Online Monitoring of Metric Temporal Logic*

Hsi-Ming Ho, Joël Ouaknine and James Worrell

Department of Computer Science, University of Oxford
Wolfson Building, Parks Road, Oxford, OX1 3QD, UK

Abstract. Current approaches to monitoring real-time properties suffer either from unbounded space requirements or lack of expressiveness. In this paper, we adapt a separation technique enabling us to rewrite arbitrary MTL formulas into LTL formulas over a set of atoms comprising bounded MTL formulas. As a result, we obtain the first trace-length independent online monitoring procedure for full MTL in a dense-time setting.

1 Introduction

In recent years, there has been increasing interest in *runtime verification* as a complement to traditional model checking techniques (see [21, 29] for surveys). Runtime monitoring, for example, may be used in situations in which we wish to evaluate a system that is either too complex to model or whose internal details are not accessible. Moreover, logics whose model-checking problems are undecidable may become tractable in this more restricted setting. The latter is the case in the present paper, which is concerned with runtime monitoring of *Metric Temporal Logic* with both forwards and backwards temporal modalities ($\mathsf{MTL}[\mathbf{U}, \mathbf{S}]$).

$\mathsf{MTL}[\mathbf{U}, \mathbf{S}]$ was introduced almost 25 years ago by Koymans [19] and has since become the most widely studied real-time temporal logic. Over the reals, it has been shown that $\mathsf{MTL}[\mathbf{U}, \mathbf{S}]$ has the same expressiveness as *Monadic First-Order Logic of Order and Metric* ($\mathsf{FO}[<, +\mathbb{Q}]$) [17]. In this paper, we study the ***monitoring*** problem for $\mathsf{MTL}[\mathbf{U}, \mathbf{S}]$ over *timed words*. This so-called *pointwise semantics* is more natural and appropriate when we consider systems modelled as timed automata. Also, monitoring timed words is often conceptually simpler and more efficient [6].

Given an $\mathsf{MTL}[\mathbf{U}, \mathbf{S}]$ formula φ and a finite timed word ρ, the *prefix* problem asks whether all infinite timed words extending ρ satisfy φ. The monitoring problem can be seen as an *online* version of the prefix problem where ρ is given incrementally, one event at a time. The monitoring procedure is required to output an answer when either (i) all infinite extensions of the current trace satisfy the specification, or (ii) no infinite extension of the current trace can possibly meet the specification. In this paper, we consider a variant of the monitoring problem, based on the notion of *informative prefixes* [20].

* More extensive technical details as well as all proofs can be found in the full version of this paper [16].

Ideally, for a monitoring procedure to be practical, we require that it be *trace-length independent* [7] in the sense that the total space requirement should not depend on the length of the input trace. With this objective in mind, the principal difficulty in monitoring $\mathsf{MTL}[\mathbf{U}, \mathbf{S}]$ is that it allows unbounded intervals and nesting of future and past operators, and hence the truth value of a formula at some point may depend on the truth values of its subformulas arbitrarily far in the future or past. For this reason, most real-time monitoring procedures in the literature impose certain syntactic or semantic restrictions, e.g., only allowing bounded future modalities[1] or assuming integer-time traces. A notable exception is [4] which handles the full logic $\mathsf{MTL}[\mathbf{U}, \mathbf{S}]$ over dense-time signals, but which unfortunately fails to be trace-length independent.

The main contribution of this paper is a new online monitoring procedure for $\mathsf{MTL}[\mathbf{U}, \mathbf{S}]$ over dense-time traces. The procedure we give handles the full logic $\mathsf{MTL}[\mathbf{U}, \mathbf{S}]$ and is trace-length independent,[2] making it suitable for traces with potentially unbounded lengths, e.g., network activity logs. For a given formula, we first adapt a separation theorem of [17] to rewrite an $\mathsf{MTL}[\mathbf{U}, \mathbf{S}]$ formula into an $\mathsf{LTL}[\mathbf{U}, \mathbf{S}]$ formula over a set of atoms comprising bounded $\mathsf{MTL}[\mathbf{U}, \mathbf{S}]$ formulas, whose truth values are computed and stored efficiently. The remaining untimed component is then handled via translation to deterministic finite automata. The resulting algorithm is free of dynamic memory allocations, linked lists, etc., and hence can be implemented efficiently.

2 Related Work

The most closely related work to the present paper is that of Finkbeiner and Kuhtz [13], which concerns monitoring MTL over a discrete-time semantics. They handle bounded formulas in a similar fashion to us and highlight the problematic role of unbounded temporal operators. However they do not exploit a syntactic rewriting of unbounded operators from the scope of bounded operators, and are forced to apply specialised constructions in this case.

Another highly relevant work is that of Nickovic and Piterman [26], in which a translation from MTL to deterministic timed automata is proposed. The essence of the method is the observation that, while the truth values of unbounded subformulas must necessarily be guessed, the truth values of bounded subformulas can be obtained via bounded look-ahead. In spirit, this is very similar to our approach. The main differences are that they consider only the future fragment, and we handle bounded subformulas explicitly rather than encoding them into clock constraints.

[1] Note in passing that, unlike for LTL, past modalities strictly increase the expressiveness of MTL [9].

[2] As shown in [22], trace-length independence necessarily requires a global bound on the *variability* of time sequences, i.e., the maximum number of events which can occur in any given unit-duration time interval. This is a standard assumption which is in practice always met by physical systems. The proof in [22] is carried out in the continuous semantics, but it goes through in the pointwise setting as well.

Regarding real-time logics with past, it is known that the non-punctual fragment of $\mathsf{MTL}[\mathbf{U},\mathbf{S}]$, called $\mathsf{MITL}[\mathbf{U},\mathbf{S}]$, can be translated into timed automata [1, 2, 11, 18, 23]. The difficulty in using such approaches for monitoring lies in the fact that timed automata cannot be determinised in general. In principle one can carry out determinisation on-the-fly for timed words of bounded variability; however, it is not clear that this approach can yield an efficient procedure.

Automata-free monitoring procedures also appear in the literature. For example, in a pioneering paper, Thati and Roşu [30] propose a rewriting-based monitoring procedure for $\mathsf{MTL}[\mathbf{U},\mathbf{S}]$. Their procedure is trace-length independent and amenable to efficient implementations. However, the procedure only works for integer-time traces and hence does not appear applicable to our setting.

Online monitoring of real-time properties is still a very active topic of research. Recently, there have been some attempts to extend temporal logics with (restricted) first-order quantifiers for monitoring (see, e.g., [5, 7, 10, 15, 28]). The work in the present paper can be seen as orthogonal to these advances.

3 Background

3.1 Metric Temporal Logic

A *time sequence* $\tau = \tau_1\tau_2\ldots$ is a non-empty strictly increasing sequence of rational numbers such that $\tau_1 = 0$. We consider both finite and infinite time sequences, denoting by $|\tau|$ the length of such a sequence. If τ is infinite we require it to be unbounded, i.e., we disallow the so-called Zeno sequences.

A *timed word* over a finite alphabet Σ is a pair $\rho = (\sigma,\tau)$, where $\sigma = \sigma_1\sigma_2\ldots$ is a non-empty finite or infinite word over Σ and τ is a time sequence of the same length. We equivalently consider a timed word as a sequence of events $(\sigma_1,\tau_1)(\sigma_2,\tau_2)\ldots$. The finite timed words considered in this paper arise as prefixes of infinite timed words, and so we sometimes use the term *prefix* to denote an arbitrary finite timed word. We write $T\Sigma^*$ and $T\Sigma^\omega$ for the respective sets of finite and infinite timed words over Σ. For a set of propositions P we write $\Sigma_P = 2^P$.

For a space-bounded online monitoring procedure to be possible, we must impose a global bound on the variability of time sequences, cf. [22]. Henceforth we assume that all timed words have variability at most k_{var} for some (*a priori* known) absolute constant k_{var}, i.e., there are at most k_{var} events in any unit time interval.

We specify properties of timed words using *Metric Temporal Logic* with both the 'Until' and 'Since' modalities, denoted $\mathsf{MTL}[\mathbf{U},\mathbf{S}]$. Given a set of propositions P, the formulas of $\mathsf{MTL}[\mathbf{U},\mathbf{S}]$ are given by the following grammar

$$\varphi ::= p \mid \mathbf{true} \mid \varphi_1 \wedge \varphi_2 \mid \neg\varphi \mid \varphi_1\,\mathbf{U}_I\,\varphi_2 \mid \varphi_1\,\mathbf{S}_I\,\varphi_2$$

where $p \in P$ and $I \subseteq (0,\infty)$ is an interval with endpoints in $\mathbb{Q}_{\geq 0} \cup \{\infty\}$. We sometime omit the subscript I if $I = (0,\infty)$. Given $x \in \mathbb{Q}$, we write $x < I$ to

mean $x < \sup(I)$. Additional temporal operators and dual operators are defined in the standard way, e.g., $\mathbf{P}_I\varphi \equiv \mathbf{true}\mathbf{S}_I\varphi$ and $\mathbf{H}_I\varphi \equiv \neg\mathbf{P}_I\neg\varphi$. For an $\mathsf{MTL}[\mathbf{U},\mathbf{S}]$ formula φ, we denote by $|\varphi|$ the number of subformulas of φ.

The satisfaction relation $\rho, i \models \varphi$ for an $\mathsf{MTL}[\mathbf{U},\mathbf{S}]$ formula φ, an infinite timed word $\rho = (\sigma, \tau)$ and a position $i \geq 1$ is defined as follows:

- $\rho, i \models p$ iff $p \in \sigma_i$
- $\rho, i \models \varphi_1 \, \mathbf{U}_I \, \varphi_2$ iff there exists $j > i$ such that $\rho, j \models \varphi_2$, $\tau_j - \tau_i \in I$, and $\rho, k \models \varphi_1$ for all k with $i < k < j$
- $\rho, i \models \varphi_1 \, \mathbf{S}_I \, \varphi_2$ iff there exists j, $1 \leq j < i$ such that $\rho, j \models \varphi_2$, $\tau_i - \tau_j \in I$ and $\rho, k \models \varphi_1$ for all k with $j < k < i$.[3]

The semantics of the Boolean connectives is defined in the expected way.

We say that ρ satisfies φ, denoted $\rho \models \varphi$, if $\rho, 1 \models \varphi$. We write $\mathcal{L}(\varphi)$ for the set of infinite timed words that satisfy φ. Abusing notation, we also write $\mathcal{L}(\psi)$ for the set of infinite (untimed) words that satisfy the $\mathsf{LTL}[\mathbf{U},\mathbf{S}]$ formula ψ, and $\mathcal{L}(\mathcal{A})$ for the set of infinite words accepted by automaton $\mathcal{A}$.

3.2 Truncated Semantics and Informative Prefixes

Since in online monitoring one naturally deals with truncated paths, it is useful to define a satisfaction relation of formulas over finite timed words. To this end we adopt a timed version of the *truncated semantics* [12] which incorporates *strong* and *weak* views on satisfaction over truncated paths. These views indicate whether the evaluation of the formula 'has completed' on the finite path, i.e., whether the truth value of the formula on the whole path is already determined. For example, the formula $\mathbf{F}_{(0,5)}p$ is weakly satisfied by any finite timed word whose time points are all strictly less than 5 since there is an extension that satisfies the formula. We also consider the *neutral* view, which extends to $\mathsf{MTL}[\mathbf{U},\mathbf{S}]$ the traditional LTL semantics over finite words [24].

The respective strong, neutral and weak satisfaction relations will be denoted by $\models_f^+$, $\models_f$ and $\models_f^-$ respectively. The definitions below closely follow [12].

Definition 1. *The satisfaction relation $\rho, i \models_f^+ \varphi$ for an $\mathsf{MTL}[\mathbf{U},\mathbf{S}]$ formula φ, a finite timed word $\rho = (\sigma, \tau)$ and a position i, $1 \leq i \leq |\rho|$ is defined as follows:*

- $\rho, i \models_f^+ p$ *iff* $p \in \sigma_i$
- $\rho, i \models_f^+ \mathbf{true}$
- $\rho, i \models_f^+ \varphi_1 \wedge \varphi_2$ *iff* $\rho, i \models_f^+ \varphi_1$ *and* $\rho, i \models_f^+ \varphi_1$
- $\rho, i \models_f^+ \neg\varphi$ *iff* $(\rho, i) \not\models_f^- \varphi$
- $\rho, i \models_f^+ \varphi_1 \mathbf{U}_I \varphi_2$ *iff there exists* j, $i < j \leq |\rho|$, *such that* $\rho, j \models_f^+ \varphi_2$, $\tau_j - \tau_i \in I$, *and* $\rho, j' \models_f^+ \varphi_1$ *for all* j' *with* $i < j' < j$
- $\rho, i \models_f^+ \varphi_1 \mathbf{S}_I \varphi_2$ *iff there exists* j, $1 \leq j < i$, *such that* $\rho, j \models_f^+ \varphi_2$, $\tau_i - \tau_j \in I$ *and* $\rho, j' \models_f^+ \varphi_1$ *for all* j' *with* $j < j' < i$.

[3] Note that we adopt *strict* interpretations to $\mathbf{U}_I$ and $\mathbf{S}_I$. It is easy to see that, e.g., weak-future until operators can be defined in strict-future ones.

Definition 2. *The satisfaction relation* $\rho, i \models_f^- \varphi$ *for an* $\mathsf{MTL}[\mathbf{U}, \mathbf{S}]$ *formula* φ*, a finite timed word* $\rho = (\sigma, \tau)$ *and a position* i*,* $1 \leq i \leq |\rho|$ *is defined as follows:*

- $\rho, i \models_f^- p$ *iff* $p \in \sigma_i$
- $\rho, i \models_f^- \mathbf{true}$
- $\rho, i \models_f^- \varphi_1 \wedge \varphi_2$ *iff* $\rho, i \models_f^- \varphi_1$ *and* $\rho, i \models_f^- \varphi_1$
- $\rho, i \models_f^- \neg\varphi$ *iff* $(\rho, i) \not\models_f^+ \varphi$
- $\rho, i \models_f^- \varphi_1 \, \mathbf{U}_I \, \varphi_2$ *iff either of the following holds:*
  - *there exists* j*,* $i < j \leq |\rho|$*, such that* $\rho, j \models_f^- \varphi_2$*,* $\tau_j - \tau_i \in I$*, and* $\rho, j' \models_f^- \varphi_1$ *for all* j' *with* $i < j' < j$
  - $\tau_{|\rho|} - \tau_i < I$ *and* $\rho, j' \models_f^- \varphi_1$ *for all* j' *with* $i < j' \leq |\rho|$
- $\rho, i \models_f^- \varphi_1 \, \mathbf{S}_I \, \varphi_2$ *iff there exists* j*,* $1 \leq j < i$*, such that* $\rho, j \models_f^- \varphi_2$*,* $\tau_i - \tau_j \in I$ *and* $\rho, j' \models_f^- \varphi_1$ *for all* j' *with* $j < j' < i$*.*

The following proposition which helps explain the terms strong, neutral and weak, can be proved by a simple induction on the structure of φ.

Proposition 1. *For a finite timed word* ρ*, a position* i *in* ρ *and an* $\mathsf{MTL}[\mathbf{U}, \mathbf{S}]$ *formula* φ*,*

$$\rho, i \models_f^+ \varphi \rightarrow \rho, i \models_f \varphi \text{ and } \rho, i \models_f \varphi \rightarrow \rho, i \models_f^- \varphi \,.$$

A closely related notion, *informative prefixes* [20], has been adopted in several works on online monitoring of untimed properties, e.g., [3, 14]. Intuitively, an informative prefix for a formula φ is a prefix that 'tells the whole story' about the fulfilment or violation of φ.[4] We give two examples before the formal definition.

Example 1. Consider the following formula over $\{p_1\}$:

$$\varphi = \mathbf{FG}(\neg p_1) \wedge \mathbf{G}(p_1 \rightarrow \mathbf{F}_{(0,3)} p_1) \,.$$

The finite timed word $\rho = (\{p_1\}, 0)(\{p_1\}, 2)(\emptyset, 5.5)$ is an informative bad prefix for φ, since no extension satisfies the second conjunct. On the other hand, while $\rho' = (\{p_1\}, 0)(\{p_1\}, 2)(\{p_1\}, 4)$ is a bad prefix for φ, it has (different) extensions that satisfy, respectively, the left and right conjuncts. Thus we do not consider it an informative bad prefix.

Example 2. Consider the following formula over $\{p_1\}$:

$$\varphi' = \mathbf{G}(\neg p_1) \wedge \mathbf{G}(p_1 \rightarrow \mathbf{F}_{(0,3)} p_1) \,.$$

This formula is equivalent to the formula φ in the previous example. However, all bad prefixes for φ' are informative.

[4] Our usage of the term *informative* slightly deviates from [20] as in that paper the term refers exclusively to bad prefixes.

If a prefix ρ strongly satisfies φ then we say that it is an *informative good prefix* for φ. Similarly we say ρ is an *informative bad prefix* for φ when it fails to weakly satisfy φ. Finally ρ is an *informative prefix* if it is either an informative good prefix or an informative bad prefix. Here we have adopted the semantic characterisation of informative prefixes in terms of the truncated semantics from [12], rather than the original syntactic definition [20].

The following proposition follows immediately from the definition of informative prefixes.

Proposition 2. *ρ is informative for φ iff ρ is informative for $\neg\varphi$.*

Since $\rho \models_f \varphi \leftrightarrow \rho \not\models_f \neg\varphi$, negating a formula essentially exchanges its set of informative good prefixes and informative bad prefixes. The following proposition says 'something good remains good' and 'something bad remains bad'.

Proposition 3. *For a finite timed word ρ, a position i in ρ and an $\mathsf{MTL}[\mathbf{U},\mathbf{S}]$ formula φ, if ρ is a prefix of the finite timed word ρ', then*

$$\rho, i \models_f^+ \varphi \to \rho', i \models_f^+ \varphi \text{ and } \rho, i \not\models_f^- \varphi \to \rho', i \not\models_f^- \varphi\,.$$

4 $\mathsf{LTL}[\mathbf{U},\mathbf{S}]$ over Bounded Atoms

In this section we present a series of logical equivalences that can be used to rewrite a given $\mathsf{MTL}[\mathbf{U},\mathbf{S}]$ formula into an equivalent formula in which no unbounded temporal operator occurs within the scope of a bounded operator. Only the rules for future modalities and open intervals are given, as the rules for past modalities are symmetric and the rules for other types of intervals are straightforward variants. Since we work in the pointwise semantics, the techniques in [17] (developed for the continuous semantics) must be carefully adapted.

4.1 Normal Form

We say an $\mathsf{MTL}[\mathbf{U},\mathbf{S}]$ formula is in *normal form* if it satisfies the following.

(i) All occurrences of unbounded temporal operators are of the form $\mathbf{U}_{(0,\infty)}$, $\mathbf{S}_{(0,\infty)}$, $\mathbf{G}_{(0,\infty)}$, $\mathbf{H}_{(0,\infty)}$.
(ii) All other occurrences of temporal operators are of the form $\mathbf{U}_I$, $\mathbf{S}_I$ with bounded I.
(iii) Negation is only applied to propositions or bounded temporal operators (except that we allow $\mathbf{G}_{(0,\infty)}$, $\mathbf{H}_{(0,\infty)}$).
(iv) In any subformula of the form $\varphi_1\,\mathbf{U}_I\,\varphi_2$, $\varphi_1\,\mathbf{S}_I\,\varphi_2$, $\mathbf{F}_I\varphi_2$, $\mathbf{P}_I\varphi_2$ where I is bounded, φ_1 is a disjunction of temporal subformulas and propositions and φ_2 is a conjunction thereof.

We describe how to rewrite a given formula into normal form. To satisfy (i) and (ii), apply the usual rules (e.g., $\mathbf{G}_I\varphi \leftrightarrow \neg\mathbf{F}_I\neg\varphi$) and the rule:

$$\varphi_1\,\mathbf{U}_{(a,\infty)}\,\varphi_2 \leftrightarrow \varphi_1\,\mathbf{U}\,\varphi_2 \wedge \big(\mathbf{F}_{(0,a]}\mathbf{true} \to \mathbf{G}_{(0,a]}(\varphi_1 \wedge \varphi_1\,\mathbf{U}\,\varphi_2)\big)\,.$$

To satisfy (iii), use the usual rules and the rule:

$$\neg(\varphi_1 \,\mathbf{U}\, \varphi_2) \leftrightarrow \mathbf{G}\neg\varphi_2 \vee \left(\neg\varphi_2 \,\mathbf{U}\, (\neg\varphi_2 \wedge \neg\varphi_1)\right).$$

For (iv), use the usual rules of Boolean algebra and the rules below:

$$\begin{aligned}
\phi \,\mathbf{U}_I\, (\varphi_1 \vee \varphi_2) &\leftrightarrow (\phi \,\mathbf{U}_I\, \varphi_1) \vee (\phi \,\mathbf{U}_I\, \varphi_2) \\
(\varphi_1 \wedge \varphi_2) \,\mathbf{U}_I\, \phi &\leftrightarrow (\varphi_1 \,\mathbf{U}_I\, \phi) \wedge (\varphi_2 \,\mathbf{U}_I\, \phi).
\end{aligned}$$

4.2 Extracting Unbounded Operators from Bounded Operators

We now provide a set of rewriting rules that extract unbounded operators from the scopes of bounded operators. In what follows, let $\varphi_{xlb} = \mathbf{false}\,\mathbf{U}_{(0,b)}\,\mathbf{true}$, $\varphi_{ylb} = \mathbf{false}\,\mathbf{S}_{(0,b)}\,\mathbf{true}$ and

$$\begin{aligned}
\varphi_{ugb} = &\Big((\varphi_{xlb} \rightarrow \mathbf{G}_{(b,2b)}\varphi_1) \wedge \left(\neg\varphi_{ylb} \rightarrow (\varphi_1 \wedge \mathbf{G}_{(0,b]}\varphi_1)\right)\Big) \\
&\mathbf{U}\left((\varphi_1 \wedge (\varphi_1 \,\mathbf{U}_{(b,2b)}\, \varphi_2)) \vee \left(\neg\varphi_{ylb} \wedge \left(\varphi_2 \vee \left(\varphi_1 \wedge (\varphi_1 \,\mathbf{U}_{(0,b]}\, \varphi_2)\right)\right)\right)\right), \\
\varphi_{ggb} = &\,\mathbf{G}\Big((\varphi_{xlb} \rightarrow \mathbf{G}_{(b,2b)}\varphi_1) \wedge \left(\neg\varphi_{ylb} \rightarrow (\varphi_1 \wedge \mathbf{G}_{(0,b]}\varphi_1)\right)\Big).
\end{aligned}$$

Proposition 4. *The following equivalences hold over infinite timed words.*

$$\begin{aligned}
\theta \,\mathbf{U}_{(a,b)}\, \left((\varphi_1 \,\mathbf{U}\, \varphi_2) \wedge \chi\right) &\leftrightarrow \theta \,\mathbf{U}_{(a,b)}\, \left((\varphi_1 \,\mathbf{U}_{(0,2b)}\, \varphi_2) \wedge \chi\right) \\
&\quad \vee \Big(\left(\theta \,\mathbf{U}_{(a,b)}\, (\mathbf{G}_{(0,2b)}\varphi_1 \wedge \chi)\right) \wedge \varphi_{ugb}\Big) \\
\theta \,\mathbf{U}_{(a,b)}\, (\mathbf{G}\varphi \wedge \chi) &\leftrightarrow \left(\theta \,\mathbf{U}_{(a,b)}\, (\mathbf{G}_{(0,2b)}\varphi \wedge \chi)\right) \wedge \varphi_{ggb} \\
\theta \,\mathbf{U}_{(a,b)}\, \left((\varphi_1 \,\mathbf{S}\, \varphi_2) \wedge \chi\right) &\leftrightarrow \theta \,\mathbf{U}_{(a,b)}\, \left((\varphi_1 \,\mathbf{S}_{(0,b)}\, \varphi_2) \wedge \chi\right) \\
&\quad \vee \Big(\left(\theta \,\mathbf{U}_{(a,b)}\, (\mathbf{H}_{(0,b)}\varphi_1 \wedge \chi)\right) \wedge \varphi_1 \,\mathbf{S}\, \varphi_2\Big) \\
\theta \,\mathbf{U}_{(a,b)}\, (\mathbf{H}\varphi \wedge \chi) &\leftrightarrow \left(\theta \,\mathbf{U}_{(a,b)}\, (\mathbf{H}_{(0,b)}\varphi \wedge \chi)\right) \wedge \mathbf{H}\varphi \\
\left((\varphi_1 \,\mathbf{U}\, \varphi_2) \vee \chi\right) \,\mathbf{U}_{(a,b)}\, \theta &\leftrightarrow \left((\varphi_1 \,\mathbf{U}_{(0,2b)}\, \varphi_2) \vee \chi\right) \,\mathbf{U}_{(a,b)}\, \theta \\
&\quad \vee \Big(\Big(\left((\varphi_1 \,\mathbf{U}_{(0,2b)}\, \varphi_2) \vee \chi\right) \,\mathbf{U}_{(0,b)}\, (\mathbf{G}_{(0,2b)}\varphi_1)\Big) \\
&\qquad \wedge \\
&\qquad \mathbf{F}_{(a,b)}\theta \wedge \varphi_{ugb}\Big) \\
\left((\mathbf{G}\varphi) \vee \chi\right) \,\mathbf{U}_{(a,b)}\, \theta &\leftrightarrow \chi \,\mathbf{U}_{(a,b)}\, \theta \\
&\quad \vee \left(\chi \,\mathbf{U}_{(0,b)}\, (\mathbf{G}_{(0,2b)}\varphi_1) \wedge \mathbf{F}_{(a,b)}\theta \wedge \varphi_{ggb}\right)
\end{aligned}$$

$$\big((\varphi_1 \,\mathbf{S}\, \varphi_2) \vee \chi\big)\, \mathbf{U}_{(a,b)}\, \theta \leftrightarrow \big((\varphi_1 \,\mathbf{S}_{(0,b)}\, \varphi_2) \vee \chi\big)\, \mathbf{U}_{(a,b)}\, \theta \vee \Bigg(\Big(\big(\big(\mathbf{H}_{(0,b)}\varphi_1 \vee (\varphi_1 \,\mathbf{S}_{(0,b)}\, \varphi_2) \vee \chi\big)\, \mathbf{U}_{(a,b)}\, \theta\big) \wedge \varphi_1 \,\mathbf{S}\, \varphi_2\Big)$$

$$\big((\mathbf{H}\varphi) \vee \chi\big)\, \mathbf{U}_{(a,b)}\, \theta \leftrightarrow \chi\, \mathbf{U}_{(a,b)}\, \theta \vee \Big(\big((\mathbf{H}_{(0,b)}\varphi \vee \chi)\, \mathbf{U}_{(a,b)}\, \theta\big) \wedge \mathbf{H}\varphi\Big).$$

Proof. We sketch the proof for the first rule as the proofs for the other rules are similar. In the following, let the current position be i and the position of an (arbitrary) event in $(\tau_i + a, \tau_i + b)$ be j.

For the forward direction, let the witness position where φ_2 holds be w. If $\tau_w < \tau_j + 2b$, the subformula $\varphi_1 \,\mathbf{U}_{(0,2b)}\, \varphi_2$ clearly holds at j and we are done. Otherwise, $\mathbf{G}_{(0,2b)}\varphi_1$ holds at j and it follows that $(\varphi_{xlb} \rightarrow \mathbf{G}_{(b,2b)}\varphi_1)$ and φ_{ylb} (and vacuously $\neg\varphi_{ylb} \rightarrow (\varphi_1 \wedge \mathbf{G}_{(0,b]}\varphi_1)$) hold at all positions j', $i < j' < j$. Let $l > j$ be the first position such that $\tau_w \in (\tau_l + b, \tau_l + 2b)$. Consider the following cases:

- There is such l: It is clear that $\big(\varphi_1 \wedge (\varphi_1 \mathbf{U}_{(b,2b)} \varphi_2)\big)$ holds at l. Since $\mathbf{G}_{(b,2b)}\varphi_1$ holds at all positions j'', $j \leq j'' < l$ by the minimality of l, $(\varphi_{xlb} \rightarrow \mathbf{G}_{(b,2b)}\varphi_1)$ also holds at these positions. For the other conjunct, note that φ_{ylb} holds at j and $\varphi_1 \wedge \mathbf{G}_{(0,b]}\varphi_1$ holds at all positions j''', $j < j''' < l$.
- There is no such l: Consider the following cases:
  - $\neg\varphi_{ylb}$ and $\neg\mathbf{P}_{[b,b]}\mathbf{true}$ hold at w: There is no event in $(\tau_w - 2b, \tau_w)$. The proof is similar to the case where l exists.
  - $\neg\varphi_{ylb}$ and $\mathbf{P}_{[b,b]}\mathbf{true}$ hold at w: Let l' be the position such that $\tau_{l'} = \tau_w - b$. There must be no event in $(\tau_{l'} - b, \tau_{l'})$. It follows that $\neg\varphi_{ylb}$ and $\big(\varphi_1 \wedge (\varphi_1 \,\mathbf{U}_{(0,b]}\, \varphi_2)\big)$ hold at l'. The proof is similar.
  - φ_{ylb} holds at w: By assumption, there is no event in $(\tau_w - 2b, \tau_w - b)$. It is easy to see that there is a position such that $\neg\varphi_{ylb} \wedge \big(\varphi_1 \wedge (\varphi_1 \,\mathbf{U}_{(0,b]}\, \varphi_2)\big)$ holds. The proof is again similar.

We prove the other direction by contraposition. Consider the interesting case where $\mathbf{G}_{(0,2b)}\varphi_1$ holds at j yet $\varphi_1 \,\mathbf{U}\, \varphi_2$ does not hold at j. If φ_2 never holds in $[\tau_j + 2b, \infty)$ then we are done. Otherwise, let $l > j$ be the first position such that both φ_1 and φ_2 do not hold at l (note that $\tau_l \geq \tau_j + 2b$). It is clear that $\Bigg(\big(\varphi_1 \wedge (\varphi_1 \,\mathbf{U}_{(b,2b)}\, \varphi_2)\big) \vee \Big(\neg\varphi_{ylb} \wedge \big(\varphi_2 \vee \big(\varphi_1 \wedge (\varphi_1 \,\mathbf{U}_{(0,b]}\, \varphi_2)\big)\big)\Big)\Bigg)$ does not hold at all positions j', $i < j' \leq l$. Consider the following cases:

- φ_{ylb} does not hold at l: $\varphi_1 \wedge \mathbf{G}_{(0,b]}\varphi_1$ does not hold at l, and hence φ_{ugb} fails to hold at i.
- φ_{ylb} holds at l: Consider the following cases:
  - There is an event in $(\tau_l - 2b, \tau_l - b)$: Let this event be at position j''. We have $j'' + 1 < l$, $\tau_{j''+1} - \tau_{j''} \geq b$ and $\tau_l - \tau_{j''+1} < b$. However, it follows that φ_{ylb} does not hold at $j'' + 1$ and $\varphi_1 \wedge \mathbf{G}_{(0,b]}\varphi_1$ holds at $j'' + 1$, which is a contradiction.

- There is no event in $(\tau_l - 2b, \tau_l - b)$: Let the first event in $[\tau_l - b, \tau_l)$ be at position j''. It is clear that φ_{ylb} does not hold at j'' and $\varphi_1 \wedge \mathbf{G}_{(0,b]}\varphi_1$ must hold at j'', which is a contradiction.

□

Proposition 5. *For an* $\mathsf{MTL}[\mathbf{U}, \mathbf{S}]$ *formula* φ*, we can use the rules above to obtain an equivalent formula* $\hat{\varphi}$ *in which no unbounded temporal operator appears in the scope of a bounded temporal operator.*

Proof. Define the *unbounding depth* $ud(\varphi)$ of a formula φ to be the modal depth of φ counting only unbounded operators. We demonstrate a rewriting process on φ which terminates in an equivalent formula $\hat{\varphi}$ such that any subformula $\hat{\psi}$ of $\hat{\varphi}$ with outermost operator bounded has $ud(\hat{\psi}) = 0$.

Assume that the input formula φ is in normal form. Let k be the largest unbounding depth among all subformulas of φ with bounded outermost operators. We pick all minimal (wrt. subformula order) such subformulas ψ with $ud(\psi) = k$. By applying the rules in Section 4.2, we can rewrite ψ into ψ' where all subformulas of ψ' with bounded outermost operators have unbounded depths strictly less than k. We then substitute these ψ' back into φ to obtain φ'. We repeat this step until there remain no bounded operators with unbounding depth k. Rules that rewrite a formula into normal form are used whenever necessary on relevant subformulas—this will never affect their unbounding depths. It is easy to see that we will eventually obtain such a formula φ^*. Now rewrite φ^* into normal form and start over again. This is to be repeated until we reach $\hat{\varphi}$. □

Given the input formula φ over propositions $P = \{p_1, \ldots, p_n\}$, we can apply the rewriting process above to obtain a formula $\hat{\varphi}$. Since each rewriting rule is a logical equivalence, we have the following theorem.

Theorem 1. $\mathcal{L}(\varphi) = \mathcal{L}(\hat{\varphi})$.

The syntactic separation of the original formula could potentially induce a non-elementary blow-up. However, such behaviour does not seem to be realised in practice. In our experience, the syntactically separated formula is often of comparable size to the original formula, which itself is typically small. For example, consider the following formula:

$$\mathbf{G}\big(\texttt{ChangeGear} \rightarrow \mathbf{F}_{(0,30)}(\texttt{InjectFuel} \wedge \mathbf{P}\texttt{InjectLubricant})\big)\,.$$

The syntactically separated version of the formula is

$$\mathbf{G}\big[\texttt{ChangeGear} \rightarrow \mathbf{F}_{(0,30)}(\texttt{InjectFuel} \wedge \mathbf{P}_{(0,30)}\texttt{InjectLubricant}) \vee \big(\mathbf{F}_{(0,30)}(\texttt{InjectFuel}) \wedge \mathbf{P}\texttt{InjectLubricant}\big)\big]\,.$$

In any case, Proposition 5 and Theorem 1 imply that we may even require the input formula to be in 'separated form' without sacrificing any expressiveness.

5 Online Monitoring Procedure

Having obtained $\hat{\varphi} = \Phi(\psi_1, \ldots, \psi_m)$ where $\psi_1, \ldots, \psi_m$ are bounded formulas over P and Φ is an $\mathsf{LTL}[\mathbf{U}, \mathbf{S}]$ formula, we now introduce new propositions $Q = \{q_1, \ldots, q_m\}$ that correspond to bounded subformulas. In this way, we can monitor Φ as an untimed property over Q, only that now we obtain the truth values of $q_1, \ldots, q_m$ by simple dynamic programming procedures. As these propositions correspond to bounded formulas, we only need to store a 'sliding window' on the input timed word.

5.1 Untimed $\mathsf{LTL}[\mathbf{U}, \mathbf{S}]$ Part

We describe briefly the standard way to construct automata that detect informative prefixes [20]. For a given LTL formula Θ, first use a standard construction [31] to obtain a language-equivalent alternating Büchi automaton $\mathcal{A}_\Theta$. Then redefine its set of accepting states to be the empty set and treat it as an automaton over finite words. The resulting automaton $\mathcal{A}_\Theta^{true}$ accepts exactly all informative good prefixes for Θ. For online monitoring, one can then determinise $\mathcal{A}_\Theta^{true}$ with the usual subset construction. The same can be done for $\neg\Theta$ to obtain a deterministic automaton detecting informative bad prefixes for Θ.

In our case, we first translate the $\mathsf{LTL}[\mathbf{U}, \mathbf{S}]$ formulas Φ and $\neg\Phi$ into a pair of *two-way* alternating Büchi automata. It is easy to see that, with the same 'tweaks', we can obtain two automata that accept informative good prefixes and informative bad prefixes for Φ (by Proposition 2). We then apply existing procedures that translate two-way alternating automata over finite words into deterministic automata, e.g., [8]. We call the resulting automata $\mathcal{D}_{good}$ and $\mathcal{D}_{bad}$ and execute them in parallel.

5.2 Bounded Metric Part

We define $fr(\varphi)$ and $pr(\varphi)$ (*future-reach* and *past-reach*) for an $\mathsf{MTL}[\mathbf{U}, \mathbf{S}]$ formula φ as follows (the cases for boolean connectives are defined as expected):

- $fr(\mathbf{true}) = pr(\mathbf{true}) = fr(p) = pr(p) = 0$ for all $p \in P$
- $fr(\varphi_1 \,\mathbf{U}_I\, \varphi_2) = \sup(I) + \max(fr(\varphi_1), fr(\varphi_2))$
- $pr(\varphi_1 \,\mathbf{S}_I\, \varphi_2) = \sup(I) + \max(pr(\varphi_1), pr(\varphi_2))$
- $fr(\varphi_1 \,\mathbf{S}_I\, \varphi_2) = \max(fr(\varphi_1), fr(\varphi_2) - \inf(I))$
- $pr(\varphi_1 \,\mathbf{U}_I\, \varphi_2) = \max(pr(\varphi_1), pr(\varphi_2) - \inf(I))$.

Intuitively, these indicate the lengths of the time horizons needed to determine the truth value of φ. We also define $l_f(\psi) = k_{var} \cdot \lceil fr(\psi) \rceil$ and $l_p(\psi) = k_{var} \cdot \lceil pr(\psi) \rceil$ (recall that we assume that timed words are of bounded variability k_{var}).

Naïve Method Suppose that we would like to obtain the truth value of q_i at position j in the input (infinite) timed word $\rho = (\sigma, \tau)$. Observe that only events occurring between $\tau_j - pr(\psi_i)$ and $\tau_j + fr(\psi_i)$ can affect the truth value of ψ_i at j. This implies that $\rho, j \models \psi_i \leftrightarrow \rho', j \models_f \psi_i$, given that ρ' is a prefix of ρ that contains all events between $\tau_j - pr(\psi_i)$ and $\tau_j + fr(\psi_i)$. Since ρ is of bounded variability k_{var}, there will be at most $l_p(\psi_i) + 1 + l_f(\psi_i)$ events between $\tau_j - pr(\psi_i)$ and $\tau_j + fr(\psi_i)$. It follows that we can simply record all events in this interval. Events outside of this interval are irrelevant as they do not affect whether $\rho', j \models_f \psi_i$. In particular, we maintain a two-dimensional array of $l_p(\psi_i) + 1 + l_f(\psi_i) + 1$ rows and $1 + |\psi|$ columns. The first column is used to store timestamps of the corresponding events.[5] The last $|\psi|$ columns are used to store the truth values of subformulas. We then use dynamic programming procedures (cf. [25]) to evaluate whether $\rho', j \models_f \psi_i$. These procedures fill up the array in a bottom-up manner, starting from minimal subformulas. The columns for boolean combinations can be filled in the natural way.

Now consider all propositions in Q. We can obtain the truth values of them at all positions in the 'sliding window' by using an array of $l_p^Q + 1 + l_f^Q + 1$ rows and $1 + |\psi_1| + \cdots + |\psi_m|$ columns, where $l_p^Q = \max_{i \in [1,m]} l_p(\psi_i)$ and $l_f^Q = \max_{i \in [1,m]} l_f(\psi_i)$. Each column can be filled in time linear in its length. Overall, we need an array of size $O(k_{var} \cdot c_{sum} \cdot |\hat{\varphi}|)$ where c_{sum} is the sum of the constants in $\hat{\varphi}$, and for each position j we need time $O(k_{var} \cdot c_{sum} \cdot |\hat{\varphi}|)$ to obtain the truth values of all propositions in Q. This method is not very efficient as for each j we need to fill all columns for temporal subformulas from scratch. Previously computed entries cannot always be reused as certain entries are 'wrong'—they were computed without the knowledge of events outside of the interval.

Incremental Evaluation We describe an optimisation which allows effective reuse of computed entries stored in the table. The idea is to treat entries that depend on future events as 'unknown' and not to fill them. By construction, these unknown entries will not be needed for the result of the evaluation.

For a past subformula, e.g, $\varphi_1 \, \mathbf{S}_{(a,b)} \, \varphi_2$, we can simply suspend the column-filling procedure when we filled all entries using the truth values of φ_1 and φ_2 (at various positions) that are currently known. We may continue when the truth values of φ_1 and φ_2 (at some other positions) that are previously unknown become available. The case for future subformulas is more involved. Suppose that we are filling a column for $p_1 \, \mathbf{U}_{(a,b)} \, p_2$ with the naïve method. Denote the corresponding timestamp of an index i in the column by $t(i)$ and the timestamp of the last acquired event by $t_{\max}$. Observe that not all of the truth values at indices j, $t(j) + b > t_{\max}$ can be reused later, as they might depend on future events. However, if we know that φ_1 does not hold at some j', $t(j') + b > t_{\max}$, then all the truth values at indices $< j'$ can be reused in the following iterations as they cannot depend on future events. Now consider the general case of filling

[5] We assume the timestamps can be finitely represented, e.g., with a built-in data type, and additions and subtractions on them can be done in constant time.

the column for $\psi = \varphi_1 \, \mathbf{U}_{(a,b)} \, \varphi_2$. We keep an index j_ψ that points to the first unknown entry in the column, and we now let $t_{\max} = \min(t(j_{\varphi_1} - 1), t(j_{\varphi_2} - 1))$. In each iteration, if j_{φ_1} and j_{φ_2} are updated to some new values, $t_{\max}$ also changes accordingly. If this happens, we first check if $t(j_\psi) + b > t_{\max}$. If this is the case, we do nothing (observe the fact that φ_1 must hold at all indices l, $t(j_\psi) < t(l) \leq t_{\max}$, thus the truth value at j_ψ must remain unknown). Otherwise we find the least index $l' > j_\psi$ such that $t(l') + b > t_{\max}$. Additionally, we check if all truth values of φ_1 between $t_{\max}$ and $t^{old}_{\max}$ are **true**, starting from $t_{\max}$. If φ_1 is not satisfied at some (maximal) position j' then start filling at $\max(l', j') - 1$. Otherwise we start filling from $l' - 1$.

Observe that we can use a variable to keep track of the least index $l' > j_\psi$ such that $t(l') + b > t_{\max}$ instead of finding it each time since it increases monotonically. Also we can keep track of the greatest index where φ_2 holds. With these variables, we can easily make the extra 'sweeping' happen only twice (once for φ_1 and once for φ_2) over newly acquired truth values. Also observe that the truth value of a subformula at a certain position will be filled only once. These observations imply that each entry in the array can be filled in amortised constant time. Assuming that each step of an deterministic automaton takes constant time, we can state the following theorem.

Theorem 2. *For an* $\mathsf{MTL}[\mathbf{U}, \mathbf{S}]$ *formula* φ*, the automata* $\mathcal{D}_{good}$ *and* $\mathcal{D}_{bad}$ *have size* $2^{2^{O(|\Phi|)}}$ *where* Φ *is the* $\mathsf{LTL}[\mathbf{U}, \mathbf{S}]$ *formula described above. Moreover, for an infinite timed word of bounded variability* k_{var}*, our procedure uses space* $O(k_{var} \cdot c_{sum} \cdot |\hat{\varphi}|)$ *and amortised time* $O(|\hat{\varphi}|)$ *per event, where* $\hat{\varphi}$ *is the syntactically separated equivalent formula of* φ *and* c_{sum} *is the sum of the constants in* $\hat{\varphi}$*.*

5.3 Correctness

One may think of the monitoring process on an infinite timed word $\rho \in T\Sigma^\omega_P$ as continuously extending a corresponding finite timed word $\rho' \in T\Sigma^*_Q$. Suppose that, instead of $\mathcal{D}_{good}$ and $\mathcal{D}_{bad}$, we now execute a deterministic ω-automaton $\mathcal{D}_\Phi$ such that $\mathcal{L}(\mathcal{D}_\Phi) = \mathcal{L}(\Phi)$. Since we are implicitly ensuring that the truth values of propositions in Q are valid along the way, it is easy to see that the corresponding run on $\mathcal{D}_\Phi$ will be accepting iff $\rho \models \varphi$. However, for the purpose of online monitoring, we will be more interested in deciding whether $\rho \models \varphi$ given only a finite prefix of ρ. In this subsection we show that our approach is both sound and complete for detecting informative prefixes.

The following proposition is immediate since three views of the truncated semantics coincide in this case.

Proposition 6. *For a bounded* $\mathsf{MTL}[\mathbf{U}, \mathbf{S}]$ *formula* ψ*, a finite timed word* $\rho = (\sigma, \tau)$ *and a position* $1 \leq i \leq |\rho|$ *such that* $\tau_i + fr(\psi) \leq \tau_{|\rho|}$ *and* $\tau_i - pr(\psi) \geq 0$*, we have*

$$\rho, i \models^+_f \psi \leftrightarrow \rho, i \models_f \psi \leftrightarrow \rho, i \models^-_f \psi \,.$$

The following lemma implies that the rewriting process outlined in Section 4 preserves the 'informativeness' of prefixes.

Lemma 1. *For an* MTL[**U**, **S**] *formula* φ*, let* φ' *be the formula obtained after applying one of the rewriting rules in Section 4 on some of its subformula. We have*

$$\rho \models_f^+ \varphi \leftrightarrow \rho \models_f^+ \varphi' \text{ and } \rho \models_f^- \varphi \leftrightarrow \rho \models_f^- \varphi' .$$

Given the lemma above, we can state the following theorem.

Theorem 3. *The set of informative good prefixes of* φ *coincides with the set of informative good prefixes of* $\hat{\varphi}$*. The same holds for informative bad prefixes.*

Now we state the main result of the paper in the following two theorems.

Theorem 4 (Soundness). *In our procedure, if we ever reach an accepting state of* $\mathcal{D}_{good}$ *(*$\mathcal{D}_{bad}$*) via a finite word* $u \in \Sigma_Q^*$*, then the finite timed word* $\rho \in T\Sigma_P^*$ *that we have read must be an informative good (bad) prefix for* φ*.*

Proof. For such u and the corresponding ρ (note that $|u| \leq |\rho|$),

$$\forall i \in [1, |u|] \left((u, i \not\models_f^- \Theta \rightarrow \rho, i \not\models_f^- \vartheta) \wedge (u, i \models_f^+ \Theta \rightarrow \rho, i \models_f^+ \vartheta) \right)$$

where Θ is a subformula of Φ and $\vartheta = \Theta(\psi_1, \ldots, \psi_m)$. This can easily be proved by structural induction. If u is accepted by $\mathcal{D}_{good}$, we have $u \models_f^+ \Phi$ by construction. By the above we have $\rho \models_f^+ \Phi(\psi_1, \ldots, \psi_m)$, as desired. The case for $\mathcal{D}_{bad}$ is symmetric. □

Theorem 5 (Completeness). *Whenever we read an informative good (bad) prefix* $\rho = (\sigma, \tau)$ *for* φ*,* $\mathcal{D}_{good}$ *(*$\mathcal{D}_{bad}$*) must eventually reach an accepting state.*

Proof. For the finite word u' obtained a bit later with $|u'| = |\rho|$,

$$\forall i \in [1, |u'|] \left((\rho, i \models_f^+ \vartheta \rightarrow u', i \models_f^+ \Theta) \wedge (\rho, i \not\models_f^- \vartheta \rightarrow u', i \not\models_f^- \Theta) \right)$$

where Θ is a subformula of Φ and $\vartheta = \Theta(\psi_1, \ldots, \psi_m)$. Again, this can be proved by structural induction (the base step holds by Proposition 3). The theorem follows. □

Remark 1. As pointed out in Example 1, is possible that some of the bad prefixes for the input formula φ are not informative. Certain syntactic restrictions can be imposed on φ to avoid such a situation. For example, it can be shown that all bad prefixes of Safety-MTL [27] formulas will inevitably be extended to informative bad prefixes.[6]

[6] As noted by Kupferman and Vardi [20], all Safety-MTL properties are either *intentionally safe* or *accidentally safe.*

6 Conclusion

We have proposed a new trace-length independent dense-time online monitoring procedure for $\mathsf{MTL}[\mathbf{U},\mathbf{S}]$, based on rewriting the input $\mathsf{MTL}[\mathbf{U},\mathbf{S}]$ formula into an $\mathsf{LTL}[\mathbf{U},\mathbf{S}]$ formula over a set of bounded $\mathsf{MTL}[\mathbf{U},\mathbf{S}]$ atoms. The former is converted into a deterministic (untimed) automaton, while the truth values of the latter are maintained through dynamic programming. We circumvent the potentially delicate issue of translating $\mathsf{MTL}[\mathbf{U},\mathbf{S}]$ to a class of deterministic timed automata.

We are currently investigating whether the procedure can be extended to support more expressive modalities. Another possible direction for future work is to improve the monitoring procedure. For example, the dynamic programming procedures in Section 5.2 can support subformulas with unbounded past. This can be exploited to use a smaller equivalent formula in place of $\hat{\varphi}$.

References

1. Alur, R., Feder, T., Henzinger, T.: The benefits of relaxing punctuality. Journal of the ACM 43(1), 116–146 (1996)
2. Alur, R., Henzinger, T.: Back to the future: towards a theory of timed regular languages. In: Proceedings of FOCS 1992. pp. 177–186. IEEE Computer Society Press (1992)
3. Armoni, R., Korchemny, D., Tiemeyer, A., Vardi, M.Y., Zbar, Y.: Deterministic dynamic monitors for linear-time assertions. In: Proceedings of FATES/RV 2006. LNCS, vol. 4262, pp. 163–177. Springer (2006)
4. Baldor, K., Niu, J.: Monitoring dense-time, continuous-semantics, metric temporal logic. In: Proceedings of RV 2012. LNCS, vol. 7687, pp. 245–259. Springer (2012)
5. Basin, D., Klaedtke, F., Müller, S., Pfitzmann, B.: Runtime monitoring of metric first-order temporal properties. In: Proceedings of FSTTCS 2008. LIPIcs, vol. 2, pp. 49–60. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik (2008)
6. Basin, D., Klaedtke, F., Zălinescu, E.: Algorithms for monitoring real-time properties. In: Proceedings of RV 2011. LNCS, vol. 7186, pp. 260–275. Springer (2011)
7. Bauer, A., Küster, J., Vegliach, G.: From propositional to first-order monitoring. In: Proceedings of RV 2013. LNCS, vol. 8174, pp. 59–75. Springer (2013)
8. Birget, J.C.: State-complexity of finite-state devices, state compressibility and incompressibility. Mathematical Systems Theory 26(3), 237–269 (1993)
9. Bouyer, P., Chevalier, F., Markey, N.: On the expressiveness of TPTL and MTL. In: Proceedings of FSTTCS 2005. LNCS, vol. 3821, pp. 432–443. Springer (2005)
10. Chai, M., Schlingloff, H.: A rewriting based monitoring algorithm for TPTL. In: Proceedings of CS&P 2013. CEUR Workshop Proceedings, vol. 1032, pp. 61–72. CEUR-WS.org (2013)
11. D'Souza, D., Matteplackel, R.: A clock-optimal hierarchical monitoring automaton construction for MITL. Tech. Rep. 2013-1, Department of Computer Science and Automation, Indian Institute of Science (2013), `http://www.csa.iisc.ernet.in/TR/2013/1/lics2013-tr.pdf`
12. Eisner, C., Fisman, D., Havlicek, J., Lustig, Y.: Reasoning with temporal logic on truncated paths. In: Proceedings of CAV 2003. LNCS, vol. 2725, pp. 27–39. Springer (2003)

13. Finkbeiner, B., Kuhtz, L.: Monitor circuits for LTL with bounded and unbounded future. In: Proceedings of RV 2009. LNCS, vol. 5779, pp. 60–75. Springer (2009)
14. Geilen, M.: On the construction of monitors for temporal logic properties. Electronic Notes in Theoretical Computer Science 55(2), 181–199 (2001)
15. Gunadi, H., Tiu, A.: Efficient runtime monitoring with metric temporal logic: A case study in the android operating system. In: Proceedings of FM 2014. LNCS, Springer (2014), to appear.
16. Ho, H.M., Ouaknine, J., Worrell, J.: Online monitoring of metric temporal logic (2014), `http://www.cs.ox.ac.uk/people/hsi-ming.ho/monitoring-full.pdf`, full version.
17. Hunter, P., Ouaknine, J., Worrell, J.: Expressive completeness of metric temporal logic. In: Proceedings of LICS 2013. pp. 349–357. IEEE Computer Society Press (2013)
18. Kini, D., Krishna, S., Pandya, P.: On construction of safety signal automata for MITL[U,S] using temporal projections. In: Proceedings of FORMATS 2011. LNCS, vol. 6919, pp. 225–239. Springer (2011)
19. Koymans, R.: Specifying real-time properties with metric temporal logic. Real-Time Systems 2(4), 255–299 (1990)
20. Kupferman, O., Vardi, M.Y.: Model checking of safety properties. Formal Methods in System Design 19(3), 291–314 (2001)
21. Leucker, M., Schallhart, C.: A brief account of runtime verification. Journal of Logic and Algebraic Programming 78(5), 293–303 (2009)
22. Maler, O., Nickovic, D., Pnueli, A.: Real time temporal logic: Past, present, future. In: Proceedings of FORMATS 2005. LNCS, vol. 3829, pp. 2–16. Springer (2005)
23. Maler, O., Nickovic, D., Pnueli, A.: From MITL to timed automata. In: Proceedings of FORMATS 2006. LNCS, vol. 4202, pp. 274–289. Springer (2006)
24. Manna, Z., Pnueli, A.: Temporal verification of reactive systems: safety, vol. 2. Springer (1995)
25. Markey, N., Raskin, J.: Model checking restricted sets of timed paths. Theoretical Computer Science 358(2-3), 273–292 (2006)
26. Nickovic, D., Piterman, N.: From MTL to deterministic timed automata. In: Proceedings of FORMATS 2010. LNCS, vol. 6246, pp. 152–167. Springer (2010)
27. Ouaknine, J., Worrell, J.: Safety metric temporal logic is fully decidable. In: Proceedings of TACAS 2006. LNCS, vol. 3920, pp. 411–425. Springer (2006)
28. Pedro, A.d.M., Pereira, D., Pinho, L.M., , Pinto, J.S.: A compositional monitoring framework for hard real-time systems. In: Proceedings of NFM 2014. LNCS, Springer (2014), to appear.
29. Sokolsky, O., Havelund, K., Lee, I.: Introduction to the special section on runtime verification. International Journal on Software Tools for Technology Transfer 14(3), 243–247 (2011)
30. Thati, P., Roşu, G.: Monitoring algorithms for metric temporal logic specifications. Electronic Notes in Theoretical Computer Science 113, 145–162 (2005)
31. Vardi, M.Y.: An automata-theoretic approach to linear temporal logic. In: Logics for Concurrency – Structure versus Automata (8th Banff Higher Order Workshop'95). LNCS, vol. 1043, pp. 238–266. Springer (1996)