


Time for Timed Monitorability

Thomas M. Grosen 

Aalborg University, Aalborg, Denmark

Sean Kauffman 

Queen's University, Kingston, Canada

Kim G. Larsen 

Aalborg University, Aalborg, Denmark

Martin Zimmermann 

Aalborg University, Aalborg, Denmark

Abstract

Monitoring is an important part of the verification toolbox, in particular in situations where exhaustive verification using, e.g., model-checking is infeasible. The goal of online monitoring is to determine the satisfaction or violation of a specification during runtime, i.e., based on finite execution prefixes. However, not every specification is amenable to monitoring, e.g., properties for which no finite execution can witness satisfaction or violation. Monitorability is the question of whether a given specification is amenable to monitoring, and has been extensively studied in discrete time.

Here, we study the monitorability problem for real-time properties expressed as Timed Automata. For specifications given by deterministic Timed Muller Automata, we prove decidability while we show that the problem is undecidable for specifications given by nondeterministic Timed Büchi automata.

Furthermore, we refine monitorability to also determine bounds on the number of events as well as the time that must pass before monitoring the property may yield an informative verdict. We prove that for deterministic Timed Muller automata, such bounds can be effectively computed. In contrast we show that for nondeterministic Timed Büchi automata such bounds are not computable.

2012 ACM Subject Classification Theory of computation → Logic and verification; Theory of computation → Modal and temporal logics

Keywords and phrases Monitorability, Monitoring, Timed Automata, MITL

Digital Object Identifier 10.4230/LIPIcs.CONCUR.2025.28

Related Version *arXiv Version*: <https://arxiv.org/abs/2504.10008> [22]

Funding T. M. Grosen and K. G. Larsen have been funded by the Villum Investigator Grant S4OS. T. M. Grosen, K. G. Larsen, and M. Zimmermann have been supported by DIREC - Digital Research Centre Denmark. T. M. Grosen has been supported by Mitacs. S. Kauffman has been supported by the Natural Sciences and Engineering Research Council of Canada.

1 Introduction

A fundamental challenge in Runtime Verification (RV) is that many properties provide no utility when monitored in an online setting. Thus, it is of utmost importance to identify those properties that do provide useful information.

Behaviors of long-running systems are typically specified as languages of infinite words, but online monitors only observe finite (prefixes of) system executions. Thus, a monitor has to determine whether such a finite prefix already implies satisfaction or violation of the property. While many different types of monitors have been proposed, most online monitors return information in the form of *verdicts* about the finite prefix. Inherently to the problem, there are at least three verdicts [10]: $\{\top, \perp, ?\}$, where \top and \perp are *conclusive* verdicts



© Thomas M. Grosen, Sean Kauffman, Kim G. Larsen, Martin Zimmermann;
licensed under Creative Commons License CC-BY 4.0

36th International Conference on Concurrency Theory (CONCUR 2025).

Editors: Patricia Bouyer and Jaco van de Pol; Article No. 28; pp. 28:1–28:20

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

that mean that the finite prefix guarantees that *every* possible infinite extension satisfies, respectively violates, the property, and the *inconclusive* verdict $?$ signifying that neither is the case.

For example, consider an arbiter granting access to a shared resource. The property φ_1 expressing “there is no request during the first second” is satisfied if the first request arrives after two seconds, no matter how the execution continues. Hence, the verdict for such a finite execution is \top . Similarly, the property φ_2 expressing “after any request, there are no requests for at least one second” is violated as soon as two consecutive requests are observed within one second, no matter how the execution continues. Hence, the verdict for such a finite execution is \perp . On the other hand, if requests arrive with a gap of two seconds between them, then the verdict for such a prefix (w.r.t. φ_2) is $?$, since there are infinite extensions satisfying the property and infinite extensions violating it.

As seen above, there are prefixes for which we get a conclusive verdict w.r.t. φ_2 . This distinguishes it from properties like φ_3 expressing “every request is eventually granted”, for which every finite prefix can be extended to satisfy the property and can be extended to violate it. Hence, every finite prefix yields the verdict $?$. Phrased concisely: monitoring the property φ_3 is futile.

However, for the user it is not transparent, while receiving the verdict $?$, whether in the future a conclusive verdict may be given, or whether every possible extension yields the verdict $?$. The concept of *monitorability* has been introduced to capture those properties that are amenable to monitoring. It comes in two variants, strong monitorability (every prefix can be extended to one that yields a conclusive verdict) and weak monitorability (some prefix yields a conclusive verdict). In this language, φ_3 is not weakly monitorable while φ_1 and φ_2 are (even strongly) monitorable. Thus, before constructing and deploying a monitor for a property, it is prudent to first check whether the property is monitorable.

The problem of deciding if a property is monitorable has been studied extensively over the last 20 years (we discuss related work in Section 7), but only in the setting of discrete time until very recently. However, many properties require real-time constraints to express, e.g., deadlines like “every request is answered within 745 ms”. In particular, safety-critical systems are nearly always real-time systems with physics-based deadlines, and these systems tend to benefit the most from formal verification methods like RV. Had the property “the Therac 25 control program must wait eight seconds before switching between X-ray and electron modes” been monitored, it might have saved lives [26]. While monitoring algorithms exist for real-time properties [34, 9, 8, 23], the problem of real-time monitorability has gone largely unexamined.

Our Contribution This work makes monitoring of real-time systems more useful by examining the monitorability problem for real-time properties and by introducing qualitative refinements of the verdict $?$. We consider real-time properties expressed by Timed Automata (TA) over infinite words [3]. Nondeterministic Timed Büchi Automata (TBA) are used for model checking tools like UPPAAL [25] and for monitoring temporal logics [23, 20, 14] and are strictly more expressive than deterministic Timed Muller Automata (DTMA). We prove that strong and weak monitorability are undecidable for nondeterministic TBA, but decidable for DTMA. Thus, one can algorithmically determine that it is futile to monitor properties like φ_3 , thereby increasing the applicability of monitoring of real-time systems.

Furthermore, we introduce monitorability with step-bounded horizons that strengthens monitorability by limiting the number of events in a timed-word before a conclusive verdict must be reached. A step-bounded horizon allows one to determine for a given property

and $n \in \mathbb{N}$, if a conclusive verdict is possible within n steps, enabling corrective actions earlier. Again, we show that monitorability with step-bounded horizons is undecidable for nondeterministic TBA, but decidable for DTMA.

Finally, we refine monitoring of real-time properties with time-horizon verdicts. Here, the verdict $?$ is enhanced with information about the minimum time until a conclusive verdict may be reached. Intuitively, before this time is reached, the monitor will only yield the inconclusive verdict $?$, i.e., no information can be gained from querying the monitor before. This notion was introduced by Grosen et al. [23] as “time-predictive” monitoring. Here, we formally prove that time-horizon queries can be computed effectively for DTMA.

Thus, our results highlight the importance of properties being given by deterministic timed automata when checking monitorability. This is in contrast to monitoring itself, where it suffices to have nondeterministic automata for the property *and* its negation [23], which is, e.g., the case when specifying properties in Metric Interval Temporal Logic [4]. Finally, we show that it is necessary to have automata for the property and the complement, as the monitoring function is otherwise not effectively computable.

2 Preliminaries

The nonnegative integers are denoted by \mathbb{N} and the nonnegative reals by $\mathbb{R}_{\geq 0}$. An alphabet is a finite nonempty set of letters.

A timed word is a pair $\rho = (\sigma, \tau)$, where σ is a (finite or infinite) word over an alphabet Σ and τ is a sequence of non-decreasing, non-negative real numbers of the same length as σ . For convenience, we often write $(\sigma_1, \tau_1)(\sigma_2, \tau_2) \cdots$ for a timed word (σ, τ) . $T\Sigma^*$ and $T\Sigma^\omega$ denote the sets of finite and infinite timed words over Σ . For $n \in \mathbb{N} \cup \{\infty\}$ we denote by $T\Sigma^{\leq n}$ the set of timed words over Σ of length at most n . Given a finite word $\rho = (\sigma_1, \tau_1)(\sigma_2, \tau_2) \cdots (\sigma_n, \tau_n)$, we denote its duration as $\tau(\rho) = \tau_n$. Slightly abusively, we write ε for the empty timed word $(\varepsilon, \varepsilon)$ and define $\tau(\varepsilon) = 0$. For a finite timed word $\rho = (\sigma_1, \tau_1)(\sigma_2, \tau_2) \cdots (\sigma_n, \tau_n)$, a finite or infinite word $\rho' = (\sigma'_1, \tau'_1)(\sigma'_2, \tau'_2) \cdots$ and a timepoint $t \geq \tau(\rho)$, we define the concatenation of ρ and ρ' at t as

$$\rho \cdot_t \rho' = (\sigma_1, \tau_1)(\sigma_2, \tau_2) \cdots (\sigma_n, \tau_n)(\sigma'_1, \tau'_1 + t)(\sigma'_2, \tau'_2 + t) \cdots$$

which is a timed word. As a shorthand, we write $\rho \cdot \rho'$ for $\rho \cdot_{\tau(\rho)} \rho'$. Given two finite words ρ, ρ' and a timepoint $t \geq \tau(\rho)$, we write $\rho \sqsubseteq_t \rho'$ if there exists a ρ'' such that $\rho \cdot_t \rho'' = \rho'$.

► **Example 1.** Consider the words $\rho = (a, 0)(b, 10)$ and $\rho' = (c, 5)(d, 15)$. We have

- $\rho \cdot \rho' = \rho \cdot_{10} \rho' = (a, 0)(b, 10)(c, 15)(d, 25)$,
- $\rho \cdot_{15} \rho' = (a, 0)(b, 10)(c, 20)(d, 30)$, and
- $\rho' \cdot \rho = \rho' \cdot_{15} \rho = (c, 5)(d, 15)(a, 15)(b, 25)$.

Timed Automata A timed Büchi automaton (TBA) is a tuple $\mathcal{A} = (Q, Q_0, \Sigma, \mathcal{X}, \Delta, F)$ where Q is a finite set of locations, $Q_0 \subseteq Q$ is the set of initial locations, Σ is an alphabet, \mathcal{X} is a finite set of clocks, $F \subseteq Q$ is a set of accepting locations, and $\Delta \subseteq Q \times Q \times \Sigma \times 2^{\mathcal{X}} \times G(\mathcal{X})$ is a set of transitions, where $G(\mathcal{X})$ is the set of clock constraints over \mathcal{X} . A transition $(q, q', \alpha, \lambda, g) \in \Delta$ is an edge from q to q' with label $\alpha \in \Sigma$, where $\lambda \in 2^{\mathcal{X}}$ is a set of clocks to be reset and $g \in G(\mathcal{X})$ is a clock constraint. A clock constraint is a finite conjunction of atomic constraints of the form $x \sim n$ where $x \in \mathcal{X}$, $n \in \mathbb{N}$, and $\sim \in \{<, \leq, =, \geq, >\}$. A state of \mathcal{A} is a pair (q, v) where $q \in Q$ and $v: \mathcal{X} \rightarrow \mathbb{R}_{\geq 0}$ is a clock valuation.

A run of \mathcal{A} from a state (q_0, v_0) over an infinite word $(\sigma, \tau) \in T\Sigma^\omega$ is a sequence of steps of the form

$$(q_0, v_0) \xrightarrow{(\sigma_1, \tau_1)} (q_1, v_1) \xrightarrow{(\sigma_2, \tau_2)} (q_2, v_2) \xrightarrow{(\sigma_3, \tau_3)} \dots$$

where for every $i \geq 1$, there is a transition $(q_{i-1}, q_i, \sigma_i, \lambda_i, g_i) \in \Delta$ such that $v_i(x) = 0$ for all $x \in \lambda_i$ and $v_i(x) = v_{i-1}(x) + \tau_i - \tau_{i-1}$ otherwise, and g is satisfied by $v_{i-1} + \tau_i - \tau_{i-1}$ (where we use $\tau_0 = 0$). For a run r , $\text{inf}(r) \subseteq Q$ denotes the set of locations visited infinitely often in r . A run r is (Büchi) accepting if $\text{inf}(r) \cap F \neq \emptyset$.

A timed Muller automaton (TMA) $\mathcal{A} = (Q, Q_0, \Sigma, \mathcal{X}, \Delta, \mathcal{F})$ is like a TBA, but the set F of accepting locations is replaced by a set $\mathcal{F} \subseteq 2^Q$ of sets of locations. A run r of \mathcal{A} is (Muller) accepting if $\text{inf}(r) \in \mathcal{F}$.

The language $L(\mathcal{A})$ of a timed (Büchi or Muller) automaton is the set of words $\rho \in T\Sigma^\omega$ such that \mathcal{A} has an accepting run over ρ .

An automaton is deterministic if the set of initial locations is a singleton and if all edges from the same location and with the same label must have disjoint clock constraints. We use the abbreviations DTBA and DTMA to refer to the two deterministic automaton models.

► **Proposition 2.** *The following results on TBA and TMA are due to Alur and Dill [3].*

1. Let $\mathcal{A} = (Q, Q_0, \Sigma, \mathcal{X}, \Delta, F)$ be a TBA and let $\mathcal{A}' = (Q, Q_0, \Sigma, \mathcal{X}, \Delta, \{F' \mid F \cap F' \neq \emptyset\})$, which is a TMA. Then, $L(\mathcal{A}) = L(\mathcal{A}')$. Furthermore, if \mathcal{A} is deterministic, then so is \mathcal{A}' .
2. For every TMA $\mathcal{A} = (Q, Q_0, \Sigma, \mathcal{X}, \Delta, \mathcal{F})$, there is a TBA \mathcal{A}' with $L(\mathcal{A}) = L(\mathcal{A}')$. The set of locations of \mathcal{A}' has the form $Q \times \mathcal{F} \times \{0, 1, \dots, |Q|\}$.
3. The class of languages accepted by DTMA is a strict subset of the class of languages accepted by TBA.

Logic We use Metric Interval Temporal Logic (MITL) to formally express properties to be monitored. The syntax of MITL formulas over a finite alphabet Σ is defined as

$$\varphi ::= p \mid \neg\varphi \mid \varphi \vee \psi \mid X_I\varphi \mid \varphi U_I\psi$$

where $p \in \Sigma$ and I ranges over non-singular intervals over $\mathbb{R}_{\geq 0}$ with endpoints in $\mathbb{N} \cup \{\infty\}$. Note that we often write $\sim n$ for $I = \{d \in \mathbb{R}_{\geq 0} \mid d \sim n\}$ where $\sim \in \{<, \leq, \geq, >\}$, and $n \in \mathbb{N}$. We also define the standard syntactic sugar $\text{true} = p \vee \neg p$, $\text{false} = \neg \text{true}$, $\varphi \wedge \psi = \neg(\neg\varphi \vee \neg\psi)$, $\varphi \rightarrow \psi = \neg\varphi \vee \psi$, $F_I\varphi = \text{true} U_I\varphi$, and $G_I\varphi = \neg F_I\neg\varphi$.

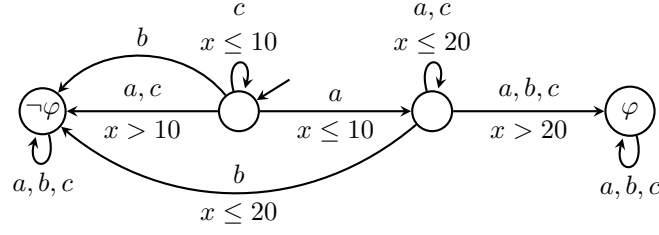
The semantics of MITL is defined over infinite timed words. Given such a timed word $\rho = (\sigma_1, \tau_1)(\sigma_2, \tau_2) \dots \in T\Sigma^\omega$, a position $i \geq 1$, and an MITL formula φ , we inductively define the satisfaction relation $\rho, i \models \varphi$ as follows:

- $\rho, i \models p$ if and only if $p = \sigma_i$.
- $\rho, i \models \neg\varphi$ if and only if $\rho, i \not\models \varphi$.
- $\rho, i \models \varphi \vee \psi$ if $\rho, i \models \varphi$ or $\rho, i \models \psi$.
- $\rho, i \models X_I\varphi$ if and only if $\rho, (i+1) \models \varphi$ and $\tau_{i+1} - \tau_i \in I$.
- $\rho, i \models \varphi U_I\psi$ if and only if there exists $k \geq i$ s.t. $\rho, k \models \psi$, $\tau_k - \tau_i \in I$, and $\rho, j \models \varphi$ for all $i \leq j < k$.

We write $\rho \models \varphi$ whenever $\rho, 1 \models \varphi$, and say that ρ satisfies φ . The language $L(\varphi)$ of an MITL formula φ is the set of all infinite timed words that satisfy φ .

► **Proposition 3** ([4, 12]). *For each MITL formula φ there is a TBA \mathcal{A} such that $L(\varphi) = L(\mathcal{A})$.*

► **Example 4.** Figure 1 illustrates the above proposition by providing a DTBA over the alphabet $\Sigma = \{a, b, c\}$ for the formula $F_{[0,10]}a \wedge G_{[0,20]}\neg b$ and its negation.



■ **Figure 1** DTBA for the language of the MITL formula $\varphi = F_{[0,10]}a \wedge G_{[0,20]}\neg b$ and its negation: If location φ ($\neg\varphi$) is accepting then it accepts $L(\varphi)$ ($L(\neg\varphi)$, respectively).

177 **Monitoring** The monitoring problem asks to make verdicts about the satisfaction or violation
 178 of properties (over infinite timed words) after having observed only a finite prefix. Here, we
 179 follow the classical approach of considering the possible extensions of a finite observations.

180 ► **Definition 5 (Observation).** An observation is a pair (ρ, t) containing a finite timed word ρ
 181 and a timepoint $t \geq \tau(\rho)$, representing the current timepoint (which might be later than the
 182 last observed event in ρ). As we use observations as inputs for algorithms, we require that all
 183 timepoints in ρ and t are rational.

184 We continue with giving some intuition for the three-valued monitoring approach. Here,
 185 one is given an observation and aims to determine whether a property φ (of infinite timed
 186 words) is already satisfied, already violated, or neither.

- 187 ■ If all possible extensions of the observation satisfy φ , then we give the corresponding
 188 verdict \top signifying that the observation conclusively witnesses satisfaction of φ .
- 189 ■ If all possible extensions of the observation violate φ , then we give the corresponding
 190 verdict \perp signifying that the observation conclusively witnesses violation of φ .
- 191 ■ Otherwise, i.e., if there is an extension of the observation that satisfies φ and there is a
 192 extension of the observation that violates φ , then we give the inconclusive verdict $?$.

193 Let us formalize this intuition.

194 ► **Definition 6 (Timed Monitoring Function).** Given a property $\varphi \subseteq T\Sigma^\omega$ and an observa-
 195 tion (ρ, t) , the monitoring function V_φ is defined as

$$196 \quad V_\varphi(\rho, t) = \begin{cases} \top & \text{if } \rho \cdot_t \mu \in \varphi \text{ for all } \mu \in T\Sigma^\omega, \\ \perp & \text{if } \rho \cdot_t \mu \notin \varphi \text{ for all } \mu \in T\Sigma^\omega, \\ ? & \text{otherwise.} \end{cases}$$

197 In the following, we use $V_\varphi(\rho)$ as a shorthand for $V_\varphi(\rho, \tau(\rho))$.

198 ► **Example 7.** Consider the specification $\varphi = F_{[0,10]}a \wedge G_{[0,20]}\neg b$ from Example 4. We have

- 199 ■ $V_\varphi((a, 3), 4) = ?$,
- 200 ■ $V_\varphi((a, 11), 11) = \perp$,
- 201 ■ $V_\varphi((a, 3)(c, 7), 13) = ?$,
- 202 ■ $V_\varphi((a, 3)(c, 7)(c, 22), 22) = \top$, but
- 203 ■ $V_\varphi((a, 3)(c, 7)(b, 12), 12) = \perp$. Also,
- 204 ■ $V_\varphi((a, 3)(c, 7), 22) = \top$ while
- 205 ■ $V_\varphi((a, 3)(c, 7)) = V_\varphi((a, 3)(c, 7), 7) = ?$, i.e., the current timepoint t can yield conclusive
 206 verdicts when time is passing, even if no new events are observed.

► **Proposition 8** (Effectiveness of Timed Monitoring [23]). V_φ is effectively computable by a zone-based online algorithm¹ that requires TBA for both φ and $T\Sigma^\omega \setminus \varphi$.

Recall that TBA are *not* closed under complement [3], so this result is not applicable to all properties accepted by TBA. However, for the important special case of properties φ specified in MITL, V_φ is effectively computable, as MITL properties are closed under complementation (as the logic allows for negations) and MITL formulas can be translated into equivalent TBA (see Proposition 3). Similarly, when φ is given by a DTMA, then V_φ is effectively computable, as DTMA are closed under complementation and can be turned into equivalent TBA [3].

However, it was previously open whether V_φ was computable if φ was given by a non-deterministic automaton, but we did not have access to an automaton for the complement $T\Sigma^\omega \setminus \varphi$. In Section 3, we answer the question negatively.

Monitorability Not every property is amenable to monitoring, e.g., for $\varphi = G_{\geq 0}F_{\geq 0}a$, we have $V_\varphi(\rho, t) = ?$ for every observation (ρ, t) . The reason is that every ρ can be extended to satisfy φ and can be extended to violate φ . In the untimed setting, much effort has been put into characterizing the monitorable properties, i.e., those for which monitoring can generate some information. Here, we consider monitorability in the timed setting.

► **Definition 9** (Timed Monitorability). Fix an observation (ρ, t) and a property $\varphi \subseteq T\Sigma^\omega$.

■ φ is strongly (ρ, t) -monitorable if and only if

for all $\rho' \in T\Sigma^*$ there exists $\rho'' \in T\Sigma^*$ such that $V_\varphi(\rho \cdot_t \rho' \cdot \rho'') \in \{\top, \perp\}$.

■ φ is weakly (ρ, t) -monitorable if and only if

there exists $\rho'' \in T\Sigma^*$ such that $V_\varphi(\rho \cdot_t \rho'') \in \{\top, \perp\}$.

■ φ is strongly monitorable if it is strongly $(\varepsilon, 0)$ -monitorable.

■ φ is weakly monitorable if it is weakly $(\varepsilon, 0)$ -monitorable.

► **Example 10.**

1. Consider the property $\varphi_1 = F_{\geq 0}a$. For every observation (ρ, t) , we have $V_{\varphi_1}(\rho \cdot_t (a, 0), t) = \top$. Hence, φ_1 is strongly monitorable.
2. Now, consider $\varphi_2 = a \rightarrow G_{\geq 0}F_{\geq 0}a$. Then, we have $V_{\varphi_2}((b, 0), 0) = \top$, as every extension of $((b, 0), 0)$ satisfies φ_2 (as the premise is violated). Hence, φ_2 is weakly monitorable. However, it is not strongly monitorable: Consider the observation $((a, 0), 0)$, for which every extension satisfies the premise. We have $(a, 0) \cdot_0 \rho'' \cdot (a, 0)(a, 1)(a, 2) \cdots \models \varphi_2$ and $(a, 0) \cdot_0 \rho'' \cdot (b, 0)(b, 1)(b, 2) \cdots \not\models \varphi_2$. Hence, $V_{\varphi_2}((a, 0) \cdot_0 \rho'') = ?$ for all ρ'' .
3. Now, consider $\varphi_3 = G_{\geq 0}F_{\geq 0}a$. Arguments as for φ_2 show that it is neither strongly nor weakly monitorable, as every finite word can be extended by $(a, 0)(a, 1)(a, 2) \cdots$ to satisfy φ_3 and can be extended by $(b, 0)(b, 1)(b, 2) \cdots$ to violate φ_3 .

► **Remark 11.** The astute reader might wonder why we quantify only over words ρ' (and ρ'') in Definition 9 and not over words and timepoints to concatenate at. The reason is that both definitions are equivalent, as $\rho_1 \cdot_t \rho_2$, for finite words ρ_1 and ρ_2 and a timepoint $t \geq \tau(\rho)$, is equal to $\rho_1 \cdot \rho'_2$, where ρ'_2 is obtained from ρ_2 by incrementing all its timepoints by $t - \tau(\rho_1)$.

Let us continue by collecting some simple consequences of Definition 9.

¹ See Page 11 for a formal definition of zones.

- 246 ▶ **Remark 12.** Let (ρ, t) and (ρ', t') be two observations with $\rho \sqsubseteq_t \rho'$, and let $\varphi \subseteq T\Sigma^\omega$.
- 247 1. If φ is strongly (ρ, t) -monitorable, then it is also weakly (ρ, t) -monitorable. Thus, if φ is
- 248 strongly monitorable, then it is also weakly monitorable.
- 249 2. If φ is strongly (ρ, t) -monitorable, then it is also strongly (ρ', t') -monitorable.
- 250 3. If φ is not weakly (ρ, t) -monitorable, then it is also not weakly (ρ', t') -monitorable.

251 In the following we study the decidability of monitorability, i.e., we consider the following

252 decision problems where properties are given by timed automata:

- 253 1. Given a property φ , is φ strongly monitorable?
- 254 2. Given a property φ , is φ weakly monitorable?
- 255 3. Given a property φ and an observation (ρ, t) , is φ strongly (ρ, t) -monitorable?
- 256 4. Given a property φ and an observation (ρ, t) , is φ weakly (ρ, t) -monitorable?

257 By definition, if Problem 3 is decidable for a class of properties, then Problem 1 is also

258 decidable for the same class of properties. Hence, if Problem 1 is undecidable for a class of

259 properties, then Problem 3 is also undecidable for the same class of properties. A similar

260 relation holds between Problem 2 and Problem 4.

261 3 Monitoring and Monitorability for TBA

262 In this section, we prove that the monitoring function is not computable and that the strong

263 types of monitorability are undecidable when the property is given by a TBA. Note that this

264 shows that the positive results for monitoring in the literature [23, 20, 14], which require

265 automata both for the property and its complement, are tight in that sense: Only giving

266 an automaton for the property, but not for its complement, is not sufficient to compute the

267 monitoring function. We start by investigating the computability of the monitoring function.

268 ▶ **Theorem 13** (Ineffectiveness of Timed Monitoring). *The function “Given a TBA \mathcal{A} and an*

269 *observation (ρ, t) , return $V_{L(\mathcal{A})}(\rho, t)$ ” is not computable.*

270 **Proof.** For every property $\varphi \subseteq T\Sigma^\omega$, we have $V_\varphi(\varepsilon, 0) = \top$ if and only if $\varphi = T\Sigma^\omega$. Hence,

271 universality of a TBA \mathcal{A} reduces to checking whether $V_{L(\mathcal{A})}(\varepsilon, 0) = \top$. As universality for

272 timed automata is undecidable [5], the monitoring function cannot be computable. ◀

273 Note that the specification automaton \mathcal{A} is part of the input in the problem considered

274 in Theorem 13. We leave it open whether $V_{L(\mathcal{A})}$ is computable for every fixed \mathcal{A} , i.e., in the

275 setting where only the observation is the input.

276 Next, we turn our attention to deciding monitorability for TBA.

277 ▶ **Theorem 14.** *Strong and weak monitorability are undecidable for properties given by TBA.*

278 **Proof.** Strong monitorability for untimed non-deterministic Büchi Automata is PSPACE-

279 complete [17], which can be shown by a reduction from the universality problem [18].

280 Analogously, we reduce the (undecidable [3]) universality problem for non-deterministic

281 timed automata (over finite words) to the problem of strong monitorability, following Diekert,

282 Muscholl, and Walukiewicz [18].

283 Let $\mathcal{A} = (Q, Q_0, \Sigma, \mathcal{X}, \Delta, F)$ be such a timed automaton, i.e., a finite run is accepting if it

284 ends in a location in F . We assume w.l.o.g. that \mathcal{A} is complete in the sense that every word

285 has a run. This can always be achieved by adding a fresh sink location and by rerouting all

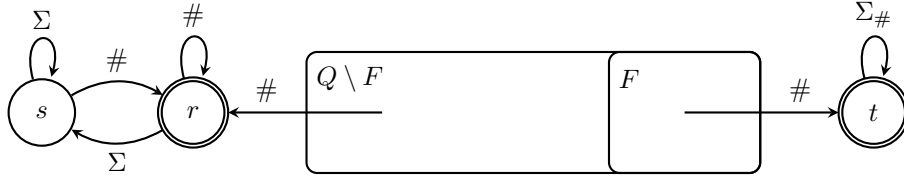
286 missing transitions to it.

287 We add a new letter $\# \notin \Sigma$ to obtain $\Sigma_\# = \Sigma \cup \{\#\}$ and construct a TBA $\mathcal{A}' =$

288 $(Q', Q_0, \Sigma_\#, \mathcal{X}', \Delta', F')$ from \mathcal{A} such that \mathcal{A}' is strongly monitorable if and only if $L(\mathcal{A}) = T\Sigma^*$.

To this end, we introduce three new locations r, s, t , i.e., $Q' = Q \cup \{r, s, t\}$. Next, we define Δ' (see Figure 2) by copying the transitions from Δ and by adding the following transitions, where we write $q \xrightarrow{a} q'$ to denote $(q, q', a, \emptyset, \text{true})$:

- $\{q \xrightarrow{\#} r \mid q \in Q \setminus F\}$: from every non-accepting location of \mathcal{A} , there is a $\#$ -transition to r .
 - $\{r \xrightarrow{a} s, s \xrightarrow{a} s \mid a \in \Sigma\}$: for every $a \neq \#$ there is an a -transition from r to s and an a -labeled self-loop on s .
 - $\{s \xrightarrow{\#} r, r \xrightarrow{\#} r\}$: there is a $\#$ -transition from s to r and a $\#$ -labeled self-loop on r .
 - $\{q \xrightarrow{\#} t \mid q \in F\}$: from every accepting location there is a $\#$ -transition to t .
 - $\{t \xrightarrow{a} t \mid a \in \Sigma_{\#}\}$: for every letter in $\Sigma_{\#}$, there is a self-loop on t labeled with that letter.
- We define the accepting locations of \mathcal{A}' as $F' = Q \cup \{r, t\}$.



■ **Figure 2** The TBA \mathcal{A}' constructed for the proof of Theorem 14.

Figure 2 shows the TBA \mathcal{A}' with the locations of the (finite-word) timed automaton \mathcal{A} partitioned into F and $Q \setminus F$. In the figure, the new locations r, s , and t and accompanying transitions are shown separately from the original automaton. Double circles denote accepting locations. Intuitively, processing a $\#$ from an accepting location of \mathcal{A} takes the run to the accepting sink t . On the other hand, processing a $\#$ from a non-accepting location of \mathcal{A} takes the run to a component where the run continuation is accepting if and only if it processes infinitely many $\#$'s.

It remains to show that the language $L(\mathcal{A}')$ is strongly monitorable if and only if $L(\mathcal{A}) = T\Sigma^*$. One direction is trivial: If $L(\mathcal{A}) = T\Sigma^*$, then $L(\mathcal{A}') = T\Sigma_{\#}^{\omega}$, as all words without a $\#$ can be processed using the states of the complete automaton \mathcal{A} (which are all accepting in \mathcal{A}') and all words with a $\#$ can be accepted by moving to state t (which is an accepting sink) when processing the first $\#$. Hence, $L(\mathcal{A}')$ is strongly monitorable.

On the other hand, if $L(\mathcal{A}) \neq T\Sigma^*$, then there exists a finite word $\rho \notin L(\mathcal{A})$. Hence, every run prefix of \mathcal{A}' processing $\rho \cdot \tau(\rho) (\#, \tau(\rho))$ must be in location r . Now, choose some $a \in \Sigma$. Then, for all $\rho'' \in T\Sigma_{\#}^*$, we have:

- $\rho \cdot \tau(\rho) (\#, \tau(\rho)) \cdot \rho'' \cdot (a, 0)(a, 1)(a, 2) \cdots \notin L(\mathcal{A}')$ (as it contains only finitely many $\#$).
- $\rho \cdot \tau(\rho) (\#, \tau(\rho)) \cdot \rho'' \cdot (\#, 0)(\#, 1)(\#, 2) \cdots \in L(\mathcal{A}')$ (as it contains infinitely many $\#$).

Hence, $V_{L(\mathcal{A}')}(\rho \cdot \tau(\rho) (\#, \tau(\rho)) \cdot \rho'') = ?$ for all ρ'' . Thus, $L(\mathcal{A}')$ is not strongly monitorable.

In the case for weak monitorability we reduce the (undecidable [3]) universality problem for TBA (not TA as we did in the strong case). Additionally, as an intermediate step, we consider a problem about Brzozowski derivatives of properties of infinite timed words: Let $\rho \in T\Sigma^*$ be a finite timed word and $\varphi \subseteq T\Sigma^{\omega}$ be such a property. We say that ρ is a universal prefix for φ if the Brzozowski derivative

$$\{\mu \in T\Sigma^{\omega} \mid \rho \cdot \mu \in \varphi\}$$

of φ and ρ is equal to $T\Sigma^{\omega}$. Note that if φ is equal to $T\Sigma^{\omega}$, then every prefix is universal for φ . However, the property $(a, 0) \cdot T\Sigma^{\omega}$ has a universal prefix (e.g., $(a, 0)$), but is not universal.

Further, we say that φ is weakly \top -monitorable (cp. Definition 21) if and only if there is a ρ such that $V_\varphi(\rho) = \top$, i.e., in comparison to (standard) weak monitorability, we only consider the verdict \top . Note that φ has a universal prefix if and only if it is weakly \top -monitorable.

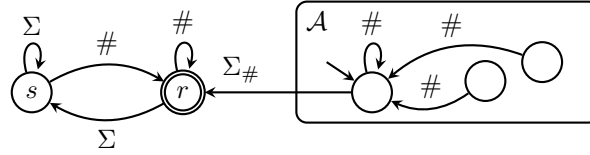
So, it remains to first reduce universality of TBA to the existence of a universal prefix for languages of TBA and then to reduce weak \top -monitorability to weak monitorability.

We begin with reducing TBA universality to the existence of universal prefixes. Intuitively, we add edges allowing to “restart” a run which allows every prefix to simulate the empty prefix. Then, every prefix behaves like the empty prefix, which is universal if and only if the automaton is universal.

Let $\mathcal{A} = (Q, Q_0, \Sigma, \mathcal{X}, \Delta, F)$ be a TBA. We add a new letter $\# \notin \Sigma$ to obtain $\Sigma_\# = \Sigma \cup \{\#\}$ and construct a TBA $\mathcal{A}' = (Q', Q_0, \Sigma_\#, \mathcal{X}, \Delta', F')$ from \mathcal{A} such that $L(\mathcal{A}) = T\Sigma^\omega$ if and only if \mathcal{A}' has a universal prefix. To this end, we introduce two new locations r and s , i.e., $Q' = Q \cup \{r, s\}$. Next, we define Δ' (see Figure 3) by copying the transitions from Δ and by adding the following transitions, where we write $q \xrightarrow{a} q'$ to denote $(q, q', a, \emptyset, \text{true})$:

- $\{q \xrightarrow{a} r \mid q \in Q_0, a \in \Sigma_\#\}$: from every initial location of \mathcal{A} , there is an a -transition to r for every $a \in \Sigma_\#$.
- $\{r \xrightarrow{a} s, s \xrightarrow{a} s \mid a \in \Sigma\}$: for every $a \neq \#$ there is an a -transition from r to s and an a -labeled self-loop on s .
- $\{s \xrightarrow{\#} r, r \xrightarrow{\#} r\}$: there is a $\#$ -transition from s to r and a $\#$ -labeled self-loop on r .
- $\{(q, q_0, \#, \mathcal{X}, \text{true}) \mid q \in Q, q \in Q_0\}$: from every location there is a $\#$ -transition to each initial location, which resets all clocks.

We define the accepting locations of \mathcal{A}' as $F' = Q \cup \{r\}$.



■ **Figure 3** The TBA \mathcal{A}' constructed for the proof of Theorem 14.

Let $L(\mathcal{A}) = T\Sigma^\omega$ and fix some $\mu' \in T\Sigma_\#^\omega$. If μ' contains infinitely many $\#$'s, then it is accepted by \mathcal{A}' using the new states r and s . On the other hand, if μ' contains only finitely many $\#$'s, then it has the form

$$\mu' = \rho_0 \cdot (\#, t_0) \cdot \rho_1 \cdot (\#, t_1) \cdots (\#, t_{n-2}) \cdot \rho_{n-1} \cdot (\#, t_{n-1}) \cdot \mu''$$

for some $n \geq 0$, where the ρ_i and μ'' are $\#$ -free. As each ρ_i is a prefix of some word in $L(\mathcal{A}) = T\Sigma^\omega$ and μ'' is in $L(\mathcal{A}) = T\Sigma^\omega$, one can construct an accepting run of \mathcal{A}' on μ' . Thus, $L(\mathcal{A}') = T\Sigma_\#^\omega$, i.e., ε is a universal prefix of \mathcal{A}' .

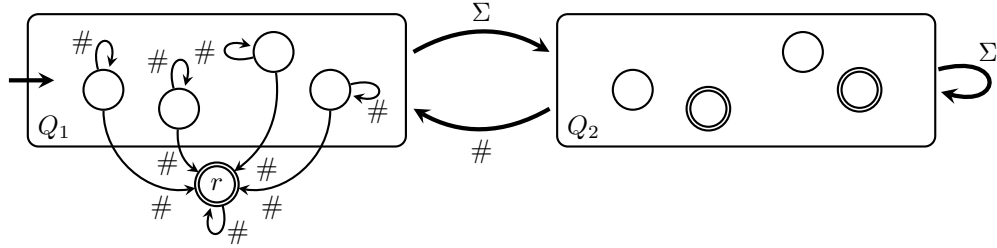
Now for the other direction. If a word ρ is a universal prefix of \mathcal{A}' , then for all $\mu \in T\Sigma^\omega$, the word $\rho \cdot (\#, 0) \cdot \mu$ is accepted by \mathcal{A}' . Hence, there is an accepting run of \mathcal{A}' on $\rho \cdot (\#, 0) \cdot \mu$. As all $\#$ -transitions lead to an initial state of \mathcal{A} and reset the clocks, and as μ does not contain any $\#$'s, the suffix of the run on μ must also be an accepting run of \mathcal{A} , i.e., we have $\mu \in L(\mathcal{A})$. Thus, $L(\mathcal{A}) = T\Sigma^\omega$. This concludes the first step of our proof.

In the second and last step of our proof we reduce weak \top -monitorability to (standard) weak monitorability. Intuitively, we manipulate the automaton so that it can never give the verdict \perp while preserving the existence of observations that yield the verdict \top .

Let $\mathcal{A} = (Q, Q_0, \Sigma, \mathcal{X}, \Delta, F)$ be a TBA. We add a new letter $\# \notin \Sigma$ to obtain $\Sigma_\# = \Sigma \cup \{\#\}$ and construct a TBA $\mathcal{A}' = (Q', Q_0', \Sigma_\#, \mathcal{X}, \Delta', F')$ from \mathcal{A} such that \mathcal{A} is weakly \top -monitorable if and only if \mathcal{A}' is weakly monitorable.

Now Q' essentially contains two copies of Q : $Q_1 = Q \times \{1\}$ and $Q_2 = Q \times \{2\}$ as well as a new additional accepting location r . The transition relation of \mathcal{A}' is defined as follows (see also Figure 4), we write $q \xrightarrow{a} q'$ to denote $(q, q', a, \emptyset, \text{true})$:

- $\{(q, 1) \xrightarrow{\#} (q, 1), (q, 1) \xrightarrow{\#} r \mid q \in Q\}$: locations of Q_1 have a $\#$ -labeled self-loop and a $\#$ -transition to r .
 - $\{((q, 1), (q', 2), a, \lambda, g) \mid (q, q', a, \lambda, g) \in \Delta\}$: locations of Q_1 have their Σ -labeled transitions redirected to Q_2 .
 - $\{(q, 2), (q', 2), a, \lambda, g) \mid (q, q', a, \lambda, g) \in \Delta\}$: locations of Q_2 have copies of the Σ -labeled transitions from \mathcal{A} .
 - $\{(q, 2) \xrightarrow{\#} (q, 1) \mid q \in Q\}$: locations of Q_2 have $\#$ -labeled transitions directed back to Q_1 .
 - $\{r \xrightarrow{\#} r\}$: The new location r has a $\#$ -labeled self-loop.
- The set Q'_0 of initial locations of \mathcal{A}' is $Q_0 \times \{1\}$ and the set F' of accepting locations of \mathcal{A}' is $F \times \{2\} \cup \{r\}$.



■ **Figure 4** The TBA \mathcal{A}' constructed for the proof of Theorem 14. Intuitively, \mathcal{A}' has two disjoint copies of the locations of \mathcal{A} and Σ -labeled transitions lead from both copies to the second copy, while $\#$ -labeled transitions from the second copy lead to the first copy. The first copy additionally has $\#$ -labeled self-loops on all locations and $\#$ -labeled transitions to r .

Now, let $\rho \in T\Sigma^*$ be a timed word witnessing weak \top -monitorability of \mathcal{A} , i.e., $\rho \cdot \mu \in L(\mathcal{A})$ for every $\mu \in T\Sigma^\omega$. We define $\rho' = \rho$. In \mathcal{A} , processing ρ' leads to a location in Q_2 . We argue that ρ' witnesses weak monitorability of \mathcal{A}' . To this end, consider any $\mu' \in T\Sigma_\#^\omega$ and let μ be obtained from μ' by removing all occurrences of $\#$. If μ is infinite, we may simply mimic the accepting run of \mathcal{A} on μ from ρ in \mathcal{A}' . Clearly, this will also be accepting. If μ is finite, μ' must have a suffix of the form $(\#^\omega, \tau)$, where τ is a sequence of timepoints. This suffix is accepted by utilizing the $\#$ transitions to r , from where the suffix $(\#^\omega, \tau)$ can be accepted.

For the opposite direction, let $\rho' \in T\Sigma_\#^*$ be a timed word witnessing weak monitorability of \mathcal{A}' . That is, either $\rho' \cdot \mu' \in L(\mathcal{A}')$ for all μ' or $\rho' \cdot \mu' \notin L(\mathcal{A}')$ for all μ' . However, note that for any finite ρ' , we have $\rho' \cdot (\#, 0)(\#, 1)(\#, 2) \cdots \in L(\mathcal{A}')$, due to the $\#$ -labeled transition to r , from where $(\#, 0)(\#, 1)(\#, 2) \cdots$ is accepted. Thus, we must be in the first case where we have $\rho' \cdot \mu' \in L(\mathcal{A}')$ for all μ' .

Now, let $\rho \in T\Sigma^*$ be obtained from ρ' by stripping all occurrences of $\#$ in ρ' . We show that ρ is a word witnessing weak \top -monitorability of \mathcal{A} . Note that any run of \mathcal{A}' on ρ' reaching a location in Q_1 or Q_2 can be mimicked by a run of \mathcal{A} on ρ reaching the corresponding location in Q . Let $\mu \in T\Sigma^\omega$. Then $\rho' \cdot \mu \in L(\mathcal{A}')$. Clearly, this must be due to an accepting run where a suffix of the run processing μ stays in Q_2 . Hence, $\rho \cdot \mu \in L(\mathcal{A})$, as we may mimic, for the prefix ρ , the prefix processing ρ' of the accepting run of \mathcal{A}' processing $\rho' \cdot \mu$, and, for the suffix μ , we simply copy the run staying in Q_2 . ◀

Due to Remark 12, we also obtain the undecidability of strong and weak (ρ, t) -monitorability.

399 ► **Corollary 15.** *Strong and weak (ρ, t) -monitorability is undecidable for properties given by*
 400 *TBA, even if (ρ, t) is fixed.*

401 **4 Monitorability for DTMA**

402 In this section, we show that (strong and weak) monitorability is decidable for properties
 403 given by DTMA. The key difference to TBA, for which we have shown that monitorability is
 404 undecidable, is that they are trivially closed under complement [3], which we rely on in our
 405 proof. This again demonstrates the importance of having automata for the property and its
 406 complement, as noted throughout this paper.

407 Let us begin by introducing some notation and definitions. Fix some TBA or TMA $\mathcal{A} =$
 408 $(Q, Q_0, \Sigma, \mathcal{X}, \Delta, \text{Acc})$. We write $(q_0, v_0) \xrightarrow{\rho}_{\mathcal{A}} (q_n, v_n)$ for a finite timed word $\rho = (\sigma, \tau) \in T\Sigma^*$
 409 to denote the existence of a finite sequence

$$410 (q_0, v_0) \xrightarrow{(\sigma_1, \tau_1)} (q_1, v_1) \xrightarrow{(\sigma_2, \tau_2)} \dots \xrightarrow{(\sigma_n, \tau_n)} (q_n, v_n)$$

411 of states, where for all $1 \leq i \leq n$ there is a transition $(q_{i-1}, q_i, \sigma_i, \lambda_i, g_i)$ such that $v_i(c) = 0$
 412 for all c in λ_i and $v_{i-1}(c) + (\tau_i - \tau_{i-1})$ otherwise, and g is satisfied by the valuation
 413 $v_{i-1} + (\tau_i - \tau_{i-1})$, where we use $\tau_0 = 0$.

414 ► **Definition 16** (Non-Empty Language States, Reach Set, Pre^*). *Let \mathcal{A} be a TBA or TMA.*

415 ■ *The set of non-empty language states of \mathcal{A} is*

$$416 S_{\mathcal{A}}^{\text{ne}} = \{s \mid s \text{ is a state of } \mathcal{A} \text{ and } L(\mathcal{A}, s) \neq \emptyset\}.$$

417 Here, $L(\mathcal{A}, s)$ is the set of infinite timed words accepted by a run starting in the state s .

418 ■ *Given an observation (ρ, t) , the set of states in which a run over ρ can end starting from*
 419 *the initial states of \mathcal{A} after time t has passed is*

$$420 \mathcal{T}_{\mathcal{A}}(\rho, t) = \bigcup_{q_0 \in Q_0} \{(q, v + (t - \tau(\rho))) \mid (q_0, v_0) \xrightarrow{\rho}_{\mathcal{A}} (q, v)\},$$

421 where v_0 is the clock valuation mapping every clock to 0. We call $\mathcal{T}_{\mathcal{A}}(\rho, t)$ the reach set
 422 of (ρ, t) in \mathcal{A} .

423 ■ *We define $\text{Pre}_{\mathcal{A}}^*(S)$ of a set S of states as*

$$424 \{(q, v) \mid (q, v) \xrightarrow{\rho}_{\mathcal{A}} (q', v') \text{ for some } (q', v' + (t - \tau(\rho))) \in S, \rho \in T\Sigma^*, \text{ and } t \geq \tau(\rho)\}.$$

425 A symbolic state is a pair (q, Z) of a location q and a zone Z . A zone is a finite
 426 conjunction of clock constraints of the form $x_1 \sim n$ or $x_1 - x_2 \sim n$, where x_1, x_2 are clocks,
 427 $\sim \in \{<, \leq, =, \geq, >\}$, and $n \in \mathbb{Q}$. Such a zone describes a convex set of clock valuations.
 428 Zones may be efficiently represented using so-called Difference-bounded Matrices (DBM) [11].

429 ► **Proposition 17** ([23]). *There are zone-based algorithms for the following problems:*

430 ■ *Given a TBA, compute its nonempty language states.*

431 ■ *Given a TBA or TMA \mathcal{A} and an observation (ρ, t) , compute the reach set $\mathcal{T}_{\mathcal{A}}(\rho, t)$.*

432 ■ *Given a TBA or TMA \mathcal{A} and a finite union S of symbolic states, compute $\text{Pre}_{\mathcal{A}}^*(S)$.*

433 We continue by showing that the non-empty language states can also be computed for
 434 TMA, relying on an analysis of Alur and Dill's translation of TMA into equivalent TBA [3].
 435 Here, we just present the parts of their construction we need to prove our results.

► **Lemma 18.** *There is a zone-based algorithm that, given a TMA, computes its nonempty language states.*

Proof. Fix a TMA $\mathcal{A} = (Q, Q_0, \Sigma, \mathcal{X}, \Delta, \mathcal{F})$. The translation from TMA to TBA relies on the fact that $L(\mathcal{A}) = \bigcup_{F \in \mathcal{F}} L(\mathcal{A}_F)$, where \mathcal{A}_F is the TMA $(Q, Q_0, \Sigma, \mathcal{X}, \Delta, \{F\})$. Alur and Dill translated each such \mathcal{A}_F into an equivalent TBA \mathcal{A}'_F with set $Q \times \{0, 1, \dots, |F|\}$ of locations and the same clocks as \mathcal{A} . Then, the disjoint union of the \mathcal{A}'_F for $F \in \mathcal{F}$ is a TBA that is equivalent to \mathcal{A} .

The TBA \mathcal{A}'_F constructed by Alur and Dill satisfy the following fact: $L(\mathcal{A}'_F, ((q, 0), v)) = L(\mathcal{A}_F, (q, v))$ for all locations q of \mathcal{A} and all clock valuations. Thus, we have

$$S_{\mathcal{A}}^{ne} = \bigcup_{F \in \mathcal{F}} \{(q, v) \mid ((q, 0), v) \in S_{\mathcal{A}'_F}^{ne}\}.$$

Hence, we can symbolically compute the non-empty language states of \mathcal{A} by symbolically computing the non-empty states of the \mathcal{A}'_F and then project the states of the form $(q, 0)$ to q , but leaving the zones unchanged. ◀

Using this result, we can decide monitorability.

► **Theorem 19.** *Strong and weak (ρ, t) -monitorability are decidable for properties given by DTMA.*

Proof. We assume w.l.o.g. that \mathcal{A} is complete in the sense that every word has a run, which then must be unique due to determinism. This can always be achieved by adding a fresh sink location and by rerouting all missing transitions to it (while preserving determinism).

Given \mathcal{A} , we compute the non-empty language states $S_{\mathcal{A}}^{ne}$. The complement of the non-empty language states is the empty language states, i.e., the set of states for which there is no accepting run starting there. Let us denote this set as $S_{\mathcal{A}}^{\emptyset} = \overline{S_{\mathcal{A}}^{ne}}$. Using backwards reachability, we compute all the possibly-empty language states, i.e., states from which there is at least one non-accepting run, i.e., $S_{\mathcal{A}}^{pe} = Pre_{\mathcal{A}}^*(S_{\mathcal{A}}^{\emptyset})$. Note that $S_{\mathcal{A}}^{\emptyset} \subseteq S_{\mathcal{A}}^{pe}$. If a state s is not in $S_{\mathcal{A}}^{pe}$, then all states reachable from s have at least one accepting run.

The sets $S_{\mathcal{A}}^{ne}$, $S_{\mathcal{A}}^{\emptyset}$ and $S_{\mathcal{A}}^{pe}$ are illustrated in Fig. 5.



■ **Figure 5** Representation of all states of an automaton \mathcal{A} . On the left, only the empty languages states $S_{\mathcal{A}}^{\emptyset}$ and non-empty language states $S_{\mathcal{A}}^{ne}$ are shown. On the right, the possibly empty language states $S_{\mathcal{A}}^{pe}$ are added in gray. The arrows are examples of possible transitions and the dashed arrows are examples of impossible transitions. A state in $S_{\mathcal{A}}^{\emptyset}$ has no accepting run, thus cannot reach $S_{\mathcal{A}}^{ne}$. A state in $S_{\mathcal{A}}^{ne}$ might reach a state in $S_{\mathcal{A}}^{\emptyset}$. A state outside $S_{\mathcal{A}}^{pe}$ cannot reach $S_{\mathcal{A}}^{\emptyset}$.

Given an observation (ρ, t) , we can compute the reach-set of \mathcal{A} over (ρ, t) as $\mathcal{T}_{\mathcal{A}}(\rho, t)$, which is a singleton set, as \mathcal{A} is deterministic and complete. Thus, we identify $\mathcal{T}_{\mathcal{A}}(\rho, t)$ with the unique state in it.

Now, we have $\mathcal{T}_{\mathcal{A}}(\rho, t) \in S_{\mathcal{A}}^{pe}$ if and only if there exists $\rho' \in T\Sigma^*$ such that for all $\mu \in T\Sigma^\omega$ we have $\rho \cdot_t \rho' \cdot \mu \notin L(\mathcal{A})$. If the latter condition is violated, then monitoring any extension of (ρ, t) will not provide the verdict \perp , i.e., $V_{L(\mathcal{A})}(\rho' \cdot_t \rho') \neq \perp$ for all $\rho' \in T\Sigma^*$.

Since DTMA are complementable by changing the acceptance condition, but not the set of locations and clocks, we can compute $S_{\mathcal{A}}^{pe}$ for the complement automaton $\bar{\mathcal{A}}$ of \mathcal{A}

and provide the dual characterization for the \top -verdict: $\mathcal{T}_A(\rho, t) \in S_A^{pe}$ if and only if there exists $\rho' \in T\Sigma^*$ such that for all $\mu \in T\Sigma^\omega$ we have $\rho \cdot_t \rho' \cdot \mu \in L(A)$. Again, if the latter condition is violated, then monitoring any extension of (ρ, t) will not provide the verdict \top , i.e., $V_{L(A)}(\rho' \cdot_t \rho') \neq \top$ for all $\rho' \in T\Sigma^*$.

With these characterizations, we can conclude whether the \top or \perp verdict are still possible for some extensions, which allows us to decide weak (ρ, t) -monitorability: $L(A)$ is weakly (ρ, t) -monitorable if and only if $\mathcal{T}_A(\rho, t) \in S_A^{pe} \cup S_A^{pe}$. Note that this characterization crucially depends on the fact that both A and \bar{A} have the same states (locations and clock valuations).

For strong (ρ, t) -monitorability, we need to compute the states that can leave the possibly empty states via a finite run: A is strongly (ρ, t) -monitorable if and only if $\mathcal{T}_A(\rho, t) \notin Pre_A^*(S_A^{pe}) \cap Pre_A^*(\bar{S}_A^{pe})$.

Due to Lemma 17 and Lemma 18, both characterizations are effectively decidable. \blacktriangleleft

Due to Remark 12, we also obtain the decidability of strong and weak monitorability.

► **Corollary 20.** *Strong and weak monitorability are decidable for properties given by DTMA.*

5 Monitorability with Step-Bounded Horizons

Strong monitorability of a property ensures that at any time during monitoring, the observation (ρ, t) made so far can be extended by some finite ρ' such that a conclusive verdict can be made after $\rho \cdot_t \rho'$. Thus, in the setting of strong monitorability, it is always meaningful to keep monitoring. In contrast, for weakly monitorable properties, the ability to make a conclusive verdict after some future (finite) monitoring is not guaranteed. Ideally, we would like to refine the rather uninformative verdict $?$ with guaranteed minimum bounds on the number of future events before a positive or negative verdict can be made. In case both these bounds are ∞ , further monitoring is useless as the verdicts reported will always be $?$. Here it is prudent to distinguish between the two definitive verdicts, as they have different decidability properties.

► **Definition 21** (Weak Monitorability with Step-Bounded Horizons). *Fix a property $\varphi \subseteq T\Sigma^\omega$, an observation (ρ, t) , and a bound $n \in \mathbb{N}$.*

■ *φ is bounded weakly \top -(ρ, t)-monitorable with respect to n if and only if*

there exists $\rho' \in T\Sigma^{\leq n}$ such that $V_\varphi(\rho \cdot_t \rho') = \top$.

■ *φ is bounded weakly \perp -(ρ, t)-monitorable with respect to n if and only if*

there exists $\rho' \in T\Sigma^{\leq n}$ such that $V_\varphi(\rho \cdot_t \rho') = \perp$.

■ *φ is bounded weakly (ρ, t) -monitorable with respect to n if and only if*

there exists $\rho' \in T\Sigma^{\leq n}$ such that $V_\varphi(\rho \cdot_t \rho') \in \{\top, \perp\}$.

■ *φ is bounded weakly \top -monitorable with respect to n if it is bounded weakly \top -($\varepsilon, 0$)-monitorable with respect to n .*

■ *φ is bounded weakly \perp -monitorable with respect to n if it is bounded weakly \perp -($\varepsilon, 0$)-monitorable with respect to n .*

■ *φ is bounded weakly monitorable with respect to n if it is bounded weakly $(\varepsilon, 0)$ -monitorable with respect to n .*

In the following, we show that bounded weak monitorability and bounded weak \top -monitorability are undecidable, but that bounded weak \perp -monitorability is decidable.

► **Theorem 22.** *Bounded weak \top -monitorability is undecidable for properties given by TBA.*

Proof. Let φ be a property given by a TBA \mathcal{A} . Then universality of \mathcal{A} is equivalent to bounded weak \top -monitorability of φ for the bound $n = 0$. As universality for timed automata is undecidable [3], 0-bounded weak \top -monitorability for TBA properties is undecidable. ◀

Next, we show that weak \perp -(ρ, t)-monitorability behaves differently.

► **Theorem 23.** *Bounded weak \perp -(ρ, t)-monitorability is decidable for properties given by TBA.*

Proof. Let (ρ, t) be an observation with $\rho = (\sigma_1, t_1)(\sigma_2, t_2) \cdots (\sigma_m, t_m)$ and let the property be given by the TBA \mathcal{A} .

Let $\pi = q_0 \xrightarrow{\sigma_1, \lambda_1, g_1} q_1 \cdots q_{m+n-1} \xrightarrow{\sigma_{m+n}, \lambda_{m+n}, g_{m+n}} q_{m+n}$ be a syntactic path of length $m+n$ of the TBA \mathcal{A} induced by transitions $(q_i, q_{i+1}, \sigma_i, \lambda_i, g_i)$. Also, let $\tau = \{\tau_1, \dots, \tau_{m+n}\}$ be the set of variables ranging over global time (i.e., $\mathbb{R}_{\geq 0}$), where τ_i denotes the time at which the i -th transition is taken. Clearly, $\tau_i = t_i$ for $i = 1, \dots, m$. Most importantly, there is a zone $Z_\pi(\tau)$ over τ that precisely captures when (σ, τ) is a timed word realizing π . In this case, we may also express the resulting clock valuation after realizing (σ, τ) on π as $v_\pi(\tau) = (v_\pi^1, \dots, v_\pi^k)$, where k is the number of clocks of \mathcal{A} and $v_\pi^i = \tau_{m+n} - \tau_{\ell(i)}$, with $\ell(i)$ being the index of the last transition when the clock x_i was reset. Now, if $(q_{m+n}, v_\pi(\tau)) \notin S_{\mathcal{A}}^{ne}$, then the run of (σ, τ) following π cannot be extended to an accepting run of \mathcal{A} .

Witnesses for weak monitorability are closed under extensions. Hence, in the following construction, we can restrict ourselves w.l.o.g. to words of length exactly n instead of words of length at most n . The following formula expresses the existence of a timed word $\rho' = (\sigma', \tau')$ of length n such that all runs on $\rho \cdot_t \rho'$ end in a state outside of $S_{\mathcal{A}}^{ne}$, witnessing $V_{L(\mathcal{A})}(\rho \cdot_t \rho') = \perp$:

$$\exists \tau_1 \dots \exists \tau_{m+n}. \bigwedge_{i=1}^m \tau_i = t_i \wedge \tau_m \leq t \leq \tau_{m+1} \wedge \left(\bigvee_{\sigma_{m+1} \cdots \sigma_{m+n} \in \Sigma^n} \bigwedge_{\pi \in \text{SP}} (Z_\pi(\tau) \rightarrow v_\pi(\tau) \notin S_{\mathcal{A}}^{ne}) \right),$$

where SP is the set of syntactic paths of the form

$$\pi = q_0 \xrightarrow{g_1, \sigma_1, \lambda_1} q_1 \cdots q_m \xrightarrow{g_m, \sigma_m, \lambda_m} q_{m+1} \cdots q_{m+n-1} \xrightarrow{\sigma_{m+n}, \lambda_{m+n}, g_{m+n}} q_{m+n}.$$

The formula above expresses the existence of values for $\tau_1, \dots, \tau_{m+n}$, with $\tau_i = t_i$ for $i = 1, \dots, m$, and $\tau_m \leq t \leq \tau_{m+1}$, ensuring that only extensions of the observation (ρ, t) are considered. The last part of the formula ensures the existence of a timed word $\rho' = (\sigma', \tau')$, where $\sigma' = \sigma_{m+1} \cdots \sigma_{m+n}$ and $\tau' = \tau_{m+1} \cdots \tau_{m+n}$, where all runs – which are captured by $Z_\pi(\tau)$ – are outside $S_{\mathcal{A}}^{ne}$. The formula is in the first order theory of real-closed fields, which is decidable [33]. Thus, bounded weak \perp -monitorability is decidable for properties given by TBA. ◀

► **Remark 24.** It follows from Theorem 23 that for properties where both the property and its complement are given by TBA, bounded weak monitorability is decidable. Hence, if the property is given by a DTMA, then bounded weak monitorability is decidable. Moreover, one can even compute the tightest bound n by doing a breadth-first search over the symbolic state graph of the deterministic timed automaton, searching for a (shortest) path to the empty language states.

Lastly, we prove bounded weak monitorability undecidable for properties given by TBA.

► **Theorem 25.** *Bounded weak monitorability is undecidable for properties given by TBA.*

Proof. In the second step of the undecidability proof of weak monitorability (see Theorem 14), we have shown how to reduce weak \top -monitorability to weak monitorability. The same construction reduces bounded weak \top -monitorability to bounded weak monitorability, as the length of the witnesses is only decreased by removing $\#$'s. ◀

Due to Remark 12, we also obtain results for the remaining cases.

► **Corollary 26.** *Bounded weak \top -(ρ, t)-monitorability and bounded weak (ρ, t)-monitorability are undecidable (even for fixed (ρ, t)) while bounded weak \perp -monitorability is decidable.*

6 Refined Monitoring with Time-Horizon Verdicts

In the previous section, we have shown that the uninformative verdict $?$ can be refined by checking whether within a bounded number of new observations, a definitive verdict may be given. In this section, we again refine the uninformative verdict $?$ by computing lower bounds on the time that needs to pass (independently of the number of events observed during that time) before a definitive verdict may be given. Again, if this is infinite, then the monitoring process can be stopped, as no amount of waiting will yield a definitive verdict. This refinement was introduced and briefly studied as “time predictive monitoring” by Grosen et al. [23]. Here we revisit (and rename) this notion by proving computability of the time-bounded monitorability for properties given by a DTMA.

Recall that time passing without any new observed events can nevertheless yield definitive verdicts (see, e.g., the last two items in Example 7). Thus, in a practical setting, one is interested in intermittently querying the monitoring function even if no events are observed. Here, we give lower bounds on the time one should let pass before the next such query is made, thereby reducing the computational overhead of these queries.

► **Definition 27** (Refined Monitoring with Time-Horizon Verdicts). *Given an observation (ρ, t) and a property $\varphi \subseteq T\Sigma^\omega$, the refined monitoring function P_φ is defined as*

$$P_\varphi(\rho, t) = \begin{cases} \top & \text{if } V_\varphi(\rho, t) = \top, \\ \perp & \text{if } V_\varphi(\rho, t) = \perp, \\ (P_\varphi^\top(\rho, t), P_\varphi^\perp(\rho, t)) & \text{otherwise,} \end{cases}$$

with

- $P_\varphi^\top(\rho, t) = \inf \{t' \mid \rho' \in T\Sigma^* \text{ such that } V_\varphi(\rho \cdot_t \rho', t') = \top \text{ and } t' \geq \tau(\rho \cdot_t \rho')\}$ and
 - $P_\varphi^\perp(\rho, t) = \inf \{t' \mid \rho' \in T\Sigma^* \text{ such that } V_\varphi(\rho \cdot_t \rho', t') = \perp \text{ and } t' \geq \tau(\rho \cdot_t \rho')\},$
- where we use the convention $\inf \emptyset = \infty$.

► **Example 28.** Consider the MITL property $\varphi = F_{[20,40]}b$. Monitoring the finite timed word $\rho = (a, 5.1)(c, 21.0)(c, 30.4)(b, 35.1)(a, 40.2)$ will result in three $?$ verdicts followed by the verdict \top when $(b, 35.1)$ is read. However, we may offer significantly more information, e.g., when reading $(a, 5.1)$ it is clear that at least 14.9 time-units must elapse before we can give the verdict \top , and at least 34.9 time-units must elapse before we can give the verdict \perp . Hence, $P_\varphi((a, 5.1), 5.1) = (14.9, 34.9)$.

► Remark 29. $V_\varphi(\rho, t) = \top$ implies $P_\varphi^\top(\rho, t) = 0$ and $P_\varphi^\perp(\rho, t) = \infty$ and $V_\varphi(\rho, t) = \perp$ implies $P_\varphi^\top(\rho, t) = \infty$ and $P_\varphi^\perp(\rho, t) = 0$, but the converse is in general not true. Consider, for example, the MITL property $\varphi = F_{\geq 0}a$ and the observation $\rho = (b, 1)$. Then, $P_\varphi^\top(\rho, 1) = 0$ (witnessed by $\rho' = (a, 0)$ for which we have $V_{\rho \cdot_t \rho'} = \top$) and $P_\varphi^\perp = \infty$, but $V_\varphi(\rho, 1) = ?$.

Time-bounded monitorability for an observation (ρ, t) refines weak (ρ, t) -monitorability.

► Lemma 30. Let (ρ, t) be an observation and φ a property. Then, φ is weakly (ρ, t) -monitorable if and only if at least one of the values $P_\varphi^\top(\rho, t)$ and $P_\varphi^\perp(\rho, t)$ is finite.

Proof. Let φ be weakly (ρ, t) -monitorable, i.e., there is a $\rho' \in T\Sigma^*$ such that $V_\varphi(\rho \cdot_t \rho') \in \{\top, \perp\}$, say it is \top (the other case is analogous). Then, ρ' witnesses $P_\varphi^\top(\rho, t) \leq \tau(\rho')$, i.e., $P_\varphi^\top(\rho, t)$ is finite. On the other hand, if (say) $P_\varphi^\top(\rho, t)$ is finite (the other case is analogous), then there is a ρ' such that $V_\varphi(\rho \cdot_t \rho') = \top$. Hence, φ is weakly (ρ, t) -monitorable. ◀

► Theorem 31 (Refined Monitoring is Effective). P_φ is effectively computable, if φ is given by a DTMA.

Proof. When online-monitoring φ over some observation (ρ, t) , we compute the reach-set $\mathcal{T}_\mathcal{A}(\rho, t)$ and check if it is a subset of the non-empty language states $S_\mathcal{A}^{ne}$ or a subset of the empty language states $S_\mathcal{A}^\emptyset$ [23]. Thus, $P_\varphi^\perp(\rho, t)$ is the infimum of the time duration of all paths from $\mathcal{T}_\mathcal{A}(\rho, t)$ to $S_\mathcal{A}^\emptyset$. This is a time-optimal reachability problem.

Asarin and Maler [7] introduced the concept of time-optimal strategies for timed game automata. Since timed game automata trivially generalize timed automata, we can adapt the computation of the optimal time bound to obtain the minimal possible time to reach an empty language state from the reach-set. This gives us the value $P_\mathcal{A}^\perp(\rho, t)$.

Since DTMA are closed under complement, the same procedure for the complement automaton gives us the value $P_\mathcal{A}^\top(\rho, t)$. ◀

As explained in the proof of Theorem 31, monitoring is implemented by keeping track of the reach-set of the observation and checking at each update whether it is contained in the non-empty language states or in the empty language states. But, as explained in the introduction of this section, one should not only update the reach-set when a new event is observed, but also intermittently when time has passed.

For this special case where only time passes, which is covered in Definition 27 by considering $\rho' = \varepsilon$ in the infimum, we do not need to solve the expensive time-optimal reachability problem described in the proof of Theorem 31. Instead, we rely on zone operations to compute $\delta_\varphi(\rho, t) = \inf\{d \mid V_\varphi(\rho, t + d) \in \{\top, \perp\}\}$. This is done by exploring all delays (removing the upper bounds of the zones) and subtracting the non-empty language states. The lower bounds in the zones of the resulting states gives the minimum time a verdict can be made by waiting. Now, having observed ρ and the current time being t such that $V_\varphi(\rho, t) = ?$, querying the monitoring function again without a new observation before time $t + \delta_\varphi(\rho, t)$ will not yield a different verdict and can thus be avoided.

7 Related Work

A formal notion of monitorability was first introduced by Pnueli and Zaks in their work on monitoring Property Specification Language (PSL) [29]. In that work, the authors defined strong monitorability given a finite prefix, called σ -monitorability, on which we base our (ρ, t) -monitorability. From this, Bauer, Leuker, and Schallhart defined the most common definition of monitorability, that is strong σ -monitorability from the initial state [10]. They proved

that safety and guarantee properties are a proper subset of the class of strongly monitorable properties. Later, Chen et al. [13] and Peled and Havelund [28] noticed that there exist properties that are not strongly monitorable but still have utility to monitor, and proposed equivalent definitions of weak monitorability. Mascle et al. showed the monitorability of LTL properties can be improved by considering robust semantics [27]. The term *strong monitorability* has been used before in the context of partially observable stochastic systems modeled as Hidden Markov Models. Sistla, Žefran, and Feng first used the term, contrasted with (standard) monitorability [32] in their work extending the results of Gondi, Patel, and Sistla on monitoring ω -regular properties of stochastic systems [21].

The complexity of monitorability problems has been studied in other untimed settings. Diekert and Leucker proposed a topological definition of strong monitorability, showing that problem is equivalent to showing that the boundary in the Cantor topology has an empty interior [16]. Diekert, Muscholl, and Walukiewicz later proved that deciding monitorability for (untimed) Büchi automata is PSPACE-complete [18]. Agrawal and Bonakdarpour proposed a definition of monitorability for hyperproperties and determined the monitorable classes for their three-valued specification language, HyperLTL [15]. Francalanza, Aceto, and Ingólfssdóttir characterized monitorable properties of the branching-time μ -Hennessy-Milner Logic [19]. This was later extended by Aceto et al. who introduced a hierarchy of monitorable fragments of the language [1].

Attempts have been made to unify these different notions of monitorability. Peled and Havelund introduced a classification for properties centered around monitorability [28]. Kauffman, Havelund, and Fischmeister defined a common notation for strong and weak monitorability for different verdict domains [24]. Aceto, Achilleos, and Francalanza provided syntactic characterizations of monitorability for classical notions of monitorability as well as for a variant of the modal μ -calculus, recHML [2].

We are aware of only one other work addressing monitorability for real-time properties. Amara et al. (very recently) introduced a new linear-time timed μ -calculus, which subsumes MTL, and therefore also MITL, and identified its largest monitorable fragment [6]. Their work differs from ours in three important respects. First, they rely on a definition of monitorability introduced by Schneider as *Execution Monitoring Enforceability* [31] while we use the more typical definition of strong and weak monitorability due to Pnueli and Zaks [29]. Second, they introduce a new calculus and characterize its maximal monitorable fragment. We instead consider the case of properties expressed as Timed Automata, which lend themselves to algorithmic manipulation. Finally, we prove decidability and undecidability results.

8 Conclusion

In this work, we have studied monitorability for timed properties specified by either (possibly nondeterministic) TBA or by deterministic TMA. In general, we proved monitorability decidable for specifications given by deterministic automata and undecidable for specifications given by nondeterministic automata. The notable exception here is bounded weak \perp -monitorability, which is even decidable for nondeterministic TBA.

Also, we provided refinements of monitoring and monitorability making the verdict $?$ more informative by providing bounds on the number of events or the amount of time that needs to pass before a conclusive verdict may be given. In practical settings, this is crucial information and also allows to optimize the monitoring process in the real-time setting.

Our decidability proof for monitorability of DTMA relies on the fact that DTMA can be complemented without changing the state space. On the other hand, monitorability is

undecidable if the property is given by a TBA. Thus, another question for further research is to consider strong monitorability when given TBA for the property *and* for its negation. This is a very natural setting, as the specification logic MITL is closed under negation and can be translated into TBA. Also, monitoring often requires TBA for both the property and its negation [23, 20, 14].

Acknowledgements. We would like to thank Corto Mascle, who brought Rampersad et al.'s work on the complexity of suffix-universality [30] to our attention, which inspired our undecidability proof for weak monitorability.

References

- 1 Luca Aceto, Antonis Achilleos, Adrian Francalanza, Anna Ingólfssdóttir, and Karoliina Lehtinen. Adventures in monitorability: From branching to linear time and back again. In *Symposium on Principles of Programming Languages (POPL'19)*, volume 3. ACM, January 2019. doi:10.1145/3290365.
- 2 Luca Aceto, Antonis Achilleos, Adrian Francalanza, Anna Ingólfssdóttir, and Karoliina Lehtinen. An operational guide to monitorability with applications to regular properties. *Software and Systems Modeling*, 20(2):335–361, April 2021. doi:10.1007/s10270-020-00860-z.
- 3 Rajeev Alur and David L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2):183–235, 1994. doi:10.1016/0304-3975(94)90010-8.
- 4 Rajeev Alur, Tomás Feder, and Thomas A. Henzinger. The benefits of relaxing punctuality. *J. ACM*, 43(1):116–146, 1996. doi:10.1145/227595.227602.
- 5 Rajeev Alur and P. Madhusudan. Decision problems for timed automata: A survey. In *Formal Methods for the Design of Real-Time Systems: International School on Formal Methods for the Design of Computer, Communication, and Software Systems*, volume 3185 of *LNCS*, pages 1–24. Springer, 2004. doi:10.1007/978-3-540-30080-9_1.
- 6 Mouloud Amara, Giovanni Bernardi, Mohammed Foughali, and Adrian Francalanza. A theory of (linear-time) timed monitors. In *39th European Conference on Object-Oriented Programming (ECOOP 2025)*, volume 333, Bergen (NO), Norway, June 2025. URL: <https://hal.science/hal-05043055>.
- 7 Eugene Asarin and Oded Maler. As soon as possible: Time optimal control for timed automata. In Frits W. Vaandrager and Jan H. van Schuppen, editors, *Hybrid Systems: Computation and Control, Second International Workshop, HSCC'99, Berg en Dal, The Netherlands, March 29-31, 1999, Proceedings*, volume 1569 of *LNCS*, pages 19–30. Springer, 1999. doi:10.1007/3-540-48983-5_6.
- 8 David Basin, Felix Klaedtke, and Eugen Zălinescu. Algorithms for monitoring real-time properties. In *Runtime Verification*, pages 260–275. Springer, 2012. doi:10.1007/978-3-642-29860-8_20.
- 9 Andreas Bauer, Martin Leucker, and Christian Schallhart. Monitoring of real-time properties. In *Foundations of Software Technology and Theoretical Computer Science*, pages 260–272, Berlin, Heidelberg, 2006. Springer. doi:10.1007/11944836_25.
- 10 Andreas Bauer, Martin Leucker, and Christian Schallhart. Runtime verification for LTL and TLTL. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 20(4):14:1–14:64, 9 2011. doi:10.1145/2000799.2000800.
- 11 Johan Bengtsson and Wang Yi. Timed automata: Semantics, algorithms and tools. In Jörg Desel, Wolfgang Reisig, and Grzegorz Rozenberg, editors, *Lectures on Concurrency and Petri Nets, Advances in Petri Nets [This tutorial volume originates from the 4th Advanced Course on Petri Nets, ACPN 2003, held in Eichstätt, Germany in September 2003. In addition to lectures given at ACPN 2003, additional chapters have been commissioned]*, volume 3098 of *Lecture Notes in Computer Science*, pages 87–124. Springer, 2003. doi:10.1007/978-3-540-27755-2_3.
- 12 Thomas Brihaye, Gilles Geeraerts, Hsi-Ming Ho, and Benjamin Monmege. MightyL: A compositional translation from MITL to timed automata. In Rupak Majumdar and Viktor

- Kuncak, editors, *CAV 2017, Part I*, volume 10426 of *LNCS*, pages 421–440, Cham, 2017. Springer. doi:10.1007/978-3-319-63387-9_21.
- 13 Zhe Chen, Yifan Wu, Ou Wei, and Bin Sheng. Deciding weak monitorability for runtime verification. In *Int. Conference on Software Engineering (ICSE’18)*, pages 163–164. ACM, 2018. doi:10.1145/3183440.3195077.
- 14 Alessandro Cimatti, Thomas Møller Grosen, Kim G. Larsen, Stefano Tonetta, and Martin Zimmermann. Exploiting assumptions for effective monitoring of real-time properties under partial observability. In Alexandre Madeira and Alexander Knapp, editors, *SEFM*, volume 15280 of *LNCS*, pages 70–88. Springer, 2024. doi:10.1007/978-3-031-77382-2_5.
- 15 Michael R. Clarkson, Bernd Finkbeiner, Masoud Kolehini, Kristopher K. Micinski, Markus N. Rabe, and César Sánchez. Temporal logics for hyperproperties. In *Int. Conference on Principles of Security and Trust (POST’14)*, volume 8414 of *LNCS*, pages 265–284. Springer, 2014. doi:10.1007/978-3-642-54792-8_15.
- 16 Volker Diekert and Martin Leucker. Topology, monitorable properties and runtime verification. *Theoretical Computer Science*, 537:29–41, 2014. doi:10.1016/j.tcs.2014.02.052.
- 17 Volker Diekert and Anca Muscholl. On distributed monitoring of asynchronous systems. In *Logic, Language, Information and Computation*, volume 7456 of *LNTCS*, pages 70–84. Springer, 2012.
- 18 Volker Diekert, Anca Muscholl, and Igor Walukiewicz. A note on monitors and Büchi automata. In *Theoretical Aspects of Computing - ICTAC 2015*, volume 9399 of *LNTCS*, pages 39–57. Springer, 2015. doi:10.1007/978-3-319-25150-9_3.
- 19 Adrian Francalanza, Luca Aceto, and Anna Ingolfsson. Monitorability for the Hennessy-Milner logic with recursion. *Formal Methods in System Design*, 51(1):87–116, August 2017. doi:10.1007/s10703-017-0273-z.
- 20 Martin Fränzle, Thomas Møller Grosen, Kim G. Larsen, and Martin Zimmermann. Monitoring real-time systems under parametric delay. In Nikolai Kosmatov and Laura Kovács, editors, *IFM 2024*, volume 15234 of *LNCS*, pages 194–213. Springer, 2024. doi:10.1007/978-3-031-76554-4_11.
- 21 Kalpana Gondi, Yogeshkumar Patel, and A. Prasad Sistla. Monitoring the full range of ω -regular properties of stochastic systems. In Neil D. Jones and Markus Müller-Olm, editors, *Verification, Model Checking, and Abstract Interpretation*, pages 105–119, Berlin, Heidelberg, 2009. Springer. doi:10.1007/978-3-540-93900-9_12.
- 22 Thomas Møller Grosen, Sean Kauffman, Kim G. Larsen, and Martin Zimmermann. Time for timed monitorability. *arXiv*, 2504.10008, 2025. arXiv:2504.10008, doi:10.48550/ARXIV.2504.10008.
- 23 Thomas Møller Grosen, Sean Kauffman, Kim Guldstrand Larsen, and Martin Zimmermann. Monitoring timed properties (revisited). In Sergiy Bogomolov and David Parker, editors, *FORMATS 2022*, volume 13465 of *LNCS*, pages 43–62. Springer, 2022. doi:10.1007/978-3-031-15839-1_3.
- 24 Sean Kauffman, Klaus Havelund, and Sebastian Fischmeister. What can we monitor over unreliable channels? *International Journal on Software Tools for Technology Transfer*, 23:579–600, 06 2021. doi:10.1007/s10009-021-00625-z.
- 25 Kim Guldstrand Larsen, Marius Mikucionis, and Brian Nielsen. Online testing of real-time systems using uppaal. In *Formal Approaches to Software Testing*, volume 3395 of *LNCS*, pages 79–94. Springer, 2004. doi:10.1007/978-3-540-31848-4_6.
- 26 N.G. Leveson and C.S. Turner. An investigation of the therac-25 accidents. *Computer*, 26(7):18–41, 1993. doi:10.1109/MC.1993.274940.
- 27 Corto Mascle, Daniel Neider, Maximilian Schwenger, Paulo Tabuada, Alexander Weinert, and Martin Zimmermann. From LTL to rltl monitoring: improved monitorability through robust semantics. *Formal Methods Syst. Des.*, 59(1):170–204, 2021. URL: <https://doi.org/10.1007/s10703-022-00398-4>, doi:10.1007/s10703-022-00398-4.

- 28 Doron Peled and Klaus Havelund. Refining the safety–liveness classification of temporal properties according to monitorability. In *Models, Mindsets, Meta: The What, the How, and the Why Not? Essays Dedicated to Bernhard Steffen on the Occasion of His 60th Birthday*, volume 11200 of *LNCS*, pages 218–234. Springer, 2019. doi:10.1007/978-3-030-22348-9_14.
- 29 A. Pnueli and A. Zaks. PSL model checking and run-time verification via testers. In Jayadev Misra, Tobias Nipkow, and Emil Sekerinski, editors, *FM 2006: Formal Methods*, pages 573–586, Berlin, Heidelberg, 2006. Springer. doi:10.1007/11813040_38.
- 30 Narad Rampersad, Jeffrey O. Shallit, and Zhi Xu. The computational complexity of universality problems for prefixes, suffixes, factors, and subwords of regular languages. *Fundam. Informaticae*, 116(1-4):223–236, 2012. doi:10.3233/FI-2012-680.
- 31 Fred B. Schneider. Enforceable security policies. *ACM Trans. Inf. Syst. Secur.*, 3(1):30–50, February 2000. doi:10.1145/353323.353382.
- 32 A. Prasad Sistla, Miloš Žefran, and Yao Feng. Monitorability of stochastic dynamical systems. In Ganesh Gopalakrishnan and Shaz Qadeer, editors, *Computer Aided Verification*, pages 720–736, Berlin, Heidelberg, 2011. Springer.
- 33 Alfred Tarski. A decision method for elementary algebra and geometry. In Bob F. Caviness and Jeremy R. Johnson, editors, *Quantifier Elimination and Cylindrical Algebraic Decomposition*, pages 24–84, Vienna, 1998. Springer Vienna.
- 34 Prasanna Thati and Grigore Roşu. Monitoring algorithms for metric temporal logic specifications. *Electronic Notes in Theoretical Computer Science*, 113:145–162, 2005. doi:10.1016/j.entcs.2004.01.029.