Foundations of Software Technology and Theoretical Computer Science (Bangalore) 2008.
Editors: R. Hariharan, M. Mukund, V. Vinay; pp -

Runtime Monitoring of Metric First-order Temporal Properties

David Basin[1], Felix Klaedtke[1], Samuel Müller[2], Birgit Pfitzmann[3]

[1]ETH Zurich, Switzerland
{basin,felixkl}@inf.ethz.ch

[2]IBM Zurich Research Lab and ETH Zurich, Switzerland
sml@zurich.ibm.com

[3]IBM Watson Research Lab, USA
bpfitzm@us.ibm.com

ABSTRACT. We introduce a novel approach to the runtime monitoring of complex system properties. In particular, we present an online algorithm for a safety fragment of metric first-order temporal logic that is considerably more expressive than the logics supported by prior monitoring methods. Our approach, based on automatic structures, allows the unrestricted use of negation, universal and existential quantification over infinite domains, and the arbitrary nesting of both past and bounded future operators. Moreover, we show how to optimize our approach for the common case where structures consist of only finite relations, over possibly infinite domains. Under an additional restriction, we prove that the space consumed by our monitor is polynomially bounded by the cardinality of the data appearing in the processed prefix of the temporal structure being monitored.

1 Introduction

Runtime monitoring [1] is an approach to verifying system properties at execution time by using an online algorithm to check whether a system trace satisfies a temporal property. While novel application areas such as compliance or business activity monitoring [13, 15] require expressive property specification languages, current monitoring techniques are restricted in the properties they can handle. They either support properties expressed in propositional temporal logics and thus cannot cope with variables ranging over infinite domains [6,16,20, 23,29], do not provide both universal and existential quantification [4,12,17,23–25] or only in restricted ways [4,28,30], do not allow arbitrary quantifier alternation [4,22], cannot handle unrestricted negation [8,22,27,30], do not provide quantitative temporal operators [22,25], or cannot simultaneously handle past and future temporal operators [8,22–24,26,27].

In this paper, we present a runtime monitoring approach for an expressive safety fragment of metric first-order temporal logic (MFOTL) [8] that overcomes most of these limitations. The fragment consists of formulae of the form $\Box\,\phi$, where ϕ is bounded, i.e., its temporal operators refer only finitely into the future. Our monitor uses automatic structures [7] to finitely represent infinite structures, which allows for the unrestricted use of negation and quantification in monitored formulae. Moreover, our monitor supports the arbitrary nesting of both (metric) past and bounded future operators. This means that complex properties can be specified more naturally than with only past operators.[1]

In a nutshell, our monitor works as follows: Given a MFOTL formula $\Box\phi$ over a signature S, where ϕ is bounded, we first transform ϕ into a first-order formula $\hat{\phi}$ over an extended signature $\hat{S}$, obtained by augmenting S with auxiliary predicates for every temporal subformula in ϕ. Our monitor then incrementally processes a temporal structure (D, τ) over S and determines for each time point i those elements in (D, τ) that violate ϕ. This is achieved by incrementally constructing a collection of automata that finitely represent the (possibly infinite) interpretations of the auxiliary predicates and by evaluating the transformed first-order formula $\neg\hat{\phi}$ over the extended $\hat{S}$-structure at every time point. In doing so, our monitor discards any information not required for evaluating $\neg\hat{\phi}$ at the current and future time points.

We also show how to adapt our monitoring approach to the common case where all relations are required to be finite and hence relational databases can serve as an alternative to automata. Under the additional (realistic) restriction that time increases after at most a fixed number of time points, our incremental construction ensures that our monitor requires only polynomial space in the cardinality of the data appearing in the processed prefix of the monitored temporal structure. This is in contrast to complexity results for other approaches, such as the logical data expiration technique proposed for 2-FOL [30]. While this logic is at least as expressive as MFOTL, the space required for monitoring (syntactically-restricted) 2-FOL formulae is non-elementary in the cardinality of the data in the processed prefix.

Overall, we see our contributions as follows. First, the presented monitor admits a substantially more expressive logic than previous monitoring approaches. In particular, by supporting arbitrary bounded MFOTL formulae, it significantly extends Chomicki's dynamic integrity checking approach for temporal databases [8]. Second, we extend runtime monitoring to automatic structures, which allows for the unrestricted use of negation and quantification in monitored formulae. Third, for the restricted setting where all relations are finite, we show how to implement our monitor using relational databases. Here, we extend the rewrite procedure of [11] to handle a larger class of temporal formulae. We then prove that, under an additional restriction, the space consumed by our monitor is polynomially bounded in the cardinality of the data appearing in the processed prefix of a monitored temporal structure. Finally, our work shows how to effectively combine ideas from different, but related areas, including database theory, runtime monitoring, model checking, and model theory.

This paper is an extended abstract. Full details are presented in [5].

2 Metric First-order Temporal Logic

In this section, we introduce metric first-order temporal logic (MFOTL) [8], which extends propositional metric temporal logic [19] in a standard way. In the forthcoming sections, we present a method for monitoring requirements formalized within MFOTL.

[1]It is unknown whether the past-only fragment of MFOTL is as expressive as the fragment with both past and bounded future operators and whether formulae in the past-only fragment can be expressed as succinctly as those in the future-bounded fragment.

Syntax and Semantics. Let $\mathbb{I}$ be the set of nonempty intervals over $\mathbb{N}$. We often write an interval in $\mathbb{I}$ as $[c, d)$, where $c \in \mathbb{N}$, $d \in \mathbb{N} \cup \{\infty\}$, and $c < d$, i.e., $[c, d) := \{a \in \mathbb{N} \mid c \leq a < d\}$. A *signature* S is a tuple $(\mathsf{C}, \mathsf{R}, a)$, where C is a finite set of constant symbols, R is a finite set of predicates disjoint from C, and the function $a : \mathsf{R} \to \mathbb{N}$ associates each predicate $r \in \mathsf{R}$ with an arity $a(r) \in \mathbb{N}$. For the rest of this paper, V denotes a countably infinite set of variables, where we assume that $\mathsf{V} \cap (\mathsf{C} \cup \mathsf{R}) = \emptyset$, for every signature $S = (\mathsf{C}, \mathsf{R}, a)$. In the following, let $S = (\mathsf{C}, \mathsf{R}, a)$ be a signature.

DEFINITION 1. *The* formulae *over S are inductively defined: (i) For $t, t' \in \mathsf{V} \cup \mathsf{C}$, $t \approx t'$ and $t \prec t'$ are formulae. (ii) For $r \in \mathsf{R}$ and $t_1, \dots, t_{a(r)} \in \mathsf{V} \cup \mathsf{C}$, $r(t_1, \dots, t_{a(r)})$ is a formula. (iii) For $x \in \mathsf{V}$, if θ and θ' are formulae then $(\neg\theta)$, $(\theta \wedge \theta')$, and $(\exists x.\, \theta)$ are formulae. (iv) For $I \in \mathbb{I}$, if θ and θ' are formulae then $(\bullet_I\, \theta)$, $(\bigcirc_I\, \theta)$, $(\theta\, \mathcal{S}_I\, \theta')$, and $(\theta\, \mathcal{U}_I\, \theta')$ are formulae.*

To define the semantics of MFOTL, we need the following notions: A *(first-order) structure D* over S consists of a domain $|D| \neq \emptyset$ and interpretations $c^D \in |D|$ and $r^D \subseteq |D|^{a(r)}$, for each $c \in \mathsf{C}$ and $r \in \mathsf{R}$. A *temporal (first-order) structure* over S is a pair (D, τ), where $D = (D_0, D_1, \dots)$ is a sequence of structures over S and $\tau = (\tau_0, \tau_1, \dots)$ is a sequence of natural numbers (time stamps), where:

1. The sequence τ is monotonically increasing (i.e., $\tau_i \leq \tau_{i+1}$, for all $i \geq 0$) and makes progress (i.e., for every $i \geq 0$, there is some $j > i$ such that $\tau_j > \tau_i$).
2. D has constant domains, i.e., $|D_i| = |D_{i+1}|$, for all $i \geq 0$. We denote the domain by $|D|$ and require that $|D|$ is linearly ordered by the relation $<$.
3. Each constant symbol $c \in \mathsf{C}$ has a rigid interpretation, i.e., $c^{D_i} = c^{D_{i+1}}$, for all $i \geq 0$. We denote the interpretation of c by c^D.

A *valuation* is a mapping $v : \mathsf{V} \to |D|$. We abuse notation by applying a valuation v also to constant symbols $c \in \mathsf{C}$, with $v(c) = c^D$. For a valuation v, a variable vector $\bar{x} = (x_1, \dots, x_n)$, and $\bar{d} = (d_1, \dots, d_n) \in |D|^n$, $v[\bar{x}/\bar{d}]$ is the valuation that maps x_i to d_i, for i such that $1 \leq i \leq n$, and the valuation of the other variables is unaltered.

DEFINITION 2. *Let (D, τ) be a temporal structure over S, with $D = (D_0, D_1, \dots)$ and $\tau = (\tau_0, \tau_1, \dots)$, θ a formula over S, v a valuation, and $i \in \mathbb{N}$. We define $(D, \tau, v, i) \models \theta$ as follows:*

$(D, \tau, v, i) \models t \approx t'$	*iff*	$v(t) = v(t')$
$(D, \tau, v, i) \models t \prec t'$	*iff*	$v(t) < v(t')$
$(D, \tau, v, i) \models r(t_1, \dots, t_{a(r)})$	*iff*	$(v(t_1), \dots, v(t_{a(r)})) \in r^{D_i}$
$(D, \tau, v, i) \models (\neg\theta_1)$	*iff*	$(D, \tau, v, i) \not\models \theta_1$
$(D, \tau, v, i) \models (\theta_1 \wedge \theta_2)$	*iff*	$(D, \tau, v, i) \models \theta_1$ *and* $(D, \tau, v, i) \models \theta_2$
$(D, \tau, v, i) \models (\exists x.\, \theta_1)$	*iff*	$(D, \tau, v[x/d], i) \models \theta_1$, *for some* $d \in \lvert D\rvert$
$(D, \tau, v, i) \models (\bullet_I\, \theta_1)$	*iff*	$i > 0$, $\tau_i - \tau_{i-1} \in I$, *and* $(D, \tau, v, i-1) \models \theta_1$
$(D, \tau, v, i) \models (\bigcirc_I\, \theta_1)$	*iff*	$\tau_{i+1} - \tau_i \in I$ *and* $(D, \tau, v, i+1) \models \theta_1$
$(D, \tau, v, i) \models (\theta_1\, \mathcal{S}_I\, \theta_2)$	*iff*	*for some* $j \leq i$, $\tau_i - \tau_j \in I$, $(D, \tau, v, j) \models \theta_2$, *and* $(D, \tau, v, k) \models \theta_1$, *for all* $k \in [j+1, i+1)$
$(D, \tau, v, i) \models (\theta_1\, \mathcal{U}_I\, \theta_2)$	*iff*	*for some* $j \geq i$, $\tau_j - \tau_i \in I$, $(D, \tau, v, j) \models \theta_2$, *and* $(D, \tau, v, k) \models \theta_1$, *for all* $k \in [i, j)$

Note that the temporal operators are augmented with lower and upper bounds. A temporal formula is only satisfied if it is satisfied within the bounds given by the temporal operator, which are relative to the current time stamp τ_i.

Terminology and Notation. We use standard syntactic sugar such as the standard conventions concerning the binding strength of operators to omit parentheses (e.g., temporal operators bind weaker than Boolean connectives and quantifiers) and we use standard temporal operators (e.g., $\blacklozenge_I \theta := \mathit{true}\ \mathcal{S}_I\ \theta$, where *true* abbreviates $\exists x.\, x \approx x$). Note that the non-metric variants of the temporal operators are easily defined (e.g., $\Box\,\theta := \Box_{[0,\infty)}\,\theta$).

We call formulae of the form $t \approx t'$, $t \prec t'$, and $r(t_1,\dots,t_{a(r)})$ *atomic*, and formulae with no temporal operators *first-order*. The outermost connective (i.e., Boolean connective, quantifier, or temporal operator) occurring in a formula θ is called the *main connective* of θ. A formula that has a temporal operator as its main connective is a *temporal* formula. A formula θ is *bounded* if the interval I of every temporal operator $\mathcal{U}_I$ occurring in θ is finite.

MFOTL denotes the set of MFOTL formulae and FOL the set of first-order formulae. For $\theta \in \mathsf{MFOTL}$, we define its immediate temporal subformulae $\mathit{tsub}(\theta)$ to be: (i) $\mathit{tsub}(\alpha)$ if $\theta = \neg\alpha$ or $\theta = \exists x.\,\alpha$; (ii) $\mathit{tsub}(\alpha) \cup \mathit{tsub}(\beta)$ if $\theta = \alpha \wedge \beta$; (iii) $\{\theta\}$ if θ is a temporal formula; and (iv) $\emptyset$ otherwise. E.g., for $\theta := (\bullet\,\alpha) \wedge ((\circ\,\beta)\ \mathcal{S}_{[1,9)}\ \gamma)$, we have that $\mathit{tsub}(\theta) = \{\bullet\,\alpha, (\circ\,\beta)\ \mathcal{S}_{[1,9)}\ \gamma\}$.

If $\theta \in \mathsf{MFOTL}$ has the free variables given by the vector $\bar{x} = (x_1,\dots,x_n)$, we define the set of satisfying assignments at time instance i as

$$\theta^{(D,\tau,i)} := \left\{\bar{d} \in |D|^n \mid (D,\tau,v[\bar{x}/\bar{d}],i) \models \theta,\ \text{for some valuation } v\right\}.$$

For $\theta \in \mathsf{FOL}$, we write $(D_i,v) \models \theta$ instead of $(D,\tau,v,i) \models \theta$ and θ^{D_i} for $\theta^{(D,\tau,i)}$. Note that $(D_i,v) \models \theta$ agrees with the standard definition of satisfaction in first-order logic.

3 Monitoring by Reduction to First-order Queries

To effectively monitor MFOTL formulae, we restrict both the formulae and the temporal structures under consideration. We discuss these restrictions in §3.1 and describe monitoring in §3.2–§3.5.

3.1 Restrictions

Throughout this section, let (D,τ) be a temporal structure over the signature $S = (\mathsf{C},\mathsf{R},a)$ and ψ the formula to be monitored. We make the following restrictions on ψ and D. First, we require ψ to be of the form $\Box\,\phi$, where ϕ is bounded. It follows that ψ describes a safety property [3]. Note though that not all safety properties can be expressed by formulae of this form [9]. This is in contrast to propositional linear temporal logic, where every safety property can be expressed as $\Box\,\beta$, where β contains only past-time operators [21].

Second, we require that each structure in D is automatic [18]. Roughly speaking, this means that each structure in D can be finitely represented by a collection of automata over finite words. Let us briefly recall some background on automatic structures [7, 18]. Let Σ be an alphabet and # a symbol not in Σ. The *convolution* of the words $w_1,\dots,w_k \in \Sigma^*$ with $w_i = w_{i1}\cdots w_{i\ell_i}$ is the word

$$w_1 \otimes \cdots \otimes w_k := \begin{bmatrix} w'_{11} \\ \vdots \\ w'_{k1} \end{bmatrix} \cdots \begin{bmatrix} w'_{1\ell} \\ \vdots \\ w'_{k\ell} \end{bmatrix} \in \left((\Sigma \cup \{\#\})^k\right)^*,$$

where $\ell = \max\{\ell_1,\dots,\ell_k\}$ and $w'_{ij} = w_{ij}$, for $j \le \ell_i$ and $w'_{ij} = \#$ otherwise. The padding symbol # is added to the words w_i to ensure that all of them have the same length.

DEFINITION 3. *A structure A over a signature $S = (\mathsf{C}, \mathsf{R}, a)$ is* automatic *if there is a regular language $\mathcal{L}_{|A|} \subseteq \Sigma^*$ and a surjective function $\nu : \mathcal{L}_{|A|} \to |A|$ such that the languages $\mathcal{L}_{\approx} := \{u \otimes v \mid u, v \in \mathcal{L}_{|A|}$ with $\nu(u) = \nu(v)\}$ and $\mathcal{L}_r := \{u_1 \otimes \cdots \otimes u_{a(r)} \mid u_1, \dots, u_{a(r)} \in \mathcal{L}_{|D|}$ with $(\nu(u_1), \dots, \nu(u_{a(r)})) \in r^A\}$, for each $r \in \mathsf{R}$, are regular.*

An *automatic representation* of the automatic structure A consists of (i) the function $\nu : \mathcal{L}_{|A|} \to |A|$, (ii) a family of words $(w_c)_{c \in \mathsf{C}}$ with $w_c \in \mathcal{L}_{|A|}$ and $\nu(w_c) = c^A$, for all $c \in \mathsf{C}$, and (iii) a collection $(\mathcal{A}_{|A|}, \mathcal{A}_{\approx}, (\mathcal{A}_r)_{r \in \mathsf{R}})$ of automata that recognize the languages $\mathcal{L}_{|A|}$, $\mathcal{L}_{\approx}$, and $\mathcal{L}_r$, for all $r \in \mathsf{R}$. In the following, we assume that for an automatic structure, we always have an automatic representation for it at hand. A relation $r^A \subseteq |A|^k$ is *regular* if the language $\{u_1 \otimes \cdots \otimes u_k \mid u_1, \dots, u_k \in \mathcal{L}_{|A|}$ with $(\nu(u_1), \dots, \nu(u_k)) \in r\}$ is regular. Note that an automaton reads the components of the convolution of a representative of $\bar{a} \in |A|^k$ synchronously.

In addition to the requirement that each structure in D is automatic, we require that D has a constant domain representation. This means that the domain of each D_i is represented by the same regular language $\mathcal{L}_{|D|}$ and each word in $\mathcal{L}_{|D|}$ represents the same element in $|D|$, i.e., each automatic representation has the same function $\nu : \mathcal{L}_{|D|} \to |D|$. Finally, we assume that $|D| = \mathbb{N}$ and that $<$ is the standard ordering on $\mathbb{N}$. This is without loss of generality whenever the function ν is injective, i.e., every element in $|D|$ has only one representative in $\mathcal{L}_{|D|}$. Furthermore, note that every automatic structure has an automatic representation in which the function ν is injective [18].

Note that for a first-order formula θ, we can effectively construct an automaton that represents the set θ^{D_i}. Moreover, various basic arithmetical relations are first-order definable in the structure $(\mathbb{N}, <)$ and thus regular. For example, the successor relation $\{(x, y) \in \mathbb{N}^2 \mid y = x + 1\}$ and the relation $\{(x, y) \in \mathbb{N}^2 \mid x + d \leq y\}$, for any $d \in \mathbb{N}$, are regular.

Before presenting our monitoring method, we give two examples of system properties expressed in the MFOTL fragment that our monitor can handle. First, the property "whenever the program variable *in* stores the input x, then x must be stored in the program variable *out* within 5 time units" can be expressed by $\Box \forall x.\, in(x) \to \Diamond_{[0,6)}\, out(x)$. Second, the property "the value of the program variable v increases by 1 in each step from an initial value 0 until it becomes 5 and then it stays constant" can be formalized as $\Box(\neg(\bullet\, true) \to v(0)) \wedge (\exists i.\, v(i) \wedge i \prec 5 \to \bigcirc\, v(i+1)) \wedge (v(5) \to \bigcirc\, v(5))$. Note that we use relations that are singletons to model program variables.

3.2 Overview of the Monitoring Method

To monitor the formula $\Box\, \phi$ over a temporal structure (D, τ), we incrementally build a sequence of structures $\hat{D}_0, \hat{D}_1, \dots$ over an extended signature $\hat{S}$. The extension depends on the temporal subformulae of ϕ. For each time point i, we determine the elements that violate ϕ by evaluating a transformed formula $\neg\hat{\phi} \in \mathsf{FOL}$ over $\hat{D}_i$. Observe that with future operators, we usually cannot do this yet when time point i occurs. Our monitor, which we present in §3.5, therefore maintains a list of unevaluated subformulae for past time points. In the following, we first describe how we extend S and transform ϕ. Afterwards, we explain how we incrementally build $\hat{D}_i$. Finally, we present our monitor and prove its correctness.

3.3 Signature Extension and Formula Transformation

In addition to the predicates in R, the extended signature $\hat{S}$ contains an auxiliary predicate p_α for each temporal subformula α of ϕ. For subformulae of the form $\beta\,\mathcal{S}_I\,\gamma$ and $\beta\,\mathcal{U}_I\,\gamma$, we introduce further predicates, which store information that allows us to incrementally update the auxiliary relations.

DEFINITION 4. *Let $\hat{S} := (\hat{\mathsf{C}}, \hat{\mathsf{R}}, \hat{a})$ be the signature with $\hat{\mathsf{C}} := \mathsf{C}$ and $\hat{\mathsf{R}}$ is the union of the sets R, $\{p_\alpha \mid \alpha$ temporal subformula of $\phi\}$, $\{r_\alpha \mid \alpha$ subformula of ϕ of the form $\beta\,\mathcal{S}_I\,\gamma$ or $\beta\,\mathcal{U}_I\,\gamma\}$, and $\{s_\alpha \mid \alpha$ subformula of ϕ of the form $\beta\,\mathcal{U}_I\,\gamma\}$. For $r \in \mathsf{R}$, let $\hat{a}(r) := a(r)$. If α is a temporal subformula with n free variables, then $\hat{a}(p_\alpha) := n$, and $\hat{a}(r_\alpha) := n+1$ and $\hat{a}(s_\alpha) := n+2$, if r_α and s_α exist. We assume that $p_\alpha, r_\alpha, s_\alpha \notin \mathsf{C} \cup \mathsf{R} \cup \mathsf{V}$.*

We transform MFOTL formulae over the signature S into first-order formulae over the extended signature $\hat{S}$ as follows.

DEFINITION 5. *For $\theta \in \mathsf{MFOTL}$, we define (i) $\hat{\theta} := \neg\hat{\beta}$ if θ is of the from $\neg\beta$, (ii) $\hat{\theta} := \hat{\beta} \wedge \hat{\gamma}$ if θ is of the form $\beta \wedge \gamma$, (iii) $\hat{\theta} := \exists y.\, \hat{\beta}$ if θ is of the form $\exists y.\, \beta$, (iv) $\hat{\theta} := p_\theta(\bar{x})$ if θ is a temporal formula with the vector of free variables $\bar{x}$, and (v) $\hat{\theta} := \theta$ if θ is an atomic formula.*

We assume throughout this section, without loss of generality, that each subformula of ϕ has the vector of free variables $\bar{x} = (x_1, \dots, x_n)$. The formula transformation has the following properties, which are easily shown by an induction over the formula structure.

LEMMA 6. *Let θ be a subformula of ϕ. For all $i \in \mathbb{N}$, the following properties hold:*

(i) If $p_\alpha^{\hat{D}_i} = \alpha^{(D,\tau,i)}$ for all $\alpha \in tsub(\theta)$, then $\hat{\theta}^{\hat{D}_i} = \theta^{(D,\tau,i)}$.

(ii) If $p_\alpha^{\hat{D}_i}$ is regular for all $\alpha \in tsub(\theta)$, then $\hat{\theta}^{\hat{D}_i}$ is regular.

3.4 Incremental Extended Structure Construction

We now show how the auxiliary relations in the $\hat{D}_i$s are incrementally constructed. Their instantiations are computed recursively both over time and over the formula structure, where evaluations of subformulae may also be needed from future time points. We later show that this is well-defined and can be evaluated incrementally.

For $c \in \mathsf{C}$ and $r \in \mathsf{R}$, we define $c^{\hat{D}_i} := c^{D_i}$ and $r^{\hat{D}_i} := r^{D_i}$. We address the auxiliary relations for each type of main temporal operator separately.

Previous and Next. For $\alpha = \bullet_I\,\beta$ with $I \in \mathbb{I}$, we define $p_\alpha^{\hat{D}_i}$ as $\hat{\beta}^{\hat{D}_{i-1}}$ if $i > 0$ and $\tau_i - \tau_{i-1} \in I$, and $p_\alpha^{\hat{D}_i} := \emptyset$ otherwise. Intuitively, a tuple $\bar{a}$ is in $p_\alpha^{\hat{D}_i}$ if $\bar{a}$ satisfies β at the previous time point $i-1$ and the difference of the two successive time stamps is in the interval I.

LEMMA 7. *Let $\alpha = \bullet_I\,\beta$. For $i > 0$, if $p_\delta^{\hat{D}_{i-1}}$ is regular and $p_\delta^{\hat{D}_{i-1}} = \delta^{(D,\tau,i-1)}$ for all $\delta \in tsub(\beta)$, then $p_\alpha^{\hat{D}_i}$ is regular and $p_\alpha^{\hat{D}_i} = \alpha^{(D,\tau,i)}$. Moreover, $p_\alpha^{\hat{D}_0}$ is regular and $p_\alpha^{\hat{D}_0} = \alpha^{(D,\tau,0)}$.*

PROOF. For $i = 0$, the lemma obviously holds. For $i > 0$, the regularity of $p_\alpha^{\hat{D}_i}$ follows from the assumption that the relations $p_\delta^{\hat{D}_{i-1}}$ are regular and Lemma 6(ii). The equality of the two sets follows from Lemma 6(i) and the semantics of the temporal operator $\bullet_I$. ∎

For $\alpha = \bigcirc_I \beta$ with $I \in \mathbb{I}$, we define $p_\alpha^{\hat{D}_i}$ as $\hat{\beta}^{\hat{D}_{i+1}}$ if $\tau_{i+1} - \tau_i \in I$, and $p_\alpha^{\hat{D}_i} := \emptyset$ otherwise. Note that the definition of $p_\alpha^{\hat{D}_i}$ depends on the relations of the next structure D_{i+1} and on the auxiliary relations for $\delta \in tsub(\beta)$ of the next extended structure $\hat{D}_{i+1}$. Hence, the monitor instantiates $p_\alpha^{\hat{D}_i}$ with a delay of at least one time step.

LEMMA 8. *Let $\alpha = \bigcirc_I \beta$. If $p_\delta^{\hat{D}_{i+1}}$ is regular and $p_\delta^{\hat{D}_{i+1}} = \delta^{(D,\tau,i+1)}$ for all $\delta \in tsub(\beta)$, then $p_\alpha^{\hat{D}_i}$ is regular and $p_\alpha^{\hat{D}_i} = \alpha^{(D,\tau,i)}$.*

Since and Until. We first address the past-time operator $\mathcal{S}_I$ with $I = [c,d) \in \mathbb{I}$. Assume that $\alpha = \beta\, \mathcal{S}_I\, \gamma$. We start with the initialization and update of the auxiliary relations for r_α. We define $r_\alpha^{\hat{D}_0} := \hat{\gamma}^{\hat{D}_0} \times \{0\}$ and for $i > 0$, we define

$$r_\alpha^{\hat{D}_i} := \big(\hat{\gamma}^{\hat{D}_i} \times \{0\}\big) \cup \big\{ (\bar{a},y) \in \mathbb{N}^{n+1} \,\big|\, \bar{a} \in \hat{\beta}^{\hat{D}_i},\ y < d, \text{ and } (\bar{a},y') \in r_\alpha^{\hat{D}_{i-1}}, \text{ for } y' = y - \tau_i + \tau_{i-1} \big\}\,.$$

Intuitively, a pair $(\bar{a},y)$ is in $r_\alpha^{\hat{D}_i}$ if $\bar{a}$ satisfies α at time point i independent of the lower bound c, where the "age" y indicates how long ago the formula γ was satisfied by $\bar{a}$. If $\bar{a}$ satisfies γ at the time point i, it is added to $r_\alpha^{\hat{D}_i}$ with the age 0. For $i > 0$, we additionally update the tuples $(\bar{a},y) \in r_\alpha^{\hat{D}_{i-1}}$. First, $\bar{a}$ must satisfy β at the time point i. Second, the age is adjusted by the difference of the time stamps τ_{i-1} and τ_i. Third, the new age must be less than d, otherwise it is too old to satisfy α.

The arithmetic constraint $y' = y - \tau_i + \tau_{i-1}$ in the definition of $r_\alpha^{\hat{D}_i}$ for $i > 0$ is first-order definable in D. Note that $\tau_i + \tau_{i-1}$ is a constant value. Now it is not hard to see that $r_\alpha^{\hat{D}_i}$ is regular if all its components are regular.

With the relation $r_\alpha^{\hat{D}_i}$, we can determine the elements that satisfy α at the time point i. We define $p_\alpha^{\hat{D}_i} := \big\{\bar{a} \in \mathbb{N}^n \,\big|\, (\bar{a},y) \in r_\alpha^{\hat{D}_i}, \text{ for some } y \geq c\big\}$.

LEMMA 9. *Let $\alpha = \beta\, \mathcal{S}_{[c,d)}\, \gamma$. Assume that $p_\delta^{\hat{D}_j}$ is regular and $p_\delta^{\hat{D}_j} = \delta^{(D,\tau,j)}$, for all $j \leq i$ and $\delta \in tsub(\beta) \cup tsub(\gamma)$. Then the following properties hold:*

(i) The relation $r_\alpha^{\hat{D}_i}$ is regular and for all $\bar{a} \in \mathbb{N}^n$ and $y \in \mathbb{N}$,

$$(\bar{a},y) \in r_\alpha^{\hat{D}_i} \quad \textit{iff} \quad \begin{array}{c} \text{there is a } j \in [0,i+1) \text{ such that } y = \tau_i - \tau_j < d\,, \bar{a} \in \gamma^{(D,\tau,j)}\,, \\ \text{and } \bar{a} \in \beta^{(D,\tau,k)}, \text{ for all } k \in [j+1, i+1)\,. \end{array}$$

(ii) The relation $p_\alpha^{\hat{D}_i}$ is regular and $p_\alpha^{\hat{D}_i} = \alpha^{(D,\tau,i)}$.

Note that the definition of $r_\alpha^{\hat{D}_i}$ only depends on the relation $r_\alpha^{\hat{D}_{i-1}}$, if $i > 0$, and on the relations in $\hat{D}_i$ for which the corresponding predicates occur in the subformulae of $\hat{\beta}$ or $\hat{\gamma}$. Furthermore, the definition of $p_\alpha^{\hat{D}_i}$ only depends on $r_\alpha^{\hat{D}_i}$.

We now address the bounded future-time operator $\mathcal{U}_I$ with $I = [c,d) \in \mathbb{I}$ and $d \in \mathbb{N}$. Assume that $\alpha = \beta\, \mathcal{U}_I\, \gamma$. For all $i \in \mathbb{N}$, let $\ell_i := \max\{j \in \mathbb{N} \mid \tau_{i+j} - \tau_i < d\}$. We call ℓ_i the lookahead offset at time point i. For convenience, let $\ell_{-1} := 0$. To instantiate the relation $p_\alpha^{\hat{D}_i}$, only the relations $p_\delta^{\hat{D}_i}, \ldots, p_\delta^{\hat{D}_{i+\ell_i}}$ are relevant, where $\delta \in tsub(\beta) \cup tsub(\gamma)$. The definition of $p_\alpha^{\hat{D}_i}$ is based on the auxiliary relations $r_\alpha^{\hat{D}_i}$ and $s_\alpha^{\hat{D}_i}$, which we first show how to initialize and update.

We define $r_\alpha^{\hat{D}_i}$ as the union of the sets N_r and U_r. N_r contains the tuples that are new in the sense that they are obtained from data at the time points $i+\ell_{i-1},\dots,i+\ell_i$; U_r contains the updated data from the time points $i,\dots,i+\ell_{i-1}-1$. Formally, we define

$$\begin{aligned} N_r &:= \big\{(\bar{a},j)\in\mathbb{N}^{n+1}\,\big|\,\ell_{i-1}\le j\le\ell_i,\ \bar{a}\in\hat{\gamma}^{\hat{D}_{i+j}},\text{ and }\tau_{i+j}-\tau_i\ge c\big\}\\ U_r &:= \begin{cases}\big\{(\bar{a},j)\in\mathbb{N}^{n+1}\,\big|\,(\bar{a},j+1)\in r_\alpha^{\hat{D}_{i-1}}\text{ and }\tau_{i+j}-\tau_i\ge c\big\} & \text{if } i>0,\\ \emptyset & \text{otherwise.}\end{cases}\end{aligned}$$

Intuitively, $r_\alpha^{\hat{D}_i}$ stores the tuples satisfying the formula $\Diamond_I\,\gamma$ at the time point i, where each tuple in $r_\alpha^{\hat{D}_i}$ is augmented by the index relative to i where the tuple satisfies γ.

Similarly to $r_\alpha^{\hat{D}_i}$, the relation $s_\alpha^{\hat{D}_i}$ is the union of a set N_s for the new elements and a set U_s for the updates. These two sets are defined as

$$N_s := \big\{(\bar{a},j,j')\in\mathbb{N}^{n+2}\,\big|\,\ell_{i-1}\le j\le j'\le\ell_i\text{ and }\bar{a}\in\hat{\beta}^{\hat{D}_{i+k}},\text{ for all }k\in[j,j'+1)\big\}$$

and $U_s:=\emptyset$ if $i=0$, and

$$\begin{aligned} U_s := {} & \big\{(\bar{a},j,j')\in\mathbb{N}^{n+2}\,\big|\,(\bar{a},j+1,j'+1)\in s_\alpha^{\hat{D}_{i-1}}\big\}\cup\\ & \big\{(\bar{a},j,j')\in\mathbb{N}^{n+2}\,\big|\,(\bar{a},j+1,\ell_{i-1})\in s_\alpha^{\hat{D}_{i-1}}\text{ and }(\bar{a},\ell_{i-1},j')\in N_s\big\}\end{aligned}$$

otherwise. Intuitively, $s_\alpha^{\hat{D}_i}$ stores the tuples and the bounds of the interval (relative to i) in which β is satisfied.

With the relations $r_\alpha^{\hat{D}_i}$ and $s_\alpha^{\hat{D}_i}$ at hand, we define

$$p_\alpha^{\hat{D}_i} := \big\{\bar{a}\in\mathbb{N}^n\,\big|\,(\bar{a},j)\in r_\alpha^{\hat{D}_i}\text{ and }(\bar{a},0,j')\in s_\alpha^{\hat{D}_i},\text{ for some }j\le j'+1\big\}\,.$$

LEMMA 10. *Let $\alpha=\beta\,\mathcal{U}_I\,\gamma$. Assume that $p_\delta^{\hat{D}_k}$ is regular and $p_\delta^{\hat{D}_k}=\delta^{(D,\tau,k)}$, for all $k\le i+\ell_i$ and $\delta\in tsub(\beta)\cup tsub(\gamma)$. Then the following properties hold:*

(i) The relation $r_\alpha^{\hat{D}_i}$ is regular and for all $\bar{a}\in\mathbb{N}$ and $j\in\mathbb{N}$,

$$(\bar{a},j)\in r_\alpha^{\hat{D}_i}\qquad\text{iff}\qquad \bar{a}\in\gamma^{(D,\tau,i+j)}\text{ and }\tau_{i+j}-\tau_i\in I\,.$$

(ii) The relation $s_\alpha^{\hat{D}_i}$ is regular and for all $\bar{a}\in\mathbb{N}^n$ and $j,j'\in\mathbb{N}$,

$$(\bar{a},j,j')\in s_\alpha^{\hat{D}_i}\qquad\text{iff}\qquad j\le j',\ \tau_{i+j'}-\tau_i<d,\text{ and }\bar{a}\in\beta^{(D,\tau,i+k)},\text{ for all }k\in[j,j'+1)\,.$$

(iii) The relation $p_\alpha^{\hat{D}_i}$ is regular and $p_\alpha^{\hat{D}_i}=\alpha^{(D,\tau,i)}$.

3.5 Monitor and Correctness

Figure 1 presents the monitor $\mathcal{M}(\phi)$. Without loss of generality, it assumes that each temporal subformula occurs only once in ϕ. In the following, we outline its operation.

The monitor uses two counters i and q. The counter i is the index of the current element (D_i,τ_i) in the input sequence $(D_0,\tau_0),(D_1,\tau_1),\dots$, which is processed sequentially. Initially, i is 0 and it is incremented at the end of each loop iteration (lines 4–16). The counter $q\le i$ is the index of the next time point q (possibly in the past, from the point of view of i) for which we evaluate $\neg\hat{\phi}$ over the structure $\hat{D}_q$. The evaluation is delayed until the relations $p_\alpha^{\hat{D}_q}$ for $\alpha\in tsub(\phi)$ are all instantiated (lines 10–13). Furthermore, the monitor uses the list[2]

[2]We abuse notation by using set notation for lists. Moreover, we assume that Q is ordered in that (α,j,S) occurs before (α',j',S'), whenever α is a proper subformula of α', or $\alpha=\alpha'$ and $j<j'$.

```
1: $i \leftarrow 0$                                   % current index in input sequence $(D_0, \tau_0), (D_1, \tau_1), \ldots$
2: $q \leftarrow 0$                                   % index of next query evaluation in sequence $(D_0, \tau_0), (D_1, \tau_1), \ldots$
3: $Q \leftarrow \{ ((\alpha, 0, \mathit{waitfor}(\alpha)) \mid \alpha$ temporal subformula of $\phi \}$
4: loop
5:    Carry over constants and relations of $D_i$ to $\hat{D}_i$.
6:    for all $(\alpha, j, \emptyset) \in Q$ do                 % respect ordering of subformulae
7:          Build relations for $\alpha$ in $\hat{D}_j$ (e.g., build $r_\alpha^{\hat{D}_j}$ and $p_\alpha^{\hat{D}_j}$ if $\alpha = \beta \mathcal{S}_I \gamma$).
8:          Discard auxiliary relations for $\alpha$ in $\hat{D}_{j-1}$ if $j - 1 \geq 0$ (e.g., discard $r_\alpha^{\hat{D}_{j-1}}$ if $\alpha = \beta \mathcal{S}_I \gamma$).
9:          Discard relations $p_\delta^{\hat{D}_j}$, where $\delta$ is a temporal subformula of $\alpha$.
10:   while all relations $p_\alpha^{\hat{D}_q}$ are built for $\alpha \in \mathit{tsub}(\phi)$ do
11:         Output valuations violating $\phi$ at time point $q$, i.e., output $(\neg\hat{\phi})^{\hat{D}_q}$ and $q$.
12:         Discard structure $\hat{D}_{q-1}$ if $q - 1 \geq 0$.
13:         $q \leftarrow q + 1$
14:   $Q \leftarrow \{ (\alpha, i+1, \mathit{waitfor}(\alpha)) \mid \alpha$ temporal subformula of $\phi \} \cup$
            $\{ (\alpha, j, \bigcup_{\theta \in \mathit{update}(S, \tau_{i+1} - \tau_i)} \mathit{waitfor}(\theta)) \mid (\alpha, j, S) \in Q$ and $S \neq \emptyset \}$
15:   $i \leftarrow i + 1$                               % process next element in input sequence $(D_{i+1}, \tau_{i+1})$
16: end loop
```

Figure 1: Monitor $\mathcal{M}(\phi)$

Q to ensure that the auxiliary relations of $\hat{D}_0, \hat{D}_1, \ldots$ are built at the right time: if $(\alpha, j, \emptyset)$ is an element of Q at the beginning of a loop iteration, enough time has elapsed to build the relations for the temporal subformula α of the structure $\hat{D}_j$. The monitor initializes Q in line 3. The function *waitfor* extracts the subformulae that cause a delay of the formula evaluation. We define $\mathit{waitfor}(\theta)$ to be: (i) $\mathit{waitfor}(\beta)$ if $\theta = \neg\beta$, $\theta = \exists x.\, \beta$, or $\theta = \bullet_I \beta$; (ii) $\mathit{waitfor}(\beta) \cup \mathit{waitfor}(\gamma)$ if $\theta = \beta \wedge \gamma$ or $\theta = \beta \mathcal{S}_I \gamma$, (iii) $\{\theta\}$ if $\theta = \circ_I \beta$ or $\theta = \beta \mathcal{U}_I \gamma$, and (iv) $\emptyset$ otherwise. The list Q is updated in line 14 before we increment i and start a new loop iteration. For the update we use the function *update* that is defined as

$$\begin{aligned}\mathit{update}(U, \Delta) := {} & \{\beta \mid \circ_I \beta \in U\} \cup \{\beta \, \mathcal{U}_{[\max\{0, c-\Delta\}, d-\Delta)} \, \gamma \mid \beta \, \mathcal{U}_{[c,d)} \, \gamma \in U, \text{with } d - \Delta > 0\} \cup \\ & \{\beta \mid \beta \, \mathcal{U}_{[c,d)} \, \gamma \in U \text{ or } \gamma \, \mathcal{U}_{[c,d)} \, \beta \in U, \text{with } d - \Delta \leq 0\} ,\end{aligned}$$

for a formula set U and $\Delta \in \mathbb{N}$. The update adds a new tuple $(\alpha, i+1, \mathit{waitfor}(\alpha))$ to Q, for each temporal subformula α of ϕ, and it removes the tuples of the form $(\alpha, j, \emptyset)$ from Q. Moreover, for tuples (α, j, S) with $S \neq \emptyset$, the set S is updated using the functions *waitfor* and *update* by taking into account the elapsed time to the next time point, i.e. $\tau_{i+1} - \tau_i$.

In lines 6–9, we build the relations for which enough time has elapsed, i.e., the auxiliary relations for α in $\hat{D}_j$ with $(\alpha, j, \emptyset) \in Q$. Since a tuple $(\alpha', j, \emptyset)$ does not occur before a tuple $(\alpha, j, \emptyset)$ in Q, where α is a subformula of α', the relations in $\hat{D}_j$ for α are built before those for α'. To build the relations, we use the incremental constructions described earlier in this section. We thus discard certain relations after we have built the relations for α in $\hat{D}_j$ to reduce space consumption. For instance, if $j > 0$ and $\alpha = \beta \mathcal{S}_I \gamma$, we discard the relation $r_\alpha^{\hat{D}_{j-1}}$, and we discard $r_\alpha^{\hat{D}_{j-1}}$ and $s_\alpha^{\hat{D}_{j-1}}$ when $\alpha = \beta \mathcal{U}_I \gamma$.

In lines 10–13, the valuations violating ϕ at time point q are output together with q, for all q where the relations $p_\alpha^{\hat{D}_q}$ of all immediate temporal subformulae α of ϕ have been built. After an output, the remainder of the extended structure $\hat{D}_{q-1}$ is discarded and q is incremented by 1.

THEOREM 11. *The monitor $\mathcal{M}(\phi)$ from Figure 1 has the following properties:*

(i) Whenever $\mathcal{M}(\phi)$ outputs $(\neg\hat{\phi})^{\hat{D}_q}$, then $(\neg\hat{\phi})^{\hat{D}_q} = (\neg\phi)^{(\bar{D},\bar{\tau},q)}$. Furthermore, the set $(\neg\hat{\phi})^{\hat{D}_q}$ is effectively constructable and finitely representable.

(ii) For every $n \in \mathbb{N}$, $\mathcal{M}(\phi)$ eventually sets the counter q to n in some loop iteration.

4 MFOTL Monitoring with Finite Relations

In this section, we sketch how to use relational databases as an alternative to automata for implementing our monitor and analyze its space complexity. Details are provided in [5].

In the following, we assume that all relations are finite and thus can be stored in a relational database. When replacing "regular" by "finite", however, our constructions from §3.4, in particular Lemmas 7–10, become invalid. The problem is that the auxiliary relations constructed for the temporal subformulae are possibly infinite. We overcome this problem by extending work from database theory on domain independence [14]. In particular, we generalize the solutions for first-order queries [2] and non-metric first-order temporal logic [8, 10, 11] to MFOTL formulae by trying to rewrite the given MFOTL formula ϕ so that all temporal subformulae and their direct subformulae have only finitely many satisfying valuations. After rewriting the formula ϕ, we check, based on the syntax of the result ψ, whether each $\theta \in \{\alpha \mid \alpha = \psi, \alpha$ is a temporal subformula of ψ, or α is a direct subformula of a temporal subformula of $\psi\}$ is *temporal domain independent*. If ψ passes this check, we know that it can be handled by our monitor for finite relations. Otherwise, no conclusions can be drawn. For the rest of this section, we assume that ϕ, all temporal subformulae of ϕ, and all direct subformulae of temporal subformulae of ϕ are temporal domain independent.

We now analyze the memory consumption of our monitor for finite relations. To obtain a polynomial bound on the memory consumption, we modify $\mathcal{M}(\phi)$ as follows: (i) the counters i and q are replaced by the relative counter $i - q$ and (ii) the update constructions for subformulae of the form $\alpha = \beta\, \mathcal{S}_{[c,\infty)}\, \gamma$ are modified to prevent the "age" y of a tuple $(\bar{a}, y) \in r_\alpha^{\hat{D}_{i-1}}$ from increasing forever. The analyze the resources consumed by monitors in general, we introduce the following abstract notion. Let C be a class of temporal structures over the signature $S = (\mathsf{C}, \mathsf{R}, a)$ and let $pre(C)$ denote the set of nonempty finite prefixes of the temporal structures in C.

DEFINITION 12. *Let $f, g : pre(C) \to \mathbb{N}$ and $s : \mathbb{N} \to \mathbb{N}$ be functions. We write $f \lhd^s g$ if $f(\bar{D},\bar{\tau}) < s(g(\bar{D},\bar{\tau}))$, for all $(\bar{D},\bar{\tau}) \in pre(C)$.*

In our context, the function $f : pre(C) \to \mathbb{N}$ measures the consumption of a particular resource (e.g., storage) of a monitor after it has processed the finite prefix $(\bar{D},\bar{\tau})$. The function $g : pre(C) \to \mathbb{N}$ measures the size of the prefix $(\bar{D},\bar{\tau})$. Intuitively, $f \lhd^s g$ means that, at any time point, the resource consumption (measured by f) of the monitor is bounded by the function $s : \mathbb{N} \to \mathbb{N}$ with respect to the size of the processed prefix (measured by g) of an input from C. We use the following concrete functions f and g. Let $(\bar{D},\bar{\tau}) \in pre(C)$ with $\bar{D} = (D_0, \dots, D_i)$ and $\bar{\tau} = (\tau_0, \dots, \tau_i)$.

– We define $g(\bar{D},\bar{\tau}) := |adom(\bar{D})|$, where $adom(\bar{D})$ is the active domain of $(\bar{D},\bar{\tau})$, i.e., $adom(\bar{D}) := \{c^{D_0} \mid c \in \mathsf{C}\} \cup \bigcup_{0 \le k \le i} \bigcup_{r \in \mathsf{R}} \{d_j \mid (d_1, \dots, d_{a(r)}) \in r^{D_k} \text{ and } 1 \le j \le a(r)\}$.

Note that g only counts the number of elements of $\bar{D}$ that are constants or that occur in some of $\bar{D}$'s relations. It ignores the sizes of these elements as well as the number of times and where an element appears in $\bar{D}$. It also ignores the time stamps in $\bar{\tau}$.

– We define $f(\bar{D}, \bar{\tau})$ to be the sum of the cardinalities of the relations for $r \in \hat{\mathsf{R}}$ stored by $\mathcal{M}(\phi)$ after the $(i+1)$st loop iteration, having processed the input $(D_0, \tau_0), \ldots, (D_i, \tau_i)$.

Note that $f \lhd^s g$ is a desirable property of a monitor. It says that the amount of data stored does not depend on how long the monitor has been running but only on the number of domain elements that appeared so far, and that the stored data is bounded by the function s. We remark that the property of a (polynomially) bounded history encoding [8] can be formalized as $f \lhd^s g$, for some (polynomial) $s : \mathbb{N} \to \mathbb{N}$.

THEOREM 13. *Let C be a class of temporal databases. Assume that there is some $\ell \in \mathbb{N}$ such that* $\max\{j \mid \tau_i = \tau_{i+1} = \ldots = \tau_{i+j}\} < \ell$*, for all $(D, \tau) \in C$ and all $i \in \mathbb{N}$. Then, we have that $f \lhd^s g$, where $s : \mathbb{N} \to \mathbb{N}$ is a polynomial of degree* $\max\{a(r) \mid r \in \hat{\mathsf{R}}\}$.

Note that if such a bound ℓ on the sequence τ of time stamps does not exist, we cannot guarantee any upper bound on f. It is open whether Theorem 13 can be carried over to temporal structures with possibly infinite relations and automatic representations.

5 Conclusion and Future Work

We have presented an automata-based monitoring approach for an expressive fragment of a metric first-order temporal logic. The use of automata substantially generalizes both the kinds of structures and the class of formulae that can be monitored. Moreover, it eliminates the limitations that arise in databases, where relations must be finite. An interesting question here is to what extent the use of automatic structures can be carried over to other monitoring approaches, thereby solving the problems they have with infinite relations.

One direction for future work is to explore whether our approach can be used to monitor temporal first-order logics that have an interval-based semantics instead of a point-based semantics, or a combined interval and point-based semantics, which is useful for modeling state and event predicates. Another direction is to conduct a refined complexity analysis for our algorithm with automatic structures and to validate our results by implementation and testing. In particular, we plan to design and evaluate data structures and algorithms for efficiently incrementally updating relations, which is at the heart of our monitoring algorithm.

References

[1] *Proceedings of the 1st to 8th Workshop on Runtime Verification (RV)*, 2001–2008.

[2] S. Abiteboul, R. Hull, and V. Vianu. *Foundations of Databases*. Addison-Wesley, 1995.

[3] B. Alpern and F. Schneider. Defining liveness. *Inf. Process. Lett.*, 21(4):181–185, 1985.

[4] H. Barringer, A. Goldberg, K. Havelund, and K. Sen. Rule-based runtime verification. In *Verification, Model Checking, and Abstract Interpretation (VMCAI'04)*, vol. 2937 of *LNCS*, pp. 44–57.

[5] D. Basin, F. Klaedtke, S. Müller, and B. Pfitzmann. Runtime monitoring of metric first-order temporal properties. Technical Report RZ 3702, IBM Research and ETH Zurich, 2008.

[6] A. Bauer, M. Leucker, and C. Schallhart. Monitoring of real-time properties. In *Foundations of Software Technology and Theoretical Computer Science (FSTTCS'06)*, vol. 4337 of *LNCS*, pp. 260–272.

[7] A. Blumensath and E. Grädel. Finite presentations of infinite structures: Automata and interpretations. *Theory Comput. Syst.*, 37(6):641–674, 2004.

[8] J. Chomicki. Efficient checking of temporal integrity constraints using bounded history encoding. *ACM Trans. Database Syst.*, 20(2):149–186, 1995.

[9] J. Chomicki and D. Niwiński. On the feasibility of checking temporal integrity constraints. *J. Comput. Syst. Sci.*, 51(3):523–535, 1995.

[10] J. Chomicki and D. Toman. Implementing temporal integrity constraints using an active DBMS. *IEEE Trans. on Knowl. and Data Eng.*, 7(4):566–582, 1995.

[11] J. Chomicki, D. Toman, and M. Böhlen. Querying ATSQL databases with temporal logic. *ACM Trans. Database Syst.*, 26(2):145–178, 2001.

[12] B. D'Angelo, S. Sankaranarayanan, C. Sánchez, W. Robinson, B. Finkbeiner, H. Sipma, S. Mehrotra, and Z. Manna. LOLA: Runtime monitoring of synchronous systems. In *Temporal Representation and Reasoning (TIME'05)*, pp. 166–174.

[13] N. Dinesh, A. Joshi, I. Lee, and O. Sokolsky. Checking traces for regulatory conformance. In *Runtime Verification (RV'08)*.

[14] R. Fagin. Horn clauses and database dependencies. *J. ACM*, 29(4):952–985, 1982.

[15] C. Giblin, A. Liu, S. Müller, B. Pfitzmann, and X. Zhou. Regulations expressed as logical models (REALM). In *Legal Knowledge and Information Systems (JURIX'05)*, vol. 134 of *Frontiers in Artificial Intelligence and Applications*, pp. 37–48.

[16] K. Havelund and G. Rosu. Efficient monitoring of safety properties. *Int. J. Softw. Tools Technol. Transf.*, 6(2):158–173, 2004.

[17] J. Håkansson, B. Jonsson, and O. Lundqvist. Generating online test oracles from temporal logic specifications. *Int. J. Softw. Tools Technol. Transf.*, 4(4):456–471, 2003.

[18] B. Khoussainov and A. Nerode. Automatic presentations of structures. In *Logical and Computational Complexity*, vol. 960 of *LNCS*, pp. 367–392, 1995.

[19] R. Koymans. Specifying real-time properties with metric temporal logic. *Real-Time Systems*, 2(4):255–299, 1990.

[20] K. Kristoffersen, C. Pedersen, and H. Andersen. Runtime verification of timed LTL using disjunctive normalized equation systems. *Electr. Notes Theor. Comput. Sci.*, 89(2):210–225, 2003.

[21] O. Lichtenstein, A. Pnueli, and L. Zuck. The glory of the past. In *Logic of Programs*, vol. 193 of *LNCS*, pp. 196–218, 1985.

[22] U. Lipeck and G. Saake. Monitoring dynamic integrity constraints based on temporal logic. *Inf. Syst.*, 12(3):255–269, 1987.

[23] O. Maler, D. Nickovic, and A. Pnueli. From MITL to timed automata. In *Formal Modeling and Analysis of Timed Systems (FORMATS'06)*, vol. 4202 of *LNCS*, pp. 274–289.

[24] D. Nickovic and O. Maler. AMT: A property-based monitoring tool for analog systems. In *Formal Modeling and Analysis of Timed Systems (FORMATS'07)*, vol. 4763 of *LNCS*, pp. 304–319.

[25] M. Roger and J. Goubault-Larrecq. Log auditing through model-checking. In *Computer Security Foundations Workshop (CSFW'01)*, pp. 220–234.

[26] G. Rosu and K. Havelund. Rewriting-based techniques for runtime verification. *Autom. Softw. Eng.*, 12(2):151–197, 2005.

[27] A. Sistla and O. Wolfson. Temporal triggers in active databases. *IEEE Trans. Knowl. Data Eng.*, 7(3):471–486, 1995.

[28] O. Sokolsky, U. Sammapun, I. Lee, and J. Kim. Run-time checking of dynamic properties. *Electr. Notes Theor. Comput. Sci.*, 144(4):91–108, 2006.

[29] P. Thati and G. Rosu. Monitoring algorithms for metric temporal logic specifications. *Electr. Notes Theor. Comput. Sci.*, 113:145–162, 2005.

[30] D. Toman. Logical data expiration. In *Logics for Emerging Applications of Databases*, pp. 203–238, 2003.