engine, upon which are build independently a declarative process discovery module and the present declarative rules measurement module. All the process specifications used in the following experiments are discovered with this discovery module implementing the technique presented in [14]. In the remainder of this section, we first report on the results of a time and space analysis with simulated data. Then, we investigate the computational performance on real-world event log data sets. The results demonstrate the practical feasibility and applicability of our approach.

5.1. Time analysis

To assess the efficiency of our implemented technique, we measure its time performance against an increase in the data size (i.e., the cardinality of the event log and the length of its traces) and the model size (i.e., the number of rules) with synthetic event logs.

We repeated every experiment 10 times to smooth random factors. The reported results average over the ones of the single repetitions. The machine used for the experiments was equipped with an Intel Core i5-7300U CPU at 2.60 GHz, quad-core, 16 Gb of RAM and an Ubuntu 18.04 LTS operating system.

To test the response of our implemented framework against the input data size, we set up a controlled experiment in which we first generate logs of varying

Table 13: The set of DECLARE rules used in the experiments.

INIT(a)	RESPONSE(e, f)	CHAINRESPONSE(o, p)
END(b)	PRECEDENCE(g, h)	CHAINPRECEDENCE(q, r)
ATMOSTONE(c)	ALTERNATEPRECEDENCE(i, l)	RESPONDEDEXISTENCE(s, t)
PARTICIPATION(d)	ALTERNATERESPONSE(m, n)	COEXISTENCE(u, v)
SUCCESSION(w, x)	ALTERNATESUCCESSION(y, z)	CHAINSUCCESSION(j, k)
NOTCOEXISTENCE(0, 1)		

sizes that are compliant with a fixed set of rules, resorting to the simulation engine of MINERful [57]. Thereupon, we compute the measures listed in Table 3 against all the rules of a larger test specification (not fully compliant with the event log). For every run, we recorded the wall-clock time of our prototype.

The starting set of rules stems from the DECLARE repertoire of templates [5] and is provided in Table 13. Notice that the set contains all the rule templates seen in Table 1 and is designed in a way that every constraint insists on different tasks. The test model consists of 649 constraints extracted by the discovery algorithm of Janus (setting the Support and Confidence threshold parameters to 0.05 and 0.8) from a synthetic event log of 834 963 events, 500 traces and tasks in $\{a, \ldots, z, 0, 1\}$ that is compliant with the initial model.[2] Given the test specification obtained as described above, we performed two tests of 65 iterations each, based on synthetic event logs that comply with the rules of Table 13, by: (1) increasing the length of the traces (with a step of 100 events per iteration, keeping the number of traces per event log equal to 500), and (2) increasing the number of traces in the event log (with a step of 50 new traces per iteration, keeping the trace lengths between 900 and 1000 events). Figure 2 illustrates the results of both experiments. We observe that the factor actually influencing the wall-clock time is the total amount of events rather than the trace length:

[2] Available at https://oneiroe.github.io/DeclarativeMeasurements-static

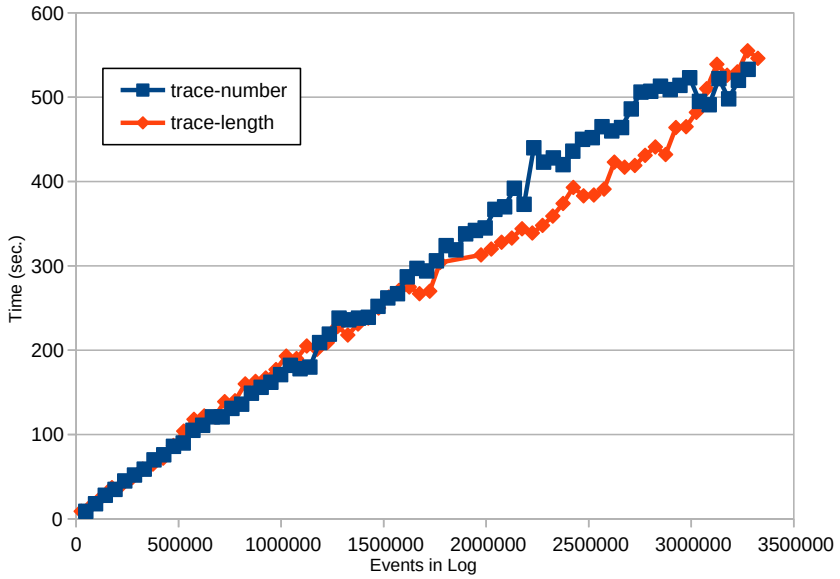

Figure 2: The computation time is linearly dependent on the total number of events in the event log.

indeed, Fig. 2 shows that the recorded timings of both experiments tend to lie on the same line. This experimental result confirms the linear relation between the total number of events in the log and the computational performance illustrated in Section 4.1.

Next, we investigate the response of the framework to an increase in the model size. To do so, we first generate an event log containing 1000 traces with a trace length between 100 and 500 events from the simulation of the rules in Table 13. Thereupon, we use the discovery algorithm of Janus to automatically retrieve different test models with varying levels of compliance. To that extent, we make the Confidence threshold range from 1.0 (full model compliance), down to 0.0 with a step of 0.05. The rationale is, the lower the Confidence threshold, the higher the number of constraints in the test model. Then, we calculate all the measures in Table 3 for every constraint of each test model. The time taken for the measurements are shown in Fig. 3. Notice that the computation time is linearly dependent on the number of rules to check, thus in line with the

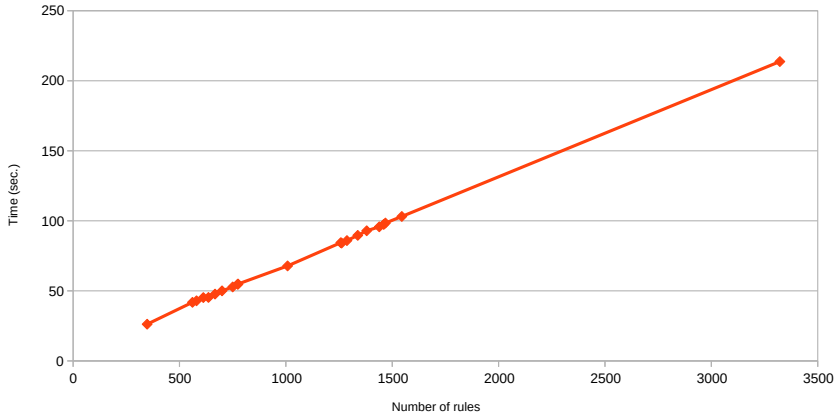

Figure 3: The computation time is linearly dependent on the total number of rules to check.

theoretical computational cost exposed in Section 4.1.

5.2. Space analysis

The space consumption of our technique depends on the data structures required to store the multi-level results. More specifically, four multidimensional matrices are used, containing respectively (1) the evaluation at the event level, (2) the measures at the trace level, (3) their descriptive statistics over the log, and finally (4) the measures at the event log level. Considering $|L|$ the number of traces in the log, $|E|$ the total number of events in a log, $|R|$ the number of constraints, $|M|$ the number of measures, the sizes of the matrices are respectively the following:

1) Events evaluation: $|E| \times |R|$, wherein each cell contains two boolean values (i.e., the evaluation of the activator and target of the constraint);

2) Trace measures: $|L| \times |R| \times |M|$, having a real number in every cell;

3) Trace measures statistics: $|R| \times |M|$, containing seven real numbers per cell (for the mean, geometric mean, variance, population variance, standard deviation, maximum value, and minimum value, respectively);

4) Event log measures: $|R| \times |M|$, with a real number each.

The events matrix is optimized as a bit matrix, where two bits are sufficient to store the boolean results of the evaluation of both the activator and target for one event. We implemented our framework in Java, so we employ 1-byte `Byte` objects and 4-byte `Float` numbers (6 decimal digits are sufficiently accurate for our purpose). Taking these indicators into account, we can estimate the space consumption. For example, assuming that $|L| = 1000$, the maximum number of events in a trace n is 50, $|R| = 100$, $|M| = 30$, the expectation for the space demands are distributed as follows:

1) Events evaluation: $1000 \times 20 \times 260 \times 1 = 5\,200\,000$ bit $= 5.2$ Mb;

2) Trace measures: $1000 \times 260 \times 37 \times 4 = 38\,480\,000$ bit $= 38.48$ Mb;

3) Trace measures statistics: $260 \times 37 \times 7 \times 4 = 269\,360$ bit $= 269.36$ Kb;

4) Event log measures: $260 \times 37 \times 4 = 38\,480$ bit $= 38.48$ Kb.

Therefore, the most memory-demanding data structures are those that pertain to the events evaluation and the trace measurements matrices. The former is bigger than the latter only if the average number of events per trace is greater than 4 times the number of measures used, i.e., $\frac{|E|}{|L|} > 4 \times |M|$. In our experiments, even using all the 37 measures of Table 3, this has not occurred.

As with the experiments for the evaluation of time, we analyze empirically the space consumption through simulations, controlling the number of events n per trace, the number of traces $|L|$, and the number of constraints under analysis $|M|$. To measure the memory consumed by the data structures, we perform a Bit serialization of the matrix objects listed above. This allows us to have a precise measure of the space consumed by every object, though it unavoidably requires that the available memory is twice as much as the strictly necessary amount.

Figure 4 illustrates the results of our experiments. As it can be seen, the resulting linear trends are in line with the expectations, modulo the constant factors introduced by the Java Virtual Machine objects. It can be noticed that the number of constraints to check, being a common factor among all the objects,

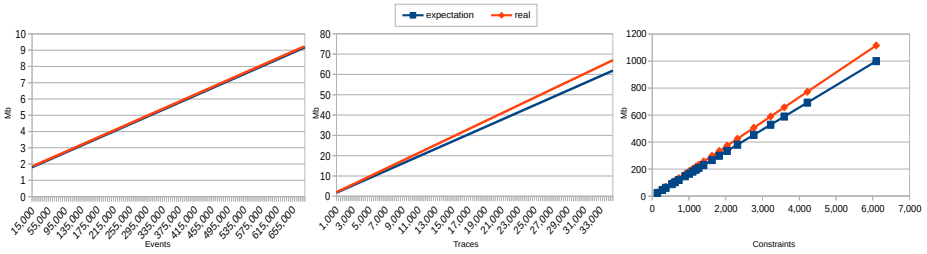

Figure 4: The space consumption is linearly dependent on the total number of events in the log (left), number of traces in the log (center), and number of rules to check (right)

increases the overall required memory quicker than the other parameters. We remark that depending on the desired outcome, not all the measures nor all the matrices are necessary. For example, if the log measures are desired, the trace measures and their statistics can be ignored and vice-versa. The events evaluation is the only mandatory object upon which all other computations are based.

At present, our implementation works in-memory, thus it is assumed that all the objects fit in main memory (which proved to be sufficient for all the real life log under analysis). However, we would remark that every measure calculation (both at log and trace levels) is independent from the other one, thus it is possible to either *(i)* compute one measure or constraint at the time in pipeline, in order to reduce the memory load, or *(ii)* to distribute the workload, by making each system compute independently one measure per constraint. Both are interesting directions for the future upgrades of our implementation.

5.3. Application on real-world event logs

To test the performance also in real settings, we compute the rule measures on 13 event logs, whose characteristics are exposed in Table 14. Twelve of those event logs are openly available[3] and belong to the Business Process Intelligence Challange (BPIC) collection, a Road-Traffic Fines Management Process (RTFMP) and the aforementioned Sepsis event log. In addition, we

[3] https://data.4tu.nl/

analyze the performance of our prototype on an event log stemming from a partner of a smart-city project in which the authors are involved (labeled as "Smart city" in Table 14). We included the Smart city event log due to its considerable size: as it can be noticed from the table, it is the one bearing the largest amount of events in this experiment.

For each log, we ran the discovery algorithm of Janus [14] in order to extract a test model to check the event log against. We tuned the parameters of the discovery algorithm to obtain a set of rules to which the event log complies for the most part (Confidence threshold of 0.8), even though the constrained tasks are possibly infrequently co-occurring (Support threshold of 0.05). Table 14 illustrates the results. For each event log we report, along with the number of traces, the occurring tasks, the events, the number of constraints in the test model, the total time from the launch to the termination of the software ("Time"), the time to evaluate the rules on events ("Checks"), the time to compute the measures both at the trace and log levels ("Measures"), and the total space consumed by the data structures of our tool ("Space") against their expected value ("Expectation"). We remark that the space consumption is consistent with our theoretical expectation and that the wall-clock time remains within acceptable ranges as the slowest run takes around 4.5 minutes to check about 600 constraints in a considerably big event log such as BPIC17 [58] handling around 2.5 Gb of data.

5.4. Analysis of custom rules

In order to demonstrate that our framework can handle any Reactive Constraints, beyond the standard DECLARE repertoire, we applied our approach to compute the discussed measures of a custom rule on the Sepsis event log. We name the custom rule BIDIRECTIONALTIMECONSEQUENT(a, b, c) as its RCon formulation is $\mathsf{a} \boxdot\!\!\rightarrow \Diamond \mathsf{b} \wedge \Diamond\!\!\!-\, \mathsf{c}$. It states that if a occurs, it is expected that either c occurred before it or b will occurs afterwards. Table 15 reports the measures at log and trace level for BIDIRECTIONALTIMECONSEQUENT(Admission NC, CRP, IV Liquid) calculated on the Sepsis real-life log [34]. As it can be noticed, Confidence and

Table 14: Performance records on real-life datasets.

Event log	Traces	Tasks	Events	Rules	Time [sec]	Checks [msec]	Measures [msec]	Space [Mb]	Expectation [Mb]
BPIC12 [59]	13087	36	262200	519	106.7	19 237.1	83 160.6	1222.76	1141.93
BPIC13_cp [60]	1487	7	6660	20	1.2	221.2	338.3	5.10	4.56
BPIC13_i [60]	7554	13	65533	14	3.6	584.9	1064.3	18.09	16.59
BPIC14_f [61]	41353	9	369485	51	29	3977.1	21 277.4	357.10	331.04
BPIC15_1f [62]	902	70	21656	3856	42.2	13 686	25 974.4	670.45	602.83
BPIC15_2f [62]	681	82	24678	5889	57.8	19 790.6	35 029.5	832.38	745.84
BPIC15_3f [62]	1369	62	43786	4098	81.2	26 468	51 764.6	1105.65	1014.59
BPIC15_4f [62]	860	65	29403	4690	60.4	18 997.8	38 492.4	818.55	740.39
BPIC15_5f [62]	975	74	30030	5164	69.7	22 947.1	43 684.2	999.51	906.35
BPIC17_f [58]	21861	41	714198	611	275.9	58 909.8	207 182.2	2561.93	2413.94
RTFMP [63]	150370	11	561470	49	80.1	7887.5	65 214	1210.04	1118.05
Sepsis [34]	1050	16	15214	260	3.8	765.8	1842.6	49.80	44.67
Smart city [64]	4347	20	692333	292	61.5	23 261.8	32 301.8	398.74	390.37

Recall are relatively high (0.82 and 0.79, respectively) and the values of Coverage and Prevalence (0.76 and 0.79, respectively) suggest a frequent occurrence of activator and target. The value of Lift is greater than 1, which denotes dependency between activator and target (especially at a trace level). The detailed results of the evaluation on each trace can be found at `oneiroe.github.io/DeclarativeMeasurements-static`.

The capability of our framework to handle non-standard rules opens up new possibilities for the claimed extendibility of DECLARE as a declarative specification language, claimed from its very inception to be open to customization through the definition of new rules according to the process analyst needs [65].

6. Evaluation

In this section, we report on experiments that show interesting implications of having a vast availability of measures with customization options. Specifically, Section 6.1 investigates over which measures can be of interest in the scope of declarative specification discovery, and Section 6.2 shows how the properties of measures can be exploited to characterize the alterations of constraints when the underlying process changes.

Table 15: Measures resulting from the evaluation of constraint BidirectionalTimeConsequent(Admission NC, CRP, IV Liquid) on the Sepsis event log [34].

Measure	Log	Trace	Measure	Log	Trace	Measure	Log	Trace
Support	0.62	0.06	Prevalence	0.79	0.66	One-way Support	0.04	0.51
Confidence	0.82	0.82	Added Value	0.03	0.18	Two-way Support	0.03	0.05
Recall	0.79	0.09	Relative Risk	1.17	1.36	Two-Way Support Variation	0.01	0.02
Lovinger	-0.16	0.05	Jaccard	0.67	0.09	Linear Correlation Coefficient	0.13	0.13
Specificity	0.30	0.36	Ylue Q	0.32	0.63	Piatetsky-Shapiro	0.02	0.01
Accuracy	0.70	0.39	Ylue Y	0.17	0.63	Cosine	0.80	0.30
Lift	1.04	1.24	Klosgen	0.02	0.07	Information Gain	0.04	0.30
Leverage	0.22	0.76	Conviction	1.16	Infinity	Sebag-Schoenauer	4.56	Infinity
Compliance	0.86	0.98	Interestingness Weighting Dependency	0.03	0.01	Least Contradiction	0.62	0.00
Odds Ratio	1.95	Infinity	Collective Strength	6.47	Infinity	Odd Multiplier	1.20	Infinity
Gini Index	0.01	0.03	Laplace Correction	0.82	0.60	Example and Counterexample Rate	0.78	0.81
Certainty factor	0.14	0.67	J Measure	0.00	0.01	Zhang	0.17	0.63
Coverage	0.76	0.07						

All the experimental data (code, input data, results) can be found at https://oneiroe.github.io/DeclarativeMeasurements-static. In the following experiments, we resort to the following tools: *(i)* The Janus discovery algorithm [14] for the discovery of declarative models from events logs; *(ii)* The simulation engine of MINERful [57] for the generation of event logs complying with given declarative specifications; *(iii)* The error injection engine of MINERful [66] for the controlled insertion of noise into event logs; *(iv)* The declarative model simplification technique of MINERful [33] for the removal of redundancies from declarative specifications.

6.1. Ranking experiment

The objectives of this experiment are the following: (1) Empirically showing that relying on more measures than the sole Support and Confidence measures is effective to characterize process rules in an event log; (2) Highlighting insightful measures in a declarative process mining context.

To achieve both objectives we rank the measures according to how many correct and interesting rules they are able to recognize. We take inspiration from

the *correct rules at N* experiment introduced by the seminal work of Le and Lo [6]. Given an event log L, a ground truth set of rules R_G satisfied in L, a set of rules $R_D \supset R_G$ containing also loosely satisfied rules in L, the set of interestingness measures M of Table 3, and a predefined threshold N, we compute the value of each measure $m \in M$ for all the rules $r \in R_D$ on L. Then, for every measure $m \in M$ we create sets of rules that are associated to a common value and sort those sets accordingly. This leads to a separate sorting of the rules for every measure. If, for instance, rules r_1 and r_2 have a Confidence of 1.0, rule r_3 has a Confidence of 0.9, and rules r_4, r_5 and r_6 have a Confidence of 0.8, the top-N sets are: $\{r_1, r_2\}$ in the top-1, $\{r_1, r_2, r_3\}$ in the top-2, and $r_1, r_2, r_3, r_4, r_5, r_6$ in the top-3 for Confidence. Intuitively, a good measure should assign high scores to correct rules. Therefore, we finally count how many of the rules in R_G are within the top-N sets. We repeated the experiment 10 times and considered the average of the results to avoid fluctuations caused by the random factors of simulation. We performed the experiment with N set to 1, 5, 10, 25, 50, 100, 200, 500, 1000, and 1500, i.e., ranging from considering only the best-scoring rules to considering all the rules in R_D. The final ranking of a measure is computed as the average of its ranking for each N. Table 16(a) shows the final rankings for this experiment. Together with the ranking, for each measure we report also the number of correct rules ("Correct" column) and the average ratio of correct rules over the total number ("Ratio" column) in the top-N sets, averaged over the 10 repetitions of the experiment.

We run our experiments with three different event logs: *(i)* a simulation of a synthetic process specification (Section 6.1.1); *(ii)* a simulation of a synthetic process specification with random changes in the event log so as to mimic partial non-compliance (Section 6.1.2); *(iii)* a real-life event log (Section 6.1.3). At the end of this section, we draw some conclusions from the obtained results.

6.1.1. Process simulation

More specifically, we simulate the specification described in Table 13. Notice that the rules are designed to not interfere with one another and each of them

constrains different tasks. The simulation produces an event log that is fully compliant with the rules in Table 13. From the simulated event log, we discover a new process specification with loose bounds (Support and Confidence thresholds set to 0.05 and 0.5, respectively) in order to discover also infrequent and seldom violated rules. We simplify the resulting set of rules by removing those that do not match yet strictly subsume or are entailed by the ground-truth rules, in order to avoid misleading results – for example, if ChainResponse(o, p) and Response(o, p) both belong to the returned set of rules, the former is retained and the latter is removed because ChainResponse(o, p) is part of the ground-truth specification (see Table 13) and is subsumed by Response(o, p). The full detail of the technique that deals with the removal of redundant rules is out of scope for this paper. The interested reader can find a detailed description of the problem and the approach in [33]. The simplified discovered model consists of 1310 rules on average. Thereupon, we apply our measurement framework to compute the measures in Table 3 at the event log level for all the discovered rules. Finally, we sort the rules according to each measure, and rank the measures according to how many of the original rules are among in top N sets.

Notice that Support ranks only sixteenth. Confidence, by contrast, is at the top of the ranking. It should be observed, however, that the experiment considers by design never violated rules (i.e., those with maximal Confidence), hence the top position of this measure. Nevertheless, there are two measures that match Confidence, namely *(i)* Example and Counterexample Rate and *(ii)* the Sebag-Schoenauer measure. This is in accordance with **P5**, as the absence of counterexamples for the rules makes measures with known maximal values highly rank the original, correct rules. The "Ratio" column reported in Table 16(a) helps to distinguish the accuracy of the results. For example, Odd Multiplier and Odd Ratio have the same rank (i.e., the same amount of correct rules identified), although the Odd Ratio returns 3 times more rules than the Odd Multiplier within the top-N sets.

Table 16: Ranking of measures according to the simulation experiment with a fully compliant simulated event log (left), with an altered event log (center), and with a real-life event log (right).

Rank	Measure	Correct	Ratio
1	Confidence	11.00	24.79%
1	Example and Counterexample Rate	11.00	24.79%
1	Sebag-Schoenauer	11.00	24.81%
4	Laplace Correction	8.65	17.70%
5	Accuracy	6.62	10.69%
6	Least Contradiction	6.35	10.55%
7	Cosine	6.24	10.51%
8	Jaccard	6.23	10.51%
8	Prevalence	6.15	10.17%
10	Conviction	7.00	24.90%
10	Odd Multiplier	7.00	22.24%
10	Odds Ratio	7.00	7.57%
10	Ylue Q	7.00	7.57%
10	Ylue Y	7.00	7.57%
15	Support	5.18	6.15%
16	Lift	4.94	9.55%
17	Certainty factor	4.48	3.66%
18	One-way Support	4.39	3.24%
18	Zhang	4.35	4.15%
20	Two-way Support	4.26	1.79%
21	Interestingness Weighting Dependency	4.27	1.79%
22	Added Value	4.25	1.95%
22	Klosgen	4.24	2.25%
24	Piatetsky-Shapiro	4.19	1.65%
25	Recall	4.64	3.69%
26	Information Gain	4.33	3.91%
27	Linear Correlation Coefficient	3.92	2.44%
28	Leverage	3.40	1.20%
29	Coverage	3.24	0.81%
30	Specificity	2.68	0.38%
31	Gini Index	3.01	0.84%
31	Lovinger	2.23	0.32%
33	Relative Risk	2.48	0.33%
34	Collective Strength	2.02	0.24%
35	J Measure	0.00	0.00%
35	Two-Way Support Variation	0.00	0.00%

(a) Simulation

Rank	Measure	Correct	Ratio
1	Confidence	5.58	6.21%
1	Example and Counterexample Rate	5.58	6.21%
3	Laplace Correction	5.53	6.72%
4	Odd Multiplier	5.42	11.16%
5	Conviction	5.43	11.20%
6	Lift	4.58	11.38%
7	Prevalence	4.42	4.31%
8	Accuracy	4.61	4.06%
9	Least Contradiction	4.31	3.98%
10	Recall	4.20	0.51%
11	Jaccard	4.22	3.97%
12	Cosine	4.18	3.97%
13	One-way Support	4.09	3.12%
14	Certainty factor	4.50	2.95%
15	Added Value	4.01	2.21%
16	Information Gain	4.05	3.40%
17	Support	3.85	3.88%
18	Zhang	4.44	2.77%
19	Two-way Support	4.05	1.99%
20	Klosgen	4.02	2.02%
21	Interestingness Weighting Dependency	4.04	1.80%
22	Piatetsky-Shapiro	3.94	1.73%
23	Sebag-Schoenauer	3.30	3.19%
24	Coverage	3.28	0.88%
25	Ylue Q	2.99	0.46%
25	Ylue Y	2.99	0.46%
27	Leverage	3.28	0.97%
28	Two-Way Support Variation	3.96	2.70%
29	Lovinger	3.87	0.93%
30	Linear Correlation Coefficient	3.58	1.38%
31	Gini Index	3.00	0.83%
32	J Measure	2.77	0.78%
33	Specificity	2.14	0.28%
34	Odds Ratio	1.94	0.24%
35	Collective Strength	1.98	0.23%
36	Relative Risk	2.06	0.25%

(b) Simulation with noise

Rank	Measure	Correct	Ratio
1	Recall	77.67	14.64%
2	Confidence	76.67	51.00%
2	Example and Counterexample Rate	76.67	51.00%
4	Sebag-Schoenauer	74.78	50.39%
5	Least Contradiction	70.56	56.43%
6	Cosine	70.44	56.40%
6	Jaccard	70.44	56.40%
8	Odds Ratio	72.33	19.88%
8	Ylue Q	72.33	19.88%
8	Ylue Y	72.33	19.88%
11	Accuracy	70.11	56.03%
12	Laplace Correction	42.89	31.07%
12	Relative Risk	71.33	20.46%
12	Specificity	71.33	20.46%
15	Conviction	65.33	54.59%
16	Odd Multiplier	65.00	54.50%
17	Certainty factor	52.56	36.07%
18	Linear Correlation Coefficient	41.00	37.66%
19	Gini Index	34.78	39.72%
19	One-way Support	30.44	27.55%
21	Interestingness Weighting Dependency	31.67	18.93%
22	Information Gain	29.78	25.97%
22	Piatetsky-Shapiro	30.78	19.64%
24	Zhang	38.33	27.05%
25	Added Value	29.67	17.07%
25	Leverage	29.78	19.62%
27	Prevalence	31.56	26.90%
28	Two-way Support	30.22	16.00%
29	Lift	29.44	25.21%
30	Klosgen	29.56	14.75%
31	Support	28.22	19.04%
32	Coverage	19.89	4.68%
33	Lovinger	27.00	7.44%
34	Collective Strength	20.78	5.81%
35	J Measure	7.22	4.10%
36	Two-Way Support Variation	2.11	1.48%

(c) Real-world based

6.1.2. Process simulation with noise

The previous experiment tests a perfectly compliant setup where the reference rules are never violated. For this reason, we conduct a modified version of the experiment, this time injecting noise in the event log in order to check if the

ranking is preserved in non-optimal situations. Specifically, we randomly delete or duplicate 5% of the occurrences of every task. The results can be found in Table 16(b). It can be seen that also in the presence of partial non-compliance, the measures of Confidence and Example and Counterexample Rate continue to be on top of the ranking, while Sebag-Schoenauer measure drops in the second half of the list, together with Ylue Q, Ylue Y and Odds Ratio. This sudden change due to noise is motivated by the fact that these measures are convex, as we discussed in regard with **P6**, so they rapidly decrease in presence of counterexamples. Support persists in the middle of the ranking, with 17 measures scoring better than it. Lift and Information Gain perform by far better in presence of noise. Understanding how measures react to changes in the data (noise in this case), is key for process drift analysis [67], where the evolution of the process is the main objective. We study more in details the influence of noise on the measurements in Section 6.2.

6.1.3. Real-life event log

We replicate our ranking experiment based on a real-life event log, specifically the Sepsis data-set [34]. In [35], Mannhardt and Blinde illustrate a procedural model discovered with the help of domain experts representing the sepsis treatment process at a hospital. We manually translated the model into DECLARE rules that we use as a ground-truth specification to replicate the ranking experiment. The model consists of 100 rules, listed in Table 17. Because it is a real life event log, we do not inject any noise. Table 16(c) illustrates the results. It can be seen that the measures of Confidence and Example and Counterexample Rate remain at the top of the ranking. Also, while Recall conquers the first position for the amount of correct rules reported, it also returns a high number of other incorrect rules as signaled by its low average ratio score. In this case, Support drops at the bottom of the ranking.

These experiments suggest possible candidate measures to be used for declarative process discovery. We tested different scenarios, thereby showing how the

Table 17: DECLARE model of SEPSIS data-set based on [35].

INIT(ER Registration)	ATMOSTONE(ER Registration)	ATMOSTONE(ER Sepsis Triage)
ATMOSTONE(ER Triage)	SUCCESSION(ER Registration, ER Triage)	SUCCESSION(ER Triage, ER Sepsis Triage)
PRECEDENCE(ER Registration, CRP)	PRECEDENCE(ER Registration, Leucocytes)	PRECEDENCE(ER Registration, LacticAcid)
ATMOSTONE(IV Liquid)	ATMOSTONE(IV Antibiotics)	PRECEDENCE(ER Sepsis Triage, IV Liquid)
PRECEDENCE(ER Sepsis Triage, IV Antibiotics)	COEXISTENCE(IV Liquid, IV Antibiotics)	PRECEDENCE(ER Sepsis Triage, Admission NC)
PRECEDENCE(ER Sepsis Triage, Admission IC)	NOTSUCCESSION(Admission IC, IV Antibiotics)	NOTSUCCESSION(Admission IC, IV Liquid)
NOTSUCCESSION(Admission NC, IV Antibiotics)	NOTSUCCESSION(Admission NC, IV Liquid)	PRECEDENCE(ER Sepsis Triage, Return ER)
ATMOSTONE(Return ER)	NOTSUCCESSION(Return ER, Admission IC)	NOTSUCCESSION(Return ER, Admission NC)
NOTSUCCESSION(Return ER, CRP)	NOTSUCCESSION(Return ER, ER Sepsis Triage)	NOTSUCCESSION(Return ER, IV Antibiotics)
NOTSUCCESSION(Return ER, IV Liquid)	NOTSUCCESSION(Return ER, LacticAcid)	NOTSUCCESSION(Return ER, Leucocytes)
NOTSUCCESSION(Return ER, Release A)	NOTSUCCESSION(Return ER, Release B)	NOTSUCCESSION(Return ER, Release C)
NOTSUCCESSION(Return ER, Release D)	NOTSUCCESSION(Return ER, Release E)	ATMOSTONE(Release A)
ATMOSTONE(Release B)	ATMOSTONE(Release C)	ATMOSTONE(Release D)
ATMOSTONE(Release E)	NOTCOEXISTENCE(Release A, Release B)	NOTCOEXISTENCE(Release A, Release C)
NOTCOEXISTENCE(Release A, Release D)	NOTCOEXISTENCE(Release A, Release E)	NOTCOEXISTENCE(Release B, Release A)
NOTCOEXISTENCE(Release B, Release C)	NOTCOEXISTENCE(Release B, Release D)	NOTCOEXISTENCE(Release B, Release E)
NOTCOEXISTENCE(Release C, Release B)	NOTCOEXISTENCE(Release C, Release A)	NOTCOEXISTENCE(Release C, Release D)
NOTCOEXISTENCE(Release C, Release E)	NOTCOEXISTENCE(Release D, Release A)	NOTCOEXISTENCE(Release D, Release C)
NOTCOEXISTENCE(Release D, Release B)	NOTCOEXISTENCE(Release D, Release E)	NOTCOEXISTENCE(Release E, Release B)
NOTCOEXISTENCE(Release E, Release C)	NOTCOEXISTENCE(Release E, Release D)	NOTCOEXISTENCE(Release E, Release A)
NOTSUCCESSION(Release A, Admission IC)	NOTSUCCESSION(Release A, Admission NC)	NOTSUCCESSION(Release A, CRP)
NOTSUCCESSION(Release A, ER Sepsis Triage)	NOTSUCCESSION(Release A, IV Antibiotics)	NOTSUCCESSION(Release A, IV Liquid)
NOTSUCCESSION(Release A, LacticAcid)	NOTSUCCESSION(Release A, Leucocytes)	NOTSUCCESSION(Release B, Admission IC)
NOTSUCCESSION(Release B, Admission NC)	NOTSUCCESSION(Release B, CRP)	NOTSUCCESSION(Release B, ER Sepsis Triage)
NOTSUCCESSION(Release B, IV Antibiotics)	NOTSUCCESSION(Release B, IV Liquid)	NOTSUCCESSION(Release B, LacticAcid)
NOTSUCCESSION(Release B, Leucocytes)	NOTSUCCESSION(Release C, Admission IC)	NOTSUCCESSION(Release C, Admission NC)
NOTSUCCESSION(Release C, CRP)	NOTSUCCESSION(Release C, ER Sepsis Triage)	NOTSUCCESSION(Release C, IV Antibiotics)
NOTSUCCESSION(Release C, IV Liquid)	NOTSUCCESSION(Release C, LacticAcid)	NOTSUCCESSION(Release C, Leucocytes)
NOTSUCCESSION(Release D, Admission IC)	NOTSUCCESSION(Release D, Admission NC)	NOTSUCCESSION(Release D, CRP)
NOTSUCCESSION(Release D, ER Sepsis Triage)	NOTSUCCESSION(Release D, IV Antibiotics)	NOTSUCCESSION(Release D, IV Liquid)
NOTSUCCESSION(Release D, LacticAcid)	NOTSUCCESSION(Release D, Leucocytes)	NOTSUCCESSION(Release E, Admission IC)
NOTSUCCESSION(Release E, Admission NC)	NOTSUCCESSION(Release E, CRP)	NOTSUCCESSION(Release E, ER Sepsis Triage)
NOTSUCCESSION(Release E, IV Antibiotics)	NOTSUCCESSION(Release E, IV Liquid)	NOTSUCCESSION(Release E, LacticAcid)
NOTSUCCESSION(Release E, Leucocytes)		

measures' scores vary in each of them and how the evaluation of rules can benefit from the perspectives of multiple measures that were not previously available

for temporal specifications. However, we remark that some measures should be handled in a specific manner. For example, Coverage is a descriptive measure reporting on the portion of the event log in which the rule activator occurs: this characteristic determines its low ranking in all the previous experiments. A low score in these experiments does not imply that a measure is of scarce use in general, though. Indeed, we can only highlight the average top ranking measures of these experiments as excellent options, as we do in the next section while depicting the current results. The informed proposal of best practices for the selection of measures and combinations thereof depending on the analysis purposes (e.g., discovery) constitutes a highly interesting outlook for research.

6.2. Sensitivity and resilience to noise

In this section, we study in details the effect of noise injection in the event log on the constraints log measurements. We experimentally observed in Section 6.1.2 that measures exhibit different changes upon the presence of alterations from the expected behavior in an event log. A measure may "sense" the alteration in the data with respect to a rule or remain unaffected. Also, if the alteration is perceived, the magnitude of the measures reaction may be different, in light of the different properties that the measures enjoy, as discussed in Section 3.2. Markedly, we will empirically demonstrate that the properties originally for association rules hold also in the context of temporal rules. The possibility to characterize a constraint evolution is crucial in continuous measurement settings such as streaming analysis [68] or drift analysis [67]. Informed decisions on the measures to monitor based on the characteristic they have and the properties they enjoy is, therefore, key. Providing a set of guidelines to support such decisions is out of scope for this paper and paves the path for future research avenues. Nevertheless, with this experiment, we hope to provide useful preliminary indications in that sense. We also remark that while we call uncommon or unexpected events as "noise" or "error" (reflecting our experimental setting), the same measures evolution would occur in the case of process improvements or changes in the normative aspects the process is subject to.

We conducted the experiment as follows. We took as a reference model the set of rules in Table 13 and we simulated it to generate a clean event log that is compliant wit it. We remark again how the rules are designed to not interfere with one another. In this way, it is possible to observe the response of measures at varying noise levels targeting one constraint at a time, thus limiting the effect of cross-interference. Thereupon, we injected noise in the event log and calculated all the measures in Table 3 for the reference model. In particular, we made use of the following types of noise [66]:

Events insertion: Spurious events are included in the traces (mimicking, e.g., double records, alien events, etc.);

Events deletion: Events are expunged from the event log (mimicking, e.g., missing records, uncommitted transactions, etc.);

White noise: Events are randomly inserted and deleted.

Addressing one rule at a time, we studied (1) the direct effect of noise on that constraint by altering the occurrences of its activator and target via insertions and deletions, and (2) the indirect effect, by altering the occurrences of the other tasks in the event log with white noise. We made the noise spread all over the event log according to a controlled probability variable. For instance, setting the noise injection as the deletion of occurrences of task `a` with a probability of 20 % results in the removal of 20 % of the occurrences of task `a` from the event log, picked at random.

More specifically, for every rule in the set of Table 13, we ran a separate experiment for (i) event insertion noise affecting the activator or (ii) the target, (iii) event deletion noise affecting the activator or (iv) the target, (v) white noise affecting neither the activator nor the target. For each of the combinations above, we let the error-injection probability range from 0 to 100 % with a step of 10 %. Because of the random factor, we repeated each experiment 10 times and recorded the average results.

Figure 5 shows the results of our experiments on constraint Response(`e`, `f`). The Response constraint imposes that the target occurs eventually after every occurrence of the activator. Plotting all the measures together in a static image

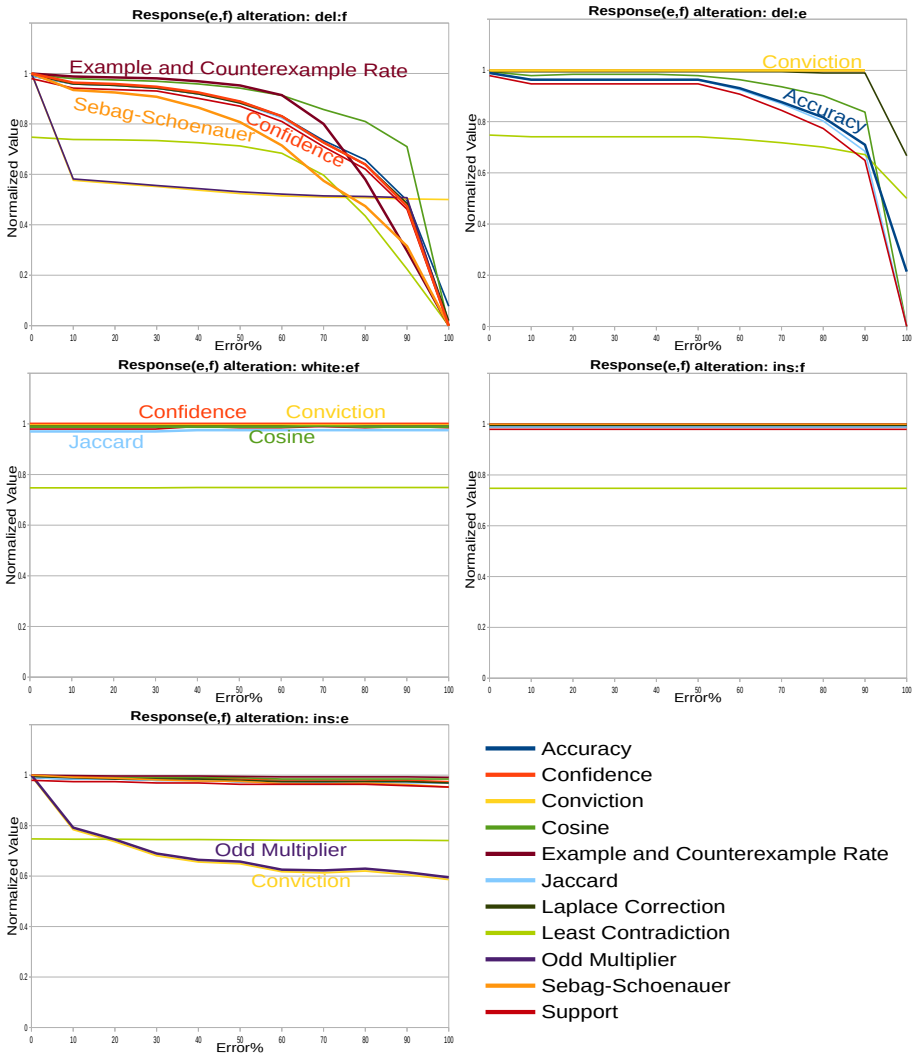

Figure 5: Effect of error injection on constraint Response(e, f) for the best 10 scoring measures from the ranking experiment in Section 6.1.

within the boundaries of a page results in a complex intertwining of lines, hampering the readability of results. Thus we report here only the top 10 measures among the averaged results of the ranking experiment presented in Section 6.1, namely: Confidence, Example and Counterexample Rate, Laplace Correction, Least Contradiction, Accuracy, Cosine, Jaccard, Sebag-Schoenauer, Conviction, and Odd Multiplier plus Support as a baseline reference. Furthermore, as the measures have different ranges, we normalize them between 0 and 1 in order to compare the different trends. The normalization formula used is

$$\frac{\left(\frac{v_m}{1+|v_m|} - \frac{\min_m}{1+|\min_m|}\right)}{\left(\frac{\max_m}{1+|\max_m|} - \frac{\min_m}{1+|\min_m|}\right)}$$

where v_m is the value of measure, $|v_m|$ its absolute value, $\max_m$ and $\min_m$ the maximum and minimum values of the range of a measure, respectively (in case of infinite ranges, the maximum value supported by the software is considered). The full set of plots with all the measures and constraints can be interactively explored at https://github.com/Oneiroe/DeclarativeMeasurements-static.

For the RESPONSE constraint, the selected measures are overall particularly sensitive to the deletions of the target and influenced by both the deletions and insertions of the activator, while mostly insensitive to spurious insertions of the target and white noise. More specifically, we can derive the following observations.

- The deletion of events that satisfy the target leads to more violations of the rule, i.e., lower $P(\varphi_\tau|\varphi_\alpha, L)$ and higher $P(\neg\varphi_\tau|\varphi_\alpha, L)$. The negative effect is thus reflected in the constant decrease of all measures. The rapidity at which this phenomenon occurs is particularly interesting. In accordance with property **P6**, the decrease of concave measures (e.g., Example and Counterexample Rate) is slower than that of linear ones (e.g., Confidence), and convex measures decrease faster than all the others (e.g., Sebag-Schoenauer).
- The deletion of events that satisfy the activator, instead, does not bring further violations, but a higher $P(\neg\varphi_\alpha, L)$. As a consequence, the mea-

sures are mostly stable until the error rate is close to 100 %. As per **P5**, measures with a decreasing trend (e.g., Laplace Correction or Accuracy) are susceptible to the frequency of the rule activators in the event logs, while measures depending only on rule satisfaction (e.g., Conviction and Odd Multiplier) are constant at all the error rates except 100 % (in which case, they are no longer defined due to a zero-by-zero division in their formula).

- The insertion of more events that satisfy the target lead to a higher $P(\varphi_\tau, L)$ but does not influence the constraint satisfactions, as $P(\varphi_\tau|\varphi_\alpha, L)$ remains constant. We remark that the target φ_τ of RESPONSE(e, f) is $\Diamond$f, thus injecting more occurrences of f does imply an increase of $P(\varphi_\tau, L)$ proportional to the error rate. Following from this consideration and in light of **P3**, all the selected measures remain constant.
- The insertion of more events that satisfy the activator is less definite, as the new tasks may bring either to more rule satisfactions, i.e., higher $P(\varphi_\tau|\varphi_\alpha, L)$, or violations, i.e., higher $P(\neg\varphi_\tau|\varphi_\alpha, L)$. As RESPONSE(e, f) requires the occurrence of f eventually after e at any distance in the trace, in this experiment the injections mostly leads to satisfactions. That is why most of the measures show only a slightly decreasing trend. Notably, Conviction and Odd Multiplier measures sense this alteration in a marked way with respect to all the other rules.
- Lastly, the insertion of events not related to both the target and the activation mostly do not alter the measures. The satisfactions and violations remain constant, whilst the only increase is in $P(\neg\varphi_\alpha, L)$. Nevertheless, considering **P1** and **P5**, we can distinguish the fluctuating measures due to the frequencies change (e.g., Cosine and Jaccard) from the unaffected ones (e.g., Confidence and Conviction).

Because of space limits, we cannot illustrate the outcome for the other rules of Table 13. The interested reader can find the full set of experimental data and results at https://oneiroe.github.io/DeclarativeMeasurements-static in a digital interactive format, which is more suitable for data exploration and

browsing.

6.3. Discussion

The applications of our framework presented in Sections 6.1 and 6.2 allows for some reflections on the employment and availability of the measures. First, the measures evolution reflects whether the alteration in the data affects either the target or the activator of the rule (or both). This gives more insights into the analyzed phenomena than knowing whether a rule is satisfied or not at a given moment.

Nevertheless, given a constraint, not all the data alterations are perceived by all measures. This implies that depending on the requirements of the analysis the choice of the measure is crucial. We reported, for example, an experiment to identify measures suitable for discovery. Furthermore, given a certain alteration, it is possible to identify groups of measures with similar trends that focus on the same aspects of a rule, as it can be easily noticed in Fig. 5. Within such groups, it is then possible to select one representative measure.

To this extent, the properties discussed in Section 3.2 are clearly a guiding criterion for the selection of measures, markedly reflected in our experiments in which measures with similar trends could be distinctly identified by the properties they enjoy. Ultimately, this choice is strictly related to the requirements and goals of the event log analysis.

For instance, **P6** turned out to be particularly relevant. While measures may have similar trends, the magnitude of such trends, reflected in the steepness of the curves with which the measure evolves (i.e., concave, convex, or linear), indicates how quickly the measure react to changes in the data. Furthermore, it shows the range of tolerance before the alteration in the data becomes too large to recognize the specific constraint behavior. A resilient measure with a slower decrease is desirable to sense if the fundamental characteristics of a log are still visible despite the deviations (e.g., to implement discovery algorithms that are robust to noise). On the other hand, a sensitive measure with a faster decrease is desirable when exceptions to the rules are only accepted in a very

limited manner (e.g., to monitor normative processes).

An approach enabling such a vast number of measures for temporal specifications is presented in the seminal work of Le and Lo [6]. We extend their investigation along two main lines: first, our approach can handle arbitrary temporal formulae as activator and target, as opposed to single-task variables only (which also restrict the analysis to the sole RESPONSE and PRECEDENCE patterns). Second, our computation of the rule probabilities processes the whole trace at once, while the calculation scheme in [6] relies on a sliding window, whose size has to be manually set up thereby influencing the results.

7. Conclusion

In this paper, we presented a comprehensive measurement framework for declarative specifications modeled as Reactive Constraints. Given an event log and a set of custom probabilistic measures, the framework accepts in input any RCon and returns as output the evaluation of the rule for each event of the log, the computed measures for all the traces and their statistics over the entire event log, and the computed measures over the entire log. The framework goes beyond the current state of the art as it is not limited to a specific set of measures or rules. The experiments conducted reveal the possibility to characterize the behavior of a given constraint through the combination of different measures, which sense differently the behavior recorded in the log. Also, while the choice of the measures to employ is highly context-dependent, we showed how the measures properties can be used to guide the selection, as their effects are clearly visible in the results.

Future work. Different possibilities are now open upon the foundations of this measurement framework. It is possible to exploit the possibility to characterize a phenomenon by studying the evolution of different measures for, e.g., the dynamic recognition of exceptions in process monitoring [69], the identification of process drifts [67] or the analysis of process variants [70]. Also, it can be employed as a post-processing tool for multi-measure filtering of the results of

declarative process discovery techniques like [3, 4, 14].

As measures react differently to diverse stimuli for distinct types of rules, a method to find the best combination of measures depending on the analysis context turns out to be key. To this extent, future research could resort to existing techniques like [71] or develop novel multi-measure heuristics. Also, the measures can be integrated for the assessment of multi-constraint specifications as a whole as in [1, 20].

While the analysis of multiple measures at once may be overwhelming for a human, machine learning techniques could benefit from the availability of the great amount of information returned by the proposed framework, as they are designed to deal with large sets of multidimensional data. Therefore, it seems to be also promisingly exploitable for feature selection tasks in sequence classification [24].

Finally, we observe that the implementation of this framework can largely benefit from run-time optimization for the verification of the rules' automata, particularly for as far as the recognition of permanent violations and satisfactions is concerned [72]. The design and integration of such dedicated techniques serves as an impulse for future research.

Acknowledgments

The authors want to thank Arik Senderovich for the fruitful discussion and useful observations on this research.

The work of Claudio Di Ciccio was partly supported by the Italian Ministry of University and Research (MUR) under grant "Dipartimenti di eccellenza 2018-2022" of the Department of Computer Science at Sapienza University of Rome and by the Sapienza SPECTRA research project.

References

[1] M. de Leoni, F. M. Maggi, W. M. P. van der Aalst, An alignment-based framework to check the conformance of declarative process models and to

preprocess event-log data, Inf. Syst. 47 (2015) 258–277. doi:10.1016/j.is.2013.12.005.

[2] L. T. Ly, F. M. Maggi, M. Montali, S. Rinderle-Ma, W. M. P. van der Aalst, Compliance monitoring in business processes: Functionalities, application, and tool-support, Inf. Syst. 54 (2015) 209–234.

[3] F. M. Maggi, R. P. J. C. Bose, W. M. P. van der Aalst, Efficient discovery of understandable declarative process models from event logs, in: CAiSE, 2012, pp. 270–285. doi:10.1007/978-3-642-31095-9_18.

[4] C. Di Ciccio, M. Mecella, On the discovery of declarative control flows for artful processes, ACM Trans. Management Inf. Syst. 5 (4) (2015) 24:1–24:37. doi:10.1145/2629447.

[5] W. M. P. van der Aalst, M. Pesic, DecSerFlow: Towards a truly declarative service flow language, in: WS-FM, 2006, pp. 1–23. doi:10.1007/11841197_1.

[6] T. B. Le, D. Lo, Beyond support and confidence: Exploring interestingness measures for rule-based specification mining, in: SANER, 2015, pp. 331–340. doi:10.1109/SANER.2015.7081843.

[7] W. Hämäläinen, G. I. Webb, A tutorial on statistically sound pattern discovery, Data Min. Knowl. Discov. 33 (2) (2019) 325–377. doi:10.1007/s10618-018-0590-x.

[8] S. Brin, R. Motwani, C. Silverstein, Beyond market baskets: Generalizing association rules to correlations, in: J. Peckham (Ed.), SIGMOD 1997, Proceedings ACM SIGMOD International Conference on Management of Data, May 13-15, 1997, Tucson, Arizona, USA, ACM Press, 1997, pp. 265–276. doi:10.1145/253260.253327.

[9] G. Piatetsky-Shapiro, Discovery, analysis, and presentation of strong rules, in: G. Piatetsky-Shapiro, W. J. Frawley (Eds.), Knowledge Discovery in Databases, AAAI/MIT Press, 1991, pp. 229–248.

[10] F. B. Galiano, I. J. Blanco, D. Sánchez, M. A. V. Miranda, Measuring the accuracy and interest of association rules: A new framework, Intell. Data Anal. 6 (3) (2002) 221–235.

[11] L. Geng, H. J. Hamilton, Interestingness measures for data mining: A survey, ACM Comput. Surv. 38 (3) (2006) 9. doi:10.1145/1132960.1132963.

[12] C. C. Aggarwal, P. S. Yu, A new framework for itemset generation, in: A. O. Mendelzon, J. Paredaens (Eds.), Proceedings of the Seventeenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, June 1-3, 1998, Seattle, Washington, USA, ACM Press, 1998, pp. 18–24. doi:10.1145/275487.275490.

[13] S. Brin, R. Motwani, J. D. Ullman, S. Tsur, Dynamic itemset counting and implication rules for market basket data, in: J. Peckham (Ed.), SIGMOD 1997, Proceedings ACM SIGMOD International Conference on Management of Data, May 13-15, 1997, Tucson, Arizona, USA, ACM Press, 1997, pp. 255–264. doi:10.1145/253260.253325.

[14] A. Cecconi, C. Di Ciccio, G. De Giacomo, J. Mendling, Interestingness of traces in declarative process mining: The Janus LTLp_f approach, in: BPM, 2018, pp. 121–138. doi:10.1007/978-3-319-98648-7_8.

[15] P. Lenca, P. Meyer, B. Vaillant, S. Lallich, On selecting interestingness measures for association rules: User oriented description and multiple criteria decision aid, Eur. J. Oper. Res. 184 (2) (2008) 610–626. doi:10.1016/j.ejor.2006.10.059.

[16] A. Cecconi, G. De Giacomo, C. Di Ciccio, F. M. Maggi, J. Mendling, A temporal logic-based measurement framework for process mining, in: B. F. van Dongen, M. Montali, M. T. Wynn (Eds.), 2nd International Conference on Process Mining, ICPM 2020, Padua, Italy, October 4-9, 2020, IEEE, 2020, pp. 113–120. doi:10.1109/ICPM49681.2020.00026.

[17] R. Agrawal, R. Srikant, Fast algorithms for mining association rules in large databases, in: J. B. Bocca, M. Jarke, C. Zaniolo (Eds.), VLDB'94, Proceedings of 20th International Conference on Very Large Data Bases, September 12-15, 1994, Santiago de Chile, Chile, Morgan Kaufmann, 1994, pp. 487–499.

[18] S. Debois, T. T. Hildebrandt, P. H. Laursen, K. R. Ulrik, Declarative process mining for DCR graphs, in: SAC, 2017, pp. 759–764. doi:10.1145/3019612.3019622.

[19] W. M. P. van der Aalst, H. T. de Beer, B. F. van Dongen, Process mining and verification of properties: An approach based on temporal logic, in: OTM to Meaningful Internet Systems, 2005, pp. 130–147. doi:10.1007/11575771_11.

[20] A. Burattin, F. M. Maggi, W. M. P. van der Aalst, A. Sperduti, Techniques for a posteriori analysis of declarative processes, in: EDOC, 2012, pp. 41–50. doi:10.1109/EDOC.2012.15.

[21] J. Yang, D. Evans, D. Bhardwaj, T. Bhat, M. Das, Perracotta: mining temporal API rules from imperfect traces, in: ICSE, 2006, pp. 282–291. doi:10.1145/1134285.1134325.

[22] M. Gabel, Z. Su, Online inference and enforcement of temporal properties, in: ICSE, 2010, pp. 15–24. doi:10.1145/1806799.1806806.

[23] C. Lemieux, D. Park, I. Beschastnikh, General LTL specification mining (T), in: ASE, 2015, pp. 81–92. doi:10.1109/ASE.2015.71.

[24] Z. Xing, J. Pei, E. J. Keogh, A brief survey on sequence classification, SIGKDD Explorations 12 (1) (2010) 40–48. doi:10.1145/1882471.1882478.

[25] E. Egho, D. Gay, M. Boullé, N. Voisine, F. Clérot, A user parameter-free approach for mining robust sequential classification rules, Knowl. Inf. Syst. 52 (1) (2017) 53–81. doi:10.1007/s10115-016-1002-4.

[26] J. M. Fowkes, C. Sutton, A subsequence interleaving model for sequential pattern mining, in: B. Krishnapuram, M. Shah, A. J. Smola, C. C. Aggarwal, D. Shen, R. Rastogi (Eds.), Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016, ACM, 2016, pp. 835–844. doi:10.1145/2939672.2939787.

[27] J. D. Smedt, G. Deeva, J. D. Weerdt, Mining behavioral sequence constraints for classification, IEEE Trans. Knowl. Data Eng. 32 (6) (2020) 1130–1142. doi:10.1109/TKDE.2019.2897311.

[28] G. De Giacomo, M. Y. Vardi, Linear temporal logic and linear dynamic logic on finite traces, in: IJCAI, 2013, pp. 854–860.

[29] M. Pesic, D. Bosnacki, W. M. P. van der Aalst, Enacting declarative languages using LTL: avoiding errors and improving performance, in: SPIN, Vol. 6349 of LNCS, Springer, 2010, pp. 146–161.

[30] C. Baier, J.-P. Katoen, K. Guldstrand Larsen, Principles of Model Checking, The MIT Press, 2008.

[31] M. Dumas, M. La Rosa, J. Mendling, H. A. Reijers, Fundamentals of Business Process Management, Second Edition, Springer, 2018. doi:10.1007/978-3-662-56509-4.

[32] G. De Giacomo, R. De Masellis, M. Montali, Reasoning on LTL on finite traces: Insensitivity to infiniteness, in: AAAI Conference on Artificial Intelligence, AAAI Press, 2014, pp. 1027–1033.

[33] C. Di Ciccio, F. M. Maggi, M. Montali, J. Mendling, Resolving inconsistencies and redundancies in declarative process models, Inf. Syst. 64 (2017) 425–446. doi:10.1016/j.is.2016.09.005.

[34] F. Mannhardt, Sepsis cases - event log (Dec 2016). doi:10.4121/uuid:915d2bfb-7e84-49ad-a286-dc35f063a460.

[35] F. Mannhardt, D. Blinde, Analyzing the trajectories of patients with sepsis using process mining, in: J. Gulden, S. Nurcan, I. Reinhartz-Berger, W. Guédria, P. Bera, S. Guerreiro, M. Fellmann, M. Weidlich (Eds.), BPMDS, Vol. 1859 of CEUR Workshop Proceedings, CEUR-WS.org, 2017, pp. 72–80.

[36] M. Ohsaki, S. Kitaguchi, K. Okamoto, H. Yokoi, T. Yamaguchi, Evaluation of rule interestingness measures with a clinical dataset on hepatitis, in: J. Boulicaut, F. Esposito, F. Giannotti, D. Pedreschi (Eds.), PKDD, Vol. 3202 of Lecture Notes in Computer Science, Springer, 2004, pp. 362–373. doi:10.1007/978-3-540-30116-5_34.

[37] P.-N. Tan, V. Kumar, J. Srivastava, Selecting the right objective measure for association analysis, Information Systems 29 (4) (2004) 293 – 313, knowledge Discovery and Data Mining (KDD 2002). doi:https://doi.org/10.1016/S0306-4379(03)00072-3.

[38] C. L. Sistrom, C. W. Garvan, Proportions, odds, and risk, Radiology 230 (1) (2004) 12–19, pMID: 14695382. doi:10.1148/radiol.2301031028.

[39] C. J. van Rijsbergen, Information Retrieval, Butterworth, 1979.

[40] F. Mosteller, Association and estimation in contingency tables, Journal of the American Statistical Association 63 (321) (1968) 1–28. doi:10.1080/01621459.1968.11009219.

[41] G. U. Yule, K. Pearson, Vii. on the association of attributes in statistics: with illustrations from the material of the childhood society, Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character 194 (252-261) (1900) 257–319. doi:10.1098/rsta.1900.0019.

[42] G. U. Yule, On the methods of measuring association between two attributes, Journal of the Royal Statistical Society 75 (6) (1912) 579–652.

[43] W. Klösgen, Problems for knowledge discovery in databases and their treatment in the statistics interpreter explora, Int. J. Intell. Syst. 7 (7) (1992) 649–673. doi:10.1002/int.4550070707.

[44] B. Gray, M. E. Orlowska, CCAIIA: clustering categorial attributed into interseting accociation rules, in: X. Wu, K. Ramamohanarao, K. B. Korb (Eds.), Research and Development in Knowledge Discovery and Data Mining, Second Pacific-Asia Conference, PAKDD-98, Melbourne, Australia, April 15-17, 1998, Proceedings, Vol. 1394 of Lecture Notes in Computer Science, Springer, 1998, pp. 132–143. doi:10.1007/3-540-64383-4_12.

[45] P. Clark, R. Boswell, Rule induction with CN2: some recent improvements, in: Y. Kodratoff (Ed.), Machine Learning - EWSL-91, European Working Session on Learning, Porto, Portugal, March 6-8, 1991, Proceedings, Vol. 482 of Lecture Notes in Computer Science, Springer, 1991, pp. 151–163. doi:10.1007/BFb0017011.

[46] L. Breiman, J. H. Friedman, R. A. Olshen, C. J. Stone, Classification and Regression Trees, Wadsworth, 1984.

[47] P. Smyth, R. M. Goodman, An information theoretic approach to rule induction from databases, IEEE Trans. Knowl. Data Eng. 4 (4) (1992) 301–316. doi:10.1109/69.149926.

[48] Y. Yao, N. Zhong, An analysis of quantitative measures associated with rules, in: N. Zhong, L. Zhou (Eds.), Methodologies for Knowledge Discovery and Data Mining, Third Pacific-Asia Conference, PAKDD-99, Beijing, China, April 26-28, 1999, Proceedings, Vol. 1574 of Lecture Notes in Computer Science, Springer, 1999, pp. 479–488. doi:10.1007/3-540-48912-6_64.

[49] A. Agresti, Categorical Data Analysis, Wiley, 1990.

[50] A systematic approach to the construction and evaluation of tests of ability, Psychological Monographs 61 (4) (1947) i–49, place: US Publisher: American Psychological Association. doi:10.1037/h0093565.

[51] A. A. Freitas, On objective measures of rule surprisingness, in: J. M. Zytkow, M. Quafafou (Eds.), Principles of Data Mining and Knowledge Discovery, Second European Symposium, PKDD '98, Nantes, France, September 23-26, 1998, Proceedings, Vol. 1510 of Lecture Notes in Computer Science, Springer, 1998, pp. 1–9. doi:10.1007/BFb0094799.

[52] M. Sebag, M. Schoenauer, Generation of rules with certainty and confidence factors from incomplete and incoherent learning bases, in: Proc. of EKAW, Vol. 88, 1988, p. 28.

[53] J. Azé, Y. Kodratoff, Extraction de" pépites" de connaissance dans les données: une nouvelle approche et une étude de la sensibilité au bruit, Revue RNTI, E-1 (2004) 247–270.

[54] H. Zhang, B. Padmanabhan, A. Tuzhilin, On the discovery of significant statistical quantitative rules, in: W. Kim, R. Kohavi, J. Gehrke, W. DuMouchel (Eds.), Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Seattle, Washington, USA, August 22-25, 2004, ACM, 2004, pp. 374–383. doi:10.1145/1014052.1014094.

[55] F. M. Maggi, C. Di Ciccio, C. Di Francescomarino, T. Kala, Parallel algorithms for the automated discovery of declarative process models, Inf. Syst. 74 (Part) (2018) 136–152. doi:10.1016/j.is.2017.12.002.

[56] A. Fugard, N. Pfeifer, B. Mayerhofer, G. Kleiter, How people interpret conditionals: Shifts toward the conditional event, Journal of Experimental Psychology: Learning Memory and Cognition 37 (3) (2011) 635–648. doi:10.1037/a0022329.

[57] C. Di Ciccio, M. L. Bernardi, M. Cimitile, F. M. Maggi, Generating event logs through the simulation of declare models, in: EOMAS, 2015, pp. 20–36. doi:10.1007/978-3-319-24626-0_2.

[58] B. van Dongen, BPI Challenge 2017 (2 2017). doi:10.4121/uuid:5f3067df-f10b-45da-b98b-86ae4c7a310b.

[59] B. van Dongen, BPI Challenge 2012 (Apr 2012). doi:10.4121/uuid:3926db30-f712-4394-aebc-75976070e91f.

[60] W. Steeman, BPI Challenge 2013 (Apr 2014). doi:10.4121/uuid:a7ce5c55-03a7-4583-b855-98b86e1a2b07.

[61] B. van Dongen, BPI Challenge 2014 (Apr 2014). doi:10.4121/uuid:c3e5d162-0cfd-4bb0-bd82-af5268819c35.

[62] B. van Dongen, BPI Challenge 2015 (May 2015). doi:10.4121/uuid:31a308ef-c844-48da-948c-305d167a0ec1.

[63] M. de Leoni, F. Mannhardt, Road traffic fine management process (Feb 2015). doi:10.4121/uuid:270fd440-1057-4fb9-89a9-b699b47990f5.

[64] A. Azzam, P. R. Aryan, A. Cecconi, C. Di Ciccio, F. J. Ekaputra, J. D. Fernández, S. Karampatakis, E. Kiesling, A. Musil, M. Sabou, P. Shadlau, T. Thurner, The cityspin platform: A CPSS environment for city-wide infrastructures, in: A. Longo, M. Fazio, R. Ranjan, M. Zappatore (Eds.), CPSS@IoT, Vol. 2530 of CEUR Workshop Proceedings, CEUR-WS.org, 2019, pp. 57–64.

[65] W. M. P. van der Aalst, M. Pesic, H. Schonenberg, Declarative workflows: Balancing between flexibility and support, Computer Science - R&D 23 (2) (2009) 99–113. doi:10.1007/s00450-009-0057-9.

[66] C. Di Ciccio, M. Mecella, J. Mendling, The effect of noise on mined declarative constraints, in: SIMPDA, 2013, pp. 1–24. doi:10.1007/978-3-662-46436-6_1.

[67] A. Yeshchenko, C. Di Ciccio, J. Mendling, A. Polyvyanyy, Visual drift detection for event sequence data of business processes, IEEE Transactions on Visualization and Computer Graphics (Jan. 2021). doi:10.1109/TVCG.2021.3050071.

[68] A. Burattin, M. Cimitile, F. M. Maggi, A. Sperduti, Online discovery of declarative process models from event streams, IEEE Trans. Serv. Comput. 8 (6) (2015) 833–846. doi:10.1109/TSC.2015.2459703.

[69] F. M. Maggi, M. Montali, M. Westergaard, W. M. P. van der Aalst, Monitoring business constraints with linear temporal logic: An approach based on colored automata, in: BPM, 2011, pp. 132–147.

[70] A. Cecconi, A. Augusto, C. Di Ciccio, Detection of statistically significant differences between process variants through declarative rules, in: A. Polyvyanyy, M. T. Wynn, A. V. Looy, M. Reichert (Eds.), Business Process Management Forum - BPM Forum 2021, Rome, Italy, September 06-10, 2021, Proceedings, Vol. 427 of Lecture Notes in Business Information Processing, Springer, 2021, pp. 73–91. doi:10.1007/978-3-030-85440-9_5.

[71] Z. Cao, Y. Tian, T. B. Le, D. Lo, Rule-based specification mining leveraging learning to rank, Autom. Softw. Eng. 25 (3) (2018) 501–530. doi:10.1007/s10515-018-0231-z.

[72] A. Bauer, M. Leucker, C. Schallhart, Runtime verification for LTL and TLTL, ACM Trans. Softw. Eng. Methodol. 20 (4) (2011) 14:1–14:64. doi:10.1145/2000799.2000800.

[73] D. M. Gabbay, The declarative past and imperative future: Executable temporal logic for interactive systems, in: B. Banieqbal, H. Barringer, A. Pnueli (Eds.), Temporal Logic in Specification, Altrincham, UK, April 8-10, 1987, Proceedings, Vol. 398 of Lecture Notes in Computer Science, Springer, 1987, pp. 409–448. doi:10.1007/3-540-51803-7_36.

[74] M. Reynolds, The complexity of temporal logic over the reals, Ann. Pure Appl. Log. 161 (8) (2010) 1063–1096. doi:10.1016/j.apal.2010.01.002.

[75] G. De Giacomo, A. Di Stasio, F. Fuggitti, S. Rubin, Pure-past linear temporal and dynamic logic on finite traces, in: C. Bessiere (Ed.), IJCAI,

International Joint Conferences on Artificial Intelligence Organization, 2020, pp. 4959–4965, survey track. doi:10.24963/ijcai.2020/690.

[76] J. E. Hopcroft, R. Motwani, J. D. Ullman, Introduction to automata theory, languages, and computation, 3rd Edition, Pearson international edition, Addison-Wesley, 2007.

A. Proof of linearity

The RCon evaluation can be performed efficiently based on the automaton-based technique defined in [14]. The verification is there defined for online settings (run-time analysis). In contrast, we can exploit the offline setting of the present contribution (post-mortem analysis) to enhance furthermore the performances.

Before proceeding, we need to extend the LTLp$_f$ notations presented in Section 3. In the following, we will classify $\bigcirc, \Box, \Diamond, \mathbf{U}$ as *future operators*, $\ominus, \boxminus, \diamondminus, \mathbf{S}$ as *past operators*, and the following pairs of operators as *mirror images*: *(i)* $\bigcirc$ and $\ominus$, *(ii)* $\Box$ and $\boxminus$, *(iii)* $\Diamond$ and $\diamondminus$, *(iv)* $\mathbf{U}$ and $\mathbf{S}$. A LTLp$_f$ formula φ is named: **pure past ($\varphi^{\blacktriangleleft}$)** if it contains only past operators; **pure present ($\varphi^{\blacktriangledown}$)** if it contains no temporal operators at all; **pure future ($\varphi^{\blacktriangleright}$)** if it contains only future operators [73]. For example, $\varphi^{\blacktriangleleft} = \diamondminus(\mathsf{a}\ \mathbf{S}\ \mathsf{g})$, $\varphi^{\blacktriangledown} = \mathsf{a} \wedge \mathsf{b} \vee \mathsf{c}$, and $\varphi^{\blacktriangleright} = \bigcirc(\Box\mathsf{e} \vee (\Diamond\mathsf{b})\ \mathbf{W}\ \mathsf{p})$ are pure past, pure present, and pure future formulae, respectively. Finally, the *mirror image* φ_{M} of formula φ is the temporal formula obtained by replacing all its operators with their mirror images [74].
In [14] it has been proven that

- any LTLp$_f$ formula φ can be decomposed in $\varphi \equiv \bigvee_{j=1}^{m}(\varphi^{\blacktriangleleft} \wedge \varphi^{\blacktriangledown} \wedge \varphi^{\blacktriangleright})_j$, where $\varphi^{\blacktriangleleft}$, $\varphi^{\blacktriangledown}$, and $\varphi^{\blacktriangleright}$ are respectively pure past, present, and future formulae [14, Definition 2];
- each sub-formula of the decomposed formula can be valuated through distinct automata [14, Theorem 3];
- The past-only automata can be reversed into future-only automata valuating the reverse of a trace [14, Theorem 4]

The advantage of the reversion is that it allows for the verification of past components in only one read of the trace. This was not possible for future components because the end of the trace is unknown in an online setting. In an offline setting, as the entire trace is given, it is possible to apply the same

reasoning to reverse future-only automata into past-only automata valuating a trace in one backward read from the end to the start of the trace.

In the following, we provide proof sketches of the intuitions above. We replicate the steps taken for past reversion [14] applying them for the future reversion. To this extent, we rely on mirror images and reversed automata.

Lemma A.1. *Let $t \in \Sigma^*$ be a trace of length n and t_R its reverse. Given a pure future formula $\varphi^{\blacktriangleright}$, and its mirror image $\varphi^{\blacktriangleright}_M$, then $t, 1 \models \varphi^{\blacktriangleright}$ iff $t_R, n \models \varphi^{\blacktriangleright}_M$.*

The proof follows from the semantics of future and past operators of LTLp$_f$ provided in Section 3. For instance, verifying $\varphi^{\blacktriangleright} = \Diamond \mathsf{a}$ on $t = \langle \mathsf{f}, \mathsf{e}, \mathsf{d}, \mathsf{c}, \mathsf{b}, \mathsf{a}, \mathsf{g}, \mathsf{h}, \mathsf{i} \rangle$ at instant $i = 1$ is equivalent to verifying $\varphi^{\blacktriangleright}_{\mathrm{M}} = \Diamondblack \mathsf{a}$ on $t_R = \langle \mathsf{i}, \mathsf{h}, \mathsf{g}, \mathsf{a}, \mathsf{b}, \mathsf{c}, \mathsf{d}, \mathsf{e}, \mathsf{f} \rangle$ at $i = 9$. Notice that this holds for sub-traces too, thus verifying $\varphi^{\blacktriangleright}$ on t at instant $i = 5$ is equivalent to verifying $\varphi^{\blacktriangleright}_{\mathrm{M}}$ over $t_{[5:9]R} = \langle \mathsf{i}, \mathsf{h}, \mathsf{g}, \mathsf{a}, \mathsf{b}, \mathsf{f} \rangle$ at $i = 9$.

It follows from Lemma A.1 that any pure future formula can be seen as a pure past one on a reversed trace. Therefore the automaton verifying the mirror image of $\varphi^{\blacktriangleright}$ can be used for verification on the reversed trace, as stated in the following.

Corollary A.1. *Let $A_{\varphi^{\blacktriangleright}_M}$ be the automaton verifying $\varphi^{\blacktriangleright}_M$. Then $t, n \models \varphi^{\blacktriangleright}$ iff $t_R \in \mathscr{L}\big(A_{\varphi^{\blacktriangleright}_M}\big)$.*

Notice that $\varphi^{\blacktriangleright}_{\mathrm{M}}$ is a pure past formula, therefore $A_{\varphi^{\blacktriangleright}_{\mathrm{M}}}$ can be built by applying the technique of [75]. Furthermore, it is possible to transform the obtained automaton in order to read directly the original trace t thanks to the property of *closure under reversion* of regular languages [76].
From Lemma A.1 and Corollary A.1 we derive the following.

Theorem A.1 (Valuation through $\overleftarrow{A}_{\varphi^{\blacktriangleright}_{\mathrm{M}}}$). *Let $\varphi^{\blacktriangleright}$ be a pure future formula and $\varphi^{\blacktriangleright}_M$ its mirror image. Let $A_{\varphi^{\blacktriangleright}_M} \in \mathcal{A}$ be the automaton verifying $\varphi^{\blacktriangleright}_M$. Given a trace $t \in \Sigma^*$ of length n, we have that: $t, 1 \models \varphi^{\blacktriangleright}$ iff $t \in \mathscr{L}\Big(\overleftarrow{A}_{\varphi^{\blacktriangleright}_M}\Big)$.*

Consider $t = \langle \mathsf{f}, \mathsf{e}, \mathsf{d}, \mathsf{c}, \mathsf{b}, \mathsf{a}, \mathsf{g}, \mathsf{h}, \mathsf{i} \rangle$ and the RCon $\Psi = \mathsf{a} \boxdot\!\!\rightarrow (\ominus \mathsf{b} \vee \Diamond \mathsf{c})$ and the pure future formula of its separated automata set (sep.aut.set) $\varphi^{\blacktriangleright} = \Diamond \mathsf{c}$. The RCon

is activated by $t(6)$. Its mirror image is $\varphi^{\blacktriangleright}_{\mathrm{M}} = \Diamond\mathsf{c}$. It is possible to verify $\varphi^{\blacktriangleright}_{\mathrm{M}}$ over trace $t_{[6:9]R} = \langle \mathsf{i}, \mathsf{h}, \mathsf{g}, \mathsf{a} \rangle$ at $i = 9$, thereby verifying $\varphi^{\blacktriangleright}$ over $t_{[6:9]}$ as per Lemma A.1. Thanks to Theorem A.1, $\varphi^{\blacktriangleright}_{\mathrm{M}}$ can be verified on $t_{[6:9]}$ with the reversed automaton $\overleftarrow{A}_{\varphi^{\blacktriangleright}_{\mathrm{M}}}$.

In conclusion, we proved that any pure future formula can be verified by parsing sub-traces from the end of the trace back to the activator event. As the verification of future formulae is the only potentially exponential time element [14], using the reversed verification provided in this section guarantees the optimization to linear time, as empirically evidenced in Section 5.