Online Causation Monitoring of Signal Temporal Logic

Zhenya Zhang[1](✉), Jie An[2], Paolo Arcaini[2], and Ichiro Hasuo[2]

[1] Kyushu University, Fukuoka, Japan
zhang@ait.kyushu-u.ac.jp

[2] National Institute of Informatics, Tokyo, Japan
{jiean,arcaini,hasuo}@nii.ac.jp

Abstract. Online monitoring is an effective validation approach for hybrid systems, that, at runtime, checks whether the (partial) signals of a system satisfy a specification in, e.g., *Signal Temporal Logic (STL)*. The classic STL monitoring is performed by computing a robustness interval that specifies, at each instant, how far the monitored signals are from violating and satisfying the specification. However, since a robustness interval monotonically shrinks during monitoring, classic online monitors may fail in reporting new violations or in precisely describing the system evolution at the current instant. In this paper, we tackle these issues by considering the *causation* of violation or satisfaction, instead of directly using the robustness. We first introduce a *Boolean causation monitor* that decides whether each instant is relevant to the violation or satisfaction of the specification. We then extend this monitor to a *quantitative causation monitor* that tells how far an instant is from being relevant to the violation or satisfaction. We further show that classic monitors can be derived from our proposed ones. Experimental results show that the two proposed monitors are able to provide more detailed information about system evolution, without requiring a significantly higher monitoring cost.

Keywords: online monitoring · Signal Temporal Logic · monotonicity

1 Introduction

Safety-critical systems require strong correctness guarantees. Due to the complexity of these systems, offline verification may not be able to guarantee their total correctness, as it is often very difficult to assess all possible system behaviors. To mitigate this issue, runtime verification [4, 29, 36] has been proposed as a

Z. Zhang is supported by JSPS KAKENHI Grant No. JP23K16865 and No. JP23H03372. J. An, P. Arcaini, and I. Hasuo are supported by ERATO HASUO Metamathematics for Systems Design Project (No. JPMJER1603), JST, Funding Reference number 10.13039/501100009024 ERATO. P. Arcaini is also supported by Engineerable AI Techniques for Practical Applications of High-Quality Machine Learning-based Systems Project (Grant Number JPMJMI20B8), JST-Mirai.

complementary technique that analyzes the system execution at runtime. Online monitoring is such an approach that checks whether the system execution (e.g., given in terms of signals) satisfies or violates a system specification specified in a temporal logic [28,34], e.g., *Signal Temporal Logic (STL)* [30].

Quantitative online monitoring is based on the STL *robust semantics* [17,21] that not only tells whether a signal satisfies or violates a specification φ (i.e., the classic Boolean satisfaction relation), but also assigns a value in $\mathbb{R} \cup \{\infty, -\infty\}$ (i.e., *robustness*) that indicates *how robustly* φ is satisfied or violated. However, differently from offline assessment of STL formulas, an online monitor needs to reason on *partial signals* and, so, the assessment of the robustness should be adapted. We consider an established approach [12] employed by *classic online monitors* (`ClaM` in the following). It consists in computing, instead of a single robustness value, a *robustness interval*; at each monitoring step, `ClaM` identifies an *upper bound* $[\mathrm{R}]^{\mathsf{U}}$ telling the maximal reachable robustness of any possible suffix signal (i.e., any continuation of the system evolution), and a *lower bound* $[\mathrm{R}]^{\mathsf{L}}$ telling the minimal reachable robustness. If, at some instant, $[\mathrm{R}]^{\mathsf{U}}$ becomes negative, the specification is violated; if $[\mathrm{R}]^{\mathsf{L}}$ becomes positive, the specification is satisfied. In the other cases, the specification validity is `unknown`.

Consider a simple example in Fig. 1. It shows the monitoring of the `speed` of a vehicle (in the upper plot); the specification requires the `speed` to be always below 10. The lower plot reports how the upper bound $[\mathrm{R}]^{\mathsf{U}}$ and the lower bound $[\mathrm{R}]^{\mathsf{L}}$ of the reachable robustness change over time. We observe that the initial value of $[\mathrm{R}]^{\mathsf{U}}$ is around 8 and gradually decreases.[1] The monitor allows to detect that the specification is violated at time $b = 20$ when the `speed` becomes higher than 10, and therefore $[\mathrm{R}]^{\mathsf{U}}$ goes below 0. After that, the violation severity progressively gets worse till time $b = 30$, when $[\mathrm{R}]^{\mathsf{U}}$ becomes -5. After that point, the monitor does not provide any additional useful information about the system evolution, as $[\mathrm{R}]^{\mathsf{U}}$ remains stuck at -5. However, if we observe the signal of the `speed` after $b = 30$, we notice that (i) the severity of the violation is mitigated, and the "1st violation episode" ends at time $b = 35$; however, the monitor `ClaM` does not report this type of information; (ii) a "2nd violation episode" occurs in the time interval $[40, 45]$; the monitor `ClaM` does not distinguish the new violation.

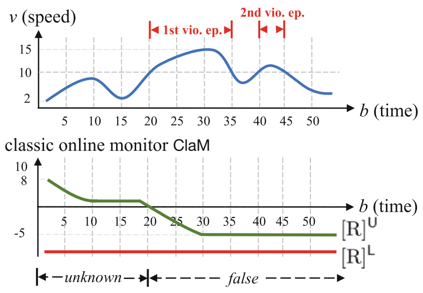


Fig. 1. `ClaM` – Robustness upper and lower bounds of $\square_{[0,100]}(v < 10)$

The reason for the issues reported in the example is that the upper and lower bounds are monotonically decreasing and increasing; this has the consequence

[1] The value of lower bound $[\mathrm{R}]^{\mathsf{L}}$ is not shown in the figure, as not relevant. In the example, it remains constant before $b = 100$, and the value is usually set either according to domain knowledge about system signals, or to $-\infty$ otherwise.

that the robustness interval at a given step is "masked" by the history of previous robustness intervals, and, e.g., it is not possible to detect mitigation of the violation severity. Moreover, as an extreme consequence, as soon as the monitor `ClaM` assesses the violation of the specification (i.e., the upper bound $[\mathrm{R}]^{\mathsf{U}}$ becomes negative), or its satisfaction (i.e., the lower bound $[\mathrm{R}]^{\mathsf{L}}$ becomes positive), the Boolean status of the monitor does not change anymore. Such characteristic directly derives from the STL semantics and it is known as the *monotonicity* [9–11] of classic online monitors. Monotonicity has been recognized as a problem of these monitors in the literature [10,37,40], since it does not allow to detect specific types of information that are "masked". We informally define two types of *information masking* that can occur because of monotonicity:

***evolution masking*:** the monitor may not properly report the evolution of the system execution, e.g., mitigation of violation severity may not be detected;
***violation masking*:** as a special case of *evolution masking*, the first violation episode during the system execution "masks" the following ones.

The information not reported by `ClaM` because of information masking, is very useful in several contexts. First of all, in some systems, the first violation of the specification does not mean that the system is not operating anymore, and one may want to continue monitoring and detect all the succeeding violations; this is the case, e.g., of the monitoring approach reported by Selyunin et al. [37] in which all the violations of the SENT protocol must be detected. Moreover, having a precise description of the system evolution is important for the usefulness of the monitoring; for example, the monitoring of the `speed` in Fig. 1 could be used in a vehicle for checking the speed and notifying the driver whenever the speed is approaching the critical limit; if the monitor is not able to precisely capture the severity of violation, it cannot be used for this type of application.

Some works [10,37,40] try to mitigate the monotonicity issues, by "resetting" the monitor at specific points. A recent approach has been proposed by Zhang et al. [40] (called `ResM` in the following) that is able to identify each "violation episode" (i.e., it solves the problem of *violation masking*), but does not solve the *evolution masking* problem. For the example in Fig. 1, `ResM` is able to detect the two violation episodes in intervals $[20, 35]$ and $[40, 45]$, but it is not able to report that the speed decreases after $b = 10$ (in a non-violating situation), and that the severity of the violation is mitigated after $b = 30$.

Contribution. In this paper, in order to provide more information about the evolution of the monitored system, we propose to monitor the *causation* of violation or satisfaction, instead of considering the robustness directly. To do this, we rely on the notion of *epoch* [5]. At each instant, the *violation (satisfaction) epoch* identifies the time instants at which the evaluation of the atomic propositions of the specification φ causes the violation (satisfaction) of φ.

Based on the notion of epoch, we define a *Boolean causation monitor* (called `BCauM`) that, at runtime, not only assesses the specification violation/satisfaction, but also tells whether each instant is relevant to it. Namely, `BCauM` marks each current instant b as (i) a *violation causation instant*, if b is added to the violation epoch; (ii) a *satisfaction causation instant*, if b is added to the satisfaction epoch;

(iii) an *irrelevant instant*, if b is not added to any epoch. We show that `BCauM` is able to detect all the violation episodes (so solving the *violation masking* issue), as violation causation instants can be followed by irrelevant instants. Moreover, we show that the information provided by the classic Boolean online monitor can be derived from that of the Boolean causation monitor `BCauM`.

However, `BCauM` just tells us whether the current instant is a (violation or satisfaction) causation instant or not, but does not report *how far* the instant is from being a causation instant. To this aim, we introduce the notion of *causation distance*, as a quantitative measure characterizing the spatial distance of the signal value at b from turning b into a causation instant. Then, we propose the *quantitative causation monitor* (`QCauM`) that, at each instant, returns its causation distance. We show that using `QCauM`, besides solving the *violation masking* problem, we also solve the *evolution masking* problem. Moreover, we show that we can derive from `QCauM` both the classic quantitative monitor `ClaM`, and the Boolean causation monitor `BCauM`.

Experimental results show that the proposed monitors, not only provide more information, but they do it in an efficient way, not requiring a significant additional monitoring time w.r.t. the existing monitors.

Outline. Section 2 reports necessary background. We introduce `BCauM` in Sect. 3, and `QCauM` in Sect. 4. Experimental assessment of the two proposed monitors is reported in Sect. 5. Finally, Sect. 6 discusses some related work, and Sect. 7 concludes the paper.

2 Preliminaries

In this section, we review the fundamentals of *signal temporal logic (STL)* in Sect. 2.1, and then introduce the existing classic online monitoring approach in Sect. 2.2.

2.1 Signal Temporal Logic

Let $T \in \mathbb{R}_+$ be a positive real, and $d \in \mathbb{N}_+$ be a positive integer. A *d-dimensional signal* is a function $\mathbf{v} \colon [0, T] \to \mathbb{R}^d$, where T is called the *time horizon* of $\mathbf{v}$. Given an arbitrary time instant $t \in [0, T]$, $\mathbf{v}(t)$ is a d-dimensional real vector; each dimension concerns a *signal variable* that has a certain physical meaning, e.g., `speed`, `RPM`, `acceleration`, etc. In this paper, we fix a set **Var** of variables and assume that a signal $\mathbf{v}$ is *spatially bounded*, i.e., for all $t \in [0, T]$, $\mathbf{v}(t) \in \Omega$, where Ω is a d-dimensional hyper-rectangle.

Signal temporal logic (STL) is a widely-adopted specification language, used to describe the expected behavior of systems. In Definition 1 and Definition 2, we respectively review the syntax and the robust semantics of STL [17,21,30].

Definition 1 (STL syntax). In STL, the *atomic propositions* α and the *formulas* φ are defined as follows:

$$\alpha ::\equiv f(w_1, \ldots, w_K) > 0 \qquad \varphi ::\equiv \alpha \mid \bot \mid \neg\varphi \mid \varphi \wedge \varphi \mid \Box_I \varphi \mid \Diamond_I \varphi \mid \varphi\, \mathcal{U}_I\, \varphi$$

Here f is a K-ary function $f : \mathbb{R}^K \to \mathbb{R}$, $w_1, \dots, w_K \in \mathbf{Var}$, and I is a closed interval over $\mathbb{R}_{\geq 0}$, i.e., $I = [l, u]$, where $l, u \in \mathbb{R}$ and $l \leq u$. In the case that $l = u$, we can use l to stand for I. $\Box, \Diamond$ and $\mathcal{U}$ are temporal operators, which are known as *always*, *eventually* and *until*, respectively. The always operator $\Box$ and eventually operator $\Diamond$ are two special cases of the until operator $\mathcal{U}$, where $\Diamond_I \varphi \equiv \top \mathcal{U}_I\, \varphi$ and $\Box_I \varphi \equiv \neg \Diamond_I \neg \varphi$. Other common connectives such as $\vee, \to$ are introduced as syntactic sugar: $\varphi_1 \vee \varphi_2 \equiv \neg(\neg\varphi_1 \wedge \neg\varphi_2)$, $\varphi_1 \to \varphi_2 \equiv \neg\varphi_1 \vee \varphi_2$.

Definition 2 (STL robust semantics). Let $\mathbf{v}$ be a signal, φ be an STL formula and $\tau \in \mathbb{R}_+$ be an instant. The *robustness* $\mathrm{R}(\mathbf{v}, \varphi, \tau) \in \mathbb{R} \cup \{\infty, -\infty\}$ of $\mathbf{v}$ w.r.t. φ at τ is defined by induction on the construction of formulas, as follows.

$$\mathrm{R}(\mathbf{v}, \alpha, \tau) := f(\mathbf{v}(\tau)) \qquad \mathrm{R}(\mathbf{v}, \bot, \tau) := -\infty \qquad \mathrm{R}(\mathbf{v}, \neg\varphi, \tau) := -\mathrm{R}(\mathbf{v}, \varphi, \tau)$$

$$\mathrm{R}(\mathbf{v}, \varphi_1 \wedge \varphi_2, \tau) := \min\left(\mathrm{R}(\mathbf{v}, \varphi_1, \tau), \mathrm{R}(\mathbf{v}, \varphi_2, \tau)\right)$$

$$\mathrm{R}(\mathbf{v}, \Box_I \varphi, \tau) := \inf_{t \in \tau + I} \mathrm{R}(\mathbf{v}, \varphi, t) \qquad \mathrm{R}(\mathbf{v}, \Diamond_I \varphi, \tau) := \sup_{t \in \tau + I} \mathrm{R}(\mathbf{v}, \varphi, t)$$

$$\mathrm{R}(\mathbf{v}, \varphi_1\, \mathcal{U}_I\, \varphi_2, \tau) := \sup_{t \in \tau + I} \min\left(\mathrm{R}(\mathbf{v}, \varphi_2, t), \inf_{t' \in [\tau, t)} \mathrm{R}(\mathbf{v}, \varphi_1, t')\right)$$

Here, $\tau + I$ denotes the interval $[l + \tau, u + \tau]$.

The original STL semantics is Boolean, which represents whether a signal $\mathbf{v}$ satisfies φ at an instant τ, i.e., whether $(\mathbf{v}, \tau) \models \varphi$. The robust semantics in Definition 2 is a quantitative extension that refines the original Boolean STL semantics, in the sense that, $\mathrm{R}(\mathbf{v}, \varphi, \tau) > 0$ implies $(\mathbf{v}, \tau) \models \varphi$, and $\mathrm{R}(\mathbf{v}, \varphi, \tau) < 0$ implies $(\mathbf{v}, \tau) \not\models \varphi$. More details can be found in [21, Proposition 16].

2.2 Classic Online Monitoring of STL

STL robust semantics in Definition 2 provides an offline monitoring approach for *complete signals*. *Online monitoring*, instead, targets a growing *partial signal* at runtime. Besides the verdicts $\top$ and $\bot$, an online monitor can also report the verdict `unknown` (denoted as ?), which represents a status when the satisfaction of the signal to φ is not decided yet. In the following, we formally define partial signals and introduce online monitors for STL.

Let T be the time horizon of a signal $\mathbf{v}$, and let $[a, b] \subseteq [0, T]$ be a sub-interval in the time domain $[0, T]$. A *partial signal* $\mathbf{v}_{a:b}$ is a function which is only defined in the interval $[a, b]$; in the remaining domain $[0, T] \setminus [a, b]$, we denote that $\mathbf{v}_{a:b} = \epsilon$, where ϵ stands for a value that is not defined.

Specifically, if $a = 0$ and $b \in (a, T]$, a partial signal $\mathbf{v}_{a:b}$ is called a *prefix* (partial) signal; dually, if $b = T$ and $a \in [0, b)$, a partial signal $\mathbf{v}_{a:b}$ is called a *suffix* (partial) signal. Given a prefix signal $\mathbf{v}_{0:b}$, a *completion* $\mathbf{v}_{0:b} \cdot \mathbf{v}_{b:T}$ of $\mathbf{v}_{0:b}$ is defined as the concatenation of $\mathbf{v}_{0:b}$ with a suffix signal $\mathbf{v}_{b:T}$.

Definition 3 (Classic Boolean STL online monitor). Let $\mathbf{v}_{0:b}$ be a prefix signal, and φ be an STL formula. An online monitor $\mathrm{M}(\mathbf{v}_{0:b}, \varphi, \tau)$ returns a verdict in $\{\top, \bot, ?\}$ (namely, `true`, `false`, and `unknown`), as follows:

$$\mathrm{M}(\mathbf{v}_{0:b}, \varphi, \tau) \ := \begin{cases} \top & \text{if } \forall \mathbf{v}_{b:T}.\, \mathrm{R}(\mathbf{v}_{0:b} \cdot \mathbf{v}_{b:T}, \varphi, \tau) > 0 \\ \bot & \text{if } \forall \mathbf{v}_{b:T}.\, \mathrm{R}(\mathbf{v}_{0:b} \cdot \mathbf{v}_{b:T}, \varphi, \tau) < 0 \\ ? & \text{otherwise} \end{cases}$$

Namely, the verdicts of $\mathrm{M}(\mathbf{v}_{0:b}, \varphi, \tau)$ are interpreted as follows:

- if any possible completion $\mathbf{v}_{0:b} \cdot \mathbf{v}_{b:T}$ of $\mathbf{v}_{0:b}$ satisfies φ, then $\mathbf{v}_{0:b}$ satisfies φ;
- if any possible completion $\mathbf{v}_{0:b} \cdot \mathbf{v}_{b:T}$ of $\mathbf{v}_{0:b}$ violates φ, then $\mathbf{v}_{0:b}$ violates φ;
- otherwise (i.e., there is a completion $\mathbf{v}_{0:b} \cdot \mathbf{v}_{b:T}$ that satisfies φ, and there is a completion $\mathbf{v}_{0:b} \cdot \mathbf{v}_{b:T}$ that violates φ), then $\mathrm{M}(\mathbf{v}_{0:b}, \varphi, \tau)$ reports `unknown`.

Note that, by Definition 3 only, we cannot synthesize a feasible online monitor, because the possible completions for $\mathbf{v}_{0:b}$ are infinitely many. A constructive online monitor is introduced in [12], which implements the functionality of Definition 3 by computing the *reachable* robustness of $\mathbf{v}_{0:b}$. We review this monitor in Definition 4.

Definition 4 (Classic Quantitative STL online monitor (`ClaM`)). Let $\mathbf{v}_{0:b}$ be a prefix signal, and let φ be an STL formula. We denote by $\mathtt{R}^{\alpha}_{\max}$ and $\mathtt{R}^{\alpha}_{\min}$ the possible *maximum* and *minimum bounds* of the robustness $\mathrm{R}(\mathbf{v}, \alpha, \tau)$[2]. Then, an *online monitor* $[\mathrm{R}](\mathbf{v}_{0:b}, \varphi, \tau)$, which returns a sub-interval of $[\mathtt{R}^{\alpha}_{\min}, \mathtt{R}^{\alpha}_{\max}]$ at the instant b, is defined as follows, by induction on the construction of formulas.

$$[\mathrm{R}](\mathbf{v}_{0:b}, \alpha, \tau) \ := \begin{cases} \left[f\left(\mathbf{v}_{0:b}(\tau)\right), f\left(\mathbf{v}_{0:b}(\tau)\right)\right] & \text{if } \tau \in [0, b] \\ \left[\mathtt{R}^{\alpha}_{\min}, \mathtt{R}^{\alpha}_{\max}\right] & \text{otherwise} \end{cases}$$

$$[\mathrm{R}](\mathbf{v}_{0:b}, \neg\varphi, \tau) \ := \ -[\mathrm{R}](\mathbf{v}_{0:b}, \varphi, \tau)$$

$$[\mathrm{R}](\mathbf{v}_{0:b}, \varphi_1 \wedge \varphi_2, \tau) \ := \ \min\Big([\mathrm{R}](\mathbf{v}_{0:b}, \varphi_1, \tau), [\mathrm{R}](\mathbf{v}_{0:b}, \varphi_2, \tau)\Big)$$

$$[\mathrm{R}](\mathbf{v}_{0:b}, \Box_I \varphi, \tau) \ := \ \inf_{t \in \tau + I}\Big([\mathrm{R}](\mathbf{v}_{0:b}, \varphi, t)\Big)$$

$$[\mathrm{R}](\mathbf{v}_{0:b}, \varphi_1 \,\mathcal{U}_I\, \varphi_2, \tau) \ := \ \sup_{t \in \tau + I} \min\Big([\mathrm{R}](\mathbf{v}_{0:b}, \varphi_2, t), \inf_{t' \in [\tau, t)} [\mathrm{R}](\mathbf{v}_{0:b}, \varphi_1, t')\Big)$$

Here, f is defined as in Definition 1, and the arithmetic rules over intervals $I = [l, u]$ are defined as follows: $-I := [-u, -l]$ and $\min(I_1, I_2) := [\min(l_1, l_2), \min(u_1, u_2)]$.

We denote by $[\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:b}, \varphi, \tau)$ and $[\mathrm{R}]^{\mathsf{L}}(\mathbf{v}_{0:b}, \varphi, \tau)$ the upper bound and the lower bound of $[\mathrm{R}](\mathbf{v}_{0:b}, \varphi, \tau)$ respectively. Intuitively, the two bounds together form the reachable robustness interval of the completion $\mathbf{v}_{0:b} \cdot \mathbf{v}_{b:T}$, under any possible suffix signal $\mathbf{v}_{b:T}$. For instance, in Fig. 2, the upper bound $[\mathrm{R}]^{\mathsf{U}}$ at $b = 20$ is 0, which indicates that the robustness of the completion of the signal `speed`, under any suffix, can never be larger than 0.

The quantitative online monitor `ClaM` in Definition 4 refines the Boolean one in Definition 3, and the Boolean monitor can be derived from `ClaM` as follows:

[2] $\mathrm{R}(\mathbf{v}, \alpha, \tau)$ is bounded because $\mathbf{v}$ is bounded by Ω. In practice, if Ω is not know, we set $\mathtt{R}^{\alpha}_{\max}$ and $\mathtt{R}^{\alpha}_{\min}$ to, respectively, ∞ and $-\infty$.

- if $[\mathrm{R}]^{\mathsf{L}}(\mathbf{v}_{0:b}, \varphi, \tau) > 0$, it implies that $\mathrm{M}(\mathbf{v}_{0:b}, \varphi, \tau) = \top$;
- if $[\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:b}, \varphi, \tau) < 0$, it implies that $\mathrm{M}(\mathbf{v}_{0:b}, \varphi, \tau) = \bot$;
- otherwise, if $[\mathrm{R}]^{\mathsf{L}}(\mathbf{v}_{0:b}, \varphi, \tau) < 0$ and $[\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:b}, \varphi, \tau) > 0$, $\mathrm{M}(\mathbf{v}_{0:b}, \varphi, \tau) = ?$.

The classic online monitors are *monotonic* by definition. In the Boolean monitor (Definition 3), with the growth of $\mathbf{v}_{0:b}$, $\mathrm{M}(\mathbf{v}_{0:b}, \varphi, \tau)$ can only turn from ? to $\{\bot, \top\}$, but never the other way around. In the quantitative one (Definition 4), as shown in Lemma 1, $[\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:b}, \varphi, \tau)$ and $[\mathrm{R}]^{\mathsf{L}}(\mathbf{v}_{0:b}, \varphi, \tau)$ are both monotonic, the former one decreasingly, the latter one increasingly. An example can be observed in Fig. 2.

Lemma 1 (Monotonicity of STL online monitor). Let $[\mathrm{R}](\mathbf{v}_{0:b}, \varphi, \tau)$ be the quantitative online monitor for a partial signal $\mathbf{v}_{0:b}$ and an STL formula φ. With the growth of the partial signal $\mathbf{v}_{0:b}$, the upper bound $[\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:b}, \varphi, \tau)$ monotonically decreases, and the lower bound $[\mathrm{R}]^{\mathsf{L}}(\mathbf{v}_{0:b}, \varphi, \tau)$ monotonically increases, i.e., for two time instants $b_1, b_2 \in [0, T]$, if $b_1 < b_2$, we have (i) $[\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:b_1}, \varphi, \tau) \geq [\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:b_2}, \varphi, \tau)$, and (ii) $[\mathrm{R}]^{\mathsf{L}}(\mathbf{v}_{0:b_1}, \varphi, \tau) \leq [\mathrm{R}]^{\mathsf{L}}(\mathbf{v}_{0:b_2}, \varphi, \tau)$.

Proof. This can be proved by induction on the structures of STL formulas. The detailed proof can be found in the full version [38]. □

3 Boolean Causation Online Monitor

As explained in Sect. 1, monotonicity of classic online monitors causes different types of *information masking*, which prevents some information from being delivered. In this section, we introduce a novel *Boolean causation (online) monitor* `BCauM`, that solves the *violation masking* issue (see Sect. 1). `BCauM` is defined based on *online signal diagnostics* [5,40], which reports the *cause* of violation or satisfaction of the specification at the atomic proposition level.

Definition 5 (Online signal diagnostics). Let $\mathbf{v}_{0:b}$ be a partial signal and φ be an STL specification. At an instant b, online signal diagnostics returns a *violation epoch* $\mathrm{E}^{\ominus}(\mathbf{v}_{0:b}, \varphi, \tau)$, under the condition $[\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:b}, \varphi, \tau) < 0$, as follows:

$$\mathrm{E}^{\ominus}(\mathbf{v}_{0:b}, \alpha, \tau) := \begin{cases} \{\langle \alpha, \tau \rangle\} & \text{if } [\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:b}, \alpha, \tau) < 0 \\ \emptyset & \text{otherwise} \end{cases}$$

$$\mathrm{E}^{\ominus}(\mathbf{v}_{0:b}, \neg\varphi, \tau) := \mathrm{E}^{\oplus}(\mathbf{v}_{0:b}, \varphi, \tau)$$

$$\mathrm{E}^{\ominus}(\mathbf{v}_{0:b}, \varphi_1 \wedge \varphi_2, \tau) := \bigcup_{\substack{i \in \{1,2\} \text{ s.t.} \\ [\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:b}, \varphi_i, \tau) < 0}} \mathrm{E}^{\ominus}(\mathbf{v}_{0:b}, \varphi_i, \tau)$$

$$\mathrm{E}^{\ominus}(\mathbf{v}_{0:b}, \Box_I \varphi, \tau) := \bigcup_{\substack{t \in \tau + I \text{ s.t.} \\ [\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:b}, \varphi, t) < 0}} \mathrm{E}^{\ominus}(\mathbf{v}_{0:b}, \varphi, t)$$

$$\mathrm{E}^{\ominus}(\mathbf{v}_{0:b}, \varphi_1 \,\mathcal{U}_I\, \varphi_2, \tau) := \bigcup_{\substack{t \in \tau + I \text{ s.t.} \\ [\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:b}, \varphi_1 \mathcal{U}_t \varphi_2, \tau) < 0}} \left(\mathrm{E}^{\ominus}(\mathbf{v}_{0:b}, \varphi_2, t) \cup \bigcup_{t' \in [\tau, t)} \mathrm{E}^{\ominus}(\mathbf{v}_{0:b}, \varphi_1, t') \right)$$

and a *satisfaction epoch* $\mathrm{E}^{\oplus}(\mathbf{v}_{0:b}, \varphi, \tau)$, under the condition $[\mathrm{R}]^{\mathsf{L}}(\mathbf{v}_{0:b}, \varphi, \tau) > 0$, as follows:

$$\mathrm{E}^{\oplus}(\mathbf{v}_{0:b}, \alpha, \tau) := \begin{cases} \{\langle \alpha, \tau \rangle\} & \text{if } [\mathrm{R}]^{\mathsf{L}}(\mathbf{v}_{0:b}, \alpha, \tau) > 0 \\ \emptyset & \text{otherwise} \end{cases}$$

$$\mathrm{E}^{\oplus}(\mathbf{v}_{0:b}, \neg\varphi, \tau) := \mathrm{E}^{\ominus}(\mathbf{v}_{0:b}, \varphi, \tau)$$

$$\mathrm{E}^{\oplus}(\mathbf{v}_{0:b}, \varphi_1 \wedge \varphi_2, \tau) := \bigcup_{\substack{i \in \{1,2\} \text{ s.t.} \\ [\mathrm{R}]^{\mathsf{L}}(\mathbf{v}_{0:b}, \varphi_i, \tau) > 0}} \mathrm{E}^{\oplus}(\mathbf{v}_{0:b}, \varphi_i, \tau)$$

$$\mathrm{E}^{\oplus}(\mathbf{v}_{0:b}, \Box_I \varphi, \tau) := \bigcup_{\substack{t \in \tau + I \text{ s.t.} \\ [\mathrm{R}]^{\mathsf{L}}(\mathbf{v}_{0:b}, \varphi, t) > 0}} \mathrm{E}^{\oplus}(\mathbf{v}_{0:b}, \varphi, t)$$

$$\mathrm{E}^{\oplus}(\mathbf{v}_{0:b}, \varphi_1 \mathcal{U}_I \varphi_2, \tau) := \bigcup_{\substack{t \in \tau + I \text{ s.t.} \\ [\mathrm{R}]^{\mathsf{L}}(\mathbf{v}_{0:b}, \varphi_1 \mathcal{U}_t \varphi_2, \tau) > 0}} \left(\mathrm{E}^{\oplus}(\mathbf{v}_{0:b}, \varphi_2, t) \cup \bigcup_{t' \in [\tau, t)} \mathrm{E}^{\oplus}(\mathbf{v}_{0:b}, \varphi_1, t') \right)$$

If the conditions are not satisfied, $\mathrm{E}^{\ominus}(\mathbf{v}_{0:b}, \varphi, \tau)$ and $\mathrm{E}^{\oplus}(\mathbf{v}_{0:b}, \varphi, \tau)$ are both $\emptyset$. Note that the definition is recursive, thus the conditions should also be checked for computing the violation and satisfaction epochs of the sub-formulas of φ.

Computation for other operators can be inferred by the presented ones and the STL syntax (Definition 1).

Intuitively, when a partial signal $\mathbf{v}_{0:b}$ violates a specification φ, a violation epoch starts collecting the evaluations (identified by pairs of atomic propositions and instants) of the signal at the atomic proposition level, that cause the violation of the whole formula φ (which also applies to the satisfaction cases in a dual manner). This is done inductively, based on the semantics of different operators:

- in the case of an atomic proposition α, if α is violated at τ, it collects $\langle \alpha, \tau \rangle$;
- in the case of a negation $\neg\varphi$, it collects the satisfaction epoch of φ;
- in the case of a conjunction $\varphi_1 \wedge \varphi_2$, it collects the union of the violation epochs of the sub-formulas violated by the partial signal;
- in the case of an *always* operator $\Box_I \varphi$, it collects the epochs of the sub-formula φ at all the instants t where φ is evaluated as being violated.
- in the case of an *until* operator $\varphi_1 \mathcal{U}_I \varphi_2$, it collects the epochs of the sub-formula φ_2 at all the instants t and the epochs of φ_1 at the instants $t' \in [\tau, t)$, in the case where the clause "φ_1 until φ_2" is violated at t.

Example 1. The example in Fig. 2 illustrates how an epoch is collected. The specification requires that whenever the `speed` is higher than 10, the car should decelerate within 5 time units. As shown by the classic monitor, the specification is violated at $b = 25$, since v becomes higher than 10 at 20 but a remains positive during $[20, 25]$. Note that the specification can be rewritten as $\varphi \equiv \Box_{[0,100]}(\neg(v > 10) \vee \Diamond_{[0,5]}(a < 0))$. For convenience, we name the sub-formulas of φ as follows:

$$\varphi' \equiv \neg(v > 10) \vee \Diamond_{[0,5]}(a < 0) \qquad \varphi_1 \equiv \neg(v > 10) \qquad \varphi_2 \equiv \Diamond_{[0,5]}(a < 0)$$
$$\alpha_1 \equiv v > 10 \qquad \alpha_2 \equiv a < 0$$

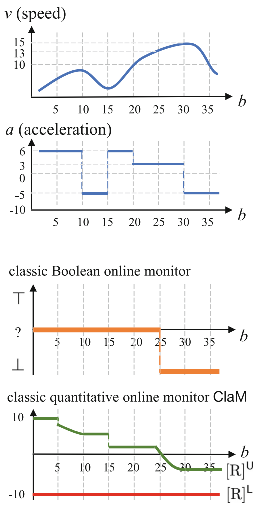


Fig. 2. Classic monitor (`ClaM`) result for the STL specification: $\Box_{[0,100]}(v > 10 \to \Diamond_{[0,5]}(a < 0))$

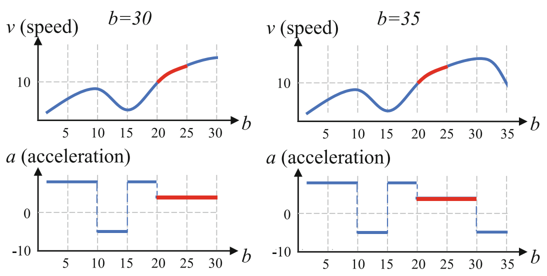


Fig. 3. The violation epochs (the red parts) respectively when $b = 30$ and $b = 35$

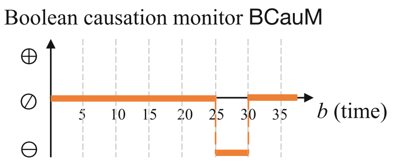


Fig. 4. Boolean causation monitor (`BCauM`) result

Figure 3 shows the violation epochs at two instants 30 and 35. First, at $b = 30$,

$$\begin{aligned}\mathrm{E}^{\ominus}(\mathbf{v}_{0:30}, \varphi, 0) &= \left(\bigcup_{t\in[20,25]} \mathrm{E}^{\oplus}(\mathbf{v}_{0:30}, \alpha_1, t)\right) \cup \left(\bigcup_{t\in[20,30]} \mathrm{E}^{\ominus}(\mathbf{v}_{0:30}, \alpha_2, t)\right)\\ &= \langle \alpha_1, [20, 25]\rangle \cup \langle \alpha_2, [20, 30]\rangle\end{aligned}$$

Similarly, the violation epoch $\mathrm{E}^{\ominus}(\mathbf{v}_{0:35}, \varphi, 0)$ at $b = 35$ is the same as that at $b = 30$. Intuitively, the epoch at $b = 30$ shows the cause of the violation of $\mathbf{v}_{0:30}$; then since signal $a < 0$ in $[30, 35]$, this segment is not considered as the cause of the violation, so the epoch remains the same at $b = 35$. $\lhd$

Definition 6 (Boolean causation monitor (`BCauM`)). Let $\mathbf{v}_{0:b}$ be a partial signal and φ be an STL specification. We denote by $\mathcal{A}$ the set of atomic propositions of φ. At each instant b, a *Boolean causation (online) monitor* `BCauM` returns a verdict in $\{\ominus, \oplus, \oslash\}$ (called *violation causation*, *satisfaction causation* and *irrelevant*), which is defined as follows,

$$\mathscr{M}(\mathbf{v}_{0:b}, \varphi, \tau) := \begin{cases} \ominus & \text{if } \exists \alpha \in \mathcal{A}.\ \langle \alpha, b\rangle \in \mathrm{E}^{\ominus}(\mathbf{v}_{0:b}, \varphi, \tau) \\ \oplus & \text{if } \exists \alpha \in \mathcal{A}.\ \langle \alpha, b\rangle \in \mathrm{E}^{\oplus}(\mathbf{v}_{0:b}, \varphi, \tau) \\ \oslash & \text{otherwise} \end{cases}$$

An instant b is called a *violation/satisfaction causation instant* if $\mathscr{M}(\mathbf{v}_{0:b}, \varphi, \tau)$ returns $\ominus/\oplus$, or an *irrelevant instant* if $\mathscr{M}(\mathbf{v}_{0:b}, \varphi, \tau)$ returns $\oslash$.

Intuitively, if the current instant b (with the related α) is included in the epoch (thus the signal value at b is relevant to the violation/satisfaction of φ), `BCauM` will report a *violation/satisfaction causation* ($\ominus/\oplus$); otherwise, it will report *irrelevant* ($\oslash$). Notably `BCauM` is non-monotonic, in that even if it reports $\ominus$ or $\oplus$ at some instant b, it may still report $\oslash$ after b. This feature allows `BCauM` to bring more information, e.g., it can detect the end of a violation episode and the start of a new one (i.e., it solves the *violation masking* issue in Sect. 1); see Example 2.

Example 2. Based on the signal diagnostics in Fig. 3, the Boolean causation monitor `BCauM` reports the result shown as in Fig. 4.

Compared to the classic Boolean monitor in Fig. 2, `BCauM` brings more information, in the sense that it detects the end of the violation episode at $b = 30$, by going from $\ominus$ to $\oslash$, when the signal a becomes negative. $\lhd$

Theorem 1 states the relation of `BCauM` with the classic Boolean online monitor.

Theorem 1. The Boolean causation monitor `BCauM` in Definition 6 refines the classic Boolean online monitor in Definition 3, in the following sense:

- $\mathrm{M}(\mathbf{v}_{0:b}, \varphi, \tau) = \bot$ *iff.* $\bigvee_{t\in[0,b]} (\mathscr{M}(\mathbf{v}_{0:t}, \varphi, \tau) = \ominus)$
- $\mathrm{M}(\mathbf{v}_{0:b}, \varphi, \tau) = \top$ *iff.* $\bigvee_{t\in[0,b]} (\mathscr{M}(\mathbf{v}_{0:t}, \varphi, \tau) = \oplus)$
- $\mathrm{M}(\mathbf{v}_{0:b}, \varphi, \tau) = \ ?$ *iff.* $\bigwedge_{t\in[0,b]} (\mathscr{M}(\mathbf{v}_{0:t}, \varphi, \tau) = \oslash)$

Proof. The proof is based on Definitions 5 and 6, Lemma 1 about the monotonicity of classic STL online monitors, and two extra lemmas in the full version [38]. $\square$

4 Quantitative Causation Online Monitor

Although `BCauM` in Sect. 3 is able to solve the *violation masking* issue, it still does not provide enough information about the evolution of the system signals, i.e., it does not solve the *evolution masking* issue introduced in Sect. 1. To tackle this issue, we propose a *quantitative (online) causation monitor* `QCauM` in Definition 7, which is a quantitative extension of `BCauM`. Given a partial signal $\mathbf{v}_{0:b}$, `QCauM` reports a *violation causation distance* $[\mathscr{R}]^{\ominus}(\mathbf{v}_{0:b}, \varphi, \tau)$ and a *satisfaction causation distance* $[\mathscr{R}]^{\oplus}(\mathbf{v}_{0:b}, \varphi, \tau)$, which, respectively, indicate *how far* the signal value at the current instant b is from turning b into a violation causation instant and from turning b into a satisfaction causation instant.

Definition 7 (Quantitative causation monitor (`QCauM`)). Let $\mathbf{v}_{0:b}$ be a partial signal, and φ be an STL specification. At instant b, the quantitative causation monitor `QCauM` returns a *violation causation distance* $[\mathscr{R}]^{\ominus}(\mathbf{v}_{0:b}, \varphi, \tau)$, as follows:

$$[\mathscr{R}]^{\ominus}(\mathbf{v}_{0:b}, \alpha, \tau) := \begin{cases} f(\mathbf{v}_{0:b}(\tau)) & \text{if } b = \tau \\ \mathrm{R}^{\alpha}_{\max} & \text{otherwise} \end{cases}$$

$$[\mathscr{R}]^{\ominus}(\mathbf{v}_{0:b}, \neg\varphi, \tau) := -[\mathscr{R}]^{\oplus}(\mathbf{v}_{0:b}, \varphi, \tau)$$

$$[\mathscr{R}]^{\ominus}(\mathbf{v}_{0:b}, \varphi_1 \wedge \varphi_2, \tau) := \min\left([\mathscr{R}]^{\ominus}(\mathbf{v}_{0:b}, \varphi_1, \tau), [\mathscr{R}]^{\ominus}(\mathbf{v}_{0:b}, \varphi_2, \tau)\right)$$

$$[\mathscr{R}]^{\ominus}(\mathbf{v}_{0:b}, \varphi_1 \vee \varphi_2, \tau) := \min\begin{pmatrix} \max\left([\mathscr{R}]^{\ominus}(\mathbf{v}_{0:b}, \varphi_1, \tau), [\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:b}, \varphi_2, \tau)\right), \\ \max\left([\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:b}, \varphi_1, \tau), [\mathscr{R}]^{\ominus}(\mathbf{v}_{0:b}, \varphi_2, \tau)\right) \end{pmatrix}$$

$$[\mathscr{R}]^{\ominus}(\mathbf{v}_{0:b}, \Box_I \varphi, \tau) := \inf_{t \in \tau + I} \left([\mathscr{R}]^{\ominus}(\mathbf{v}_{0:b}, \varphi, t) \right)$$

$$[\mathscr{R}]^{\ominus}(\mathbf{v}_{0:b}, \Diamond_I \varphi, \tau) := \inf_{t \in \tau + I} \left(\max \left([\mathscr{R}]^{\ominus}(\mathbf{v}_{0:b}, \varphi, t), [\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:b}, \Diamond_I \varphi, \tau) \right) \right)$$

$$[\mathscr{R}]^{\ominus}(\mathbf{v}_{0:b}, \varphi_1 \,\mathcal{U}_I\, \varphi_2, \tau) := \inf_{t \in \tau + I} \left(\max \left(\begin{array}{l} \min \left(\begin{array}{l} \inf_{t' \in [\tau, t)} [\mathscr{R}]^{\ominus}(\mathbf{v}_{0:b}, \varphi_1, t') \\ [\mathscr{R}]^{\ominus}(\mathbf{v}_{0:b}, \varphi_2, t) \end{array} \right) \\ [\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:b}, \varphi_1 \,\mathcal{U}_I\, \varphi_2, \tau) \end{array} \right) \right)$$

and a *satisfaction causation distance* $[\mathscr{R}]^{\oplus}(\mathbf{v}_{0:b}, \varphi, \tau)$, as follows:

$$[\mathscr{R}]^{\oplus}(\mathbf{v}_{0:b}, \alpha, \tau) := \begin{cases} f(\mathbf{v}_{0:b}(\tau)) & \text{if } b = \tau \\ \mathtt{R}^{\alpha}_{\min} & \text{otherwise} \end{cases}$$

$$[\mathscr{R}]^{\oplus}(\mathbf{v}_{0:b}, \neg\varphi, \tau) := -[\mathscr{R}]^{\ominus}(\mathbf{v}_{0:b}, \varphi, \tau)$$

$$[\mathscr{R}]^{\oplus}(\mathbf{v}_{0:b}, \varphi_1 \wedge \varphi_2, \tau) := \max \left(\begin{array}{l} \min \left([\mathscr{R}]^{\oplus}(\mathbf{v}_{0:b}, \varphi_1, \tau), [\mathrm{R}]^{\mathsf{L}}(\mathbf{v}_{0:b}, \varphi_2, \tau) \right), \\ \min \left([\mathrm{R}]^{\mathsf{L}}(\mathbf{v}_{0:b}, \varphi_1, \tau), [\mathscr{R}]^{\oplus}(\mathbf{v}_{0:b}, \varphi_2, \tau) \right) \end{array} \right)$$

$$[\mathscr{R}]^{\oplus}(\mathbf{v}_{0:b}, \varphi_1 \vee \varphi_2, \tau) := \max \left([\mathscr{R}]^{\oplus}(\mathbf{v}_{0:b}, \varphi_1, \tau), [\mathscr{R}]^{\oplus}(\mathbf{v}_{0:b}, \varphi_2, \tau) \right)$$

$$[\mathscr{R}]^{\oplus}(\mathbf{v}_{0:b}, \Box_I \varphi, \tau) := \sup_{t \in \tau + I} \left(\min \left([\mathscr{R}]^{\oplus}(\mathbf{v}_{0:b}, \varphi, t), [\mathrm{R}]^{\mathsf{L}}(\mathbf{v}_{0:b}, \Box_I \varphi, \tau) \right) \right)$$

$$[\mathscr{R}]^{\oplus}(\mathbf{v}_{0:b}, \Diamond_I \varphi, \tau) := \sup_{t \in \tau + I} \left([\mathscr{R}]^{\oplus}(\mathbf{v}_{0:b}, \varphi, t) \right)$$

$$[\mathscr{R}]^{\oplus}(\mathbf{v}_{0:b}, \varphi_1 \,\mathcal{U}_I\, \varphi_2, \tau) := \sup_{t \in \tau + I} \left(\max \left(\begin{array}{l} \min \left(\begin{array}{l} \sup_{t' \in [\tau, t)} [\mathscr{R}]^{\oplus}(\mathbf{v}_{0:b}, \varphi_1, t') \\ \inf_{t' \in [\tau, t)} [\mathrm{R}]^{\mathsf{L}}(\mathbf{v}_{0:b}, \varphi_1, t') \\ [\mathrm{R}]^{\mathsf{L}}(\mathbf{v}_{0:b}, \varphi_2, t) \end{array} \right) \\ \min \left(\begin{array}{l} \inf_{t' \in [\tau, t)} [\mathrm{R}]^{\mathsf{L}}(\mathbf{v}_{0:b}, \varphi_1, t') \\ [\mathscr{R}]^{\oplus}(\mathbf{v}_{0:b}, \varphi_2, t) \end{array} \right) \end{array} \right) \right)$$

Intuitively, a violation causation distance $[\mathscr{R}]^{\ominus}(\mathbf{v}_{0:b}, \varphi, \tau)$ is the spatial distance of the signal value $\mathbf{v}_{0:b}(b)$, at the current instant b, from turning b into a violation causation instant such that b is relevant to the violation of φ (also applied to the satisfaction case dually). It is computed inductively on the structure of φ:

- Case atomic propositions α: if $b = \tau$ (i.e., at which instant α should be evaluated), then the distance of b from being a violation causation instant is $f(\mathbf{v}_{0:b}(b))$; otherwise, if $b \neq \tau$, despite the value of $f(\mathbf{v}_{0:b}(b))$, b can never be a violation causation instant, according to Definition 5, because only $f(\mathbf{v}_{0:b}(\tau))$ is relevant to the violation of α. Hence, the distance will be $\mathtt{R}^{\alpha}_{\max}$;
- Case $\neg\varphi$: b is a violation causation instant for $\neg\varphi$ if b is a satisfaction causation instant for φ, so $[\mathscr{R}]^{\ominus}(\mathbf{v}_{0:b}, \neg\varphi, \tau)$ depends on $[\mathscr{R}]^{\oplus}(\mathbf{v}_{0:b}, \varphi, \tau)$;
- Case $\varphi_1 \wedge \varphi_2$: b is a violation causation instant for $\varphi_1 \wedge \varphi_2$ if b is a violation causation instant for either φ_1 or φ_2, so $[\mathscr{R}]^{\ominus}(\mathbf{v}_{0:b}, \varphi_1 \wedge \varphi_2, \tau)$ depends on the minimum between $[\mathscr{R}]^{\ominus}(\mathbf{v}_{0:b}, \varphi_1, \tau)$ and $[\mathscr{R}]^{\ominus}(\mathbf{v}_{0:b}, \varphi_2, \tau)$;

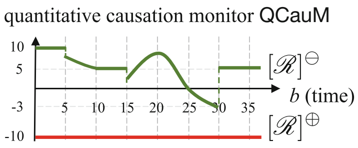


Fig. 5. Quantitative causation monitor (`QCauM`) result for Example 1

- Case $\varphi_1 \vee \varphi_2$: b is a violation causation instant for $\varphi_1 \vee \varphi_2$ if, first, $\varphi_1 \vee \varphi_2$ has been violated at b, and second, b is the violation causation instant for either φ_1 or φ_2. Hence, $[\mathscr{R}]^{\ominus}(\mathbf{v}_{0:b}, \varphi_1 \vee \varphi_2, \tau)$ depend on both the violation status (measured by $[\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:b}, \varphi_i, \tau)$) of one sub-formula and the violation causation distance of the other sub-formula;
- Case $\Box_I\varphi$: b is a violation causation instant for $\Box_I\varphi$ if b is the violation causation instant for the sub-formula φ evaluated at any instant in $\tau + I$. So, $[\mathscr{R}]^{\ominus}(\mathbf{v}_{0:b}, \Box_I\varphi, \tau)$ depends on the infimum of the violation causation distances regarding φ evaluated at the instants in $\tau + I$;
- Case $\Diamond_I\varphi$: b is a violation causation instant for $\Diamond_I\varphi$ if, first, $\Diamond_I\varphi$ has been violated at b, and second, b is a violation causation instant for the sub-formula φ evaluated at any instant in $\tau + I$. So, $[\mathscr{R}]^{\ominus}(\mathbf{v}_{0:b}, \Diamond_I\varphi, \tau)$ depends on both the violation status of $\Diamond_I\varphi$ (measured by $[\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:b}, \Diamond_I\varphi, \tau)$) and the infimum of the violation causation distances of φ evaluated in $\tau + I$.
- Case $\varphi_1\mathcal{U}_I\varphi_2$: $[\mathscr{R}]^{\ominus}(\mathbf{v}_{0:b}, \varphi_1\,\mathcal{U}_I\,\varphi_2, \tau)$ depends on, first, the violation status of the whole formula (measured by $[\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:b}, \varphi_1\mathcal{U}_I\varphi_2, \tau)$), and also, the infimum of the violation causation distances regarding the evaluation of "φ_1 holds until φ_2" at each instant in $\tau + I$.

Similarly, we can also compute the satisfaction causation distance. We use Example 3 to illustrate the quantitative causation monitor for the signals in Example 1.

Example 3. Consider the quantitative causation monitor for the signals in Example 1. At $b = 30$, the violation causation distance is computed as:

$$
\begin{aligned}
&[\mathscr{R}]^{\ominus}(\mathbf{v}_{0:30},\varphi,0)=\inf_{t\in[0,100]}[\mathscr{R}]^{\ominus}\left(\mathbf{v}_{0:30},\varphi',t\right)\\
&=\inf_{t\in[0,100]}\left(\min\left(\begin{matrix}\max\left([\mathscr{R}]^{\ominus}(\mathbf{v}_{0:30},\varphi_1,t),[\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:30},\varphi_2,t)\right),\\ \max\left([\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:30},\varphi_1,t),[\mathscr{R}]^{\ominus}(\mathbf{v}_{0:30},\varphi_2,t)\right)\end{matrix}\right)\right)\\
&=\inf_{t\in[0,100]}\left(\min\left(\begin{matrix}\max\left(-[\mathscr{R}]^{\oplus}(\mathbf{v}_{0:30},\alpha_1,t),\sup\limits_{t'\in t+[0,5]}[\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:30},\alpha_2,t')\right)\\ \max\left(-[\mathrm{R}]^{\mathsf{L}}(\mathbf{v}_{0:30},\alpha_1,t),\max\left(\begin{matrix}[\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:30},\varphi_2,t),\\ \inf\limits_{t'\in t+[0,5]}[\mathscr{R}]^{\ominus}\left(\mathbf{v}_{0:30},\alpha_2,t'\right)\end{matrix}\right)\right)\end{matrix}\right)\right)\\
&=\max\left(-[\mathrm{R}]^{\mathsf{L}}(\mathbf{v}_{0:30},\alpha_1,25),[\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:30},\varphi_2,25),\inf_{t'\in[25,30]}[\mathscr{R}]^{\ominus}\left(\mathbf{v}_{0:30},\alpha_2,t'\right)\right)\\
&=\max(-3,-3,-5)=-3.
\end{aligned}
$$

Similarly, at $b = 35$, the violation causation distance $[\mathscr{R}]^{\ominus}(\mathbf{v}_{0:35}, \varphi, 0) = 5$. See the result of `QCauM` shown in Fig. 5. Compared to `ClaM` in Fig. 2, it is evident that `QCauM` provides much more information about the system evolution, e.g., it can report that, in the interval $[15, 20]$, the system satisfies the specification "more", as the `speed` decreases. $\lhd$

By using the violation and satisfaction causation distances reported by `QCauM` jointly, we can infer the verdict of `BCauM`, as indicated by Theorem 2.

Theorem 2. The quantitative causation monitor `QCauM` in Definition 7 refines the Boolean causation monitor `BCauM` in Definition 6, in the sense that:

- if $[\mathscr{R}]^{\ominus}(\mathbf{v}_{0:b}, \varphi, \tau) < 0$, it implies $\mathscr{M}(\mathbf{v}_{0:b}, \varphi, \tau) = \ominus$;
- if $[\mathscr{R}]^{\oplus}(\mathbf{v}_{0:b}, \varphi, \tau) > 0$, it implies $\mathscr{M}(\mathbf{v}_{0:b}, \varphi, \tau) = \oplus$;
- if $[\mathscr{R}]^{\ominus}(\mathbf{v}_{0:b}, \varphi, \tau) > 0$ and $[\mathscr{R}]^{\oplus}(\mathbf{v}_{0:b}, \varphi, \tau) < 0$, it implies $\mathscr{M}(\mathbf{v}_{0:b}, \varphi, \tau) = \oslash$.

Proof. The proof is generally based on mathematical induction. First, by Definition 7 and Definition 5, it is straightforward that Theorem 2 holds for the atomic propositions.

Then, assuming that Theorem 2 holds for an arbitrary formula φ, we prove that Theorem 2 also holds for the composite formula φ' constructed by applying STL operators to φ. The complete proof for all three cases is shown in the full version [38].

As an instance, we show the proof for the first case with $\varphi' = \varphi_1 \vee \varphi_2$, i.e., we prove that $[\mathscr{R}]^{\ominus}(\mathbf{v}_{0:b}, \varphi_1 \vee \varphi_2, \tau) < 0$ implies $\mathscr{M}(\mathbf{v}_{0:b}, \varphi_1 \vee \varphi_2, \tau) = \ominus$.

$$\begin{aligned}
& [\mathscr{R}]^{\ominus}(\mathbf{v}_{0:b}, \varphi_1 \vee \varphi_2, \tau) < 0 & \\
\Rightarrow & \max\Big([\mathscr{R}]^{\ominus}(\mathbf{v}_{0:b}, \varphi_1, \tau), [\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:b}, \varphi_2, \tau)\Big) < 0 & \text{(by Def. 7 and w.l.o.g.)} \\
\Rightarrow & [\mathscr{R}]^{\ominus}(\mathbf{v}_{0:b}, \varphi_1, \tau) < 0 & \text{(by def. of max)} \\
\Rightarrow & \mathscr{M}(\mathbf{v}_{0:b}, \varphi_1, \tau) = \ominus & \text{(by assumption)} \\
\Rightarrow & \mathrm{E}^{\ominus}(\mathbf{v}_{0:b}, \varphi_1 \vee \varphi_2, \tau) \supseteq \mathrm{E}^{\ominus}(\mathbf{v}_{0:b}, \varphi_1, \tau) & \text{(by Def. 5 and Thm. 1)} \\
\Rightarrow & \exists \alpha.\ \langle \alpha, b \rangle \in \mathrm{E}^{\ominus}(\mathbf{v}_{0:b}, \varphi_1 \vee \varphi_2, \tau) & \text{(by def. of } \supseteq) \\
\Rightarrow & \mathscr{M}(\mathbf{v}_{0:b}, \varphi_1 \vee \varphi_2, \tau) = \ominus & \text{(by Def. 6)}
\end{aligned}$$

$\square$

The relation between the quantitative causation monitor `QCauM` and the Boolean causation monitor `BCauM`, disclosed by Theorem 2, can be visualized by the comparison between Fig. 5 and Fig. 4. Indeed, when the violation causation distance reported by `QCauM` is negative in Fig. 5, `BCauM` reports $\ominus$ in Fig. 4.

Next, we present Theorem 3, which states the relation between the quantitative causation monitor `QCauM` and the classic quantitative monitor `ClaM`.

Theorem 3. The quantitative causation monitor `QCauM` in Definition 7 refines the classic quantitative online monitor `ClaM` in Definition 4, in the sense that, the monitoring results of `ClaM` can be reconstructed from the results of `QCauM`, as follows:

$$[\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:b},\varphi,\tau) = \inf_{t\in[0,b]} [\mathscr{R}]^{\ominus}(\mathbf{v}_{0:t},\varphi,\tau) \tag{1}$$

$$[\mathrm{R}]^{\mathsf{L}}(\mathbf{v}_{0:b},\varphi,\tau) = \sup_{t\in[0,b]} [\mathscr{R}]^{\oplus}(\mathbf{v}_{0:t},\varphi,\tau) \tag{2}$$

Proof. The proof is generally based on mathematical induction. First, by Definition 7 and Definition 4, it is straightforward that Theorem 3 holds for the atomic propositions.

Then, we make the global assumption that Theorem 3 holds for an arbitrary formula φ, i.e., both the two cases $\inf_{t\in[0,b]} [\mathscr{R}]^{\ominus}(\mathbf{v}_{0:t},\varphi,\tau) = [\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:b},\varphi,\tau)$ and $\sup_{t\in[0,b]} [\mathscr{R}]^{\oplus}(\mathbf{v}_{0:t},\varphi,\tau) = [\mathrm{R}]^{\mathsf{L}}(\mathbf{v}_{0:b},\varphi,\tau)$ hold. Based on this assumption, we prove that Theorem 3 also holds for the composite formula φ' constructed by applying STL operators to φ.

As an instance, we prove $\inf_{t\in[0,b]} [\mathscr{R}]^{\ominus}(\mathbf{v}_{0:t},\varphi',\tau) = [\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:b},\varphi',\tau)$ with $\varphi' = \varphi_1 \vee \varphi_2$ as follows. The complete proof is presented in the full version [38].

First, if $b = \tau$, it holds that:

$$\begin{aligned}
&\inf_{t\in[0,b]} [\mathscr{R}]^{\ominus}(\mathbf{v}_{0:t},\varphi_1\vee\varphi_2,\tau) = [\mathscr{R}]^{\ominus}(\mathbf{v}_{0:\tau},\varphi_1\vee\varphi_2,\tau) \\
&= \max\Big([\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:\tau},\varphi_1,\tau), [\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:\tau},\varphi_2,\tau)\Big) && \text{(by Def. 7 and global assump.)} \\
&= [\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:b},\varphi_1\vee\varphi_2,\tau) && \text{(by Def. 4)}
\end{aligned}$$

Then, we make a local assumption that, given an arbitrary b, it holds that $\inf_{t\in[0,b]} [\mathscr{R}]^{\ominus}(\mathbf{v}_{0:t},\varphi_1\vee\varphi_2,\tau) = [\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:b},\varphi_1\vee\varphi_2,\tau)$. We prove that, for b' which is the next sampling point to b, it holds that,

$$\begin{aligned}
&\inf_{t\in[0,b']} [\mathscr{R}]^{\ominus}(\mathbf{v}_{0:t},\varphi_1\vee\varphi_2,\tau) \\
&= \min\Big([\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:b},\varphi_1\vee\varphi_2,\tau), [\mathscr{R}]^{\ominus}(\mathbf{v}_{0:b'},\varphi_1\vee\varphi_2,\tau)\Big) && \text{(by local assump.)} \\
&= \min\begin{pmatrix}
\max\Big([\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:b},\varphi_1,\tau), [\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:b},\varphi_2,\tau)\Big), \\
\max\Big([\mathscr{R}]^{\ominus}(\mathbf{v}_{0:b'},\varphi_1,\tau), [\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:b'},\varphi_2,\tau)\Big), \\
\max\Big([\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:b'},\varphi_1,\tau), [\mathscr{R}]^{\ominus}(\mathbf{v}_{0:b'},\varphi_2,\tau)\Big)
\end{pmatrix} && \text{(by Defs. 4 \& 7)} \\
&= \min\begin{pmatrix}
\max\Big([\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:b},\varphi_1,\tau), [\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:b},\varphi_2,\tau)\Big), \\
\max\Big([\mathscr{R}]^{\ominus}(\mathbf{v}_{0:b'},\varphi_1,\tau), [\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:b},\varphi_2,\tau)\Big), \\
\max\Big([\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:b},\varphi_1,\tau), [\mathscr{R}]^{\ominus}(\mathbf{v}_{0:b'},\varphi_2,\tau)\Big), \\
\max([\mathscr{R}]^{\ominus}(\mathbf{v}_{0:b'},\varphi_1,\tau), [\mathscr{R}]^{\ominus}(\mathbf{v}_{0:b'},\varphi_2,\tau))
\end{pmatrix} && \text{(by global assump.)} \\
&= \max\begin{pmatrix}
\min\Big([\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:b},\varphi_1,\tau), [\mathscr{R}]^{\ominus}(\mathbf{v}_{0:b'},\varphi_1,\tau)\Big), \\
\min\Big([\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:b},\varphi_2,\tau), [\mathscr{R}]^{\ominus}(\mathbf{v}_{0:b'},\varphi_2,\tau)\Big)
\end{pmatrix} && \text{(by def. of min, max)} \\
&= \max\Big([\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:b'},\varphi_1,\tau), [\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:b'},\varphi_2,\tau)\Big) && \text{(by global assump.)} \\
&= [\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:b'},\varphi_1\vee\varphi_2,\tau) && \text{(by Def. 4)}
\end{aligned}$$

□

$$\begin{tikzcd}
(\mathbf{v},\tau) \models \varphi & \mathrm{M}(\mathbf{v}_{0:b},\varphi,\tau) \arrow[l] & \mathscr{M}(\mathbf{v}_{0:b},\varphi,\tau) \arrow[l, "\text{Thm. 1}"'] \\
\mathrm{R}(\mathbf{v},\varphi,\tau) \arrow[u] & \substack{[\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:b},\varphi,\tau) \\ [\mathrm{R}]^{\mathsf{L}}(\mathbf{v}_{0:b},\varphi,\tau)} \arrow[l] \arrow[u] & \substack{[\mathscr{R}]^{\ominus}(\mathbf{v}_{0:b},\varphi,\tau) \\ [\mathscr{R}]^{\oplus}(\mathbf{v}_{0:b},\varphi,\tau)} \arrow[l, "\text{Thm. 3}"'] \arrow[u, "\text{Thm. 2}"']
\end{tikzcd}$$

Fig. 6. Refinement among STL monitors

Theorem 3 shows that the result $[\mathrm{R}]^{\mathsf{U}}(\mathbf{v}_{0:b},\varphi,\tau)$ of `ClaM` can be derived from the result of `QCauM` by applying $\inf_{t\in[0,b]} [\mathscr{R}]^{\ominus}(\mathbf{v}_{0:b},\varphi,t)$. For instance, comparing the results of `QCauM` in Fig. 5 and the results of `ClaM` in Fig. 2, we can find that the results in Fig. 2 can be reconstructed by using the results in Fig. 5.

Remark 1. Figure 6 shows the refinement relations between the six STL monitoring approaches. The left column lists the offline monitoring approaches derived directly from the Boolean and quantitative semantics of STL respectively. The middle column shows the classic online monitoring approaches. Our two causation monitors, namely `BCauM` and `QCauM`, are given in the column on the right. Given a pair (A, B) of the approaches, $A \leftarrow B$ indicates that the approach B refines the approach A, in the sense that B can deliver more information than A, and the information delivered by A can be derived from the information delivered by B. It is clear that the refinement relation in the figure ensures transitivity. Note that blue arrows are contributed by this paper. As shown by Fig. 6, the relation between `BCauM` and `QCauM` is analogous to that between the Boolean and quantitative semantics of STL.

5 Experimental Evaluation

We implemented a tool[3] for our two causation monitors. It is built on the top of Breach [15], a widely used tool for monitoring and testing of hybrid systems [18]. Being consistent with Breach, the monitors target the output signals given by Simulink models, as an additional block. Experiments were executed on a MacOS machine, 1.4 GHz Quad-Core Intel Core-i5, 8 GB RAM, using Breach v1.10.0.

5.1 Experiment Setting

Benchmarks. We perform the experiments on the following two benchmarks. *Abstract Fuel Control (AFC)* is a powertrain control system from Toyota [27], which has been widely used as a benchmark in the hybrid system community [18–20]. The system outputs the *air-to-fuel* ratio `AF`, and requires that the deviation of `AF` from its reference value `AFref` should not be too large. Specifically, we consider the following properties from different perspectives:

– $\varphi_1^{\mathsf{AFC}} := \Box_{[10,50]}(|\mathtt{AF} - \mathtt{AFref}| < 0.1)$: the deviation should always be small;

[3] Available at https://github.com/choshina/STL-causation-monitor, and Zenodo [39].

- $\varphi_2^{\mathsf{AFC}} := \Box_{[10,48.5]}\Diamond_{[0,1.5]}(|\mathtt{AF} - \mathtt{AFref}| < 0.08)$: a large deviation should not last for too long time;
- $\varphi_3^{\mathsf{AFC}} := \Box_{[10,48]}(|\mathtt{AF} - \mathtt{AFref}| > 0.08 \rightarrow \Diamond_{[0,2]}(|\mathtt{AF} - \mathtt{AFref}| < 0.08))$: whenever the deviation is too large, it should recover to the normal status soon.

Automatic transmission (AT) is a widely-used benchmark [18–20], implementing the transmission controller of an automotive system. It outputs the `gear`, `speed` and `RPM` of the vehicle, which are required to satisfy this safety requirement:

- $\varphi_1^{\mathsf{AT}} := \Box_{[0,27]}(\mathtt{speed} > 50 \rightarrow \Diamond_{[1,3]}(\mathtt{RPM} < 3000))$: whenever the `speed` is higher than 50, the `RPM` should be below 3000 in three time units.

Baseline and Experimental Design. In order to assess our two proposed monitors (the Boolean causation monitor `BCauM` in Definition 6, and the quantitative causation monitor `QCauM` in Definition 7), we compare them with two baseline monitors: the classic quantitative robustness monitor `ClaM` (see Definition 4); and the state-of-the-art approach *monitor with reset* `ResM` [40], that, once the signal violates the specification, resets at that point and forgets the previous partial signal.

Given a model and a specification, we generate input signals by randomly sampling in the input space and feed them to the model. The online output signals are given as inputs to the monitors and the monitoring results are collected. We generate 10 input signals for each model and specification. To account for fluctuation of monitoring times in different repetitions[4], for each signal, the experiment has been executed 10 times, and we report average results.

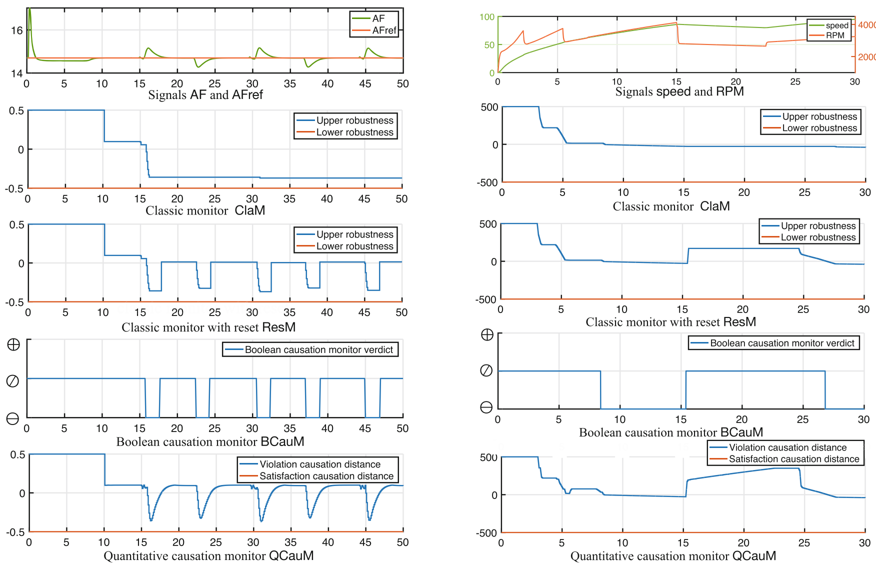
5.2 Evaluation

Qualitative Evaluation. We here show the type of information provided by the different monitors. As an example, Fig. 7 reports, for two specifications of the two models, the system output signal (in the top of the two sub-figures), and the monitoring results of the compared monitors. We notice that signals of both models (top plots) violate the corresponding specifications in multiple points. Let us consider monitoring results of φ_1^{AFC}; similar observations apply to φ_1^{AT}.

When using the `ClaM`, only the first violation right after time 15 is detected (the upper bound of robustness becomes negative); after that, the upper bound remains constant, without reporting that the system recovers from violation at around time 17, and that the specification is violated again four more times.

Instead, we notice that the monitor with reset `ResM` is able to detect all the violations (as the upper bound becomes greater than 0 when the violation episode ends), but it does not properly report the margin of robustness; indeed, during the violation episodes, it reports a constant value of around -0.4 for the upper bound, but the system violates the specification with different degrees of severity in these intervals; in a similar way, when the specification is satisfied around after time 17, the upper bound is just above 0, but actually the system

[4] Note that only the monitoring time changes across different repetitions; monitoring results are instead always the same, as monitoring is deterministic for a given signal.



(a) Specification φ_1^{AFC} and signal #4 (b) Specification φ_1^{AT} and signal #8

Fig. 7. Examples of the information provided by the different monitors

Table 1. Experimental results – Average (avg.) and standard deviation (stdv.) of monitoring and simulation times (ms)

	`ClaM`				`ResM`				`BCauM`				`QCauM`			
	monitor		total		monitor		total		monitor		total		monitor		total	
	avg.	stdv.	avg.	stdv.	avg.	stdv.	avg.	stdv.	avg.	stdv.	avg.	stdv.	avg.	stdv.	avg.	stdv.
φ_1^{AFC}	14.6	0.1	982.8	3.5	8.8	2.4	981.3	6.7	36.9	5.4	1009.7	16.5	15.1	0.1	981.9	4.4
φ_2^{AFC}	26.8	0.2	998.5	9.0	20.2	5.2	988.0	9.9	50.4	22.4	1023.9	25.1	27.4	0.2	999.5	8.2
φ_3^{AFC}	42.0	0.3	1016.5	8.9	45.5	4.8	1016.9	7.5	48.4	6.2	1021.2	7.9	81.0	1.2	1060.1	5.3
φ_1^{AT}	16.7	0.2	966.0	2.6	24.0	17.0	980.4	24.2	96.1	82.6	1065.2	93.4	31.2	0.6	985.0	7.5

satisfies the specification with different margins. As a consequence, `ResM` provides sharp changes of the robustness upper bound that do not faithfully reflect the system evolution.

We notice that the Boolean causation monitor `BCauM` only reports information about the violation episodes, but not on the degree of violation/satisfaction. Instead, the quantitative causation monitor `QCauM` is able to provide a very detailed information, not only reporting all the violation episodes, but also properly characterizing the degree with which the specification is violated or satisfied. Indeed, in `QCauM`, the violation causation distance smoothly increases from violation to satisfaction, so faithfully reflecting the system evolution.

Quantitative Assessment of Monitoring Time. We discuss the computation cost of doing the monitoring.

Table 2. Experimental results of the four monitoring approaches – Monitoring time (ms) – $\Delta A = {}^{(\texttt{QCauM}-A)}/_{A}$

φ_1^{AFC}	ClaM	ResM	BCauM	QCauM	QCauM stat. (%)		
					ΔClaM	ΔResM	ΔBCauM
#1	14.5	8.2	37.4	15.2	4.8	85.4	−59.4
#2	14.5	8.1	39.9	15.0	3.4	85.2	−62.4
#3	14.8	8.0	38.2	15.0	1.4	87.5	−60.7
#4	14.7	8.5	38.8	15.3	4.1	80.0	−60.6
#5	14.6	8.0	37.3	14.9	2.1	86.3	−60.1
#6	14.6	8.2	37.6	15.1	3.4	84.1	−59.8
#7	14.6	15.5	21.6	15.0	2.7	-3.2	−30.6
#8	14.7	7.9	39.5	15.0	2.0	89.9	−62.0
#9	14.6	7.8	39.9	15.1	3.4	93.6	−62.2
#10	14.5	8.0	38.4	15.1	4.1	88.8	−60.7

φ_2^{AFC}	ClaM	ResM	BCauM	QCauM	QCauM stat. (%)		
					ΔClaM	ΔResM	ΔBCauM
#1	26.8	19.8	45.9	27.4	2.2	38.4	−40.3
#2	27.1	27.3	27.6	27.8	2.6	1.8	0.7
#3	26.6	26.2	30.0	27.5	3.4	5.0	−8.3
#4	26.6	14.2	107.2	27.0	1.5	90.1	−74.8
#5	26.7	15.8	50.9	27.3	2.2	72.8	−46.4
#6	26.6	15.8	56.4	27.2	2.3	72.2	−51.8
#7	26.8	25.4	33.5	27.5	2.6	8.3	−17.9
#8	26.9	17.0	51.9	27.4	1.9	61.2	−47.2
#9	27.1	25.1	50.9	27.6	1.8	10.0	−45.8
#10	26.7	15.8	50.1	27.3	2.2	72.8	−45.5

φ_3^{AFC}	ClaM	ResM	BCauM	QCauM	QCauM stat. (%)		
					ΔClaM	ΔResM	ΔBCauM
#1	42.1	49.2	49.1	81.2	92.9	65.0	65.4
#2	42.5	42.2	42.2	82.1	93.2	94.5	94.5
#3	41.8	48.8	48.8	81.5	95.0	67.0	67.0
#4	42.0	34.9	63.4	78.8	87.6	125.8	24.3
#5	41.7	48.9	48.7	79.6	90.9	62.8	63.4
#6	41.7	48.5	48.7	79.7	91.1	64.3	63.7
#7	42.3	42.7	42.5	81.9	93.6	91.8	92.7
#8	42.1	42.2	42.0	81.6	93.8	93.4	94.3
#9	42.3	49.1	49.3	82.6	95.3	68.2	67.5
#10	41.6	48.6	49.1	80.8	94.2	66.3	64.6

φ_1^{AT}	ClaM	ResM	BCauM	QCauM	QCauM stat. (%)		
					ΔClaM	ΔResM	ΔBCauM
#1	16.9	30.7	29.6	32.1	89.9	4.6	8.4
#2	16.7	17.4	17.4	31.9	91.0	83.3	83.3
#3	16.7	16.8	253.4	31.0	85.6	84.5	−87.8
#4	16.9	69.7	70.2	31.8	88.2	−54.4	−54.7
#5	16.8	19.6	135.9	31.0	84.5	58.2	−77.2
#6	16.5	26.5	200.5	30.2	83.0	14.0	−84.9
#7	16.6	14.6	37.9	31.0	86.7	112.3	−18.2
#8	16.8	16.4	143.8	31.4	86.9	91.5	−78.2
#9	16.3	13.9	38.6	31.0	90.2	123.0	−19.7
#10	16.5	14.2	33.2	30.9	87.3	117.6	−6.9

In Table 1, we observe that, for all the monitors, the *monitor*ing time is much lower than the *total* time (system execution + monitoring). It shows that, for this type of systems, the monitoring overhead is negligible. Still, we compare the execution costs for the different monitors. Table 2 reports the monitoring times of all the monitors for each specification and each signal. Moreover, it reports the percentage difference between the quantitative causation monitor QCauM (the most informative one) and the other monitors.

We first observe that ResM and BCauM have, for the same specification, high variance of the monitoring times across different signals. ClaM and QCauM, instead, provide very consistent monitoring times. This is confirmed by the standard deviation results in Table 1. The consistent monitoring cost of QCauM is a good property, as the designers of the monitor can precisely forecast how long the monitoring will take, and design the overall system accordingly.

We observe that QCauM is negligibly slower than ClaM for φ_1^{AFC} and φ_2^{AFC}, and at most twice slower for the other two specifications. This additional monitoring cost is acceptable, given the additional information provided by QCauM. Compared to ResM, QCauM is usually slower (at most around the double); also in this case, as QCauM provides more information than ResM, the cost is acceptable.

Compared to the Boolean causation monitor BCauM, QCauM is usually faster, as it does not have to collect epochs, which is a costly operation. However, we observe that it is slower in φ_3^{AFC}, because, in this specification, most of the signals do not violate it (and so also BCauM does not collect epochs in this case).

To conclude, `QCauM` is a monitor able to provide much more information that exiting monitors, with an acceptable overhead in terms of monitoring time.

6 Related Work

Monitoring of STL. Monitoring can be performed either offline or online. Offline monitoring [16,30,33] targets complete traces and returns either `true` or `false`. In contrast, online monitoring deals with the partial traces, and thus a three-valued semantics was introduced for LTL monitoring [7,8], and in further for MTL and STL qualitative online monitoring [24,31], to handle the situation where neither of the conclusiveness can be made. In usual, the quantitative online monitoring provides a quantitative value or a robust satisfaction interval [12–14,25,26]. Based on them, several tools have been developed, e.g., AMT [32,33], Breach [15], S-Taliro [1], etc. We refer to the survey [3] for comprehensive introduction. Recently, in [35], Qin and Deshmukh propose clairvoyant monitoring to forecast future signal values and give probabilistic bounds on the specification validity. In [2], an online monitoring is proposed for perception systems with Spatio-temporal Perception Logic [23].

Monotonicity Issue. However, most of these works do not handle the monotonicity issue stated in this paper. In [10], Cimatti et al. propose an assumption-based monitoring framework for LTL. It takes the user expertise into account and allows the monitor *resettable*, in the sense that it can restart from any discrete time point. In [37], a recovery feature is introduced in their online monitor [25]. However, the technique is an application-specific approach, rather than a general framework. In [40], a reset mechanism is proposed for STL online monitor. However, as experimentally evaluated in Sect. 5, it essentially provides a solution for the Boolean semantics and still holds monotonicity between two resetting points.

Signal Diagnostics. Signal diagnostics [5,22,32] is originally used in an offline manner, for the purpose of fault localization and system debugging. In [22], the authors propose an approach to automatically address the single evaluations (namely, epochs) that account for the satisfaction/violation of an STL specification, for a complete trace. This information can be further used as a reference for detecting the root cause of the bugs in the CPS systems [5,6,32]. The online version of signal diagnostics, which is the basis of our Boolean causation monitor, is introduced in [40]. However, we show in Sect. 5 that the monitor based on this technique is often costly, and not able to deliver the quantitative runtime information compared to the quantitative causation monitor.

7 Conclusion and Future Work

In this paper, we propose a new way of doing STL monitoring based on causation that is able to provide more information than classic monitoring based on

STL robustness. Concretely, we propose two causation monitors, namely `BCauM` and `QCauM`. In particular, `BCauM` intuitively explains the concept of "causation" monitoring, and thus paves the path to `QCauM` that is more practically valuable. We further prove the relation between the proposed causation monitors and the classic ones.

As future work, we plan to improve the efficiency the monitoring, by avoiding some unnecessary computations for some instants. Moreover, we plan to apply it to the monitoring of real-world systems.

References

1. Annpureddy, Y., Liu, C., Fainekos, G., Sankaranarayanan, S.: S-TaLiRo: a tool for temporal logic falsification for hybrid systems. In: Abdulla, P.A., Leino, K.R.M. (eds.) TACAS 2011. LNCS, vol. 6605, pp. 254–257. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-19835-9_21
2. Balakrishnan, A., Deshmukh, J., Hoxha, B., Yamaguchi, T., Fainekos, G.: PerceMon: online monitoring for perception systems. In: Feng, L., Fisman, D. (eds.) RV 2021. LNCS, vol. 12974, pp. 297–308. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-88494-9_18
3. Bartocci, E., et al.: Specification-based monitoring of cyber-physical systems: a survey on theory, tools and applications. In: Bartocci, E., Falcone, Y. (eds.) Lectures on Runtime Verification. LNCS, vol. 10457, pp. 135–175. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-75632-5_5
4. Bartocci, E., Falcone, Y. (eds.): Lectures on Runtime Verification. LNCS, vol. 10457. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-75632-5
5. Bartocci, E., Ferrère, T., Manjunath, N., Ničković, D.: Localizing faults in Simulink/Stateflow models with STL. In: HSCC 2018, pp. 197–206. ACM (2018). https://doi.org/10.1145/3178126.3178131
6. Bartocci, E., Manjunath, N., Mariani, L., Mateis, C., Ničković, D.: CPSDebug: automatic failure explanation in CPS models. Int. J. Softw. Tools Technol. Transfer **23**(5), 783–796 (2020). https://doi.org/10.1007/s10009-020-00599-4
7. Bauer, A., Leucker, M., Schallhart, C.: Monitoring of real-time properties. In: Arun-Kumar, S., Garg, N. (eds.) FSTTCS 2006. LNCS, vol. 4337, pp. 260–272. Springer, Heidelberg (2006). https://doi.org/10.1007/11944836_25
8. Bauer, A., Leucker, M., Schallhart, C.: Runtime verification for LTL and TLTL. ACM Trans. Softw. Eng. Methodol. **20**(4), 1–64 (2011). https://doi.org/10.1145/2000799.2000800
9. Ciccone, L., Dagnino, F., Ferrando, A.: Ain't no stopping us monitoring now. arXiv preprint arXiv:2211.11544 (2022)
10. Cimatti, A., Tian, C., Tonetta, S.: Assumption-based runtime verification with partial observability and resets. In: Finkbeiner, B., Mariani, L. (eds.) RV 2019. LNCS, vol. 11757, pp. 165–184. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-32079-9_10
11. Decker, N., Leucker, M., Thoma, D.: Impartiality and anticipation for monitoring of visibly context-free properties. In: Legay, A., Bensalem, S. (eds.) RV 2013. LNCS, vol. 8174, pp. 183–200. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40787-1_11

12. Deshmukh, J.V., Donzé, A., Ghosh, S., Jin, X., Juniwal, G., Seshia, S.A.: Robust online monitoring of signal temporal logic. Formal Methods Syst. Des. **51**(1), 5–30 (2017). https://doi.org/10.1007/s10703-017-0286-7
13. Dokhanchi, A., Hoxha, B., Fainekos, G.: On-line monitoring for temporal logic robustness. In: Bonakdarpour, B., Smolka, S.A. (eds.) RV 2014. LNCS, vol. 8734, pp. 231–246. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-11164-3_19
14. Dokhanchi, A., Hoxha, B., Fainekos, G.: Metric interval temporal logic specification elicitation and debugging. In: MEMOCODE 2015, pp. 70–79. IEEE (2015). https://doi.org/10.1109/MEMCOD.2015.7340472
15. Donzé, A.: Breach, a toolbox for verification and parameter synthesis of hybrid systems. In: Touili, T., Cook, B., Jackson, P. (eds.) CAV 2010. LNCS, vol. 6174, pp. 167–170. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-14295-6_17
16. Donzé, A., Ferrère, T., Maler, O.: Efficient robust monitoring for STL. In: Sharygina, N., Veith, H. (eds.) CAV 2013. LNCS, vol. 8044, pp. 264–279. Springer, Cham (2013). https://doi.org/10.1007/978-3-642-39799-8_19
17. Donzé, A., Maler, O.: Robust satisfaction of temporal logic over real-valued signals. In: Chatterjee, K., Henzinger, T.A. (eds.) FORMATS 2010. LNCS, vol. 6246, pp. 92–106. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15297-9_9
18. Ernst, G., et al.: ARCH-COMP 2021 category report: falsification with validation of results. In: Frehse, G., Althoff, M. (eds.) 8th International Workshop on Applied Verification of Continuous and Hybrid Systems (ARCH21). EPiC Series in Computing, vol. 80, pp. 133–152. EasyChair (2021). https://doi.org/10.29007/xwl1
19. Ernst, G., et al.: ARCH-COMP 2020 category report: falsification. In: 7th International Workshop on Applied Verification of Continuous and Hybrid Systems (ARCH20). EPiC Series in Computing, vol. 74, pp. 140–152. EasyChair (2020). https://doi.org/10.29007/trr1
20. Ernst, G., et al.: ARCH-COMP 2022 category report: falsification with unbounded resources. In: Frehse, G., Althoff, M., Schoitsch, E., Guiochet, J. (eds.) Proceedings of 9th International Workshop on Applied Verification of Continuous and Hybrid Systems (ARCH22). EPiC Series in Computing, vol. 90, pp. 204–221. EasyChair (2022). https://doi.org/10.29007/fhnk
21. Fainekos, G.E., Pappas, G.J.: Robustness of temporal logic specifications for continuous-time signals. Theoret. Comput. Sci. **410**(42), 4262–4291 (2009). https://doi.org/10.1016/j.tcs.2009.06.021
22. Ferrère, T., Maler, O., Ničković, D.: Trace diagnostics using temporal implicants. In: Finkbeiner, B., Pu, G., Zhang, L. (eds.) ATVA 2015. LNCS, vol. 9364, pp. 241–258. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24953-7_20
23. Hekmatnejad, M., Hoxha, B., Deshmukh, J.V., Yang, Y., Fainekos, G.: Formalizing and evaluating requirements of perception systems for automated vehicles using spatio-temporal perception logic (2022). https://doi.org/10.48550/arxiv.2206.14372
24. Ho, H.-M., Ouaknine, J., Worrell, J.: Online monitoring of metric temporal logic. In: Bonakdarpour, B., Smolka, S.A. (eds.) RV 2014. LNCS, vol. 8734, pp. 178–192. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-11164-3_15
25. Jakšić, S., Bartocci, E., Grosu, R., Kloibhofer, R., Nguyen, T., Ničkovié, D.: From signal temporal logic to FPGA monitors. In: MEMOCODE 2015, pp. 218–227. IEEE (2015). https://doi.org/10.1109/MEMCOD.2015.7340489

26. Jakšić, S., Bartocci, E., Grosu, R., Nguyen, T., Ničković, D.: Quantitative monitoring of STL with edit distance. Formal Methods Syst. Des. **53**(1), 83–112 (2018). https://doi.org/10.1007/s10703-018-0319-x
27. Jin, X., Deshmukh, J.V., Kapinski, J., Ueda, K., Butts, K.: Powertrain control verification benchmark. In: HSCC 2014, pp. 253–262. ACM (2014). https://doi.org/10.1145/2562059.2562140
28. Koymans, R.: Specifying real-time properties with metric temporal logic. Real Time Syst. **2**(4), 255–299 (1990). https://doi.org/10.1007/BF01995674
29. Leucker, M., Schallhart, C.: A brief account of runtime verification. J. Logic Algebraic Program. **78**(5), 293–303 (2009). https://doi.org/10.1016/j.jlap.2008.08.004
30. Maler, O., Nickovic, D.: Monitoring temporal properties of continuous signals. In: Lakhnech, Y., Yovine, S. (eds.) FORMATS/FTRTFT -2004. LNCS, vol. 3253, pp. 152–166. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-30206-3_12
31. Maler, O., Ničković, D.: Monitoring properties of analog and mixed-signal circuits. Int. J. Softw. Tools Technol. Transf. **15**(3), 247–268 (2013). https://doi.org/10.1007/s10009-012-0247-9
32. Ničković, D., Lebeltel, O., Maler, O., Ferrère, T., Ulus, D.: AMT 2.0: qualitative and quantitative trace analysis with extended signal temporal logic. Int. J. Softw. Tools Technol. Transfer **22**(6), 741–758 (2020). https://doi.org/10.1007/s10009-020-00582-z
33. Nickovic, D., Maler, O.: AMT: a property-based monitoring tool for analog systems. In: Raskin, J.-F., Thiagarajan, P.S. (eds.) FORMATS 2007. LNCS, vol. 4763, pp. 304–319. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-75454-1_22
34. Pnueli, A.: The temporal logic of programs. In: FOCS 1977, pp. 46–57. IEEE (1977). https://doi.org/10.1109/SFCS.1977.32
35. Qin, X., Deshmukh, J.V.: Clairvoyant monitoring for signal temporal logic. In: Bertrand, N., Jansen, N. (eds.) FORMATS 2020. LNCS, vol. 12288, pp. 178–195. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-57628-8_11
36. Sánchez, C., et al.: A survey of challenges for runtime verification from advanced application domains (beyond software). Formal Methods Syst. Des. **54**(3), 279–335 (2019). https://doi.org/10.1007/s10703-019-00337-w
37. Selyunin, K., et al.: Runtime monitoring with recovery of the SENT communication protocol. In: Majumdar, R., Kunčak, V. (eds.) CAV 2017. LNCS, vol. 10426, pp. 336–355. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-63387-9_17
38. Zhang, Z., An, J., Arcaini, P., Hasuo, I.: Online causation monitoring of signal temporal logic. arXiv (2023). https://doi.org/10.48550/arXiv.2305.17754
39. Zhang, Z., An, J., Arcaini, P., Hasuo, I.: Online causation monitoring of signal temporal logic (Artifact). Zenodo (2023). https://doi.org/10.5281/zenodo.7923888
40. Zhang, Z., Arcaini, P., Xie, X.: Online reset for signal temporal logic monitoring. IEEE Trans. Comput. Aided Des. Integr. Circuits Syst. **41**(11), 4421–4432 (2022). https://doi.org/10.1109/TCAD.2022.3197693