Foundations for Query-based Runtime Monitoring of Temporal Properties over Runtime Models

Lucas Sakizloglou[1(✉)], Holger Giese[2], and Leen Lambers[1]

[1] Brandenburg University of Technology, Cottbus, Germany
lucas.sakizloglou@b-tu.de
[2] Hasso Plattner Institute at the University of Potsdam, Potsdam, Germany

Abstract. In model-driven engineering, runtime monitoring of systems with complex dynamic structures is typically performed via a runtime model capturing a snapshot of the system state: the model is represented as a graph and properties of interest as graph queries which are evaluated over the model online. For temporal properties, history-aware runtime models encode a trace of timestamped snapshots, which is monitored via temporal graph queries. In this case, the query evaluation needs to consider that a trace may be incomplete, thus future changes to the model may affect current answers. So far there is no formal foundation for query-based monitoring over runtime models encoding incomplete traces. In this paper, we present a systematic and formal treatment of incomplete traces. First, we introduce a new definite semantics for a first-order temporal graph logic which only returns answers if no future change to the model will affect them. Then, we adjust the query evaluation semantics of a querying approach we previously presented, which is based on this logic, to the definite semantics of the logic. Lastly, we enable the approach to keep to its efficient query evaluation technique, while returning (the more costly) definite answers.

1 Introduction

Modern safety-critical systems, *e.g.*, smart healthcare and autonomous transportation, consist of numerous interconnected technologies such as sensors, smart devices, and information systems [15]. These systems are human-in-the-loop and operate in highly dynamic environments [16]. Moreover, they are real-time, *i.e.*, their safe operation depends on the timing of their actions, and missed deadlines for these actions may lead to hazardous situations [46]. These characteristics hinder complete quality assurance during the design of such systems and increase the uncertainty about their behavior at runtime. Consequently, their safe operation relies on formally precise *Runtime Monitoring* (RM) techniques [34], which are capable of handling the complex underlying structure and its dynamic [13] as well as timing constraints when monitoring the system behavior [4].

As shown by recent surveys [9, 52], in model-driven engineering, RM of systems with complex dynamic structures is typically performed via a (structural) *Runtime Model* (RTM) [12] capturing a snapshot of the system state: the model is represented as a graph of interacting components and properties of interest

D. Beyer and A. Cavalcanti (Eds.): FASE 2024, LNCS 14573, pp. 22–55, 2024.
https://doi.org/10.1007/978-3-031-57259-3_2

as graph queries which are evaluated over the model online; query matches constitute monitoring issues. For efficiency, the evaluation of graph queries is based on methods which afford incremental and *change-driven* evaluation [54], *i.e.*, triggered only when changes to the RTM are relevant to a query.

For temporal properties, *history-aware* RTMs capture past changes to the model and their timing [11], thereby encoding a trace of timestamped snapshots. These RTMs are then monitored via the evaluation of *temporal graph queries* which specify the ordering and timing constraints that matches should satisfy. In this case, the query evaluation needs to consider that the trace encoded by the history-aware RTM may be *incomplete*, *i.e.*, the execution may be ongoing, and hence future changes to the RTM may affect current query answers. So far there is no formal foundation for temporal-query-based RM over incomplete RTMs.

In our previous work, we presented a querying approach for the evaluation of temporal graph queries over history-aware RTMs named InTempo [49]—see Section 2.3 for an overview and Fig. 1 for an illustration. InTempo advances the state-of-the-art by: enabling a *formally precise answer set* which pairs matches with their *temporal validity*, *i.e.*, the set of all time points for which a match exists and satisfies a temporal property according to a first-order temporal graph logic; featuring sound methods for *incremental and change-driven evaluation* as well as the optional *pruning* of the RTM, *i.e.*, the removal of temporally irrelevant history. Extensive experimental evaluation showed that our implementation of InTempo efficiently evaluated complex queries over considerably large models (approx. from 10K to 48M elements) [49]. The experimental evaluation included an RM application scenario, in which InTempo evaluated queries faster than an RTM-based tool and a tool from the related RM approach known as *Runtime Verification* (RV).

However, the formal foundation of InTempo assumes that the RTM encodes a complete trace. For the RM scenario, we equipped InTempo with a check that was applied to the answer set and, based on the timing constraint of the property, filtered matches that could be affected by future changes to the RTM. In this paper, we present a formal foundation for temporal-query-based RM over incomplete RTMs. The foundation entails the introduction of an answer set which formalizes the intuition behind the check and allows approaches like InTempo to maintain their efficiency while returning formally precise answers.

Specifically, our contributions are the following. First, we introduce a *definite* semantics for a temporal graph logic (Section 3), which only returns answers if they are definite, *i.e.*, no future change to the RTM will affect them; we show that the definite semantics is sound. Then, we introduce a new *definite answer set* (Section 4) for the query language of InTempo which pairs matches with their definite temporal validity and invalidity. Compared to the original (non-definite) answer set, the definite answer relies on the time point on which a query is evaluated and thus requires the re-computation of the definite temporal validity and invalidity in each evaluation. The definite answer set is thus inefficient, *i.e.*, not amenable to change-driven evaluation. However, we use this theoretical result to show that our last contribution, the *effective answer set* (Section 5), which

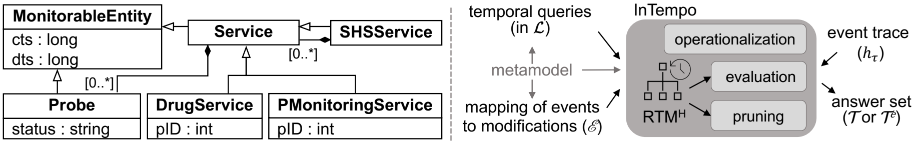

Fig. 1: An excerpt of the SHS metamodel from [49] (left) and an operational overview of the InTempo implementation where arrows denote input and output.

essentially incorporates the check mentioned above, can return definite answers while relying on the original, and thus efficient, answer set.

The presented contributions are based on unpublished material from the doctoral thesis of the first author [47]. Section 2 reiterates preliminaries and InTempo, Section 6 discusses related work, and Section 7 concludes the paper.

Running Example As a running example we will use the *Smart Healthcare System* (SHS) introduced in [49]. Fig. 1 shows an excerpt of the SHS metamodel. An SHS is an envisioned smart medical environment [45], based on the service-based exemplar in [55], which supports clinicians in medical treatments by automating tasks via smart devices. In the context of an SHS, RM may be used to verify whether treatments comply with the requirements in a guideline, which typically contain timing constraints [17]. In the SHS, services are invoked by a main service called `SHSService` to collect measurements from patient sensors, *i.e.*, `PMonitoringService`, or take medical actions via smart medical devices such as a smart pump, *i.e.*, `DrugService`. The results of service invocations are tracked via monitoring probes (`Probe`) that are attached to `Services`. `Probes` are generated periodically or upon events in the real world. Each `Probe` has a *status* attribute whose value depends on the type of `Service`. Each `Service` has a *pID* attribute which identifies the patient for whom the `Service` is invoked. The `MonitorableEntity` is explained in Section 2.1.

We focus on a property P that tracks time between triage and admission, as often done in medical guidelines [39]; in the context of an SHS, these activities are represented by the invocation of a sensor service and a drug service, respectively: *"When a sensor service is invoked for a patient, there should be a drug service invoked for the same patient within one minute and, until then, there should be no other sensor service invoked for the same patient."* The specific timing constraint is adjusted for the purpose of presentation. Assume an RTM that captures that a sensor service has just been invoked for a patient, but contains no drug invocation yet; for monitoring P, it is important to consider that a future state which contains the drug service invocation may follow in time; therefore, the present state does not yet violate P.

2 Preliminaries

In this section, we summarize preliminaries and the InTempo query language. An overview of the notation used in the paper is shown in Table 2 in Section A.

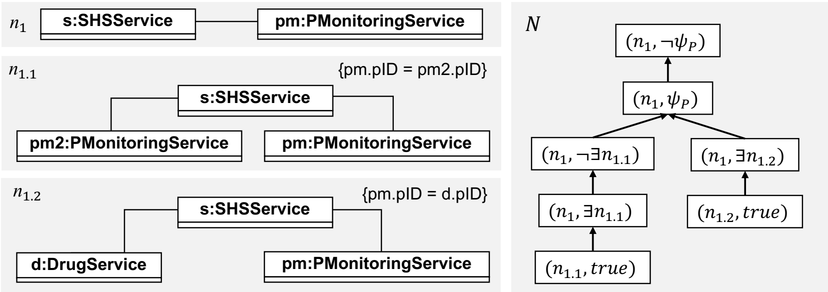

Fig. 2: Patterns for the SHS (left) and the GDN N for the query $(n, \neg\psi_P)$.

2.1 Formal Representation of Models and Queries

An RTM is typically represented as a graph, where system entities are captured by vertices, information about the entities by attributes, and relationships between entities by edges [25, 14, 24]. In this paper, for the formal representation of RTMs, we rely on the well-known *typed graphs* [20], *i.e.*, graphs *typed* over a *type graph* which defines types of vertices, edges, and valid structures for typed graphs.

Definition 1 ((typed) graph, (typed) graph morphism, type graph). *A* graph $G = (G^V, G^E, s^G, t^G)$ *consists of a set of vertices* G^V*, a set of edges* G^E*, a source function* $s^G : G^E \to G^V$*, and a target function* $t^G : G^E \to G^V$*. Given two graphs* $G = (G^V, G^E, s^G, t^G)$ *and* $K = (K^V, K^E, s^K, t^K)$*, a* graph morphism $f : G \to K$ *is a pair of mappings* $f^V : G^V \to K^V, f^E : G^E \to K^E$ *such that* $f^V \circ s^G = s^K \circ f^E$ *and* $f^V \circ t^G = t^K \circ f^E$*. A graph morphism* $f : G \to K$ *is a* monomorphism*, denoted by* $\hookrightarrow$*, if* f^V *and* f^E *are injective. A* type graph *is a distinguished graph* $TG = (TG^V, TG^E, s^{TG}, t^{TG})$*. A tuple* $(G, type)$ *consisting of a graph* G *and a graph morphism* $type : G \to TG$ *is called a* typed graph*. Given two typed graphs* $G^T = (G, type)$ *and* $K^T = (K, type')$*, a* typed graph morphism $f : G^T \to K^T$ *is a graph morphism* $f' : G \to K$ *such that* $type' \circ f' = type$*.*

Type graphs can be extended to support the well-known concepts of *inheritance* and *multiplicities* from the object-oriented paradigm [53]. Moreover, typed graphs can be extended by vertex and edge *attributes*, each associated with a data type, *i.e.*, a character string, an integer, a real number, or a boolean, to obtain *typed attributed graphs* [20]. Attribute *assignments* assign data-type-compatible values to attributes, and attribute *constraints*, *i.e.*, a boolean expression over attribute values, restrict the possible assignments. Our contributions rely on such graphs, defined in detail in our prior work [50]; to avoid the complication of presentation, here we omit these extensions from our definitions.

The metamodel in Fig. 1 may be seen as an informal representation of the type graph of the SHS, where only vertices have attributes. Correspondingly, the RTM G_7 in Fig. 3 is an informal representation of a typed attributed graph. We henceforth refer to typed attributed graphs simply as graphs or *patterns*. The RTM G_7 contains assignments, which assign values to attributes, *e.g.*, $\texttt{pm}_1.pID$

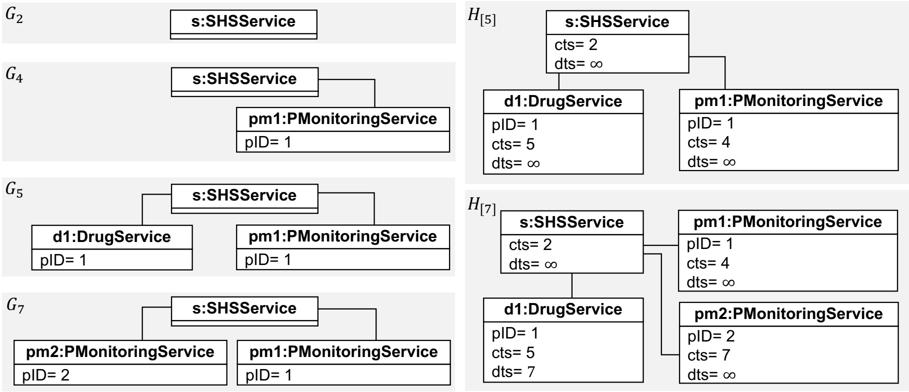

Fig. 3: Snapshots as RTMs (G_*) and traces as RTM$^{\mathrm{H}}$ instances ($H_{[*]}$).

= 1. The representation of the textual statements in property P of the running example by patterns is illustrated in Fig. 2: The invocation of a sensor service is captured in patterns n_1 and $n_{1.1}$, and the invocation of a drug service is captured in $n_{1.2}$; constraints are illustrated between braces, *e.g.*, $n_{1.1}$ requires that the values for *pID* of `pm` and `pm2` are equal; vertices with the same label refer to the same vertex in the queried RTM.

We assume that the system is instrumented to generate (instantaneous) events upon changes to its state, and identify the system execution with a possibly infinite sequence of such events. The system has a clock whose time domain is the set of non-negative real numbers $\mathbb{R}_0^+$, and uses the clock to timestamp events. We refer to an element of the time domain as a *time point*. Intuitively, an (execution) *trace* h_τ of a system with respect to an event at time point τ is the sequence of all observed events in the execution from its beginning, *i.e.*, time point 0, up to and including τ. For brevity, we group all changes with the same time point in one event. However, we require that no event groups an infinite amount of changes, thereby ruling out Zeno behaviors—in the use-cases of interest, all traces will eventually terminate and differences between measurements cannot become infinitely small. We denote the time point at position i of h_τ by τ_i, with $i \in \mathbb{N}^+$.

For a model-based representation of a trace h_τ, we rely on a *Runtime Model with History* (RTM$^{\mathrm{H}}$) [49]. An RTM$^{\mathrm{H}}$ H is a distinguished RTM where the following conditions hold. All vertices in H have a distinguished *creation timestamp cts* and a *deletion timestamp dts* to which a value is assigned—therefore in Fig. 1, all vertices inherit from the `MonitorableEntity`.[3] When a vertex is created, the time point of creation is assigned to *cts* and the value ∞ is assigned to the *dts*; the *dts* value changes when the vertex is deleted in the modeled system. As a vertex cannot have been deleted prior to its creation or deleted simultaneously to its creation, the value of *dts*, if not ∞, has to be larger than the value of *cts*.

[3] If tracking changes to attribute values or edges in an RTM is of importance, those can be modeled as vertices, which is a customary modeling technique, *e.g.*, [36].

An h_τ can be transformed to an RTM$^{\mathrm{H}}$ H based on a mapping $\mathscr{E}$ from the set of all possible events to corresponding graph modifications [48]; to capture the period covered by H in this case, we denote it by $H_{[\tau]}$. Each trace *continuation* $h_{\tau'}$ that is yielded by an event at time point τ' with $\tau' > \tau$ can be similarly transformed to a $H_{[\tau']}$ by applying the changes in the event at τ' to $H_{[\tau]}$; we refer to $H_{[\tau']}$ as a new *version* of $H_{[\tau]}$. This process generates a trace of RTMs $h^H_{\tau'}$, called an *RTM^H-trace*, which mirrors h'_τ; we refer to members of $h^H_{\tau'}$ as *instances* of the RTM$^{\mathrm{H}}$. Formally, an $H_{[\tau]}$ is a compact representation of a *timed graph sequence* [26], *i.e.*, a sequence of timestamped graphs where additions and deletions between two consecutive graphs are represented by morphisms. As an example of an RTM$^{\mathrm{H}}$, see $H_{[5]}$ in Fig. 3 which contains all changes in events up to time point 5; $H_{[5]}$ represents the timed graph sequence $G_2G_4G_5$ (left in Fig. 3; morphisms are omitted). A new event at time point 7 which contains the deletion of `d1`, and the addition of `pm2` is transformed into $H_{[7]}$; this RTM represents the sequence $G_2G_4G_5G_7$. If τ in h_τ, h^H_τ, or $H_{[\tau]}$ is irrelevant, we omit it.

2.2 Metric Temporal Graph Logic

For the specification and analysis of temporal properties in temporal queries, InTempo relies on the *Metric Temporal Graph Logic* (MTGL) [50, 26]. MTGL builds on *Nested Graph Conditions* (NGCs) [27] and *Metric Temporal Logic* (MTL) [35] to enable the formulation of *Metric Temporal Graph Conditions* (MTGCs). The language of NGCs can formulate requirements that are as expressive as first-order logic on graphs [18], as shown in [27, 44], and constitutes as such a natural formal foundation for pattern-based queries. As NGCs, MTGCs support *bindings*, *i.e.*, morphisms between patterns which bind elements in outer conditions to inner (nested) conditions, and are therefore able to track the evolution of a given binding in a sequence of graphs separately to other bindings.

In the following definition of MTGL, we focus on a subset of MTGL operators which contains the *metric*, *i.e.*, interval-based, temporal operators *until* (U_I, with I an interval in $\mathbb{R}^+_0$) and its dual *since* (S_I) from MTL. The existential quantifier features a binding between the patterns n and $\hat{n}$.

Definition 2 (metric temporal graph conditions). *Let n,$\hat{n}$ be patterns and $f : n \hookrightarrow \hat{n}$ a binding. Moreover, let I be an interval in $\mathbb{R}^+_0$. Then ψ is a* Metric Temporal Graph Condition *(MTGC) over n defined as follows.*

$$\psi_n ::= \text{true} \mid \neg\psi_n \mid \psi_n \wedge \psi_n \mid \exists(f : n \hookrightarrow \hat{n}, \psi_{\hat{n}}) \mid \psi_n \, \mathrm{U}_I \psi_n \mid \psi_n \, \mathrm{S}_I \psi_n$$

In the remainder, we abbreviate $\exists(f, \mathit{true})$ by $\exists\, f$ and, when the domain of f is clear from the context, $\exists(f : n \hookrightarrow \hat{n}, \phi_{\hat{n}})$ by $\exists(\hat{n}, \phi)$. Other abbreviations, *e.g.*, disjunction ($\vee$), *eventually* ($\Diamond_I$) can be defined as usual.

Based on the patterns in Fig. 2, property P from the running example can be reformulated into "given a binding for n_1 at a time point τ, at least one binding for $n_{1.2}$ is found at some time point $\tau' \in [\tau, \tau + 60]$, *i.e.*, at most 60 seconds later; in addition, at each time point $\tau'' \in [\tau, \tau')$ in between, no binding for $n_{1.1}$ is present." In MTGL, this property is captured by the MTGC $\psi_P \coloneqq \neg\exists\,(n_1 \hookrightarrow$

$n_{1.1}, true)\, \mathrm{U}_{[0,60]}\, \exists\, (n_1 \hookrightarrow n_{1.2}, true)$, or, abbreviated, $\neg\exists n_{1.1}\, \mathrm{U}_{[0,60]}\, \exists n_{1.2}$. The system is assumed to track time in seconds; vertices `s` and `pm` from n_1 are bound in the patterns $n_{1.1}$ and $n_{1.2}$, *i.e.*, all patterns refer to the same `s` and `pm`.

MTGL reasons over (finite) timed graph sequences. However, MTGCs can also be equivalently checked over a graph with history [26], which here corresponds to an RTM$^{\mathrm{H}}$. In the following, we define the semantics of the satisfaction relation of MTGL based on an RTM$^{\mathrm{H}}$.

Definition 3 (satisfaction of metric temporal graph conditions over an RTM). *Let H be an RTM^H, n a pattern, and $m : n \hookrightarrow H$ a binding. Moreover, let τ be a time point in $\mathbb{R}_0^+$ and ψ be an MTGC over n. Then m in H satisfies ψ at τ, written $(H, m, \tau) \models \psi$, if $max_{e \in E} e.\mathrm{cts} \leq \tau < min_{e \in E} e.\mathrm{dts}$, with E the vertices of m, and one of the following cases applies.*

- $\psi = \mathrm{true}$.
- $\psi = \neg\chi$ *and* $(H, m, \tau) \not\models \chi$.
- $\psi = \chi \wedge \omega$, $(H, m, \tau) \models \chi$, *and* $(H, m, \tau) \models \omega$.
- $\psi = \exists(f : n \hookrightarrow \hat{n}, \chi)$ *and there exists* $\hat{m} : \hat{n} \hookrightarrow H$ *such that* $\hat{m} \circ f = m$ *and* $(H, \hat{m}, \tau) \models \chi$.
- $\psi = \chi\, \mathrm{U}_I \omega$ *and there exists* τ' *with* $\tau' - \tau \in I$ *such that* $(H, m, \tau') \models \omega$ *and for all* $\tau'' \in [\tau, \tau')$ $(H, m, \tau'') \models \chi$.
- $\psi = \chi\, \mathrm{S}_I \omega$ *and there exists* τ' *with* $\tau - \tau' \in I$ *such that* $(H, m, \tau') \models \omega$ *and for all* $\tau'' \in (\tau', \tau]$ $(H, m, \tau'') \models \chi$.

Intuitively, a binding m for n in the RTM H satisfies the MTGC $\exists(f : n \hookrightarrow \hat{n}, \chi)$ at time point τ if (i) all elements of m are already created but not yet deleted at τ, and (ii) there exists a binding $\hat{m}$ for $\hat{n}$ in H such that $\hat{m}$ is *compatible* with m, *i.e.*, respects the binding between the two patterns captured in $n \hookrightarrow \hat{n}$, and $\hat{m}$ satisfies the MTGC χ at τ. The intuition behind *true*, negation, conjunction, *until*, and *since* is the usual.

2.3 InTempo: Query Language and Overview of Operation

InTempo introduces a query language, henceforth referred to as $\mathcal{L}$, which has two distinguishing features: it enables the formulation of ordering and temporal constraints in MTGL, *i.e.*, as an MTGC, thereby enabling formal precision in checking whether matches satisfy those constraints; it computes the period for which a match satisfies an MTGC, thereby enabling practical query evaluations, as the query does not have to be evaluated for each time point of interest. We summarize core concepts of graph queries and $\mathcal{L}$ below.

In its plainest form, a *graph query* is characterized by a pattern n. A *match* for this query is a binding from n to a queried graph which preserves structure and type. $\mathcal{L}$ allows for the specification of *temporal graph queries*, *i.e.*, queries of the form (n, ψ) with ψ an MTGC over n, whereby matches for n in an RTM$^{\mathrm{H}}$ H need to satisfy the temporal requirement captured in ψ. Based on the running example, the query $(n_1, \neg\psi_P)$, searches H for matches for n_1, *i.e.*, sensor services, which falsify ψ_P.

Vertices in H have lifespans, defined by their *cts* and *dts*. Similarly, a match m in H is valid only if there is a non-empty interval $\lambda^m = \cap_{e \in E}[e.cts, e.dts)$, with E the vertices of m, called the *lifespan of a match*. According to its definition, the values of regular attributes in H cannot change and, hence, cannot affect λ^m. In the special case where the pattern of a query is the empty graph $\varnothing$, an (empty) match m is always found with $\lambda^m = \mathbb{R}$. Temporal logics that reason over intervals, such as MTGL, are capable of deciding the truth value of a property for the entire time domain; in InTempo, the set of time points satisfying a property is called the *satisfaction span* and defined as $\mathcal{Y}(m, \psi) = \{\tau \mid \tau \in \mathbb{R} \wedge (H, m, \tau) \models \psi\}$ with ψ an MTGC. The *temporal validity* $\mathcal{V}(m, \psi)$ is equal to $\lambda^m \cap \mathcal{Y}(m, \psi)$ and defined as the period for which m exists in H *and* satisfies ψ.

The following computation, called the *satisfaction computation* $\mathcal{Z}$ of m for ψ, soundly computes $\mathcal{Y}$, as shown in [49]. The computation relies on interval operations defined as usual [see 41]: Let k, z be intervals; then $k \oplus z = [\ell(k) + \ell(z), r(k) + r(z)]$, $k \ominus z = [\ell(k) - r(z), r(k) - \ell(z)]$ with $\ell(k)$ and $r(k)$ the left and right end-point of k, respectively. We denote the unions $\ell(k) \cup k$ by ${}^{+}k$, and $k \cup r(k)$ by k^{+}; when $r(k) = \infty$, $k^{+} = k$. The interval k is *overlapping* z when $k \cap z \neq \emptyset$ and *adjacent* to z when $k \cap z = \emptyset$ but $k \cup z$ is an interval.

Definition 4 (satisfaction computation $\mathcal{Z}$). *Let $n, \hat{n}$ be patterns and ψ, χ, ω be MTGCs. Moreover, let m be a match for n in an RTM H, and $\hat{M}$ a set of matches for $\hat{n}$ that are* compatible *with the (enclosing) match m. The* satisfaction computation *$\mathcal{Z}(m, \psi)$ is recursively defined as follows.*

$$\mathcal{Z}(m, \text{true}) = \mathbb{R} \tag{1}$$

$$\mathcal{Z}(m, \neg\chi) = \mathbb{R} \setminus \mathcal{Z}(m, \chi) \tag{2}$$

$$\mathcal{Z}(m, \chi \wedge \omega) = \mathcal{Z}(m, \chi) \cap \mathcal{Z}(m, \omega) \tag{3}$$

$$\mathcal{Z}(m, \exists(\hat{n}, \chi)) = \bigcup_{\hat{m} \in \hat{M}} \lambda^{\hat{m}} \cap \mathcal{Z}(\hat{m}, \chi) \tag{4}$$

$$\mathcal{Z}(m, \chi \mathrm{U}_I \omega) = \begin{cases} \displaystyle\bigcup_{i \in \mathcal{Z}(m,\omega),\, j \in J_i} j \cap \big((j^{+} \cap i) \ominus I\big) & \textit{if } 0 \notin I \\ \displaystyle\bigcup_{i \in \mathcal{Z}(m,\omega)} i \cup \bigcup_{j \in J_i} j \cap \big((j^{+} \cap i) \ominus I\big) & \textit{if } 0 \in I \end{cases} \tag{5}$$

$$\mathcal{Z}(m, \chi \mathrm{S}_I \omega) = \begin{cases} \displaystyle\bigcup_{i \in \mathcal{Z}(m,\omega),\, j \in J_i} j \cap \big(({}^{+}j \cap i) \oplus I\big) & \textit{if } 0 \notin I \\ \displaystyle\bigcup_{i \in \mathcal{Z}(m,\omega)} i \cup \bigcup_{j \in J_i} j \cap \big(({}^{+}j \cap i) \oplus I\big) & \textit{if } 0 \in I \end{cases} \tag{6}$$

with J_i the set of all intervals in $\mathcal{Z}(m, \chi)$ that are either overlapping or adjacent to some $i \in \mathcal{Z}(m, \omega)$.

The intuition behind the equations for *true*, negation, and conjunction is clear. Regarding *exists*, the satisfaction span is the union of the temporal validity of all matches $\hat{m}$ for $\hat{n}$ which are compatible with m. Regarding *until*, if $0 \notin I$, the satisfaction includes every time point τ in the intersection of some $i' \in Z(m, \omega)$ with a $j' \in \mathcal{Z}(m, \chi)$ for which a time point $\tau' \in i'$ occurs within I. Furthermore, j'

needs to overlap i', *e.g.*, $j' = [1, 3]$, $i' = [2, 4]$ or be adjacent to i', *e.g.*, $j' = [1, 2)$, $i' = [2, 4]$. If j' and i' are adjacent, during the computation j becomes right-closed to ensure that their intersection produces a non-empty set. If $0 \in I$, then, according to Definition 3, it may be that j' is empty, *i.e.*, does not exist, and *until* is satisfied by every $i' \in \mathcal{Z}(m, \omega)$. Therefore, the computation includes every i' and remains unchanged otherwise. The intuition behind *since* is analogous.

The intersection of two intervals is always an interval, whereas the union of two intervals may result in disjoint sets. Hence, technically $\mathcal{Z}$ and $\mathcal{V}$ are *interval sets* which may contain disjoint or empty intervals.

We define below the answer set $\mathcal{T}$ for a query in $\mathcal{L}$.

Definition 5 (query answer set $\mathcal{T}$). *Given a pattern n, an MTGC ψ, and an RTM^H H, the* answer set *$\mathcal{T}$ of a query in $\mathcal{L}$ over H is given by:*

$$\mathcal{T}(H) = \{(m, \mathcal{V}(m, \psi)) | m \text{ is a match for } n \wedge \mathcal{V}(m, \psi) \neq \emptyset\}$$

Regarding the operation of InTempo (see Fig. 1), the approach expects a metamodel, a set of queries in $\mathcal{L}$, a mapping $\mathscr{E}$ from events to modifications, and an event trace h_τ as input—see definitions earlier. InTempo operationalizes queries (see Section 5). For each event events in h_τ, InTempo performs the corresponding changes to an RTM$^{\mathrm{H}}$ and, after each change, evaluates the queries. Pruning may follow, which triggers another query evaluation to update stored matches. Finally, InTempo returns the answer set $\mathcal{T}$ or, for RM, performs the check described in Section 1 and essentially returns matches in the effective answer set $\mathcal{T}^e$ (see Section 5). In our implementation of InTempo, the metamodel, the queries, and the mapping are defined based on model-based technologies [48].

We present an example that demonstrates that $\mathcal{T}$ may contain imprecise answers in the context of an incomplete trace.

Example 1 (imprecision over incomplete trace). Evaluated over $H_{[7]}$ in Fig. 3, the query $(n_1, \neg\psi_P)$ returns an answer set $\mathcal{T}(H_{[7]})$ which contains a pair $(m_2, [7, \infty))$; m_2 is a match for n_1 involving the vertex `pm2`, and $[7, \infty)$ is the temporal validity $\mathcal{V}$ which states that m_2 falsifies ψ_P from time point 7 onward. $\mathcal{V}$ is the result of the intersection of $\lambda^{m_2} = [7, \infty)$ with $\mathcal{Z}(m_2, \neg\psi_P) = \mathbb{R}$. The satisfaction span $\mathcal{Z}$ is computed according to Definition 4—see Table 1 for details.

This computation is definite only if $H_{[7]}$ is the last instance in an RTM$^{\mathrm{H}}$-trace; if the trace is incomplete, and it is to be continued by a new $H_{[\tau]}$ with $\tau \leq 67$, the match m_2 may still satisfy ψ_P, as there is still time for a `DrugService` to be created timely, *i.e.*, a match for the pattern $n_{1.2}$, which is compatible with m_2, to be found—assuming that until then there would be no match for $n_{1.1}$.

3 Definite Semantics for Metric Temporal Graph Logic

This section presents our contribution to MTGL. Specifically, we introduce a new semantics, called *definite*, which only returns answers if they are definite, *i.e.*, no future change to the RTM$^{\mathrm{H}}$ will affect them. Similarly to temporal logics which account for RM over incomplete traces [8, 21], the definite semantics is *three-valued*, as they return the value *unknown* when the result of the satisfaction check is not definite. We show the soundness of the definite semantics in Theorem 1 based on the regular semantics in Definition 3. Moreover, we show that for a certain period the definite and the regular semantics are equivalent (Theorem 2); this equivalence enables our contribution in Section 5, *i.e.*, it allows InTempo to return definite answers efficiently. Finally, we demonstrate an intrinsic limitation of the definite semantics: we show that for unsatisfiable properties, the semantics may return decisions with a delay, compared to the earliest time point on which the decisions could have been returned. We compute the maximum possible magnitude of the delay (Corollary 2).

We begin with the definition of the definite semantics. In the context of an $\mathrm{RTM}^{\mathrm{H}}$ $H_{[c]}$, a satisfaction decision for time point $\tau \in [0,c]$ is *definite* if the decision for τ remains the same in all possible future versions of $H_{[c]}$. We obtain the definite satisfaction span by adjusting the satisfaction relation of MTGL from Definition 3 to this notion of definiteness. Moreover, we obtain the definite falsification by negating the statements in the cases of the definite satisfaction. We present the adjusted satisfaction relation, called *definite satisfaction relation*, and the *definite falsification relation* over an $\mathrm{RTM}^{\mathrm{H}}$ below.

Definition 6 (definite satisfaction and definite falsification of metric temporal graph conditions over an $\mathbf{RTM^H}$). *Let $H_{[c]}$ be a RTM^H, n a pattern, and $m : n \hookrightarrow H_{[c]}$ a match. Moreover, let $\tau \in \mathbb{R}$ be a time point and ψ be an MTGC over n. Then the* definite satisfaction relation $\models^d$ *and* definite falsification relation $\models^d_F$ *are defined via mutual recursion as follows. The match* m definitely satisfies *ψ at τ, written $(H_{[c]}, m, \tau) \models^d \psi$, iff $\tau \in \lambda^m \cap [0, c]$, or m is the empty match, and one of the following cases applies.*

- $\psi = \text{true}$.
- $\psi = \neg\chi$ *and* $(H_{[c]}, m, \tau) \models^d_F \chi$.
- $\psi = \chi \wedge \omega$, $(H_{[c]}, m, \tau) \models^d \chi$, *and* $(H_{[c]}, m, \tau) \models^d \omega$.
- $\psi = \exists(f : n \hookrightarrow \hat{n}, \chi)$ *and there exists* $\hat{m} : \hat{n} \hookrightarrow H_{[c]}$ *such that* $\hat{m} \circ f = m$ *and* $(H_{[c]}, \hat{m}, \tau) \models^d \chi$.
- $\psi = \chi\, \mathrm{U}_I \omega$ *and there exists* τ' *with* $\tau' - \tau \in I$ *such that* $(H_{[c]}, m, \tau') \models^d \omega$ *and for all* $\tau'' \in [\tau, \tau')$ $(H_{[c]}, m, \tau'') \models^d \chi$.
- $\psi = \chi\, \mathrm{S}_I \omega$ *and there exists* τ' *with* $\tau - \tau' \in I$ *such that* $(H_{[c]}, m, \tau') \models^d \omega$ *and for all* $\tau'' \in (\tau', \tau]$ $(H_{[c]}, m, \tau'') \models^d \chi$.

The definite falsification relation is based on a logical negation of the statements in the cases of the definite satisfaction relation. The match m definitely falsifies *ψ at τ, written $(H_{[c]}, m, \tau) \models^d_F \psi$, iff $\tau \in \lambda^m \cap [0, c]$, or m is the empty match, and one of the following cases applies.*

- $\psi = \neg\chi$ *and* $(H_{[c]}, m, \tau) \models^d \chi$.
- $\psi = \chi \wedge \omega$ *and* $(H_{[c]}, m, \tau) \models^d_F \chi$ *or* $(H_{[c]}, m, \tau) \models^d_F \omega$.
- $\psi = \exists(f : n \hookrightarrow \hat{n}, \chi)$ *and either there does not exist an* $\hat{m} : \hat{n} \hookrightarrow H_{[c]}$ *such that* $\hat{m} \circ f = m$, *or there exists* $\hat{m}$ *and* $(H_{[c]}, \hat{m}, \tau) \models^d_F \chi$.

- $\psi = \chi\, \mathrm{U}_I \omega$ *and for all* τ' *with* $\tau' - \tau \in I$ $(H_{[c]}, m, \tau') \models^d_F \omega$ *or there exists* $\tau'' \in [\tau, \tau')$ *such that* $(H_{[c]}, m, \tau'') \models^d_F \chi$.
- $\psi = \chi\, \mathrm{S}_I \omega$ *and for all* τ' *with* $\tau - \tau' \in I$ $(H_{[c]}, m, \tau') \models^d_F \omega$ *or there exists* $\tau'' \in (\tau', \tau]$, $(H_{[c]}, m, \tau'') \models^d_F \chi$.

In comparison to $\models$, $\models^d$ confines the lifespans of matches and the satisfaction of *exists* to the period that has been observed, *i.e.*, $[0, c]$. Moreover, $\models^d$ relies on $\models^d_F$ for the satisfaction of a negation. Similarly to $\models^d$, $\models^d_F$ confines the decisions for matches to $[0, c]$, and relies on $\models^d$ for the falsification of negation. The match m never falsifies *true*. We note that $\models^d_F$ and $\not\models^d$ are not equivalent; $\not\models^d$ returns true for time points that do not definitely satisfy the operator, *i.e.*, points that falsify it but also points for which a definite decision cannot yet be made.

The following theorem shows the soundness of the *definite relations* $\models^d$ and $\models^d_F$ by relating them to the regular satisfaction relation $\models$ from Definition 3 and its negation $\not\models$. The theorem refers to observed prefixes of a possibly infinite RTM$^{\mathrm{H}}$-trace h^H and their possible continuations; an RTM$^{\mathrm{H}}$ $H_{[\tau_i]}$ in h^H is associated with the τ of the event with index $i \in \mathbb{N}^+$ in the execution h—see Section 2.1. The theorem states that a *definite decision*, *i.e.*, a decision made by either $\models^d$ or $\models^d_F$, for a certain time point τ over an $H_{[\tau_i]}$ in h^H implies that the same decision is made by $\models$ (or $\not\models$) for τ over $H_{[\tau_i]}$; moreover, $\models$ makes the same decision for τ over all possible future versions of $H_{[\tau_i]}$ in h^H.

Theorem 1 (definite relations imply satisfaction relation over trace). *Let* ψ *be an MTGC over a pattern* n. *Moreover, let* $h^H_{\tau_{\mathcal{D}}}$ *be RTM*^{H}*-trace, with* $\mathcal{D} \in \mathbb{N}^+$. *For all* $i \in [1, \mathcal{D}] \cap \mathbb{N}^+$, *if* m *is a match for* n *in* $H_{[\tau_i]}$ *and* $\tau \in [0, \tau_i]$, *then for all* $k \in [i, \mathcal{D}] \cap \mathbb{N}^+$, *(i) if* $(H_{[\tau_i]}, m, \tau) \models^d \psi$, *then* $(H_{[\tau_k]}, m, \tau) \models \psi$, *and (ii) if* $(H_{[\tau_i]}, m, \tau) \models^d_F \psi$, *then* $(H_{[\tau_k]}, m, \tau) \not\models \psi$.

Proof (idea). By mutual structural induction over ψ. The implication is shown to hold for each MTGL operator. See Section B.1 for the complete proof. □

In the following, we discuss the second important result of this section, *i.e.*, the equivalence of the definite and regular semantics.

The satisfaction decision for future temporal operators at time point τ may depend on a $\tau' > \tau$. The upper bound of the distance between τ' and τ is given by the *non-definiteness window*, defined below.

Definition 7 (non-definiteness window w). *Given an MTGC* ψ, *the* non-definiteness window w, i.e., *the period for which a satisfaction decision for* ψ *at a time point* τ *may be non-definite, is defined as follows.*

$$w(\psi) = \begin{cases} r(I) + \max\left(w(\chi), w(\omega)\right) & \textit{if } \psi = \chi \mathrm{U}_I\, \omega \\ \max\left(w(\chi), w(\omega)\right) & \textit{if } \psi = \chi \mathrm{S}_I\, \omega \\ \max\left(w(\chi), w(\omega)\right) & \textit{if } \psi = \chi \wedge \omega \\ w(\chi) & \textit{if } \psi = \neg\chi \\ w(\chi) & \textit{if } \psi = \exists(n, \chi) \\ 0 & \textit{if } \psi = \mathrm{true} \end{cases} \tag{7}$$

As usual in (online) RM, we assume that $w \neq \infty$, *i.e.*, MTGCs contain no unbounded future operators which may render a property non-monitorable [42].

Based on w, we present a variation of Theorem 1 which states that, given an $H_{[\tau_i]}$, if $\tau \in [0, \tau_i - w]$, with i an index in a RTM$^{\mathrm{H}}$-trace, then definite decisions made by either the definite satisfaction relation $\models^d$ or definite falsification relation $\models^d_F$ are equivalent to the decisions of the satisfaction relation $\models$. If $w \neq 0$, in order for $[0, \tau_i - w]$ to be a valid interval, it is implicitly required that $\tau_i \geq w$, *i.e.*, $H_{[\tau_i]}$ covers a period that is larger than the non-definiteness window.

Theorem 2 (definite relations are equivalent to satisfaction relation over certain period of trace). *Let ψ be an MTGC over a pattern n and w the non-definiteness window of ψ. Moreover, let $h^H_{\tau_{\mathcal{D}}}$ be an RTM^H-trace, with $\mathcal{D} \in \mathbb{N}^+$. For all $i \in [1, \mathcal{D}] \cap \mathbb{N}^+$, if m is a match for n in $H_{[\tau_i]}$ and $\tau \in [0, \tau_i - w]$, then for all $k \in [i, \mathcal{D}] \cap \mathbb{N}^+$, (i) $(H_{[\tau_i]}, m, \tau) \models^d \psi$ iff $(H_{[\tau_k]}, m, \tau) \models \psi$, and (ii) $(H_{[\tau_i]}, m, \tau) \models^d_F \psi$ iff $(H_{[\tau_k]}, m, \tau) \not\models \psi$.*

Proof (idea). By mutual structural induction over ψ. The equivalence is shown to hold for each MTGL operator. See Section B.2 for the complete proof. □

Theorem 2 enables our contribution to change-driven evaluation in Section 5.

Finally, we present the third important result of the section, *i.e.*, the limitation of the semantics. The following corollary states that all time points for which a definite decision cannot be made belong to a certain period in the observed trace.

Corollary 1 (period in trace with non-definite decisions). *Let ψ be an MTGC, w be the non-definiteness window of ψ, $H_{[\tau_i]}$ be an RTM^H instance associated with the time point τ_i, m be a match for a pattern n, and τ a time point in $[0, \tau_i]$. If $(H_{[\tau_i]}, m, \tau) \not\models^d \psi$ and $(H_{[\tau_i]}, m, \tau) \not\models^d_F \psi$, then $\tau \in (\tau_i - w, \tau_i]$.*

Proof (idea). Follows from Theorem 2—see Section B.3 for the complete proof. □

We demonstrate below that, in case an MTGC is unsatisfiable (or unfalsifiable), the definite relations may return an answer with a delay. The maximum possible delay depends on the non-definiteness window w from Definition 7.

Let $\models_{\mathbb{T}}$ and $\models_{F,\mathbb{T}}$ be respectively a satisfaction and falsification relation for MTGL that reflect the *timeliest knowledge*: Given a match m, an MTGC ψ, an RTM$^{\mathrm{H}}$ instance $H_{[\tau_i]}$ from a sequence of instances, and a time point $\tau \in [0, \tau_i]$, $(H_{[\tau_i]}, m, \tau) \models_{\mathbb{T}} \psi$ if $(H_{[\tau_i]}, m, \tau) \models \psi$ and there exists no possible successor of $H_{[\tau_i]}$ in the sequence that could falsify ψ at τ; analogously, $(H_{[\tau_i]}, m, \tau) \models_{F,\mathbb{T}} \psi$ if $(H_{[\tau_i]}, m, \tau) \not\models \psi$ and there exists no possible successor of $H_{[\tau_i]}$ that could satisfy ψ at τ. These timeliest relations can only make decisions for m over the observed trace, as m may not exist in the parts covered by successors of $H_{[\tau_i]}$, *i.e.*, in time points larger than τ_i.

Given a sequence of RTM$^{\mathrm{H}}$ instances h^H with $H_{[\tau_i]}$ an instance in h^H, let $H_{[\tau_k]}$ be the first successor of $H_{[\tau_i]}$ in h^H for which $\tau_k \geq \tau_i + w$. The following corollary states that, contrary to $\models_{\mathbb{T}}$ and $\models_{F,\mathbb{T}}$, the definite relations may have to wait for $H_{[\tau_k]}$ to be able to make a definite decision for $\tau \in (\tau_i - w, \tau_i]$.

Corollary 2 (maximum possible delay before definite decision). *Let ψ be an MTGC, w be the non-definiteness window of ψ, m be a match for a pattern n, and $H_{[\tau_i]}$ be an RTM^H instance from a sequence of RTM^H instances $h^H_{\tau_\mathcal{D}}$ with $i \in [1, \mathcal{D}] \cap \mathbb{N}^+$. Moreover, let $\tau \in (\tau_i - w, \tau_i]$ and k be the smallest index in $[i, \mathcal{D}] \cap \mathbb{N}^+$ such that $\tau_k \geq \tau_i + w$. If $(H_{[\tau_i]}, m, \tau) \not\models^d \psi$ and $(H_{[\tau_i]}, m, \tau) \not\models^d_F \psi$, then a definite decision for τ can be made over $H_{[\tau_k]}$.*

Proof. Follows from Corollary 1. □

Thus, compared to $\models_\mathbb{T}$ and $\models_{F,\mathbb{T}}$, the definite relations may make a decision for $\tau \in (\tau_i - w, \tau_i]$ with a delay of at most $(\tau_k - \tau_i)$ time points.

Example 2. (delay in definite decision) Let $\psi_c := \lozenge_{[0,1]}(\neg\exists\, n_1 \wedge \exists\, n_1)$. Consider an RTM$^\mathrm{H}$-trace comprising two RTM$^\mathrm{H}$ instances: $H_{[7]}$ in Fig. 3 and a hypothetical $H_{[9]}$ which is yielded by an unrelated change and all elements from $H_{[7]}$ are unchanged. Therefore, a match m_1 exists in both instances. The check $(H_{[7]}, m_1, 7) \models_{F,\mathbb{T}} \psi_c$ returns true, as $(H_{[7]}, m_1, 7) \not\models \psi_c$ and there is no possible successor of $H_{[7]}$ that could satisfy ψ_c; on the other hand, $(H_{[7]}, m_1, 7) \models^d_F \psi_c$ makes no decision, as according to its definition, the relation waits first for a duration of history that covers the timing constraint of *until* to be observed. The check $(H_{[9]}, m_1, 7) \models^d_F \psi_c$ returns true, as enough time has elapsed. Thus, compared to $\models_{F,\mathbb{T}}$, this decision has been made with a delay of two time points.

Avoiding this delay would require that the definite relations recognize whether an MTGC is satisfiable which is undecidable for NGCs and thus MTGCs. The delay is not observed with the running example, *i.e.*, $\psi_P := \neg\exists\, n_{1.1}\, \mathrm{U}_{[0,60]}\, \exists\, n_{1.2}$ or similar MTGCs, *e.g.*, $(\lozenge_{[0,2]}\exists\, n_{1.1}) \wedge (\lozenge_{[0,3]}\exists\, n_{1.2})$.

4 Computations and Answer Set for Definite Semantics

This section presents our contribution to the semantics of $\mathcal{L}$, the query language of InTempo. Specifically, we adjust the satisfaction computation presented in Definition 4 to the definite satisfaction relation ($\models^d$) from Definition 6. Moreover, we introduce the analogous concepts for the definite falsification relation ($\models^d_F$). Theorem 3 shows the soundness of the introduced computations. Based on these computations, we introduce a definite answer set for $\mathcal{L}$.

In the context of a temporal query (n, ψ) the *definite satisfaction span* related to a match m for n in $H_{[c]}$ is defined similarly to the satisfaction span $\mathcal{Y}$ in Section 2.3, *i.e.*, $\mathcal{Y}^d = \{\tau | \tau \in \mathbb{R} \wedge (H_{[c]}, m, \tau) \models^d \psi\}$. The *definite falsification span* is defined as $\mathcal{F}^d = \{\tau | \tau \in \mathbb{R} \wedge (H_{[c]}, m, \tau) \models^d_F \psi\}$. Any time point in the time domain not in $\mathcal{Y}^d$ or $\mathcal{F}$ belongs to the *unknown span* X. The sets $\mathcal{Y}^d, \mathcal{F}^d$, and X are disjoint. It also holds that $\mathbb{R} = \mathcal{Y}^d \uplus \mathcal{F}^d \uplus X$. The *definite satisfaction computation* $\mathcal{Z}^d$ and the *definite falsification computation* F^d for an MTGC are defined below.

Definition 8 (definite satisfaction computation $\mathcal{Z}^d$ and definite falsification computation F^d). *Let $n, \hat{n}$ be patterns and ψ, χ, ω be MTGCs. Moreover, let m be a match for n in an RTM^H H, and $\hat{M}$ a set of matches for $\hat{n}$ that are* compatible *with the (enclosing) match m. The* definite satisfaction computation *$\mathcal{Z}^d(m, \psi)$ and* definite falsification computation *$F^d(m, \psi)$ are defined via mutual recursion as follows.*

$$\mathcal{Z}^d(m, \text{true}) = \mathbb{R} \tag{8}$$

$$\mathcal{Z}^d(m, \neg\chi) = F^d(m, \chi) \tag{9}$$

$$\mathcal{Z}^d(m, \chi \wedge \omega) = \mathcal{Z}^d(m, \chi) \cap \mathcal{Z}^d(m, \omega) \tag{10}$$

$$\mathcal{Z}^d(m, \exists(\hat{n}, \chi)) = (-\infty, \tau] \cap \bigcup_{\hat{m} \in \hat{M}} \lambda^{\hat{m}} \cap \mathcal{Z}^d(\hat{m}, \chi) \tag{11}$$

$$\mathcal{Z}^d(m, \chi \mathrm{U}_I \omega) = \begin{cases} \bigcup\limits_{i \in \mathcal{Z}^d(m,\omega),\, j \in J_i^d} j \cap \big((j^+ \cap i) \ominus I\big) & \text{if } 0 \notin I \\ \bigcup\limits_{i \in \mathcal{Z}^d(m,\omega)} i \cup \bigcup\limits_{j \in J_i^d} j \cap \big((j^+ \cap i) \ominus I\big) & \text{if } 0 \in I \end{cases} \tag{12}$$

$$\mathcal{Z}^d(m, \chi \mathrm{S}_I \omega) = \begin{cases} \bigcup\limits_{i \in \mathcal{Z}^d(m,\omega),\, j \in J_i^d} j \cap \big(({}^+j \cap i) \oplus I\big) & \text{if } 0 \notin I \\ \bigcup\limits_{i \in \mathcal{Z}^d(m,\omega)} i \cup \bigcup\limits_{j \in J_i^d} j \cap \big(({}^+j \cap i) \oplus I\big) & \text{if } 0 \in I \end{cases} \tag{13}$$

with J_i^d the set of all intervals in $\mathcal{Z}^d(m, \chi)$ that are either overlapping or adjacent to some $i \in \mathcal{Z}^d(m, \omega)$.

Based on $\mathbb{R} = \mathcal{Y}^d \uplus \mathcal{F}^d \uplus X$, the definite falsification computation $F^d(m, \psi)$ can be generally defined as $F^d = \mathbb{R} \setminus (\mathcal{Z}^d \uplus X)$, which leads to the following equations.

$$F^d(m, \text{true}) = \emptyset \tag{14}$$

$$F^d(m, \neg\chi) = \mathcal{Z}^d(m, \chi) \tag{15}$$

$$F^d(m, \chi \wedge \omega) = F^d(m, \chi) \cup F^d(m, \omega) \tag{16}$$

$$F^d(m, \exists(\hat{n}, \chi)) = (-\infty, \tau] \cap \big(\mathbb{R} \setminus \mathcal{Z}^d(m, \exists(\hat{n}, \chi))\big) \tag{17}$$

$$F^d(m, \chi \mathrm{U}_I \omega) = \begin{cases} \mathbb{R} \setminus \left(\bigcup\limits_{i \in \mathcal{Z}^d(m,\omega) \uplus X(m,\omega),\, j \in J_i^d} j \cap \big((j^+ \cap i) \ominus I\big) \right) & \text{if } 0 \notin I \\ \mathbb{R} \setminus \left(\bigcup\limits_{i \in \mathcal{Z}^d(m,\omega) \uplus X(m,\omega)} i \cup \bigcup\limits_{j \in J_i^d} j \cap \big((j^+ \cap i) \ominus I\big) \right) & \text{if } 0 \in I \end{cases} \tag{18}$$

$$F^d(m, \chi \mathrm{S}_I \omega) = \begin{cases} \mathbb{R} \setminus \left(\bigcup\limits_{i \in \mathcal{Z}^d(m,\omega) \uplus X(m,\omega),\, j \in J_i^d} j \cap \big(({}^+j \cap i) \oplus I\big) \right) & \text{if } 0 \notin I \\ \mathbb{R} \setminus \left(\bigcup\limits_{i \in \mathcal{Z}^d(m,\omega) \uplus X(m,\omega)} i \cup \bigcup\limits_{j \in J_i^d} j \cap \big(({}^+j \cap i) \oplus I\big) \right) & \text{if } 0 \in I \end{cases} \tag{19}$$

where J_i^d *is the set of all intervals in* $\mathcal{Z}^d(m,\chi) \uplus X(m,\chi)$ *that are either overlapping or adjacent to some* $i \in \mathcal{Z}^d(m,\omega) \uplus X(m,\omega)$.

Regarding $\mathcal{Z}^d$, the equations for conjunction, *until*, and *since* have the same structure with their corresponding equations in Definition 4, but rely on $\mathcal{Z}^d$ instead of $\mathcal{Z}$. Analogously to $\models^d$, the computation for negation relies on F^d. The computation for *exists* confines its decisions to the period that has been observed.

Regarding F^d, a match m never falsifies *true*; analogously to $\models_F^d$, F^d relies on $\mathcal{Z}^d$ for the falsification of negation; the operator *exists* confines its computation to the observed period; the equations for *until* and *since* complement their respective definite satisfaction computations, whereby the definite satisfaction computation for their operands χ and ω instead of considering only time points that definitely satisfy χ and ω, *i.e.*, their satisfaction spans $\mathcal{Z}^d(m,\chi)$ and $\mathcal{Z}^d(m,\omega)$, considers time points *that do not definitely falsify* χ and ω, *i.e.*, $\mathcal{Z}^d(m,\chi) \uplus X(m,\chi)$ and $\mathcal{Z}^d(m,\omega) \uplus X(m,\omega)$.

The following theorem states that the set of time points in the definite satisfaction span $\mathcal{Y}^d$ and definite falsification span $\mathcal{F}^d$ are equal to the sets of time points obtained by the definite satisfaction computation $\mathcal{Z}^d$ and definite falsification computation F^d, respectively.

Theorem 3 (equality of definite spans and definite computations for satisfaction and falsification). *Given a match* m *in an RTM*^{H} $H_{[\tau]}$ *and an MTGC* ψ, *it holds that* $\mathcal{Y}^d(m,\psi) = \mathcal{Z}^d(m,\psi)$ *and* $\mathcal{F}^d(m,\psi) = F^d(m,\psi)$.

Proof (idea). The proof for $\mathcal{Z}^d$ proceeds by structural induction over ψ. The proof for F^d is based on the application of $F^d = \mathbb{R} \setminus (\mathcal{Z}^d \uplus X)$ for each MTGL operator. See Section B.4 for the complete proof. $\square$

Based on the definite computations, we now extend $\mathcal{L}$ with a notion of definite answers by adjusting the answer set $\mathcal{T}$ in Definition 5. To this end, we define the notion of *temporal invalidity* $\mathcal{IV}$ as the dual notion of temporal validity $\mathcal{V}$ from Section 2.3, *i.e.*, the intersection of the lifespan λ^m of a match m with the falsification span. Moreover, we define the *definite temporal validity* $\mathcal{V}^d$ as $\lambda^m \cap \mathcal{Z}^d$, and the *definite temporal invalidity* $\mathcal{IV}^d$ as $\lambda^m \cap F^d$.

Definition 9 (definite answer set $\mathcal{T}^d$). *Given a pattern* n, *an MTGC* ψ, *and an RTM*^{H} H, *the* definite answer set $\mathcal{T}^d$ *of a query in* $\mathcal{L}$ *over* H *is given by:*

$$\mathcal{T}^d(H) = \{(m, \mathcal{V}^d(m,\psi), \mathcal{IV}^d(m,\psi)) | m \textit{ is a match for } n \wedge (\mathcal{V}^d \neq \emptyset \vee \mathcal{IV}^d \neq \emptyset\}$$

Example 3 (precision of definite computations over incomplete trace). As in Example 1, the query $(n_1, \neg\psi_P)$ is evaluated over $H_{[7]}$. This time however, we obtain the definite answer set $\mathcal{T}^d(H_{[7]})$. The match m_2 for n_1, that involves the object `pm2`, is not contained in $\mathcal{T}^d$; m_2 is matched and its lifespan is computed to be $\lambda^{m_2} = [7,\infty)$ but no compatible match for $n_{1.2}$ is found; As shown in Table 1, $\mathcal{Z}^d(m_2,\psi_P) = (-\infty,-53]$ and $F^d(m_2,\psi_P) = \emptyset$. Therefore, both $\mathcal{V}^d$ and $\mathcal{IV}^d$ are empty, and the match is excluded from $\mathcal{T}^d$. Note that $\mathcal{T}^d(H_{[7]})$ contains

Table 1: Computations $\mathcal{Z}$, $\mathcal{Z}^d$, and F^d for two matches for $(n_1, \neg\psi_P)$ over $H_{[7]}$.

	m_1			m_2		
MTGC	$\mathcal{Z}$	$\mathcal{Z}^d$	F^d	$\mathcal{Z}$	$\mathcal{Z}^d$	F^d
true	$\mathbb{R}$	$\mathbb{R}$	$\emptyset$	$\mathbb{R}$	$\mathbb{R}$	$\emptyset$
$\exists n_{1.1}$	$\emptyset$	$\emptyset$	$(-\infty, 7]$	$\emptyset$	$\emptyset$	$(-\infty, 7]$
$\neg\exists n_{1.1}$	$\mathbb{R}$	$(-\infty, 7]$	$\emptyset$	$\mathbb{R}$	$(-\infty, 7]$	$\emptyset$
true	$\mathbb{R}$	$\mathbb{R}$	$\emptyset$	$\mathbb{R}$	$\mathbb{R}$	$\emptyset$
$\exists n_{1.2}$	$[5, 7)$	$[5, 7)$	$\{(-\infty, 5), [7, 7]\}$	$\emptyset$	$\emptyset$	$(-\infty, 7]$
ψ_P	$[-55, 7)$	$[-55, 7)$	$\{(-\infty, -55), [7, 7]\}$	$\emptyset$	$\emptyset$	$(-\infty, -53]$
$\neg\psi_P$	$\{(-\infty, -55), [7, \infty)\}$	$\{(-\infty, -55), [7, 7]\}$	$[-55, 7)$	$\mathbb{R}$	$(-\infty, -53]$	$\emptyset$

a match m_1 for n_1 that involves `pm1`, as its $\mathcal{V}^d$ is non-empty (see Table 1), *i.e.*, there are time points for which m_1 definitely falsifies $\neg\psi_P$, or definitely satisfies ψ_P. All computations in Table 1 are interval sets (see Section 2.3), however, for presentation purposes, singletons are displayed as intervals.

Let $H_{[67]}$ be an RTM$^{\mathrm{H}}$ that is yielded by an event at time point 67; the changes by this event do not affect vertices or nodes in $H_{[7]}$; m_2 would be returned by $\mathcal{T}^d$, paired with $\mathcal{V}^d = [7, 7]$, as there would be no future version of the RTM$^{\mathrm{H}}$ which could satisfy ψ_P at time point 7.

5 Keeping to Change-driven Evaluation

The operationalization of queries in InTempo (see also Fig. 1) is based on *Generalized Discrimination Networks* (GDNs) [28, 10]. Specifically, a query in $\mathcal{L}$ is decomposed into a suitable ordering, *i.e.*, a *network*, N of simple sub-queries. N is a tree where each node represents a query and each edge a dependency between queries—see Fig. 2 (right) for the GDN for ψ_P. N is executed bottom-up, *i.e.*, the execution starts with leaves and proceeds upward. The root of N computes the answer set $\mathcal{T}(H)$ of q. Each node in N stores intermediate matches paired with their $\mathcal{Z}$; therefore N is amenable to *change-driven* and incremental execution: changes to H are propagated through N, whose nodes only recompute their stored matches if the change is relevant to them or one of their dependencies. Moreover, InTempo offers a method to remove temporally irrelevant history from the RTM$^{\mathrm{H}}$, thereby rendering the query evaluation memory-efficient.

Based on these features, an extensive experimental evaluation of our implementation of InTempo showed efficient performance in the evaluation of temporal graph queries over considerably large models (approximately from 10K to 48M elements) [49]. InTempo also evaluated queries faster than the established RV tool MonPoly [6] as well as the RTM-based tool Hawk [24] in an RM application scenario. In the scenario, incomplete traces were handled by performing a check for each match which, based on the timing constraints of the property, postponed returning the match if future changes could affect it.

The definite answer set $\mathcal{T}^d$ from Definition 9 handles incomplete traces comprehensively, as it only includes matches and time points which no future change can affect. However, $\mathcal{T}^d$ relies on the definite MTGL semantics from Definition 6 which, contrary to the regular semantics from Definition 3, considers the time point on which a query is evaluated; consequently, adjusting N to compute the definite computations $\mathcal{Z}^d$ and F^d, and thus to return $\mathcal{T}^d$, would imply that every new version of $H_{[\tau]}$ would trigger a re-computation of all spans stored in N. Therefore, $\mathcal{T}^d$ is not amenable to change-driven evaluation.

Based on the intuition behind the check from above, we lastly present a new answer set, called *effective*, that contains definite results while relying on $\mathcal{T}$, which *is* amenable to change-driven evaluation. Specifically, based on the equivalence in Theorem 2, we show that $\mathcal{T}$ is equivalent to a subset of $\mathcal{T}^d$ if the $\mathcal{V}$ of matches in $\mathcal{T}$ is restricted to a period with definite decisions (see Corollary 1). This last contribution formalizes the intuition behind the check from above, and allows approaches like InTempo to maintain their efficiency while returning sound results. We define the effective answer set $\mathcal{T}^e$ for $\mathcal{L}$ based on $\mathcal{T}$ below.

Definition 10 (effective answer set $\mathcal{T}^e$). *Given a pattern n, an MTGC ψ with w the non-definiteness window of ψ, an RTM^H $H_{[\tau]}$, and an answer set $\mathcal{T}(H_{[\tau]})$ of a query in $\mathcal{L}$, the* effective answer set $\mathcal{T}^e(H_{[\tau]})$ *of the query is the set of all tuples $(m, \mathcal{V} \cap [0, \tau - w])$ such that (i) $(m, \mathcal{V}(m, \psi)) \in \mathcal{T}(H_{[\tau]})$ and (ii) $\mathcal{V}(m, \psi) \cap [0, \tau - w] \neq \emptyset$.*

The following theorem states that $\mathcal{T}^e$ is equal to a restricted version of $\mathcal{T}^d$ whose $\mathcal{V}^d$ excludes a period equal to w. We assume that the trace duration is larger than w and that the trace has more than one member.

Theorem 4 (equality of effective answer set and restricted definite temporal validity answer set over trace). *Let (n, ψ) be a query with ψ an MTGC, w be the non-definiteness window of ψ, and $h^H_{\tau_\mathcal{D}}$ be a RTM^H-trace with $\mathcal{D} \in [2, \infty] \cap \mathbb{N}^+$, and i be an index in $[k, \mathcal{D} - 1] \cap \mathbb{N}^+$ such that $\tau_k \geq w$. Moreover, let $\mathcal{T}^d_{\mathcal{V},r}(H_{[\tau_i]})$ be the* restricted definite temporal validity answer set *over $H_{[\tau_i]}$ which has been obtained from the definite answer set $\mathcal{T}^d$ but contains (i) only pairs of matches with their temporal validity $\mathcal{V}^d$, with $\mathcal{V}^d \neq \emptyset$ and (ii) $\mathcal{V}^d$ is intersected with $[0, \tau_i - w]$. Then, $\mathcal{T}^e(H_{[\tau_i]}) = \mathcal{T}^d_{\mathcal{V},r}(H_{[\tau_i]})$.*

Proof (idea). Based on the more general Theorem 2. See Section B.5 for the complete proof. □

Theorem 4 shows how InTempo returns definite results while using the change-driven evaluation for $\mathcal{T}$ described above. On the other hand, as $\mathcal{T}^d_{\mathcal{V},r}$ excludes F^d, obtaining F^d with $\mathcal{T}^e$ requires the evaluation of a separate query $(n, \neg\psi)$ in parallel to (n, ψ). Moreover, due to postponing returning answers that may be non-definite, $\mathcal{T}^e$ may return answers with a delay; although this is not observed in ψ_P from the running example, it may affect other properties, as demonstrated in Example 4. Hence, $\mathcal{T}^e$ is intended for application scenarios where this impact is either absent or acceptable.

Example 4 (Delay in detection). Let $\psi_D := (\neg\exists n_{1.1}) \wedge (\neg\Diamond_{[0,2]}\exists n_{1.2})$ be an MTGC and $(n_1, \neg\psi_D)$ a query in $\mathcal{L}$. Let $H_{[5]}$ be a hypothetical RTM$^{\mathrm{H}}$ that contains a match for n_1 and a match for $n_{1.1}$, whose lifespans are $[5, \infty)$. The time point 5 is contained in $\mathcal{V}^d(m_1, \neg\psi_D)$, *i.e.*, the decision for 5 is definite; however, this time point is not admitted to $\mathcal{T}^e(H_{[5]})$ due to the intersection with $[0, 5-w]$, where, for ψ_D, $w = 2$. The time point will be admitted to $\mathcal{T}^e$ when w has elapsed.

6 Related Work

In our previous work, we presented an analysis procedure with preliminary support for RM of MTGL, as the procedure can be adjusted so that it returns true either as soon as a falsification is detected or only when it has become definite [51]. When a falsification is detected, the procedure returns the time point on which the procedure was last executed. The result abstracts the interval-based semantics of MTGL into a point-based interpretation which lacks precision. The definite semantics from Section 3 supports RM of MTGL directly, *i.e.*, at the level of semantics. Moreover, it enables the computations of the definite falsification and satisfaction spans, which in turn enable practical query evaluations.

Compared to InTempo and its advancement we presented, other query-based approaches for RM over structural RTMs either lack a formal treatment of monitoring, *e.g.*, [24, 1], or do not support other key features, *e.g.*, first-order quantification [19], temporal operators [14, 13], or timing constraints [40]. On the other hand, these approaches have their own advantages over the foundations we presented, *e.g.*, support for distributed query evaluation [14] and more temporal primitives [24].

Runtime Verification (RV) is also concerned with formally precise online RM over incrementally processed, and thus possibly incomplete, traces of events. Despite the similarity of their aim, RV and RTMs are different in their applications and characteristics: for instance, state representations in RV focus on a low level of abstraction and are typically inaccessible during monitoring. Conversely, an RTM aims at a richer knowledge representation [14] and has to be accessible to end-users or other technologies during monitoring, as it acts as an interface to manage the system [23]—see [47, 49] for a more elaborate comparison. In RV, properties may be specified using various formalisms, *e.g.*, temporal logics and regular expressions [3], comparisons among which are non-trivial [33, 43]. In the following, we focus on approaches based on temporal logics. According to a recent classification, no approach simultaneously supports key features of InTempo such as first-order quantification, metric temporal constraints, interval-based interpretations, and native support for graph queries and bindings [22].

The RV approach most relevant to our work is MonPoly [6]. MonPoly, an established tool that has been among the top-performers in an RV competition [2], is an implementation of an incremental monitoring algorithm based on *Metric First-Order Temporal Logic* (MFOTL) [7]. The semantics of MFOTL is point-based, *i.e.*, the logic assesses the truth of a formula only for the time points of events in a trace, which means the logic cannot support the computation of a temporal validity or represent the lifespan of a match straightforwardly. MonPoly cannot always encode complex graph queries: for instance, expressing the MTGC from the running example, which prohibits the existence of a pattern, is not possible as MonPoly restricts the use of negation in this place at the formula for reasons of monitorability. Even when possible, this encoding may become overly technical and, as indicated by the performance comparison of InTempo to MonPoly [49] as well as another similar comparison [19], may affect performance: for instance, emulating graph pattern matching requires that partial orderings of match candidates are explicitly formulated in MFOTL which may bloat the size of the formula.

The RV tool DejaVu [31, 30] monitors properties specified in a first-order metric past-only logic with point-based semantics. Translating MTGCs in this logic would require emulating graph-based encodings and bindings (similar to MonPoly) and, moreover, reformulating MTGCs such that they feature only past operators. Such reformulations are not always possible and could be significantly less compact [37, 32]. Monitoring algorithms for interval-based propositional or signal logics with metric timing constraints [5, 38] are capable of interval-based interpretations; although inapplicable to a graph-based first-order setting, they are therefore based on interval computations which are similar to ours. Havelund et al. present a monitoring approach for a logic defined over intervals; properties in the logic refer to interval relations, *e.g.*, requiring that two intervals overlap, where the intervals my contain data [29]. The logic supports quantification over intervals but does not support quantification over the data.

7 Conclusion and Future Work

We present a formal and systematic treatment of incomplete traces in query-based runtime monitoring of temporal properties over structural runtime models. First, we introduce a new semantics for a first-order temporal graph logic, called definite, which only returns decisions if no future change to the model will affect them. Then, based on the definite semantics, we introduce a new definite answer set for the query language of InTempo, a querying scheme we previously presented. Lastly, we present the effective answer set which, contrary to the definite answer set, is amenable to change-driven evaluation. This answer set allows approaches like InTempo to maintain their efficiency while returning definite answers.

Our plans for future work include a consideration of a rewriting procedure for properties in MTGL, such that the rewritten properties avoid or minimize possible delays in returning results, while allowing for a comparable performance to the property before rewriting. We plan to extend the API of the InTempo implementation with the option to return the effective answer set directly. Moreover, we plan to implement the definite answer set and investigate its impact on performance. Although not as efficient as the effective answer set, we also plan to use the definite answer set for testing the answers in the effective answer set. Finally, we plan to extend InTempo with a decision procedure that, depending on the property, switches to the answer set that is more appropriate.

A Overview of Notation

The overview is shown in Table 2.

B Proofs

Following are the proofs for the theorems in the paper, as presented in the doctoral thesis of the first author [47].

B.1 Theorem 1: definite relations imply satisfaction relation over trace

Following is the proof for Theorem 1 (see [47, Section A.3.2]), *i.e.*, given an MTGC ψ over a pattern n and an RTM$^{\mathrm{H}}$-trace $h^{H}_{\tau_{\mathcal{D}}}$ with $\mathcal{D} \in \mathbb{N}^+$ the last index, for all $i \in [1, \mathcal{D}] \cap \mathbb{N}^+$, if m a match for n in $H_{[\tau_i]}$ and $\tau \in [0, \tau_i]$, then for all $k \in [i, \mathcal{D}] \cap \mathbb{N}^+$, (i) if $(H_{[\tau_i]}, m, \tau) \models^d \psi$, then $(H_{[\tau_k]}, m, \tau) \models \psi$, and (ii) if $(H_{[\tau_i]}, m, \tau) \models^d_F \psi$, then $(H_{[\tau_k]}, m, \tau) \not\models \psi$.

Proof. By definition of the RTM$^{\mathrm{H}}$, a match m in $H_{[\tau_i]}$ will be structurally present in all $H_{[\tau_k]}$ with $k \in [i, \mathcal{D}] \cap \mathbb{N}^+$—what may change (once) in future versions of $H_{[\tau_i]}$ is the lifespan of m, *i.e.*, if the *dts* of all matched elements is ∞ and one of these elements is updated to a value less than ∞; even then, this change will not affect the lifespan of m in the period $[0, \tau_i]$, that is, in $H_{[\tau_i]}$, the observation on whether m is present in $\lambda^m \cap [0, \tau_i]$ will never be refuted.

The proof proceeds by mutual structural induction over ψ. In the base case, we show the theorem to be true for the MTGL operator *true*. We omit the straightforward step for conjunction.

- *Base case*: *true*.
  We begin with the definite satisfaction. We assume $(H_{[\tau_i]}, m, \tau) \models^d$ *true* and show that $(H_{[\tau_k]}, m, \tau) \models$ *true* for an arbitrary $k \in [i, \mathcal{D}] \cap \mathbb{N}^+$. By the semantics of MTGL, *true* is always satisfied. Therefore, m in $H_{[\tau_k]}$ also satisfies *true* at τ. We have shown that the implication is true.
  We proceed with the definite falsification. Based on the semantics of the definite falsification relation, a match m never falsifies *true*. Therefore, the antecedent $(H_{[\tau_i]}, m, \tau) \models^d_F$ *true* is false, making the consequent $(H_{[\tau_k]}, m, \tau) \not\models$ *true* true.
- *Induction step*: $\psi = \neg\chi$.
  We begin with the definite satisfaction. Assume that $(H_{[\tau_i]}, m, \tau) \models^d_F \chi \Rightarrow (H_{[\tau_k]}, m, \tau) \not\models \chi$ for an arbitrary $k \in [i, \mathcal{D}] \cap \mathbb{N}^+$. By the semantics of negation and the definite relations, $(H_{[\tau_i]}, m, \tau) \models^d_F \chi \Leftrightarrow (H_{[\tau_i]}, m, \tau) \models^d \neg\chi$. Similarly, $(H_{[\tau_k]}, m, \tau) \not\models \chi \Leftrightarrow (H_{[\tau_k]}, m, \tau) \models \neg\chi$. Therefore, it also holds that $(H_{[\tau_i]}, m, \tau) \models^d \neg\chi \Rightarrow (H_{[\tau_k]}, m, \tau) \models \neg\chi$.
  We proceed with the definite falsification. Assume that $(H_{[\tau_i]}, m, \tau) \models^d \chi \Rightarrow (H_{[\tau_k]}, m, \tau) \models \chi$. Analogously to the definite satisfaction, $(H_{[\tau_i]}, m, \tau) \models^d \chi \Leftrightarrow (H_{[\tau_i]}, m, \tau) \models^d_F \neg\chi$ and $(H_{[\tau_k]}, m, \tau) \models \chi \Leftrightarrow (H_{[\tau_k]}, m, \tau) \not\models \neg\chi$. Therefore, $(H_{[\tau_i]}, m, \tau) \models^d_F \neg\chi \Rightarrow (H_{[\tau_k]}, m, \tau) \not\models \neg\chi$.

Symbol	Concept	Formal Representation	Def.
P	temporal property from running example	-	p. 3
G_τ	runtime model, at time point τ	typed attributed graph	p. 4
τ	time point	real number	p. 5
h_τ	event trace, spanning the interval $[0, \tau]$	sequence of events	p. 5
i	index of sequence member	natural number	p. 5
τ_i	time point at i-th member of sequence	real number	p. 5
$\mathscr{E}$	mapping from events to graph modifications	function	p. 6
$H_{[\tau]}$	runtime model with history, spanning the interval $[0, \tau]$	typed attributed graph	p. 6
h^H_τ	RTM$^{\mathrm{H}}$-trace, spanning the interval $[0, \tau]$	sequence of runtime models with history	p. 6
ψ, χ, ω	temporal property	metric temporal graph condition	p. 6
$n, \hat{n}$	(graph) pattern	typed attributed graph	p. 6
$\models$	(regular) satisfaction relation of metric temporal graph logic	relation	p. 7
$m, \hat{m}$	match	morphism	p. 7
$\mathcal{L}$	query language of InTempo	set of queries	p. 7
E	set of matched vertices	set of vertices in given match	p. 7
e	matched vertex	vertex in E	p. 7
λ^m	lifespan of a match m	interval	p. 8
$\mathcal{Y}$	satisfaction span	interval set	p. 8
$\mathcal{Z}$	satisfaction computation	interval set	p. 8
$\mathcal{V}$	temporal validity	interval set	p. 8
$\hat{M}$	set of matches of $\hat{m}$ compatible to m	set of matches	p. 8
$\mathcal{T}$	(regular) answer set of $\mathcal{L}$	set of $(m, \mathcal{V})$ pairs	p. 9
$\models^d$	definite satisfaction relation	relation	p. 10
$\models^d_F$	definite falsification relation	relation	p. 10
c	current time point	real number	p. 10
$\mathcal{D}$	last member of sequence	natural number	p. 11
w	non-definiteness window	interval	p. 11
$\models_{\mathbb{T}}$	timeliest satisfaction relation	relation	p. 12
$\models_{F,\mathbb{T}}$	timeliest falsification relation	relation	p. 12
$\mathcal{Y}^d$	definite satisfaction span	interval set	p. 13
$\mathcal{Z}^d$	definite satisfaction computation	interval set	p. 13
$\mathcal{F}^d$	definite falsification span	interval set	p. 13
F^d	definite falsification computation	interval set	p. 13
X	unknown span	interval set	p. 13
$\mathcal{V}^d$	definite temporal validity	interval set	p. 15
$\mathcal{IV}$	temporal invalidity	interval set	p. 15
$\mathcal{IV}^d$	definite temporal invalidity	interval set	p. 15
$\mathcal{T}^d$	definite answer set of $\mathcal{L}$	set of $(m, \mathcal{V}^d, \mathcal{IV}^d)$ triples	p. 15
N	network	generalized discrimination network	p. 16
$\mathcal{T}^d_{\mathcal{V},r}$	restricted temporal validity answer set of $\mathcal{L}$	subset of $\mathcal{T}^d$ only with $\mathcal{V}^d$	p. 17
$\mathcal{T}^e$	effective answer set of $\mathcal{L}$	subset of $\mathcal{T}$ with $\mathcal{V}$ capped based on w	p. 17

Table 2: Main symbols, their denoted concept, and formal representation; the rightmost column shows the page on which the symbol was first defined.

– *Induction step*: $\psi = \exists(\hat{n}, \chi)$.
  Let the induction hypothesis be $(H_{[\tau_i]}, \hat{m}, \tau) \models^d \chi \Rightarrow (H_{[\tau_k]}, \hat{m}, \tau) \models \chi$ and $(H_{[\tau_i]}, \hat{m}, \tau) \models^d_F \chi \Rightarrow (H_{[\tau_k]}, \hat{m}, \tau) \not\models \chi$, where $\hat{m}$ is a match for the pattern $\hat{n}$ and k an arbitrary index in $[i, \mathcal{D}] \cap \mathbb{N}^+$.
  We begin with the definite satisfaction. We assume $(H_{[\tau_i]}, m, \tau) \models^d \exists(\hat{n}, \chi)$ and show this implies $(H_{[\tau_k]}, m, \tau) \models \exists(\hat{n}, \chi)$. Since $(H_{[\tau_i]}, m, \tau) \models^d \exists(\hat{n}, \chi)$, there exists matches m and $\hat{m}$ such that $\hat{m}$ is compatible with m and $\tau \in \lambda^m \cap \lambda^{\hat{m}}$. The matches $m, \hat{m}$ will be structurally present and $\hat{m}$ will be compatible with m in all future versions of $H_{[\tau_i]}$. Moreover, there will be no changes in $\lambda^m, \lambda^{\hat{m}}$ for the period $[0, \tau]$. Also, by the induction hypothesis, $\hat{m}$ satisfies χ at τ. Therefore, by the semantics of the satisfaction relation for *exists*, $(H_{[\tau_k]}, m, \tau) \models \exists(\hat{n}, \chi)$. We have shown that the implication is true.
  We proceed with the definite falsification. We assume that $(H_{[\tau_i]}, m, \tau) \models^d_F \exists(\hat{n}, \chi)$ and show that this implies $(H_{[\tau_k]}, m, \tau) \not\models \exists(\hat{n}, \chi)$. Since $(H_{[\tau_i]}, m, \tau) \models^d_F \exists(\hat{n}, \chi)$, (i) either there exists no $\hat{m}$ in $H_{[\tau_i]}$ such that $\hat{m}$ is compatible with m, or (ii) there exists $\hat{m}$ compatible with m, but $\tau \notin \lambda^m \cap \lambda^{\hat{m}}$, or (iii) there exists $\hat{m}$ compatible with m with $\tau \in \lambda^m \cap \lambda^{\hat{m}}$ but $\hat{m}$ definitely falsifies χ at τ. If (i) is true, it will be true in all future versions of $H_{[\tau_i]}$, as matches cannot be found retrospectively. If (ii) is true, the lifespan of $\lambda^{\hat{m}}$ in the period $[0, \tau_i]$ will not change in all future versions of $H_{[\tau_i]}$. Finally, if (iii) is true, we know from the induction hypothesis that $(\hat{m}, \tau) \not\models \chi$ also over $H_{[\tau_k]}$. Therefore, in any case, $(H_{[\tau_k]}, m, \tau) \not\models \exists(\hat{n}, \chi)$. We have shown that the implication is true.
– *Induction step*: $\psi = \chi \mathrm{U}_I \omega$.
  We begin with the definite satisfaction. Induction hypothesis: $(H_{[\tau_i]}, m, \tau) \models^d \chi \Rightarrow (H_{[\tau_k]}, m, \tau) \models \chi$ and $(H_{[\tau_i]}, m, \tau) \models^d \omega \Rightarrow (H_{[\tau_k]}, m, \tau) \models \omega$ with k an arbitrary index in $[i, \mathcal{D}] \cap \mathbb{N}^+$.
  We assume $(H_{[\tau_i]}, m, \tau) \models^d \chi \mathrm{U}_I \omega$ and show this implies $(H_{[\tau_k]}, m, \tau) \models \chi \mathrm{U}_I \omega$. Since $(H_{[\tau_i]}, m, \tau) \models^d \chi \mathrm{U}_I \omega$, there exists τ such that $\tau' - \tau \in I$ and $(H_{[\tau_i]}, m, \tau') \models^d \omega$, and for all $\tau'' \in [\tau, \tau')$ $(H_{[\tau_i]}, m, \tau'') \models^d \chi$. The decisions for the time point τ' and for all time points τ'' either concern a match or not: if they do concern a match, then they are confined to $[0, \tau_i]$ and remain unaltered throughout the trace; if they do not concern a match, *e.g.*, they concern *true* or $\neg$*true*, then they again remain unaltered. Therefore, also over $H_{[\tau_k]}$ it will hold that at τ' $(H_{[\tau_k]}, m, \tau') \models \omega$, and for every τ'' $(H_{[\tau_k]}, m, \tau'') \models \chi$. Thus, by the semantics of the satisfaction relation for *until*, $(H_{[\tau_k]}, m, \tau) \models \chi \mathrm{U}_I \omega$. We have shown that the implication is true.
  We proceed with the definite falsification. Let the induction hypothesis be $(H_{[\tau_i]}, m, \tau) \models^d_F \chi \Rightarrow (H_{[\tau_k]}, m, \tau) \not\models \chi$ and $(H_{[\tau_i]}, m, \tau) \models^d_F \omega \Rightarrow (H_{[\tau_k]}, m, \tau) \not\models \omega$.
  We assume $(H_{[\tau_i]}, m, \tau) \models^d_F \chi \mathrm{U}_I \omega$ and show that this implies $(H_{[\tau_k]}, m, \tau) \not\models \chi \mathrm{U}_I \omega$. Since $(H_{[\tau_i]}, m, \tau) \models^d_F \chi \mathrm{U}_I \omega$, for all τ' such that $\tau' - \tau \in I$, either (i) $(H_{[\tau_i]}, m, \tau') \models^d_F \omega$ or (ii) there exists $\tau'' \in [\tau, \tau')$ such that $(H_{[\tau_i]}, m, \tau'') \models^d \chi$. Regardless of which is the case, *i.e.*, (i) or (ii) or both, analogously to the definite satisfaction, if the decisions for all τ' and at τ'' concern a match,

they will remain unaltered, and so will they if they do not concern a match. Therefore, the case will also hold over $H_{[\tau_k]}$. Therefore, $(H_{[\tau_k]}, m, \tau) \not\models \chi \mathrm{U}_I \omega$. We have shown that the implication is true.

– *Induction step*: $\psi = \chi \mathrm{S}_I \omega$.
The proof proceeds analogously to *until*. We begin with the definite satisfaction. Let the induction hypothesis be $(H_{[\tau_i]}, m, \tau) \models^d \chi \Rightarrow (H_{[\tau_k]}, m, \tau) \models \chi$ and $(H_{[\tau_i]}, m, \tau) \models^d \omega \Rightarrow (H_{[\tau_k]}, m, \tau) \models \omega$ with k an arbitrary index in $[i, \mathcal{D}] \cap \mathbb{N}^+$. We assume $(H_{[\tau_i]}, m, \tau) \models^d \chi \mathrm{S}_I \omega$ and show this implies $(H_{[\tau_k]}, m, \tau) \models \chi \mathrm{S}_I \omega$. Since $(H_{[\tau_i]}, m, \tau) \models^d \chi \mathrm{S}_I \omega$, there exists τ' such that $\tau - \tau' \in I$ and $(H_{[\tau_i]}, m, \tau') \models^d \omega$, and for all $\tau'' \in (\tau', \tau]$ $(H_{[\tau_i]}, m, \tau'') \models^d \chi$. The decisions for the time point τ' and all time points τ'' either concern a match or not: if they do concern a match, then they are confined to $[0, \tau_i]$ and remain unaltered throughout the trace; if they do not concern a match, then they will again remain unaltered. Therefore, also over $H_{[\tau_k]}$ it will hold that at τ' $(H_{[\tau_k]}, m, \tau') \models \omega$, and for all τ'' $(H_{[\tau_k]}, m, \tau'') \models \chi$. Thus by the semantics of the satisfaction relation for *since*, $(H_{[\tau_k]}, m, \tau) \models \chi \mathrm{S}_I \omega$. We have shown that the implication is true.
We proceed with the definite falsification. Let the induction hypothesis be $(H_{[\tau_i]}, m, \tau) \models^d_F \chi \Rightarrow (H_{[\tau_k]}, m, \tau) \not\models \chi$ and $(H_{[\tau_i]}, m, \tau) \models^d_F \omega \Rightarrow (H_{[\tau_k]}, m, \tau) \not\models \omega$.
We assume $(H_{[\tau_i]}, m, \tau) \models^d_F \chi \mathrm{S}_I \omega$ and show that this implies $(H_{[\tau_k]}, m, \tau) \not\models \chi \mathrm{S}_I \omega$. Since $(H_{[\tau_i]}, m, \tau) \models^d_F \chi \mathrm{S}_I \omega$, for all τ' such that $\tau - \tau' \in I$, either (i) $(H_{[\tau_i]}, m, \tau') \models^d_F \omega$ or (ii) there exists $\tau'' \in (\tau', \tau]$ such that $(H_{[\tau_i]}, m, \tau'') \models^d \chi$. Regardless of which is the case, *i.e.*, (i) or (ii) or both, analogously to the definite satisfaction, if the decisions for all τ' and at τ'' concern a match, they will remain unaltered, and so will they if they do not concern a match. Therefore, the case will also hold over $H_{[\tau_k]}$. Therefore, $(H_{[\tau_k]}, m, \tau) \not\models \chi \mathrm{S}_I \omega$. We have shown that the implication is true.

From the base case and induction steps, it follows that Theorem 1 holds. □

B.2 Theorem 2: definite relations are equivalent to satisfaction relation over certain period of trace

Following is the proof for Theorem 2 (see [47, Section A.3.3]), that is, given an MTGC ψ over a pattern n, the non-definiteness w window of ψ, and a sequence of RTM$^\mathrm{H}$ instances $h^H_{\tau_\mathcal{D}}$ with $\mathcal{D} \in \mathbb{N}^+$ the last index, for all $i \in [1, \mathcal{D}] \cap \mathbb{N}^+$, if m a match for n in $H_{[\tau_i]}$ and $\tau \in [0, \tau_i - w]$, then for all $k \in [i, \mathcal{D}] \cap \mathbb{N}^+$, (i) $(H_{[\tau_i]}, m, \tau) \models^d \psi$ iff $(H_{[\tau_k]}, m, \tau) \models \psi$, and (ii) $(H_{[\tau_i]}, m, \tau) \models^d_F \psi$ iff $(H_{[\tau_k]}, m, \tau) \not\models \psi$.

By definition of the RTM$^\mathrm{H}$, a match m in $H_{[\tau_i]}$ will be structurally present in all $H_{[\tau_k]}$ with $k \in [i, \mathcal{D}] \cap \mathbb{N}^+$—what may change (once) in future versions of $H_{[\tau_i]}$ is the lifespan of m, *i.e.*, if the *dts* of all matched elements is ∞ and one of these elements is updated to a value less than ∞; even then, this change will not affect the lifespan of m in the period $[0, \tau_i]$, that is, in $H_{[\tau_i]}$, the observation on whether m is present in $\lambda^m \cap [0, \tau_i]$ will never be refuted.

Proof. The direction $\Rightarrow$ of the equivalence has been shown by the more general Theorem 1, which concerned an arbitrary τ. We therefore focus on direction $\Leftarrow$ of the equivalence. As m is present in $H_{[\tau_i]}$, its lifespan λ^m in the period $[0,\tau_i]$ will remain unchanged in subsequent versions of $H_{[\tau_i]}$. In the following, the non-definiteness window w is computed according to Definition 7.

The proof proceeds by mutual structural induction over ψ. In the base case, we show the theorem to be true for the MTGL operator *true*. We omit the straightforward step for conjunction.

- *Base case*: *true*.
  We begin with the satisfaction. We assume $(H_{[\tau_k]}, m, \tau) \models$ *true* for an arbitrary $k \in [i, \mathcal{D}] \cap \mathbb{N}^+$ and $\tau \in [0, \tau_i - w]$ with $w^{nd} = 0$, and show that this implies $(H_{[\tau_i]}, m, \tau) \models^d$ *true*. As *true* is always satisfied, m in $H_{[\tau_i]}$ definitely satisfies *true* at τ. Hence, the implication to be true.
  We proceed with the falsification. Based on the semantics of satisfaction, a match m never satisfies $\not\models$ *true*. Therefore, the antecedent $(H_{[\tau_k]}, m, \tau) \not\models$ *true* is false, making the consequent $(H_{[\tau_i]}, m, \tau) \models^d_F$ *true* true.
- *Induction step*: $\psi = \neg\chi$.
  We begin with the satisfaction. Let $(H_{[\tau_k]}, m, \tau) \not\models \chi \Rightarrow (H_{[\tau_i]}, m, \tau) \models^d_F \chi$ for an arbitrary $k \in [i, \mathcal{D}] \cap \mathbb{N}^+$ and $\tau \in [0, \tau_i - w]$ with $w(\neg\chi) = w(\chi)$. By the semantics of negation and the satisfaction relation, $(H_{[\tau_k]}, m, \tau) \not\models \chi \Leftrightarrow (H_{[\tau_k]} m, \tau) \models \neg\chi$. Similarly, $(H_{[\tau_i]}, m, \tau) \models^d_F \chi \Leftrightarrow (H_{[\tau_i]}, m, \tau) \models^d \neg\chi$. Therefore, it also holds that $(H_{[\tau_k]}, m, \tau) \models \neg\chi \Rightarrow (H_{[\tau_i]}, m, \tau) \models^d \neg\chi$.
  We proceed with the falsification. Assume $(H_{[\tau_k]}, m, \tau) \models \chi \Rightarrow (H_{[\tau_i]}, m, \tau) \models^d \chi$. Analogously to the satisfaction, $(H_{[\tau_k]}, m, \tau) \models \chi \Leftrightarrow (H_{[\tau_i]}, m, \tau) \not\models \neg\chi$ and $(H_{[\tau_k]}, m, \tau) \models^d \chi \Leftrightarrow (H_{[\tau_k]}, m, \tau) \models^d_F \neg\chi$. Therefore, $(H_{[\tau_k]}, m, \tau) \not\models \neg\chi \Rightarrow (H_{[\tau_i]}, m, \tau) \models^d_F \neg\chi$.
- *Induction step*: $\psi = \exists(\hat{n}, \chi)$.
  Let the induction hypothesis be $(H_{[\tau_k]}, \hat{m}, \tau) \models \chi \Rightarrow (H_{[\tau_i]}, \hat{m}, \tau) \models^d \chi$ and $(H_{[\tau_k]}, \hat{m}, \tau) \not\models \chi \Rightarrow (H_{[\tau_i]}, \hat{m}, \tau) \models^d_F \chi$, where $\hat{m}$ is a match for the pattern $\hat{n}$, k an arbitrary index in $[i, \mathcal{D}] \cap \mathbb{N}^+$, and $\tau \in [0, \tau_i - w]$. The non-definiteness window w is given by $w(\exists(\hat{n}, \chi)) = w(\chi)$.
  We begin with the satisfaction. We assume that $(H_{[\tau_k]}, m, \tau) \models \exists(\hat{n}, \chi)$ and show that this implies $(H_{[\tau_i]}, m, \tau) \models^d \exists(\hat{n}, \chi)$. Since $(H_{[\tau_k]}, m, \tau) \models \exists(\hat{n}, \chi)$, there exists matches m and $\hat{m}$ in $H_{[\tau_k]}$ such that $\hat{m}$ is compatible with m and $\tau \in \lambda^m \cap \lambda^{\hat{m}}$. The match m is present in $H_{[\tau_i]}$ and, according to the induction hypothesis, the match $\hat{m}$ is also present in $H_{[\tau_i]}$. As the matches are structurally the same, $\hat{m}$ is also compatible with m in $H_{[\tau_i]}$. Moreover, as there are no changes in $\lambda^m, \lambda^{\hat{m}}$ for the period $[0, \tau_i]$, $\tau \in \lambda^m \cap \lambda^{\hat{m}}$ over $H_{[\tau_i]}$. We also know that $\tau \leq \tau_i$ and, by the induction hypothesis, that $\hat{m}$ satisfies χ at τ. Therefore, by the semantics of the definite satisfaction relation for *exists*, $(H_{[\tau_i]}, m, \tau) \models^d \exists(\hat{n}, \chi)$. We have shown that the implication is true.
  We proceed with the falsification. We assume that $(H_{[\tau_k]}, m, \tau) \not\models \exists(\hat{n}, \chi)$ and show that this implies $(H_{[\tau_i]}, m, \tau) \models^d_F \exists(\hat{n}, \chi)$. Since $(H_{[\tau_k]}, m, \tau) \not\models \exists(\hat{n}, \chi)$, (i) either there exists no $\hat{m}$ in $H_{[\tau_k]}$ such that $\hat{m}$ is compatible with m, or (ii)

there exists $\hat{m}$ compatible with m, but $\tau \notin \lambda^m \cap \lambda^{\hat{m}}$, or (iii) there exists $\hat{m}$ compatible with m with $\tau \in \lambda^m \cap \lambda^{\hat{m}}$ but $\hat{m}$ falsifies χ at τ. If (i) is true, it will be true in all future versions of $H_{[\tau_i]}$, as matches cannot be found retrospectively. If (ii) is true, the lifespan of $\lambda^{\hat{m}}$ in the period $[0, \tau_i]$ will not change in all future versions of $H_{[\tau_i]}$. Finally, if (iii) is true, we know from the induction hypothesis that $(\hat{m}, \tau) \models^d_F \chi$ also over $H_{[\tau_i]}$ and that $\tau \leq \tau_i$. Therefore, in any case, $(H_{[\tau_i]}, m, \tau) \models^d_F \exists(\hat{n}, \chi)$. We have shown that the implication is true.

– *Induction step*: $\psi = \chi \mathrm{U}_I \omega$.
We begin with the satisfaction. Let the induction hypothesis be $(H_{[\tau_k]}, m, \tau) \models \chi \Rightarrow (H_{[\tau_i]}, m, \tau) \models^d \chi$ and $(H_{[\tau_k]}, m, \tau) \models \omega \Rightarrow (H_{[\tau_i]}, m, \tau) \models^d \omega$ with k an arbitrary index in $[i, \mathcal{D}] \cap \mathbb{N}^+$ and $\tau \in [0, \tau_i - w]$. The non-definiteness window w is given by $max(w(\chi), w(\omega)) + r(I)$.
We assume $(H_{[\tau_k]}, m, \tau) \models \chi \mathrm{U}_I \omega$ and show $(H_{[\tau_i]}, m, \tau) \models^d \chi \mathrm{U}_I \omega$. Since $(H_{[\tau_k]}, m, \tau) \models \chi \mathrm{U}_I \omega$, there exists τ' such that $\tau' - \tau \in I$ and $(H_{[\tau_k]}, m, \tau') \models \omega$, and for all $\tau'' \in [\tau, \tau')$ $(H_{[\tau_k]}, m, \tau'') \models \chi$. From $\tau \in [0, \tau_i - w]$ and $\tau' \in [\tau + \ell(I), \tau + r(I)]$, it follows that $\tau' \leq \tau_i - max(w(\chi), w(\omega))$. Based on this and the induction hypothesis, $(H_{[\tau_i]}, m, \tau') \models^d \omega$. Moreover, as τ' stems from a period outside the non-definiteness window of ω, the decision at τ', whether it concerns a match or not, will remain unaltered once made.
The decision at τ' as well as the preceding period $[\tau, \tau')$ are also outside the non-definiteness window of χ. Thus, all $\tau'' \in [\tau, \tau')$ stem from a period covered by $H_{[\tau_i]}$, and decisions for χ made in this period are definite. Therefore, for all $[\tau + \ell(I), \tau + \tau')$ $(H_{[\tau_i]}, m, \tau'') \models^d \chi$, and, by the definite semantics, $(H_{[\tau_i]}, m, \tau) \models^d \chi \mathrm{U}_I \omega$. We have shown that the implication is true.
We proceed with the falsification. Let the induction hypothesis be that $(H_{[\tau_k]}, m, \tau) \not\models \chi \Rightarrow (H_{[\tau_i]}, m, \tau) \models^d_F \chi$ and $(H_{[\tau_k]}, m, \tau) \not\models \omega \Rightarrow (H_{[\tau_i]}, m, \tau) \models^d_F \omega$.
We assume $(H_{[\tau_k]}, m, \tau) \not\models \chi \mathrm{U}_I \omega$ and show $(H_{[\tau_i]}, m, \tau) \models^d_F \chi \mathrm{U}_I \omega$. Since $(H_{[\tau_k]}, m, \tau) \not\models \chi \mathrm{U}_I \omega$, it holds that for all τ' such that $\tau' - \tau \in I$ either (i) $(H_{[\tau_k]}, m, \tau') \not\models \omega$ or (ii) there exists $\tau'' \in [\tau, \tau')$ such that $(H_{[\tau_k]}, m, \tau'') \models \chi$. Regardless of which is the case, *i.e.*, (i) or (ii) or both, analogously to the satisfaction, the decisions for all τ' and at τ'' stem from a period that is covered by $H_{[\tau_i]}$, and decisions made in this period regarding χ and ω are definite. Therefore, the case will also hold over $H_{[\tau_i]}$. Therefore, $(H_{[\tau_i]}, m, \tau) \models^d_F \chi \mathrm{U}_I \omega$. We have shown that the implication is true.

– *Induction step*: $\psi = \chi \mathrm{S}_I \omega$.
We begin with the satisfaction. Let the induction hypothesis be $(H_{[\tau_k]}, m, \tau) \models \chi \Rightarrow (H_{[\tau_i]}, m, \tau) \models^d \chi$ and $(H_{[\tau_k]}, m, \tau) \models \omega \Rightarrow (H_{[\tau_i]}, m, \tau) \models^d \omega$ with k an arbitrary index in $[i, \mathcal{D}] \cap \mathbb{N}^+$ and $\tau \in [0, \tau_i - w]$. The non-definiteness window w is given by $max(w(\chi), w(\omega))$.
We assume $(H_{[\tau_k]}, m, \tau) \models \chi \mathrm{S}_I \omega$ and show $(H_{[\tau_i]}, m, \tau) \models^d \chi \mathrm{S}_I \omega$. Since $(H_{[\tau_k]}, m, \tau) \models \chi \mathrm{S}_I \omega$, there exists τ' such that $\tau - \tau' \in I$ and $(H_{[\tau_k]}, m, \tau') \models \omega$, and for all $\tau'' \in (\tau', \tau]$ $(H_{[\tau_k]}, m, \tau'') \models \chi$. From $\tau \in [0, \tau_i - w]$ and $\tau' \in [\tau - r(I), \tau - \ell(I)]$, it follows that $\tau' \leq \tau_i - max(w(\chi), w(\omega))$. Hence, the

decision at τ' can already be made over $H_{[\tau_i]}$, and, moreover, as τ' stems from a period outside the non-definiteness window of ω, the decision at τ', whether it concerns a match or not, will remain unaltered once made. Therefore, $(H_{[\tau_i]}, m, \tau') \models^d \omega$. The decision at τ' as well as the succeeding period $(\tau', \tau]$ is also outside the non-definiteness window of χ. Thus, all $\tau'' \in (\tau', \tau]$ stem from a period covered by $H_{[\tau_i]}$, and decisions for χ made in this period are definite. Therefore, for all $\tau'' \in (\tau', \tau]$ $(H_{[\tau_i]}, m, \tau'') \models^d \chi$, and, by the definite semantics, $(H_{[\tau_i]}, m, \tau) \models^d \chi \mathrm{S}_I \omega$. We have shown that the implication is true.
We proceed with the falsification. Let the induction hypothesis be that $(H_{[\tau_k]}, m, \tau) \not\models \chi \Rightarrow (H_{[\tau_i]}, m, \tau) \models^d_F \chi$ and $(H_{[\tau_k]}, m, \tau) \not\models \omega \Rightarrow (H_{[\tau_i]}, m, \tau) \models^d_F \omega$.
We assume $(H_{[\tau_k]}, m, \tau) \not\models \chi \mathrm{S}_I \omega$ and show $(H_{[\tau_i]}, m, \tau) \models^d_F \chi \mathrm{S}_I \omega$. Since $(H_{[\tau_k]}, m, \tau) \not\models \chi \mathrm{S}_I \omega$, it holds that for all τ' such that $\tau - \tau' \in I$ either (i) $(H_{[\tau_k]}, m, \tau') \not\models \omega$ or (ii) there exists $\tau'' \in (\tau', \tau]$ such that $(H_{[\tau_k]}, m, \tau'') \models \chi$. Regardless of which is the case, *i.e.*, (i) or (ii) or both, analogously to the satisfaction, the decisions for all τ' and at τ'' stem from a period that is covered by $H_{[\tau_i]}$, and decisions made in this period regarding χ and ω are definite. Therefore, the case will also hold over $H_{[\tau_i]}$. Therefore, $(H_{[\tau_i]}, m, \tau) \models^d_F \chi \mathrm{S}_I \omega$. We have shown that the implication is true.

From the base case and induction steps, it follows that Theorem 2 holds. □

B.3 Corollary 1: Period in trace with non-definite decisions

Following is the proof for Corollary 1 (see [47, p. 32]), that is, if ψ is an MTGC, w is the non-definiteness window of ψ, $H_{[\tau_i]}$ is a RTM$^{\mathrm{H}}$ instance associated with the time point τ_i, m is a match for a pattern n, and τ a time point in $[0, \tau_i]$, then if $(H_{[\tau_i]}, m, \tau) \not\models^d \psi$ and $(H_{[\tau_i]}, m, \tau) \not\models^d_F \psi$, then $\tau \in (\tau_i - w, \tau_i]$.

Proof. The proof follows from Theorem 2. The satisfaction relation and its negation make a decision for every time point in $[0, \tau_i - w]$, *i.e.*, the relation does not support the value *unknown*; Theorem 2 shows that the decisions made by the satisfaction relation and its negation for $[0, \tau_i - w]$ are equivalent to the decisions made by the definite relations. Consequently, if no definite decision is made for $\tau \in [0, \tau_i]$, then $\tau \notin [0, \tau_i - w]$. □

B.4 Theorem 3: Equality of definite spans and definite computations for satisfaction and falsification

Following is the proof for Theorem 3 (see [47, Section A.3.4]), *i.e.*, given a match m over a RTM$^{\mathrm{H}}$ $H_{[\tau]}$ and an MTGC ψ, the definite satisfaction span $\mathcal{Y}^d$ of m for ψ over $H_{[\tau]}$ is given by the definite satisfaction computation $\mathcal{Z}^d$ of m for ψ over $H_{[\tau]}$ in Definition 8, that is, $\mathcal{Y}^d(m, \psi) = \mathcal{Z}^d(m, \psi)$. Moreover, the definite falsification span $\mathcal{F}$ of m for ψ over $H_{[\tau]}$ is given by the definite falsification computation F of m for ψ over $H_{[\tau]}$ in Definition 8, that is, $\mathcal{F}(m, \psi) = F(m, \psi)$.

Proof. The proof for the definite satisfaction span $\mathcal{Z}^d$ proceeds almost identically to the proof for Theorem 1 for $\mathcal{Z}$ in [47, Section A.3.1], *i.e.*, by structural induction over ψ, and therefore omitted. For *true*, conjunction, *exists*, *until*, and *since* in Definition 8, inclusion can be shown in both directions—the proof for the negation relies on a reasoning analogous to the one presented below for negation for the definite falsification span.

The proof for the definite falsification F is based on the application of $F = \mathbb{R} \setminus (\mathcal{Z}^d \uplus X)$ for each MTGL operator—which follows from $\mathbb{R} = \mathcal{Y}^d \uplus \mathcal{F} \uplus X$. The unknown span X for *true* is $X = \emptyset$, whereas for *exists*, by definition of the RTM$^{\mathrm{H}}$ $H_{[\tau]}$, it is $X = (\tau, \infty)$. If F is known, it can be used to compute $\mathcal{Z}^d \uplus X$.

- $\psi = \mathit{true}$: From Equation 8 in Definition 8, we have $\mathcal{Z}^d(m, \mathit{true}) = \mathbb{R}$, therefore $F(m, \mathit{true}) = \emptyset$.
- $\psi = \neg\chi$: It holds that

$$\overline{F}(m, \neg\chi) = \mathcal{Z}^d(m, \neg\chi) \uplus X(m, \neg\chi)$$

and

$$\overline{\mathcal{Z}^d}(m, \chi) = \mathcal{Z}^d(m, \neg\chi) \uplus X(m, \neg\chi)$$

Therefore,

$$F(m, \neg\chi) = \overline{\overline{\mathcal{Z}^d}}(m, \chi) = \mathcal{Z}^d(m, \chi)$$

- $\psi = \chi \wedge \omega$: Let each time point that does not definitely falsify the MTGC a that χ encloses to be assumed to satisfy the a. In practice, this includes all time points in $\mathcal{Z}^d(m, \chi) \uplus X(m, \chi)$ for a. Subtracting this maximal satisfaction span from the time domain $\mathbb{R}$ yields the set of time points that definitely falsify χ. Let the satisfaction span of ω be defined analogously. If the satisfaction span of conjunction is computed based on these maximal satisfaction spans of χ and ω, *i.e.*, by $(\mathcal{Z}^d(m, \chi) \uplus X(m, \chi)) \cap (\mathcal{Z}^d(m, \omega) \uplus X(m, \omega))$, the definite falsification span of conjunction can be computed analogously.

$$\begin{aligned} F(m, \chi \wedge \omega) &= \mathbb{R} \setminus \big((\mathcal{Z}^d(m, \chi) \uplus X(m, \chi)) \cap (\mathcal{Z}^d(m, \omega) \uplus X(m, \omega))\big) \\ &= \mathbb{R} \setminus \big((\mathbb{R} \setminus F(m, \chi)) \cap (\mathbb{R} \setminus F(m, \omega))\big) \\ &= F(m, \chi) \cup F(m, \omega) \end{aligned}$$

- $\psi = \exists(\hat{n}, \chi)$: Let τ be the time point of the RTM$^{\mathrm{H}}$ $H_{[\tau]}$. As $\mathcal{Z}(m, \exists(\hat{n}, \chi))$ is known and $X(m, \exists(\hat{n}, \chi)) = (\tau, \infty)$, to obtain the falsification computation, we can directly solve $\mathbb{R} \setminus (\mathcal{Z}^d \uplus X)$.

$$\begin{aligned} F(m, \exists(\hat{n}, \chi)) &= \mathbb{R} \setminus \big(\mathcal{Z}^d(m, \exists(\hat{n}, \chi)) \cup (\tau, \infty)\big) \\ &= \big(\mathbb{R} \setminus (\tau, \infty)\big) \cap \big(\mathbb{R} \setminus \mathcal{Z}^d(m, \exists(\hat{n}, \chi))\big) \\ &= (-\infty, \tau] \cap \big(\mathbb{R} \setminus \mathcal{Z}^d(m, \exists(\hat{n}, \chi))\big) \end{aligned}$$

- $\psi = \chi \mathrm{U}_I \omega$ and $0 \notin I$: The computation for *until* relies on the reasoning explained in the case of conjunction. The satisfaction span of *until* is computed based on the maximal satisfaction spans of ω, *i.e.*, $\mathcal{Z}^d(m, \omega) \uplus X(m, \omega)$, and χ, that is, J_i^X is obtained by $\mathcal{Z}^d(m,\omega) \uplus X(m,\omega)$ and $\mathcal{Z}^d(m,\chi) \uplus X(m,\chi)$, thus the *until* satisfaction span is similarly maximal. Therefore, complementing this maximal satisfaction span yields all time points that definitely falsify *until*. Therefore, we have:

$$F(m, \chi \mathrm{U}_I \omega) = \mathbb{R} \setminus \left(\bigcup_{i \in \mathcal{Z}^d(m,\omega) \cup X(m,\omega),\, j \in J_i^X} j \cap \left((j^+ \cap i) \ominus I\right) \right)$$

- $\psi = \chi \mathrm{U}_I \omega$ and $0 \in I$: The reasoning is similar to the case where $0 \notin I$.
- $\psi = \chi \mathrm{S}_I \omega$ and $0 \notin I$: The case proceeds analogously to the corresponding case of *until*.
- $\psi = \chi \mathrm{S}_I \omega$ and $0 \in I$: The case proceeds analogously to the corresponding case of *until*.

By showing that $\mathcal{Y}^d(m,\psi) = \mathcal{Z}^d(m,\psi)$ and the equations for $F(m,\psi)$, we have shown that theorem holds.

B.5 Theorem 4: Equality of effective answer set and restricted definite temporal validity answer set over trace

Following is the proof for Theorem 4 (see [47, p. 57]), which states that, if $\zeta \coloneqq (n,\psi)$ is a temporal query with ψ an MTGC, w is the non-definiteness window of ψ, $h^H_{\tau_\mathcal{D}}$ is a RTM$^\mathrm{H}$-trace with $\mathcal{D} \in [2,\infty] \cap \mathbb{N}^+$, i is an index in $[k, \mathcal{D}-1] \cap \mathbb{N}^+$ such that $\tau_k \geq w$. $\mathcal{T}^d_{\mathcal{V},r}(H_{[\tau_i]})$ is the restricted definite temporal validity answer set over $H_{[\tau_i]}$ which has been obtained from the definite answer set $\mathcal{T}^d$ but contains (i) only pairs of matches with their temporal validity $\mathcal{V}^d$ with $\mathcal{V}^d \neq \emptyset$ and (ii) $\mathcal{V}^d$ is intersected with $[0, \tau_i - w]$, then the effective answer set $\mathcal{T}^e(H_{[\tau_i]})$ is equal to $\mathcal{T}^d_{\mathcal{V},r}(H_{[\tau_i]})$.

Proof. Based on the more general Theorem 2 which shows that, for $\tau \in [0, \tau_i - w]$, the satisfaction decision for τ in $H_{[\tau_i]}$ is equivalent to definite satisfaction decision for τ in $H_{[\tau_i]}$. The computations of $\mathcal{V}$ and $\mathcal{V}^d$ over $H_{[\tau_i]}$ rely on the computations of $\mathcal{Z}$ and $\mathcal{Z}^d$ over $H_{[\tau_i]}$, respectively. Theorem 1 in [47, Section A.3.1] and Theorem 3 show that satisfaction relation and definite satisfaction relation over $H_{[\tau_i]}$ are soundly reflected in $\mathcal{Z}$ and $\mathcal{Z}^d$ over $H_{[\tau_i]}$, respectively. □

References

[1] Gala Barquero, Javier Troya, and Antonio Vallecillo. "Improving Query Performance on Dynamic Graphs". In: *Softw Syst Model* 20.4 (Aug. 1, 2021), pp. 1011–1041. ISSN: 1619-1374. DOI: 10.1007/s10270-020-00832-3.

[2] Ezio Bartocci et al. "First International Competition on Runtime Verification: Rules, Benchmarks, Tools, and Final Results of CRV 2014". In: *Int J Softw Tools Technol Transfer* 21.1 (Feb. 1, 2019), pp. 31–70. ISSN: 1433-2787. DOI: 10.1007/s10009-017-0454-5.

[3] Ezio Bartocci et al. “Introduction to Runtime Verification”. In: *Lectures on Runtime Verification: Introductory and Advanced Topics*. Ed. by Ezio Bartocci and Yliès Falcone. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2018, pp. 1–33. ISBN: 978-3-319-75632-5. DOI: 10.1007/978-3-319-75632-5_1.

[4] Ezio Bartocci et al. “Specification-Based Monitoring of Cyber-Physical Systems: A Survey on Theory, Tools and Applications”. In: *Lectures on Runtime Verification*. Ed. by Ezio Bartocci and Yliès Falcone. Vol. 10457. Cham: Springer International Publishing, 2018, pp. 135–175. ISBN: 978-3-319-75631-8. URL: http://link.springer.com/10.1007/978-3-319-75632-5_5.

[5] David Basin, Felix Klaedtke, and Eugen Zălinescu. “Algorithms for Monitoring Real-Time Properties”. In: *Acta Informatica* 55.4 (June 1, 2018), pp. 309–338. ISSN: 1432-0525. DOI: 10.1007/s00236-017-0295-4.

[6] David Basin, Felix Klaedtke, and Eugen Zălinescu. “The MonPoly Monitoring Tool”. In: *Kalpa Publications in Computing*. RV-CuBES 2017. An International Workshop on Competitions, Usability, Benchmarks, Evaluation, and Standardisation for Runtime Verification Tools. Vol. 3. EasyChair, Dec. 14, 2017, pp. 19–28. DOI: 10.29007/89hs.

[7] David Basin et al. “Monitoring Metric First-Order Temporal Properties”. In: *J. ACM* 62.2 (May 6, 2015), 15:1–15:45. ISSN: 0004-5411. DOI: 10.1145/2699444.

[8] Andreas Bauer, Martin Leucker, and Christian Schallhart. “The Good, the Bad, and the Ugly, But How Ugly Is Ugly?” In: *Runtime Verification*. Ed. by Oleg Sokolsky and Serdar Taşıran. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2007, pp. 126–138. ISBN: 978-3-540-77395-5. DOI: 10.1007/978-3-540-77395-5_11.

[9] Nelly Bencomo, Sebastian Götz, and Hui Song. “Models@run.Time: A Guided Tour of the State of the Art and Research Challenges”. In: *Softw Syst Model* 18.5 (Oct. 1, 2019), pp. 3049–3082. ISSN: 1619-1374. DOI: 10.1007/s10270-018-00712-x.

[10] Thomas Beyhl et al. “On the Operationalization of Graph Queries with Generalized Discrimination Networks”. In: *Graph Transformation*. Ed. by Rachid Echahed and Mark Minas. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2016, pp. 170–186. ISBN: 978-3-319-40530-8. DOI: 10.1007/978-3-319-40530-8_11.

[11] Robert Bill et al. “On the Need for Temporal Model Repositories”. In: *Software Technologies: Applications and Foundations*. Ed. by Martina Seidl and Steffen Zschaler. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2018, pp. 136–145. ISBN: 978-3-319-74730-9. DOI: 10.1007/978-3-319-74730-9_11.

[12] Gordon Blair, Nelly Bencomo, and Robert B. France. “Models@ Run.Time”. In: *Computer* 42.10 (Oct. 2009), pp. 22–27. ISSN: 1558-0814. DOI: 10.1109/MC.2009.326.

[13] Márton Búr. "Query-Based Runtime Monitoring in Real-Time and Distributed Systems". PhD thesis. Canada: McGill University, 2021. URL: https://escholarship.mcgill.ca/concern/theses/w95055572.

[14] Márton Búr et al. "Distributed Graph Queries Over Models@run.Time for Runtime Monitoring of Cyber-Physical Systems". In: *Int J Softw Tools Technol Transfer* 22.1 (Feb. 1, 2020), pp. 79–102. ISSN: 1433-2787. DOI: 10.1007/s10009-019-00531-5.

[15] L. Catarinucci et al. "An IoT-Aware Architecture for Smart Healthcare Systems". In: *IEEE Internet of Things Journal* 2.6 (Dec. 2015), pp. 515–526. ISSN: 2327-4662. DOI: 10.1109/JIOT.2015.2417684.

[16] Federico Ciccozzi et al. "Model-Driven Engineering for Mission-Critical IoT Systems". In: *IEEE Software* 34.1 (Jan. 2017), pp. 46–53. ISSN: 1937-4194. DOI: 10.1109/MS.2017.1.

[17] Carlo Combi et al. "Modelling Temporal, Data-Centric Medical Processes". In: *Proceedings of the 2nd ACM SIGHIT International Health Informatics Symposium*. IHI '12. New York, NY, USA: Association for Computing Machinery, Jan. 28, 2012, pp. 141–150. ISBN: 978-1-4503-0781-9. DOI: 10.1145/2110363.2110382.

[18] Bruno Courcelle. "The Expression of Graph Properties and Graph Transformations in Monadic Second-Order Logic". In: *Handbook of Graph Grammars and Computing by Graph Transformation: Volume I. Foundations*. USA: World Scientific Publishing Co., Inc., Feb. 1, 1997, pp. 313–400. ISBN: 978-981-02-2884-2. DOI: 10.1142/9789812384720_0005.

[19] Wei Dou, Domenico Bianculli, and Lionel Briand. "A Model-Driven Approach to Trace Checking of Pattern-Based Temporal Properties". In: *Proceedings of the ACM/IEEE 20th International Conference on Model Driven Engineering Languages and Systems*. MODELS '17. Austin, Texas: IEEE Press, Sept. 17, 2017, pp. 323–333. ISBN: 978-1-5386-3492-9. DOI: 10.1109/MODELS.2017.9.

[20] Hartmut Ehrig, Ulrike Prange, and Gabriele Taentzer. "Fundamental Theory for Typed Attributed Graph Transformation". In: *Graph Transformations*. Ed. by Hartmut Ehrig et al. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2004, pp. 161–177. ISBN: 978-3-540-30203-2. DOI: 10.1007/978-3-540-30203-2_13.

[21] Cindy Eisner et al. "Reasoning with Temporal Logic on Truncated Paths". In: *Computer Aided Verification*. Ed. by Warren A. Hunt and Fabio Somenzi. Berlin, Heidelberg: Springer, 2003, pp. 27–39. ISBN: 978-3-540-45069-6. DOI: 10.1007/978-3-540-45069-6_3.

[22] Yliès Falcone et al. "A Taxonomy for Classifying Runtime Verification Tools". In: *Int J Softw Tools Technol Transfer* 23.2 (Apr. 1, 2021), pp. 255–284. ISSN: 1433-2787. DOI: 10.1007/s10009-021-00609-z.

[23] Robert France and Bernhard Rumpe. "Model-Driven Development of Complex Software: A Research Roadmap". In: *Future of Software Engineering (FOSE '07)*. May 2007, pp. 37–54. DOI: 10.1109/FOSE.2007.14.

[24] Antonio García-Domínguez et al. "Querying and Annotating Model Histories with Time-Aware Patterns". In: *2019 ACM/IEEE 22nd International Conference on Model Driven Engineering Languages and Systems (MODELS)*. Sept. 2019, pp. 194–204. DOI: 10.1109/MODELS.2019.000-2.

[25] Sona Ghahremani, Holger Giese, and Thomas Vogel. "Improving Scalability and Reward of Utility-Driven Self-Healing for Large Dynamic Architectures". In: *ACM Trans. Auton. Adapt. Syst.* 14.3 (Feb. 25, 2020), 12:1–12:41. ISSN: 1556-4665. DOI: 10.1145/3380965.

[26] Holger Giese et al. "Metric Temporal Graph Logic over Typed Attributed Graphs". In: *Fundamental Approaches to Software Engineering*. Ed. by Reiner Hähnle and Wil van der Aalst. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2019, pp. 282–298. ISBN: 978-3-030-16722-6. DOI: 10.1007/978-3-030-16722-6_16.

[27] Annegret Habel and Karl-Heinz Pennemann. "Correctness of High-Level Transformation Systems Relative to Nested Conditions". In: *Mathematical Structures in Computer Science* 19.2 (Apr. 2009), pp. 245–296. ISSN: 1469-8072, 0960-1295. DOI: 10.1017/S0960129508007202.

[28] E. N. Hanson, S. Bodagala, and U. Chadaga. "Trigger Condition Testing and View Maintenance Using Optimized Discrimination Networks". In: *IEEE Transactions on Knowledge and Data Engineering* 14.2 (Mar. 2002), pp. 261–280. ISSN: 1558-2191. DOI: 10.1109/69.991716.

[29] Klaus Havelund, Moran Omer, and Doron Peled. "Monitoring First-Order Interval Logic". In: *Software Engineering and Formal Methods*. Ed. by Radu Calinescu and Corina S. Păsăreanu. Cham: Springer International Publishing, 2021, pp. 66–83. ISBN: 978-3-030-92124-8. DOI: 10.1007/978-3-030-92124-8_4.

[30] Klaus Havelund and Doron Peled. "BDDs for Representing Data in Runtime Verification". In: *Runtime Verification*. Ed. by Jyotirmoy Deshmukh and Dejan Ničković. Vol. 12399. Cham: Springer International Publishing, 2020, pp. 107–128. ISBN: 978-3-030-60508-7. DOI: 10.1007/978-3-030-60508-7_6.

[31] Klaus Havelund and Doron Peled. "First-Order Timed Runtime Verification Using BDDs". In: *Automated Technology for Verification and Analysis*. Ed. by Dang Van Hung and Oleg Sokolsky. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2020, pp. 3–24. ISBN: 978-3-030-59152-6. DOI: 10.1007/978-3-030-59152-6_1.

[32] Klaus Havelund and Doron Peled. "Runtime Verification: From Propositional to First-Order Temporal Logic". In: *Runtime Verification*. Ed. by Christian Colombo and Martin Leucker. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2018, pp. 90–112. ISBN: 978-3-030-03769-7. DOI: 10.1007/978-3-030-03769-7_7.

[33] Klaus Havelund et al. "Monitoring Events That Carry Data". In: *Lectures on Runtime Verification: Introductory and Advanced Topics*. Ed. by Ezio Bartocci and Yliès Falcone. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2018, pp. 61–102. ISBN: 978-3-319-75632-5. DOI: 10.1007/978-3-319-75632-5_3.

[34] Kerianne L. Hobbs et al. "Runtime Assurance for Safety-Critical Systems: An Introduction to Safety Filtering Approaches for Complex Control Systems". In: *IEEE Control Syst.* 43.2 (Apr. 2023), pp. 28–65. ISSN: 1066-033X, 1941-000X. DOI: 10.1109/MCS.2023.3234380.

[35] Ron Koymans. "Specifying Real-Time Properties with Metric Temporal Logic". In: *Real-Time Syst* 2.4 (Nov. 1, 1990), pp. 255–299. ISSN: 1573-1383. DOI: 10.1007/BF01995674.

[36] Christian Krause et al. "An SQL-Based Query Language and Engine for Graph Pattern Matching". In: *Graph Transformation*. Ed. by Rachid Echahed and Mark Minas. Vol. 9761. Cham: Springer International Publishing, 2016, pp. 153–169. ISBN: 978-3-319-40530-8. DOI: 10.1007/978-3-319-40530-8_10.

[37] F. Laroussinie, N. Markey, and P. Schnoebelen. "Temporal Logic with Forgettable Past". In: *Proceedings 17th Annual IEEE Symposium on Logic in Computer Science*. July 2002, pp. 383–392. DOI: 10.1109/LICS.2002.1029846.

[38] Oded Maler and Dejan Ničković. "Monitoring Properties of Analog and Mixed-Signal Circuits". In: *Int J Softw Tools Technol Transfer* 15.3 (June 1, 2013), pp. 247–268. ISSN: 1433-2787. DOI: 10.1007/s10009-012-0247-9.

[39] Felix Mannhardt and Daan Blinde. "Analyzing the Trajectories of Patients with Sepsis Using Process Mining". In: *RADAR+EMISA@CAiSE, Essen, Germany, June 12-13, 2017*. Ed. by Jens Gulden et al. Vol. 1859. CEUR Workshop Proceedings. CEUR-WS.org, 2017, pp. 72–80. URL: http://ceur-ws.org/Vol-1859/bpmds-08-paper.pdf.

[40] Diego Marmsoler and Ana Petrovska. "Runtime Verification for Dynamic Architectures". In: *Journal of Logical and Algebraic Methods in Programming* 118 (Jan. 1, 2021), p. 100618. ISSN: 2352-2208. DOI: 10.1016/j.jlamp.2020.100618.

[41] Ramon E. Moore, R. Baker Kearfott, and Michael J. Cloud. *Introduction to Interval Analysis*. Society for Industrial and Applied Mathematics, Jan. 2009. ISBN: 978-0-89871-771-6. DOI: 10.1137/1.9780898717716.

[42] 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*. Ed. by Tiziana Margaria, Susanne Graf, and Kim G. Larsen. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2019, pp. 218–234. ISBN: 978-3-030-22348-9. URL: https://doi.org/10.1007/978-3-030-22348-9_14.

[43] Giles Reger and David Rydeheard. "From First-order Temporal Logic to Parametric Trace Slicing". In: *Runtime Verification*. Ed. by Ezio Bartocci and Rupak Majumdar. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2015, pp. 216–232. ISBN: 978-3-319-23820-3. DOI: 10.1007/978-3-319-23820-3_14.

[44] Arend Rensink. "Representing First-Order Logic Using Graphs". In: *Graph Transformations*. Ed. by Hartmut Ehrig et al. Lecture Notes in Computer

Science. Berlin, Heidelberg: Springer, 2004, pp. 319–335. ISBN: 978-3-540-30203-2. DOI: 10.1007/978-3-540-30203-2_23.

[45] Patrice C Roy, Samina Raza Abidi, and Syed Sibte Raza Abidi. "Monitoring Medication Adherence in Smart Environments in the Context of Patient Self-Management: A Knowledge-Driven Approach". In: *Smart Technologies in Healthcare*. CRC Press, 2017, pp. 195–223. ISBN: 978-1-315-14568-6. DOI: 10.1201/9781315145686-8.

[46] John Rushby. "Critical System Properties: Survey and Taxonomy". In: *Reliability Engineering & System Safety*. Special Issue on Software Safety 43.2 (Jan. 1, 1994), pp. 189–219. ISSN: 0951-8320. DOI: 10.1016/0951-8320(94)90065-5.

[47] Lucas Sakizloglou. "Evaluating Temporal Queries over History-Aware Architectural Runtime Models". PhD thesis. Universität Potsdam, 2023. DOI: 10.25932/publishup-60439.

[48] Lucas Sakizloglou, Matthias Barkowsky, and Holger Giese. "Keeping Pace with the History of Evolving Runtime Models". In: *Fundamental Approaches to Software Engineering*. Ed. by Esther Guerra and Mariëlle Stoelinga. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2021, pp. 262–268. ISBN: 978-3-030-71500-7. DOI: 10.1007/978-3-030-71500-7_13.

[49] Lucas Sakizloglou et al. "Incremental Execution of Temporal Graph Queries over Runtime Models with History and Its Applications". In: *Softw Syst Model* 21.5 (Oct. 1, 2022), pp. 1789–1829. ISSN: 1619-1374. DOI: 10.1007/s10270-021-00950-6.

[50] Sven Schneider et al. "Formal Testing of Timed Graph Transformation Systems Using Metric Temporal Graph Logic". In: *Int J Softw Tools Technol Transfer* 23.3 (June 2021), pp. 411–488. ISSN: 1433-2779, 1433-2787. DOI: 10.1007/s10009-020-00585-w.

[51] Sven Schneider et al. "Optimistic and Pessimistic On-the-fly Analysis for Metric Temporal Graph Logic". In: *Graph Transformation*. Ed. by Fabio Gadducci and Timo Kehrer. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2020, pp. 276–294. ISBN: 978-3-030-51372-6. DOI: 10.1007/978-3-030-51372-6_16.

[52] Michael Szvetits and Uwe Zdun. "Systematic Literature Review of the Objectives, Techniques, Kinds, and Architectures of Models at Runtime". In: *Softw Syst Model* 15.1 (Feb. 1, 2016), pp. 31–69. ISSN: 1619-1374. DOI: 10.1007/s10270-013-0394-9.

[53] Gabriele Taentzer and Arend Rensink. "Ensuring Structural Constraints in Graph-Based Models with Type Inheritance". In: *Fundamental Approaches to Software Engineering*. Ed. by Maura Cerioli. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2005, pp. 64–79. ISBN: 978-3-540-31984-9. DOI: 10.1007/978-3-540-31984-9_6.

[54] Dániel Varró et al. "Road to a Reactive and Incremental Model Transformation Platform: Three Generations of the Viatra Framework". In:

Softw Syst Model 15.3 (July 1, 2016), pp. 609–629. ISSN: 1619-1374. DOI: 10.1007/s10270-016-0530-4.

[55] Danny Weyns and Radu Calinescu. "Tele Assistance: A Self-Adaptive Service-Based System Exemplar". In: *2015 IEEE/ACM 10th International Symposium on Software Engineering for Adaptive and Self-Managing Systems*. May 2015, pp. 88–92. DOI: 10.1109/SEAMS.2015.27.