Proceedings of the 23rd International Conference on Principles of Knowledge Representation and Reasoning
Main Track

Inferring High-Level Events from Timestamped Data: Complexity and Medical Applications

Yvon K. Awuklu[1,2,3,5], **Meghyn Bienvenu**[1], **Katsumi Inoue**[4], **Vianney Jouhet**[2,3], **Fleur Mougin**[3]
[1]Univ. Bordeaux, CNRS, Bordeaux INP, LaBRI, UMR 5800, F-33400, Talence, France
[2]CHU de Bordeaux, Service d'Information Médicale, F-33000, Bordeaux, France
[3]Univ. Bordeaux, INSERM, BPH, U1219, F-33000, Bordeaux, France
[4]National Institute of Informatics, Tokyo, Japan
[5]RIKEN BDR, Kobe, 650-0047, Japan

Abstract

In this paper, we develop a novel logic-based approach to detecting high-level temporally extended events from timestamped data and background knowledge. Our framework employs logical rules to capture existence and termination conditions for simple temporal events and to combine these into meta-events. In the medical domain, for example, disease episodes and therapies are inferred from timestamped clinical observations, such as diagnoses and drug administrations stored in patient records, and can be further combined into higher-level disease events. As some incorrect events might be inferred, we use constraints to identify incompatible combinations of events and propose a repair mechanism to select preferred consistent sets of events. While reasoning in the full framework is intractable, we identify relevant restrictions that ensure polynomial-time data complexity. Our prototype system implements core components of the approach using answer set programming. An evaluation on a lung cancer use case supports the interest of the approach, both in terms of computational feasibility and positive alignment of our results with medical expert opinions. While strongly motivated by the needs of the healthcare domain, our framework is purposely generic, enabling its reuse in other areas.

1 Introduction

The adoption of electronic health records (EHRs) has significantly improved access to vast amounts of clinical data. While this access is invaluable for healthcare delivery and research, it also introduces new challenges due to the inherent complexity of medical data (Tsai et al. 2020; Rance et al. 2016), and in particular, its inherently temporal nature (Augusto 2005; Zhou and Hripcsak 2007; Li et al. 2020; Awuklu et al. 2025). In practice, data in EHRs is recorded as a sequence of time-stamped observations, such as test results, diagnoses, and treatments. Each observation implicitly refers to a clinical event—an underlying phenomenon such as a disease episode or a therapeutic intervention—that is relevant to the patient's care (Hripcsak and Albers 2013), but these events are not explicitly documented in EHRs. Physicians routinely infer clinical events using their medical expertise, which is rarely formalized in the data. For example, observing repeated antibiotic intake may lead a clinician to deduce the presence of a bacterial infection. Here, the data shows the observation (antibiotic intake), while the associated clinical events (bacterial infection, antibiotic therapy) and the reasoning process remain implicit. However, as the volume of observations per patient can overwhelm clinicians, there is a need for information systems to adopt a higher-level, event-based representation more closely aligned with clinical reasoning and decision-making. Moreover, given the high-stakes nature of healthcare decisions and medical research, the event inference process should also be transparent, making it possible to trace inferred high-level events back to the original observations. These considerations motivate us to develop a new logic-based framework for identifying high-level temporally extended events, designed with the medical domain in mind yet sufficiently generic to enable reuse in other areas.

Overview of the Logical Framework We briefly outline the main intuitions and components of our framework, which borrows ideas from various existing KR formalisms (see Section 6 for a comparison with related work). Recall that our aim is to be able to infer temporally-extended events like $\mathsf{ABTherapy}(p, d, [t_1, t_2])$, expressing that patient p receives antibiotic therapy with drug d during the time period $[t_1, t_2]$, from the timestamped observations in medical data. The starting point for our proposal is the realisation that while it can be quite difficult for medical experts to directly specify the interval endpoints (t_1, t_2), it is typically much easier for them to supply *existence conditions* which ensure (or suggest) that a given event is ongoing at timepoint t, e.g. the observation $\mathsf{DrugAdmin}(p, d, t)$ available in patient p's records indicates that a therapy with drug d is ongoing at time t. In some cases, experts may also be able to formulate *termination conditions* that indicate that an event (might or must) end at a given timepoint. For example, hospital records may contain $\mathsf{DrugStopNotification}(p, d, t)$, stating that the prescription for drug d was terminated at time t, indicating an end to the therapy with d. In our work, we will use the term *simple event* to refer to events for which we can define existence conditions (and optionally termination conditions) in terms of data predicates, without reference to other events.

How can we use the existence and termination conditions to identify the intervals of simple events? Here, we shall distinguish two kinds of simple events: *persistent* and *non-persistent*. Intuitively, persistent events continue once initiated until some termination condition is met (similar to the

principle of inertia in reasoning about actions (Reiter 2001; Shanahan 1997)), while non-persistent events require regular observations to endure. For persistent events, we consider the maximal intervals starting from a timepoint verifying the existence condition and continuing until some termination condition (if present) is reached (else marking the event as ongoing). For non-persistent events, the idea is to group together timepoints satisfying existence conditions if they are sufficiently close (with 'closeness' being determined by a provided time window), ending an event when a termination condition is reached or there are no further timepoints in the vicinity that satisfy existence conditions.

Our framework also supports *meta-events*, defined from simple events and possibly other meta-events. Such higher-level events allow one e.g. to group disease episodes or identify concurrent conditions or treatments. They also make it possible to present events at different levels of abstraction (e.g. DrugTherapy generalizes ABTherapy). Moreover, since it is difficult to provide existence and termination conditions which are both fully accurate and ensure sufficient coverage, our framework allows for event facts to be annotated with confidence levels, based upon the rules which created them. Constraints can also be used to specify consistency requirements, and a *repair mechanism* employed to select consistent combinations of events.

Contributions Our first contribution is the formal definition of a novel logic-based language for temporal event detection, which uses rules to specify the existence and termination conditions for simple events and for defining meta-events in terms of other events, as well as constraints to define consistency requirements. The semantics defines four different kinds of timelines (naïve, consistent, preferred, cautious), with each timeline consisting of simple event and meta-event facts that can be inferred from the rules, possibly taking into account the constraints and confidence levels.

As our second contribution, we explore the computational properties of our framework, presenting algorithms and complexity results for recognizing and generating the different kinds of timeline. Given the expressivity of the framework, we show unsurprisingly that the consistent, preferred, and cautious cannot be tractably recognized. However, we also identify a relevant fragment for which there is a unique preferred timeline, and for which both the preferred timeline and cautious timeline can be efficiently computed.

As a third contribution, we implemented core components of the framework using answer set programming (ASP), a well-known declarative programming paradigm (Brewka, Eiter, and Truszczynski 2011; Gebser et al. 2012; Lifschitz 2019). An experimental evaluation on a cancer use case involving hospital data demonstrates computational feasibility, and feedback from medical experts supports the clinical plausibility of the inferred events.

Paper Organization Section 2 defines the syntax and semantics of our logical framework, while Section 3 provides algorithms and complexity results for the relevant reasoning tasks. Sections 4 and 5 present respectively the implemented system and its evaluation on a medical use case. Discussions of related approaches and future work are given in Sections 6 and 7. Omitted proofs and further details on the modelling and evaluation of the medical case study are provided in the appendix of the long version (Awuklu et al. 2026).

2 A Logical Framework for Event Detection

We formalize our logical framework for specifying high-level events from temporal observations and expert knowledge, motivated and illustrated by medical applications. We shall assume that readers are acquainted with the basics of first-order logic and logic programming.

2.1 Logical Vocabulary

We use three disjoint sets of first-order predicates to describe the domain: a set $\mathbf{R}_{\mathsf{A}}$ of atemporal predicates, a set of $\mathbf{R}_{\mathsf{O}}$ of observation predicates, and a set $\mathbf{R}_{\mathsf{E}}$ of event predicates, further partitioned into the sets $\mathbf{R}_{\mathsf{PS}}$, $\mathbf{R}_{\mathsf{NS}}$, and $\mathbf{R}_{\mathsf{M}}$ of persistent simple event predicates, non-persistent simple event predicates, and meta-event predicates. Each predicate $R \in \mathbf{R}_{\mathsf{A}} \cup \mathbf{R}_{\mathsf{O}} \cup \mathbf{R}_{\mathsf{E}}$ has an arity $k \geq 0$, corresponding to its number of atemporal arguments. We use $\mathbf{R}^k_{\mathsf{A}}$ (resp. $\mathbf{R}^k_{\mathsf{O}}$, $\mathbf{R}^k_{\mathsf{E}}$) for the set of k-ary predicates in $\mathbf{R}_{\mathsf{A}}$ (resp. $\mathbf{R}_{\mathsf{O}}, \mathbf{R}_{\mathsf{E}}$). The atemporal and observation predicates correspond to the predicates occurring in the data, the key difference being that observation predicates have a timepoint as final argument (to capture timestamped facts). The event predicates are used for the inferred events and will contain a temporal interval as argument to indicate the start and end of the event.

For simplicity, we use natural numbers to represent timepoints (and positive integers for confidence levels and temporal windows), so we assume the set $\mathbf{C}$ of *constants* contains $\mathbb{N}$. *Temporal intervals* take the form $[t_1, t_2]$ where $t_1 \in \mathbb{N}$, $t_2 \in \mathbb{N} \cup \{*\}$, and $t_1 \leq t_2$ if $t_2 \neq *$, where the special symbol $*$ is used to indicate events which are still ongoing. Our rules will use *variables* drawn from a set $\mathbf{V}$. To ensure variables are appropriately instantiated, we assume $\mathbf{V}$ is partitioned into four subsets $\mathbf{V}_{\mathsf{d}}$, $\mathbf{V}_{\mathsf{n}}$, $\mathbf{V}^+_{\mathsf{n}}$, and $\mathbf{V}^*_{\mathsf{n}}$ whose variables must be instantiated respectively by elements of $\mathbf{C}$, $\mathbb{N}$, $\mathbb{N}^+$, and $\mathbb{N} \cup \{*\}$. We also allow for sets $\mathbf{F}_{\mathsf{n}}$, $\mathbf{F}^+_{\mathsf{n}}$, and $\mathbf{F}_{\mathsf{int}}$ of functions to manipulate (positive) natural numbers or intervals. For example, we will use max to aggregate confidence levels and inter to compute the intersection of two intervals. Finally, we can define the following sets of *terms*: (i) $\mathbf{T}_{\mathsf{d}} = \mathbf{C} \cup \mathbf{V}_{\mathsf{d}}$, (ii) $\mathbf{T}_{\mathsf{n}}$ (resp. $\mathbf{T}^+_{\mathsf{n}}$) is defined as the closure of $\mathbb{N} \cup \mathbf{V}_{\mathsf{n}}$ (resp. $\mathbb{N}^+ \cup \mathbf{V}^+_{\mathsf{n}}$) under applications of functions in $\mathbf{F}_{\mathsf{n}}$ (resp. $\mathbf{F}^+_{\mathsf{n}}$), and (iii) $\mathbf{T}_{\mathsf{int}}$ is obtained by closing $\{[t_1, t_2] \mid t_1 \in \mathbb{N} \cup \mathbf{V}_{\mathsf{n}}, t_2 \in \mathbb{N} \cup \{*\} \cup \mathbf{V}^*_{\mathsf{n}}, t_1 \leq t_2 \text{ if } t_1, t_2 \in \mathbb{N}\}$ under functions in $\mathbf{F}_{\mathsf{int}}$.

An *atemporal atom* has the form $R(u_1, \ldots, u_k)$ where $R \in \mathbf{R}^k_{\mathsf{A}}$, and $u_1, \ldots, u_k \in \mathbf{T}_{\mathsf{d}}$. An *observation atom* has the form $R(u_1, \ldots, u_k, t)$ where $R \in \mathbf{R}^k_{\mathsf{O}}$, $u_1, \ldots, u_k \in \mathbf{T}_{\mathsf{d}}$ and $t \in \mathbf{T}_{\mathsf{n}}$. An *event atom* has the form $R(u_1, \ldots, u_k, \iota)$, where $R \in \mathbf{R}^k_{\mathsf{E}}$, $u_1, \ldots, u_k \in \mathbf{T}_{\mathsf{d}}$, $\iota \in \mathbf{T}_{\mathsf{int}}$. Atoms without variables are called *facts*. A *dataset* is a finite set of atemporal and observation facts.

We will attach *confidence levels* (from $\mathbb{N}^+$) to event facts based upon how they were generated. Note that to easily identify the most reliable facts, we fix 1 as the best confidence level (*thus, higher numbers will denote lower confi-*

dence). A *(confidence-)annotated event atom* takes the form $R(\mathbf{u}, \iota, \ell)$, where $R(\mathbf{u}, \iota)$ is an event atom and $\ell \in \mathbf{T}_{\mathsf{n}}^{+}$.

Finally, as will be detailed next, we shall employ special auxiliary predicates exists, ends, and window to define existence and termination conditions and temporal windows.

2.2 Specifying Events via Rules

We now introduce the syntax of rules used to define simple and meta-event predicates.

Simple Events For simple non-persistent events, we must provide the conditions that allow us to infer that such the event holds (or ends) at a given timepoint, as well as defining a temporal window in order to know how to define the event intervals. Formally, a *ruleset for a non-persistent simple event predicate* $R \in \mathbf{R}_{\mathsf{NS}}^{k}$ is a set of rules consisting of:

- one or more *existence rules* of the form[1]

$$\mathsf{exists}(R(u_1, \ldots, u_k), t, \ell) \leftarrow B$$

- zero or more *termination rules* of the form

$$\mathsf{ends}(R(u_1, \ldots, u_k), t, \ell) \leftarrow B$$

- one or more *(expansion) window rules* of the form

$$\mathsf{window}(R(u_1, \ldots, u_k), w) \leftarrow B$$

where $u_1, \ldots, u_k \in \mathbf{T}_{\mathsf{d}}$, $t \in \mathbf{T}_{\mathsf{n}}$ (specifying a timepoint), $\ell \in \mathbb{N}^{+}$ (giving the confidence level of the rule, with the convention that 1 denotes greatest confidence), and $w \in \mathbf{T}_{\mathsf{n}}^{+}$ (defining a temporal window). Rule bodies B take the form of conjunctions whose conjuncts may be (possibly negated) atemporal and observation atoms, inequality atoms $z \neq z'$ between terms $z, z' \in \mathbf{T}_{\mathsf{d}}$, or inequality / comparison atoms $z \bowtie z'$ for $z, z' \in \mathbf{T}_{\mathsf{n}}$ and $\bowtie \in \{\neq, <, \leq\}$. Rules are required to be *safe*: every variable in a rule must occur either in some unnegated atemporal or observation body atom. Additionally, window rules should provide a unique window value $w > 0$ for each $R(c_1, \ldots, c_k)$ that exists due to the existence rules (this is made formal in Section 2.3). The simplest way to accomplish this is to assign each predicate R a fixed window, but it can be useful to be able to assign different windows based upon the event arguments.

Example 1. *For the simple event* ABTherapy *(shortened to* ABTh*), we could use the following rules:*

$$\mathsf{exists}(\mathsf{ABTh}(p, d), t, 1) \leftarrow \mathsf{Adm}(p, d, t) \wedge \mathsf{AB}(d)$$
$$\mathsf{ends}(\mathsf{ABTh}(p, d), t, 1) \leftarrow \mathsf{Stop}(p, d, t) \wedge \mathsf{AB}(d)$$
$$\mathsf{window}(\mathsf{ABTh}(p, d), 48) \leftarrow \mathsf{P}(p) \wedge \mathsf{AB}(d)$$

*The existence condition looks for administrations (*Adm*) of an antibiotic (*AB*) drug. The termination rule applies when there is a drug stop notification (*Stop*). A fixed window of 48 hours is defined, but one could use multiple rules with different windows to e.g. differentiate by the class of antibiotics.*

[1] In line with standard notations for logics for reasoning about actions, auxiliary predicates may have atoms $R(u_1, \ldots, u_n)$ of arbitrary arity as arguments. Alternatively, we could express the same thing in classical logic using reification and multiple copies of the auxiliary predicates to accommodate atoms of different arities. We write rules right-to-left as typical in logic programming.

We proceed similarly for persistent simple events, the main difference being that no window is needed. A *ruleset for a persistent simple event predicate* $R \in \mathbf{R}_{\mathsf{PS}}^{k}$ thus comprises one or more existence rules and (optionally) termination rules (having the same syntactic form as before).

Example 2. *Tyrosine kinase inhibitor (TKI) therapy is used in specific cases of lung cancer. It is intended to be taken for life unless it leads to toxicity or proves ineffective, in which case a patient is switched to another TKI drug. We model TKI therapy (*TKITh*) as a persistent simple event:*

$$\mathsf{exists}(\mathsf{TKITh}(p, d), t, 1) \leftarrow \mathsf{Adm}(p, d, t) \wedge \mathit{TKI}(d)$$
$$\mathsf{exists}(\mathsf{TKITh}(p, d), t, 2) \leftarrow \mathsf{Presc}(p, d, t) \wedge \mathit{TKI}(d)$$
$$\mathsf{ends}(\mathsf{TKITh}(p, d), t, 1) \leftarrow \mathsf{Adm}(p, d', t') \wedge \mathit{TKI}(d') \wedge \mathit{TKI}(d) \wedge d' \neq d$$

Both administration and prescription of a TKI can be used to infer existence, but prescription is less reliable.

Meta-Events Unlike simple events, which are defined directly from the data, the definition of meta-event predicates may refer to simple events and other meta-events. A ruleset for $\mathbf{R}_{\mathsf{M}}$ contains rules of the following form, for $R \in \mathbf{R}_{\mathsf{M}}^{k}$:

$$R(u_1, \ldots, u_k, \iota, \ell) \leftarrow B$$

where $u_1, \ldots, u_k \in \mathbf{T}_{\mathsf{d}}$, $\iota \in \mathbf{T}_{\mathsf{int}}$ (specifying the temporal interval), $\ell \in \mathbf{T}_{\mathsf{n}}^{+}$ (specifying confidence level), and the rule body B may use the same kinds of conjuncts as for simple event rules, as well as (possibly negated) confidence-annotated event atoms. Safety of rules is defined as before, except that now unnegated event atoms may also be used to restrict the range of variables.

Note that a predicate $R \in \mathbf{R}_{\mathsf{M}}$ may occur both in the head of rules and (possibly negated) in rule bodies. In our considered medical scenarios, we found it sufficient to consider *stratified rulesets*, for which there exists a total preorder $\preceq$ on the predicates such that if a rule has head predicate R' and body predicate R' then $R' \preceq R$, and if R' occurs in negated body atom, then $R' \prec R$ (Apt, Blair, and Walker 1988). We thus impose this stratification condition, which ensures that the meta-event facts are uniquely determined from the simple event facts and dataset.

Example 3. *Negation can model exclusion conditions. Hyperglycemia (*HyperGlyc*) during pregnancy (*Preg*) should not yield gestational diabetes (*GestDiab*) when pre-existing diabetes (*PreDiab*) is already active at onset. We encode this by introducing an auxiliary predicate detecting diabetes at hyperglycemia onset and excluding such cases in the definition of* GestDiab*:*

$$\mathsf{PreDiabAtOnset}(p, [t_1, t_2], \ell_1) \leftarrow \mathsf{HyperGlyc}(p, [t_1, t_2], \ell_1) \wedge \mathsf{PreDiab}(p, [t_1', t_2'], \ell_2) \wedge t_1' \leq t_1 \wedge t_1 \leq t_2' \wedge \ell_2 \leq \ell_1$$
$$\mathsf{GestDiab}(p, \mathsf{inter}([t_1, t_2], [t_3, t_4]), \mathsf{max}(\ell_1, \ell_2)) \leftarrow \mathsf{Preg}(p, [t_1, t_2], \ell_1) \wedge \mathsf{HyperGlyc}(p, [t_3, t_4], \ell_2) \wedge\ t_1 \leq t_3 \wedge t_3 \leq t_2 \wedge \neg\, \mathsf{PreDiabAtOnset}(p, [t_3, t_4], \ell_2)$$

Constraints on Events To enforce consistency of the inferred events, we consider two types of constraints: domain-independent temporal constraints and domain-specific constraints. The fixed set of *temporal constraints* Υ_{temp} ensures that simple events having the same predicate and atemporal arguments (but possibly different confidence levels) cannot have intervals that non-trivially overlap. Υ_{temp} contains the following constraints for every $R \in \mathbf{R}^k_{\mathsf{PS}} \cup \mathbf{R}^k_{\mathsf{NS}}$:

$$\bot \leftarrow R(\boldsymbol{u}, [t_1, t_2]) \ \wedge R(\boldsymbol{u}, [t'_1, t'_2]) \wedge t_1 < t'_1 \wedge t'_1 < t_2$$
$$\bot \leftarrow R(\boldsymbol{u}, [t_1, t_2]) \ \wedge R(\boldsymbol{u}, [t_1, t'_2]) \wedge t_2 \neq t'_2$$
$$\bot \leftarrow R(\boldsymbol{u}, [t_1, t_2]) \ \wedge R(\boldsymbol{u}, [t'_1, t_2]) \wedge t_1 \neq t'_1$$

where $\boldsymbol{u}$ abbreviates the tuple of variables $u_1, \ldots, u_k$ (for distinct $u_i \in \mathbf{V}_{\mathsf{d}}$). We assume that *domain-specific constraints* (if any) have the form $\bot \leftarrow C$, with C a conjunctive formula, whose conjuncts are (possibly negated) atemporal, observation, or (unannotated) event atoms, inequality or comparison atoms, subject to the usual safety condition.

Example 4. *In a medical setting, we may use the following domain constraint enforce that a patient cannot simultaneously undergo two targeted therapies with different TKIs:*

$$\bot \leftarrow \mathsf{TKITh}(p, d_1, [t_1, t_2]) \wedge \mathsf{TKITh}(p, d_2, [t'_1, t'_2]) \\ \wedge d_1 \neq d_2 \wedge t_1 < t'_1 \wedge t'_1 < t_2 \wedge t_2 < t'_2$$

Temporal Event Specifications We now have all the elements needed to define temporal event specifications:

Definition 1. *A* temporal event specification (TES) *takes the form* $\Sigma = (\Pi_{\mathsf{SE}}, \Pi_{\mathsf{ME}}, \Upsilon_{\mathsf{temp}}, \Upsilon_{\mathsf{dom}})$*, where:*

- $\Pi_{\mathsf{SE}} = \bigcup_{R \in \mathbf{R}_{\mathsf{NS}} \cup \mathbf{R}_{\mathsf{PS}}} \Pi_R$*, with* Π_R *a ruleset for* R
- Π_{ME} *is a ruleset for* $\mathbf{R}_{\mathsf{M}}$
- Υ_{temp} *is the fixed set of temporal constraints*
- Υ_{dom} *is a set of domain-specific constraints*

2.3 Semantics of the Framework

It remains to make precise which event facts are generated from a given TES and dataset. We shall present the semantics in stages, starting with simple events.

Semantics of Simple Events To define the exists-, ends-, window-facts that hold in a dataset $\mathcal{D}$, we simply evaluate rule bodies (seen as first-order formulas) in $\mathcal{D}$ (seen as a first-order structure). Given a rule body B over variables $\boldsymbol{u} \cup \boldsymbol{v}$, we let $\textsc{Eval}(B, \boldsymbol{v}, \mathcal{D}) = \{\boldsymbol{c} \mid \mathcal{D} \models \exists \boldsymbol{u}\, B[\boldsymbol{v} : \boldsymbol{c}]\}$, where $B[\boldsymbol{v} : \boldsymbol{c}]$ is B with variables $\boldsymbol{v}$ replaced by the constants in $\boldsymbol{c}$.

Definition 2. *Given a TES* $\Sigma = (\Pi_{\mathsf{SE}}, \Pi_{\mathsf{ME}}, \Upsilon_{\mathsf{temp}}, \Upsilon_{\mathsf{dom}})$ *and dataset* $\mathcal{D}$*, we define* $\Pi_{\mathsf{SE}}(\mathcal{D})$ *as the set of all facts* $\mathsf{p}(\boldsymbol{c})$ *s.t.* $\boldsymbol{c} \in \textsc{Eval}(B, \boldsymbol{v}, \mathcal{D})$ *for some rule* $\mathsf{p}(\boldsymbol{v}) \leftarrow B(\boldsymbol{u} \cup \boldsymbol{v}) \in \Pi_{\mathsf{SE}}$*. For a predicate* $R \in \mathbf{R}^k_{\mathsf{NS}}$*,* k*-tuple* $\boldsymbol{d}$ *of data constants, and confidence level* ℓ*, we define:*

$$T^{\ell}_{\exists}(R(\boldsymbol{d})) = \{t \mid \mathsf{exists}(R(\boldsymbol{d}), t, \ell') \in \Pi_{\mathsf{SE}}(\mathcal{D}), \ell' \leq \ell\}$$
$$T^{\ell}_{\times}(R(\boldsymbol{d})) = \{t \mid \mathsf{ends}(R(\boldsymbol{d}), t, \ell') \in \Pi_{\mathsf{SE}}(\mathcal{D}), \ell' \leq \ell\}$$

We say that $\Pi_{\mathsf{SE}}(\mathcal{D})$ *is* valid *if whenever* $\mathsf{exists}(R(\boldsymbol{d}), t, \ell) \in \Pi_{\mathsf{SE}}(\mathcal{D})$*, there is a unique* $\mathsf{window}(R(\boldsymbol{d}), w) \in \Pi_{\mathsf{SE}}(\mathcal{D})$.

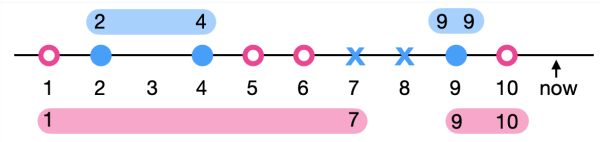
Recall that for non-persistent simple events the idea is to construct intervals for an event by starting from the timepoints where the event is stated to exist (given by $T^{\ell}_{\exists}(R(\boldsymbol{d}))$), then iteratively 'expanding' these intervals to incorporate nearby timepoints (with 'near' defined by $\mathsf{window}(R(\boldsymbol{d}), w)$) stopping either when no further such timepoint is encountered, or when a termination condition ($T^{\ell}_{\times}(R(\boldsymbol{d}))$) is reached. As we shall assume that $\Pi_{\mathsf{SE}}(\mathcal{D})$ is valid (cf. previous definition), it is always clear which window to use. The following definition formalizes this idea.

Definition 3. *A fact* $R(\boldsymbol{d}, [t_1, t_2])$ *is* inferred from $(\Pi_{\mathsf{SE}}, \mathcal{D})$ with confidence ℓ*, denoted* $\Pi_{\mathsf{SE}}, \mathcal{D} \models_{\ell} R(\boldsymbol{d}, [t_1, t_2])$*, if:*

1. $\mathsf{window}(R(\boldsymbol{d}), w) \in \Pi_{\mathsf{SE}}(\mathcal{D})$
2. *there exists* $t'_0, \ldots, t'_n \in T^{\ell}_{\exists}(R(\boldsymbol{d}))$ *such that* $t'_0 = t_1$ *and* $t'_{i+1} - t'_i \leq w$ *for every* $0 \leq i < n$
3. *there is no* $t^{\dagger} \in T^{\ell}_{\times}(R(\boldsymbol{d}))$ *such that* $t_1 \leq t^{\dagger} < t_2$
4. *for every* $t^{\sharp}_1 \in T^{\ell}_{\exists}(R(\boldsymbol{d}))$ *with* $t_1 - w \leq t^{\sharp}_1 < t_1$*, there exists* $t_e \in T^{\ell}_{\times}(R(\boldsymbol{d}))$ *with* $t^{\sharp}_1 \leq t_e < t_1$
5. *if* $t_2 = t'_n$*, then there is no* $t^{\dagger} \in T^{\ell}_{\exists}(R(\boldsymbol{d})) \cup T^{\ell}_{\times}(R(\boldsymbol{d}))$ *with* $t'_n < t^{\dagger} \leq t'_n + w$
6. *if* $t_2 \neq t'_n$*, then* $t_2 \in T^{\ell}_{\times}(R(\boldsymbol{d}))$ *and* $t_2 - t'_n \leq w$
7. *there is no* $\ell' < \ell$ *such that* $\Pi_{\mathsf{SE}}, \mathcal{D} \models_{\ell'} R(\boldsymbol{d}, [t_1, t_2])$

We briefly explain the role of the different items of the preceding definition. Item 1 defines the (unique) time window w associated with the event predicate R. This window determines the maximal temporal distance allowed between consecutive observations that belong to the same event occurrence. Item 2 ensures that there is a sequence of timepoints in $T^{\ell}_{\exists}(R(\boldsymbol{d}))$ that are sufficiently dense (adjacent timepoints are within distance w) and cover the interval $[t_1, t'_n]$. Item 3 prevents the inferred interval from crossing a termination condition by requiring that no termination point lies strictly between its start and end. Items 4 and 5 ensure that the interval could not have been extended further in either direction by using an earlier or later timepoint from $T^{\ell}_{\exists}(R(\boldsymbol{d}))$. Item 6 makes sure that if $t_2 \neq t'_n$ then t_2 satisfies a termination condition (e.g., a drug stop notification). Finally, item 7 prevents redundant inference of the exact same event with different confidence levels.

Example 5. *Suppose for event* E*, we have* $T^1_{\exists}(E) = \{2, 4, 9\}$*,* $T^2_{\exists}(E) = \{1, 5, 6, 10\}$*, and* $T^1_{\times}(E) = \{7, 8\}$*:*



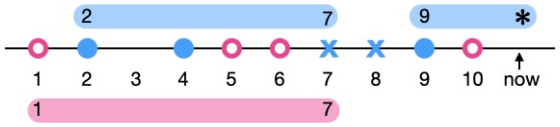
In this timeline, each ***dot*** *represents a timepoint in* $T^{\ell}_{\exists}(E)$ *where an* existence condition *for* E *is satisfied with confidence level* ℓ *(blue dots for* $\ell = 1$*, pink dots for* $\ell = 2$*), while each* ***cross*** *marks a timepoint in* $T^{\ell}_{\times}(E)$ *where a* termination condition *holds. With a window of* 2 *and current time* 11*, we get the blue intervals* $[2, 4]$ *and* $[9, 9]$ *with confidence* 1 *and the pink intervals* $[1, 7]$ *and* $[9, 10]$ *with confidence* 2*.*

Defining intervals for persistent simple events is simpler since they continue until a termination condition is reached.

Definition 4. *Consider $R \in \mathbf{R}^k_{\mathsf{PS}}$ defined in Π_{SE}, and let $\mathcal{D}$, $\boldsymbol{d}$, $[t_1, t_2]$, $T^\ell_\exists(R(\boldsymbol{d}))$ and $T^\ell_\times(R(\boldsymbol{d}))$ be as in Definitions 2 and 3. Then $R(\boldsymbol{d}, [t_1, t_2])$ is* inferred from $(\Pi_{\mathsf{SE}}, \mathcal{D})$ with confidence ℓ, *denoted $\Pi_{\mathsf{SE}}, \mathcal{D} \models_\ell R(\boldsymbol{d}, [t_1, t_2])$, if:*

1. $t_1 \in T^\ell_\exists(R(\boldsymbol{d}))$
2. *if* $t_2 \neq *$, *then* $t_2 \in T^\ell_\times(R(\boldsymbol{d}))$
3. *for every* $t_1^\sharp \in T^\ell_\exists(R(\boldsymbol{d}))$ *with* $t_1^\sharp < t_1$, *there exists* $t_e \in T^\ell_\times(R(\boldsymbol{d}))$ *with* $t_1^\sharp \leq t_e < t_1$
4. *if* $t_2 \neq *$, *there is no* $t_2^\sharp \in T^\ell_\times(R(\boldsymbol{d}))$ *with* $t_1 \leq t_2^\sharp < t_2$
5. *if* $t_2 = *$, *there is no* $t_2^\sharp \in T^\ell_\times(R(\boldsymbol{d}))$ *with* $t_1 \leq t_2^\sharp$
6. *there is no* $\ell' < \ell$ *such that* $\Pi_{\mathsf{SE}}, \mathcal{D} \models_{\ell'} R(\boldsymbol{d}, [t_1, t_2])$

Example 6. *Take the preceding example but now let E be a persistent event:*



The inferred intervals become $[2, 7]$ and $[9, *]$ for confidence 1, and $[1, 7]$ for confidence 2.

To facilitate later definitions, we introduce some notation for referring to the set of inferred simple event facts (with and without their associated confidence levels):

$$\mathsf{SE}(\mathcal{D}, \Sigma) = \{R(\boldsymbol{d}, [t_1, t_2], \ell) \,|\, \Pi_{\mathsf{SE}}, \mathcal{D} \models_\ell R(\boldsymbol{d}, [t_1, t_2])\}$$
$$\mathsf{SE}^-(\mathcal{D}, \Sigma) = \{R(\boldsymbol{d}, [t_1, t_2]) \,|\, \Pi_{\mathsf{SE}}, \mathcal{D} \models_\ell R(\boldsymbol{d}, [t_1, t_2])\}$$

Given any set $\mathcal{S}$ of confidence-annotated facts, we let $\mathcal{S}^-$ be the result of removing the confidence levels from all facts in $\mathcal{S}$, and let $\mathcal{S}_\ell$ be the set of facts in $\mathcal{S}$ with confidence ℓ.

Inferring Meta-Events Due to our decision to use stratified rulesets to define meta-event predicates, there will be a unique set of meta-event facts that can be inferred from a dataset and a given set of simple event facts:

Definition 5. *Consider a dataset $\mathcal{D}$, set $\mathcal{S}$ of confidence-annotated simple event facts, and TES Σ with ruleset Π_{ME} for $\mathbf{R}_{\mathsf{M}}$. The set $\mathsf{ME}(\mathcal{D}, \mathcal{S}, \Sigma)$ of* inferred confidence-annotated meta-event facts *contains all facts $R(\boldsymbol{d}, [t_1, t_2], \ell)$ with $R \in \mathbf{R}_{\mathsf{M}}$ appearing in the unique stratified model[2] of $(\mathcal{D}, \mathcal{S}, \Pi_{\mathsf{ME}})$. The set $\mathsf{ME}^-(\mathcal{D}, \mathcal{S}, \Sigma)$ is obtained from $\mathsf{ME}(\mathcal{D}, \mathcal{S}, \Sigma)$ by removing the confidence levels.*

Repairing Sets of Simple Events So far we have defined inferred events without paying attention to the constraints. Even if the underlying dataset is accurate, the set of inferred simple events may violate the domain constraints due to lower confidence rules, which may sometimes incorrectly suggest the existence or termination of a simple event. Additionally, the temporal constraints may be violated by 'duplicate' events derived with different confidence levels.

[2]Stratified models are defined by evaluating the rules according to the ordering of the predicates (Apt, Blair, and Walker 1988; Dantsin et al. 2001), so that it is always clear how to interpret negated atoms in rule bodies.

We thus propose to repair the set of inferred simple events so that they satisfy the constraints. First, we make clear how we define consistency w.r.t. a TES:

Definition 6. *Given a TES $\Sigma = (\Pi_{\mathsf{SE}}, \Pi_{\mathsf{ME}}, \Upsilon_{\mathsf{temp}}, \Upsilon_{\mathsf{dom}})$, we say a set $\mathcal{S}$ of confidence-annotated simple event facts is* Σ-consistent *if $\Upsilon_{\mathsf{temp}}, \Upsilon_{\mathsf{dom}}, \mathcal{D}, \mathcal{S}^-, \mathsf{ME}^-(\mathcal{D}, \mathcal{S}, \Sigma) \not\models \bot$ (i.e. there is no constraint $\bot \leftarrow C$ such that C evaluates to true w.r.t. $\mathcal{D} \cup \mathcal{S}^- \cup \mathsf{ME}^-(\mathcal{D}, \mathcal{S}, \Sigma)$).*

Observe that we need the annotated simple event facts ($\mathcal{S}$) to determine the associated meta-event facts ($\mathsf{ME}(\mathcal{D}, \mathcal{S}, \Sigma)$), but we strip the simple and meta-event facts of their annotations when we check for constraint violations (as the constraints involve unannotated event atoms).

Our first notion of repair simply considers the inclusion-maximal consistent sets of facts, in line with the subset repairs previously studied for databases and knowledge bases, cf. (Bertossi 2011; Bienvenu and Bourgaux 2016).

Definition 7. *Let $\mathcal{S}$ and Σ be as in Definition 6. Then $\mathcal{R} \subseteq \mathcal{S}$ is a* repair *of $\mathcal{S}$ w.r.t. Σ if (i) $\mathcal{R}$ is Σ-consistent, and (ii) there is no $\mathcal{U} \subseteq \mathcal{S}$ such that $\mathcal{R} \subsetneq \mathcal{U}$ and $\mathcal{U}$ is Σ-consistent. We use $\mathsf{Reps}(\mathcal{S}, \Sigma)$ for the set of repairs of $\mathcal{S}$ w.r.t. Σ.*

Example 7. *Consider again the running example where $T^1_\exists(E) = \{2, 4, 9\}$, $T^2_\exists(E) = \{1, 5, 6, 10\}$, and $T^1_\times(E) = \{7, 8\}$, yielding the intervals $[2, 4]$ and $[9, 9]$ (confidence 1) and $[1, 7]$ and $[9, 10]$ (confidence 2). Here, both pairs of intervals—$[2, 4]$ and $[1, 7]$, as well as $[9, 9]$ and $[9, 10]$—violate the temporal constraints because they overlap for the same event E. To restore consistency, one interval from each conflicting pair has to be selected, yielding four repairs:*

$$\begin{array}{lcl} \mathcal{R}_1 & = & \{E([2,4],1),\ E([9,9],1)\} \\ \mathcal{R}_2 & = & \{E([2,4],1),\ E([9,10],2)\} \\ \mathcal{R}_3 & = & \{E([1,7],2),\ E([9,9],1)\} \\ \mathcal{R}_4 & = & \{E([1,7],2),\ E([9,10],2)\} \end{array}$$

We also consider preferred repairs, which preferentially retain facts with better confidence levels, inspired by the $\subseteq_P$-repairs of (Bienvenu, Bourgaux, and Goasdoué 2014). Recall that the notation $\mathcal{R}_\ell$ denotes the set of facts in $\mathcal{R}$ having confidence ℓ, with $\ell = 1$ giving the most reliable facts.

Definition 8. *Let $\mathcal{S}$ and Σ be as in Definition 7, and let n be the maximum confidence level appearing in $\mathcal{S}$. Then $\mathcal{R} \subseteq \mathcal{S}$ is a* preferred repair *of $\mathcal{S}$ w.r.t. Σ if $\mathcal{R}$ is Σ-consistent and there does not exist $\mathcal{U} \subseteq \mathcal{S}$ and $1 \leq k \leq n$ such that (i) $\mathcal{U}$ is Σ-consistent, (ii) $\mathcal{U}_\ell = \mathcal{R}_\ell$ for every $1 \leq \ell < k$, and (iii) $\mathcal{R}_k \subsetneq \mathcal{U}_k$. We use $\mathsf{PrefReps}(\mathcal{S}, \Sigma)$ for the set of repairs of $\mathcal{S}$ w.r.t. Σ.*

Example 8. *Under the preferred repair semantics, conflicts between facts are resolved using the confidence level of facts. In our running example, there is a unique preferred repair, $\mathcal{R}_1$, which retains the two facts with confidence level 1.*

Note that we repair the set of simple events, using meta-events only to determine consistency, in order to avoid situations in which a repair contains a meta-event but the simple events needed to create it have been removed.

Semantics of Temporal Event Specifications We are now ready to define the semantics of a TES and dataset:

Definition 9. *Given a TES* $\Sigma = (\Pi_{\mathsf{SE}}, \Pi_{\mathsf{ME}}, \Upsilon_{\mathsf{temp}}, \Upsilon_{\mathsf{dom}})$ *and dataset* $\mathcal{D}$*:*

- *the* naïve timeline *is* $\mathsf{SE}(\mathcal{D}, \Sigma) \cup \mathsf{ME}(\mathcal{D}, \mathsf{SE}(\mathcal{D}, \Sigma), \Sigma)$
- *the* consistent timelines *take the form* $\mathcal{R} \cup \mathsf{ME}(\mathcal{D}, \mathcal{R}, \Sigma)$*, where* $\mathcal{R} \in \mathsf{Reps}(\mathsf{SE}(\mathcal{D}, \Sigma), \Sigma)$
- *the* preferred timelines *take the form* $\mathcal{P} \cup \mathsf{ME}(\mathcal{D}, \mathcal{P}, \Sigma)$*, where* $\mathcal{P} \in \mathsf{PrefReps}(\mathsf{SE}(\mathcal{D}, \Sigma), \Sigma)$
- *the* cautious timeline *takes the form* $\mathcal{I} \cup \mathsf{ME}(\mathcal{D}, \mathcal{I}, \Sigma)$*, where* $\mathcal{I} = \cap_{\mathcal{R} \in \mathsf{Reps}(\mathsf{SE}(\mathcal{D}, \Sigma), \Sigma)} \mathcal{R}$

The (unique) naïve timeline ignores the constraints and infers all annotated simple event and meta-event facts. The consistent and preferred timelines are obtained by taking a (preferred) repair and completing it with the associated meta-event facts. Finally, the (unique) cautious timeline first intersects all repairs, then adds in the inferable meta-event facts. Note that in the absence of negated event atoms, the naïve and cautious timelines provide upper and lower bounds, respectively, on the facts appearing in consistent and preferred timelines (but this does not hold in general).

3 Complexity & Algorithms

In this section, we examine the computational properties of our framework. As is common for data-centric tasks, our complexity analysis will employ *data complexity*, where only the set(s) of facts are treated as input, while the rules and constraints are treated as fixed.

In order to generate the different kinds of timelines, we first need to be able to compute the set of inferred simple event and meta-event facts. This can be done efficiently:

Theorem 1. *The sets* $\mathsf{SE}(\mathcal{D}, \Sigma)$ *and* $\mathsf{ME}(\mathcal{D}, \mathcal{S}, \Sigma)$ *can be computed in* PTIME *in data complexity.*

Proof sketch. Computing $\Pi_{\mathsf{SE}}(\mathcal{D})$ essentially corresponds to evaluation of first-order queries over a database, a task well known to be computable in (sub)polynomial time[3] in data complexity (more precisely, in $\mathsf{AC}_0 \subseteq$ LOGSPACE). Likewise, $\mathsf{ME}(\mathcal{D}, \mathcal{S}, \Sigma)$ is PTIME-computable once we have already computed $\mathsf{SE}(\mathcal{D}, \Sigma)$, as this essentially corresponds to evaluating a stratified Datalog program, which has PTIME data complexity, cf. (Dantsin et al. 2001).

It therefore only remains to explain how to compute $\mathsf{SE}(\mathcal{D}, \Sigma)$ from the sets $T^{\ell}_{\exists}(R(\boldsymbol{d}))$ and $T^{\ell}_{\times}(R(\boldsymbol{d}))$ and the window provided by $\mathsf{window}(R(\boldsymbol{d}), w) \in \Pi_{\mathsf{SE}}(\mathcal{D})$. A naïve yet polynomial-time procedure would consider all quadratically many possible intervals $[t, t']$ for a given $R(\boldsymbol{d})$ and check whether each condition of Definition 3 (resp. Definition 4 for persistent events) is satisfied. Naturally, one can devise more efficient algorithms that consider fewer candidate intervals (we describe in the appendix of the long version how simple events are computed in our system). □

[3]We direct interested readers to Chapter 17.1 of (Abiteboul, Hull, and Vianu 1995) for more information on the complexity of first-order query evaluation and a proof of membership in AC_0.

It follows that the naïve timeline can be efficiently computed. By contrast, there could be exponentially many different (preferred) repairs, so it may not be feasible to compute all consistent and preferred timelines. In fact, by suitably adapting complexity results for atemporal repairs, we can show it is intractable even to recognize such timelines:

Theorem 2. *It is* coNP*-complete in data complexity to decide, given a TES* Σ*, dataset* $\mathcal{D}$*, and set of facts* $\mathcal{S}$*, whether* $\mathcal{S}$ *is a consistent (or preferred) timeline for* $\Sigma, \mathcal{D}$*.*

Proof sketch. To establish the coNP upper bound for consistent timelines, consider the following guess-and-check procedure, whose input is a TES Σ, dataset $\mathcal{D}$, and set $\mathcal{S}$ of simple event and meta-event facts:

1. Compute $\mathsf{SE}(\mathcal{D}, \Sigma)$ and $\mathcal{S}_{\mathsf{SE}} = \mathcal{S} \cap \mathsf{SE}(\mathcal{D}, \Sigma)$.
2. Guess a subset $\mathcal{S}'_{\mathsf{SE}}$ of $\mathsf{SE}(\mathcal{D}, \Sigma)$.
3. Check if $\mathcal{S}_{\mathsf{SE}}$ and $\mathcal{S}'_{\mathsf{SE}}$ are Σ-consistent.
4. Return 'yes' if one of the following holds (else 'no'):
   (a) $\mathcal{S} \neq \mathcal{S}_{\mathsf{SE}} \cup \mathsf{ME}(\mathcal{D}, \mathcal{S}_{\mathsf{SE}}, \Sigma)$
   (b) $\mathcal{S}_{\mathsf{SE}}$ is *not* Σ-consistent, or
   (c) $\mathcal{S}'_{\mathsf{SE}}$ is Σ-consistent and $\mathcal{S}_{\mathsf{SE}} \subsetneq \mathcal{S}'_{\mathsf{SE}}$

It can be shown that some execution of this nondeterministic procedure returns 'yes' iff $\mathcal{S}$ is *not* a consistent timeline. Moreover, the procedure is easily adapted to preferred repairs by replacing (c) by the following condition: $\mathcal{S}'_{\mathsf{SE}}$ is Σ-consistent and there exists k such that (i) $(\mathcal{S}'_{\mathsf{SE}})_\ell = (\mathcal{S}_{\mathsf{SE}})_\ell$ for every $1 \leq \ell < k$, and (ii) $(\mathcal{S}_{\mathsf{SE}})_k \subsetneq (\mathcal{S}'_{\mathsf{SE}})_k$.

For the lower bound, we reduce 3SAT to the problem of testing whether a set of facts is *not* a consistent timeline. Consider a propositional 3CNF $\varphi = \lambda_1 \wedge \ldots \wedge \lambda_m$ over variables $v_1, \ldots, v_k$, where $\lambda_i = l_{i,1} \vee l_{i,2} \vee l_{i,3}$. We associate with each clause λ_i a corresponding vector $(v_{i,1}, b_{i,1}, v_{i,2}, b_{i,2}, v_{i,3}, b_{i,3})$ where $v_{i,j}$ is the variable in literal $l_{i,j}$ and $b_{i,j} = 1$ (resp. $b_{i,j} = 0$) if $l_{i,j} = v_{i,j}$ (resp. $l_{i,j} = \neg v_{i,j}$). We encode φ using the following dataset $\mathcal{D}_\varphi$:

$$\begin{aligned}\mathcal{D}_\varphi = &\{\mathsf{Var}(v_i, 0) \mid 1 \leq i \leq k\} \\ &\cup \{\mathsf{Clause}(v_{i,1}, b_{i,1}, v_{i,2}, b_{i,2}, v_{i,3}, b_{i,3}) \mid 1 \leq i \leq m\}\end{aligned}$$

We define a TES $\Sigma = (\Pi_{\mathsf{SE}}, \emptyset, \Upsilon_{\mathsf{temp}}, \Upsilon_{\mathsf{dom}})$, which uses two simple persistent events Q (arity 0) and Value (arity 2). The set Π_{SE} consists of the following three existence rules:

$$\begin{aligned}\mathsf{exists}(\mathsf{Q}, t, 1) &\leftarrow \mathsf{Var}(x, t) \\ \mathsf{exists}(\mathsf{Value}(x, 1), t, 1) &\leftarrow \mathsf{Var}(x, t) \\ \mathsf{exists}(\mathsf{Value}(x, 0), t, 1) &\leftarrow \mathsf{Var}(x, t)\end{aligned}$$

The set Υ_{dom} contains the following three constraints:

$$\begin{aligned}&\leftarrow \mathsf{Value}(x, 1, [t, t']) \wedge \mathsf{Value}(x, 0, [t, t']) \\ &\leftarrow \mathsf{Var}(x, t) \wedge \mathsf{Q}([t, t']) \\ &\qquad \wedge \neg\mathsf{Value}(x, 1, [t, t']) \wedge \neg\mathsf{Value}(x, 0, [t, t']) \\ &\leftarrow \mathsf{Clause}(x_1, y_1, x_2, y_2, x_3, y_3) \wedge \mathsf{Q}([t, t']) \\ &\qquad \wedge \neg\mathsf{Value}(x_1, y_1, [t, t']) \wedge \neg\mathsf{Value}(x_2, y_2, [t, t']) \\ &\qquad \wedge \neg\mathsf{Value}(x_3, y_3, [t, t'])\end{aligned}$$

Importantly, Σ does not depend on the instance φ, as required for a data complexity reduction. It is easy to see that:

$$\mathsf{SE}(\mathcal{D}_\varphi, \Sigma) = \{\mathsf{Q}([0,*],1)\} \cup \{\mathsf{Value}(v_i, b, [0,*], 1) \mid b \in \{0,1\}, 1 \le i \le k\}$$

To complete the proof, one can verify that $\{\mathsf{Q}([0,*],1)\}$ is *not* a consistent timeline of $(\Sigma, \mathcal{D}_\varphi)$ iff φ is satisfiable. □

The reduction used to show coNP-hardness employed a TES with negated event atoms. We show this is necessary, as the recognition problems are tractable if we disallow negated event atoms in rules and constraints (note that negation can still be applied to the atemporal and observation atoms).

Theorem 3. *It can be decided in* PTIME *in data complexity whether a set of facts $\mathcal{S}$ is a consistent (or preferred) timeline for a TES Σ* without negated event atoms *and dataset $\mathcal{D}$.*

Proof idea. The key to obtaining tractability is to show that Σ-inconsistency is monotonic, which implies that if $\mathcal{S} \subseteq \mathsf{SE}(\mathcal{D}, \Sigma)$ is Σ-consistent and *not* a repair, then there exists $\varphi \in \mathsf{SE}(\mathcal{D}, \Sigma) \setminus \mathcal{S}$ that can be added while retaining consistency. It thus suffices to iterate over all such φ and perform a Σ-consistency check to determine (non-)maximality of the candidate consistent timeline (in line with procedures for recognizing subset repairs, cf. Lemma 1 of (Bienvenu and Bourgaux 2016)). A similar but slightly more complex strategy can be employed for preferred timelines. □

For the cautious timeline, however, the absence of negated event atoms does not suffice to ensure tractability.

Theorem 4. *It is* coNP*-hard in data complexity to recognize or compute the cautious timeline, even in the absence of negated event atoms.*

Proof idea. We again proceed by reduction from 3SAT, adapting the proof of Theorem 2. We modify $\mathcal{D}_\varphi$ by adding a constant c_i to the Clause fact encoding the ith clause and add atemporal facts $\mathsf{First}(c_1)$, $\mathsf{Last}(c_m)$, and $\mathsf{Next}(c_i, c_{i+1})$ $(1 \le i < m)$. We keep the same set Π_{SE} and retain the constraint that enforces a single truth value (0 or 1) per variable. We add six meta-rules which serve to derive $\mathsf{Sat}(c_i, [0,*])$ if the truth assignment selected via the Value facts makes $c_1, \ldots, c_i$ hold (using the First and Next facts to 'iterate' over the clauses). Finally, a second (negation-free) domain constraint $\leftarrow \mathsf{Q}([t,t']) \wedge \mathsf{Last}(z) \wedge \mathsf{Sat}(z, [t,t'])$ ensures φ is satisfiable iff $\{\mathsf{Q}([0,*],1)\}$ is *not* the cautious timeline. □

Interestingly, however, if we consider the special case in which we only have the fixed set of temporal constraints (i.e. no domain constraints) and all termination rules have the same confidence, then there is a unique preferred repair, which moreover is efficiently computable:

Theorem 5. *When $\Upsilon_{\mathsf{dom}} = \emptyset$ and termination rules all have confidence* 1*, there is a unique preferred repair, and both the preferred timeline and cautious timeline can be computed in* PTIME *in data complexity.*

Algorithm 1 Preferred timeline (Theorem 5)

Input: TES $\Sigma = (\Pi_{\mathsf{SE}}, \Pi_{\mathsf{ME}}, \Upsilon_{\mathsf{temp}}, \Upsilon_{\mathsf{dom}})$ s.t. $\Upsilon_{\mathsf{dom}} = \emptyset$ and all termination rules have confidence 1, dataset $\mathcal{D}$
Output: unique preferred timeline $\mathcal{T}^*$

$\mathcal{S} \leftarrow \mathsf{SE}(\mathcal{D}, \Sigma)$ // inferred simple events //
$m \leftarrow \min\{\ell \mid R(\mathbf{d}, [t_1, t_2], \ell) \in \mathcal{S}\}$
$n \leftarrow \max\{\ell \mid R(\mathbf{d}, [t_1, t_2], \ell) \in \mathcal{S}\}$
Partition $\mathcal{S}$ into $\mathcal{S}_m, \ldots, \mathcal{S}_n$ by confidence level
$\mathcal{R}^* \leftarrow \mathcal{S}_m$ // initialize with top-confidence facts //
for $\ell \leftarrow m+1$ **to** n **do**
  for all $\varphi \in \mathcal{S}_\ell$ **do**
    if NoTemporalConflict$(\varphi, \mathcal{R}^*)$ **then**
      $\mathcal{R}^* \leftarrow \mathcal{R}^* \cup \{\varphi\}$
$\mathcal{T}^* \leftarrow \mathcal{R}^* \cup \mathsf{ME}(\mathcal{D}, \mathcal{R}^*, \Sigma)$
return $\mathcal{T}^*$

Proof sketch. When $\Upsilon_{\mathsf{dom}} = \emptyset$, the cautious timeline can be computed in PTIME by (i) removing those $R(\boldsymbol{u}, [t_1, t_2])$ from $\mathsf{SE}(\mathcal{D}, \Sigma)$ such that there exists some $R(\boldsymbol{u}, [t_1', t_2']) \in \mathsf{SE}(\mathcal{D}, \Sigma)$ with $[t_1', t_2'] \neq [t_1, t_2]$ where $[t_1, t_2]$ and $[t_1', t_2']$ non-trivially overlap, then (ii) applying the meta-event rules.

The PTIME result for preferred timelines is obtained by analyzing how inferred intervals are related. Indeed, when $\Upsilon_{\mathsf{dom}} = \emptyset$ and all termination rules have confidence 1, we can show that if distinct $R(\boldsymbol{d}, [t_1, t_2])$ and $R(\boldsymbol{d}, [t_1', t_2'])$ are inferred at the same confidence level, then $[t_1, t_2]$ and $[t_1', t_2']$ cannot overlap, so no repair is needed within a single confidence level. Moreover, if $\Pi_{\mathsf{SE}}, \mathcal{D} \models_\ell R(\boldsymbol{d}, [t_1, t_2])$ and $\Pi_{\mathsf{SE}}, \mathcal{D} \models_{\ell'} R(\boldsymbol{d}, [t_1', t_2'])$ with $\ell' > \ell$, then $[t_1, t_2]$ and $[t_1', t_2']$ can only overlap if $[t_1', t_2']$ fully contains $[t_1, t_2]$. This implies uniqueness and allows us to greedily build a repair level by level, as formalized in Algorithm 1. □

4 HEVA System

To evaluate the interest of our proposed approach, we implemented core components of our framework using answer set programming (ASP), a prominent declarative programming paradigm[4]. Our prototype system **HEVA** (**H**igh-level **Ev**ents with **A**SP) currently only supports temporal constraints (i.e. $\Upsilon_{\mathsf{dom}} = \emptyset$). The system accepts termination rules with multiple confidence levels, but the computation of preferred repairs uses Algorithm 1, which requires that termination rules have confidence level 1. The source code, documentation, and examples are publicly available on GitHub (https://github.com/yvoawk/HEVA).

System Inputs **HEVA** takes three forms of input: event rules, atemporal facts, and observation facts. The event rules of the TES $(\Pi_{\mathsf{SE}}, \Pi_{\mathsf{ME}})$ are specified as ASP rules using head predicates `exists`, `exists_pers` (existence conditions for persistent events), `terminates`, `pt_window` (provided time window), and `m_event`. Atemporal facts are encoded as usual ASP facts, e.g. `tki(ceritinib)`,

[4]We assume basic familiarity with ASP, see e.g. (Brewka, Eiter, and Truszczynski 2011; Gebser et al. 2012; Lifschitz 2019)

while observation facts are encoded using the `obs` predicate, e.g. `obs(has_adm, p, d, t)` (note the reified predicate name). In our experiments, these facts were generated by an external Python script using a mapping file that links `obs` predicates to fields in a relational database.

System Components **HEVA** is composed of five interacting ASP modules. The *non-persistent simple event module* computes non-persistent simple events facts from the input facts and `exists` and `terminates` rules, by expanding intervals iteratively and pruning non-maximal intervals. The *persistent simple event module* computes persistent simple events facts from the input facts and `exists_pers` and `terminates` rules.The *temporal predicate module* defines standard temporal relations between intervals, like Allen interval relations (Allen 1983), interval manipulation predicates (e.g. to compute the intersect function), and further helper predicates that simplify rule writing. These defined predicates can be used in meta-event rule bodies, while additional helper predicates, internal to **HEVA**, are provided by the *auxiliary module*. Finally, the *temporal repair module* computes the set of repairs or the unique preferred repair.

System Functionalities Based upon the options that have been selected by the user, **HEVA** passes the required ASP programs and facts to the Clingo[5] ASP system (Gebser et al. 2019), which produces *stable model(s)* corresponding to the desired timeline(s). By default, it returns a single answer set (naïve timeline), as the repair mechanism is disabled. When the `repair` option is enabled, **HEVA** may return multiple answer sets, each corresponding to a consistent timeline, or the (unique) preferred or cautious timelines, if the `preferred` or `cautious` mode is specified. Note that the restriction to termination rules of confidence 1 is only required for the `preferred` mode.

5 Experimental Evaluation

We evaluate our approach on a medical use case based on real clinical data, focusing on both computational performance and the quality of the inferred events.

5.1 Lung Cancer Use Case

We formalized a lung cancer use case within our logical framework[6]. The objective was to identify six clinical events of interest. Five were modelled as simple events: (a) *primary lung cancer episode*, inferred from diagnostic codes (ADICAP and ICD-10), ordered by confidence level ℓ: ADICAP codes ($\ell = 1$), specific ICD-10 codes ($\ell = 2$), and non-specific ICD-10 codes ($\ell = 3$), (b) *secondary cancer episode*, inferred from ICD-10 codes, (c) *EGFR and ALK mutations*, both inferred from DNA sequencing results, and (d) *TKI therapy*, inferred from administration ($\ell = 1$) or prescription records ($\ell = 2$). The sixth event, *lung cancer disease*, is a meta-event, inferred using the primary and secondary episode events. The formalization involved 16 event rules with 3 confidence levels and 497 atemporal facts.

[5]https://potassco.org/clingo/

[6]Further details on this use case, including the event rules, are provided in the appendix of (Awuklu et al. 2026).

	Min.	Q1	Q2	Mean	Q3	Max.
Obs. facts	3.00	23.00	45.00	83.65	99.50	815.00
Grnd. rules	538	708	841.5	983.9	1,066.5	3,918
Models	1.00	2.00	2.00	3.42	4.00	24.00
Time (s)	0.14	0.21	0.31	0.37	0.41	4.92

Table 1: Statistics for lung cancer use case (322 patients). Execution times for computing all stable models for consistent timelines. Q1, Q2, and Q3 indicate first, second (median), and third quartiles.

Observation facts were extracted from the clinical data warehouse of Bordeaux University Hospital. For this use case, we focus on the 322 patients with EGFR- or ALK-mutated lung cancer treated with TKIs, selected from approx. 16,800 lung cancer cases. Table 1 gives statistics on the number of observation facts and ground rules per patient.

5.2 System Performance

All experiments were run on a machine equipped with an 12th Gen Intel(R) Core(TM) i3-12100T @2.20GHz ×4, 8GB RAM, under Windows 10 Professional 64 bits, with runtimes averaged over 5 executions. This use case involves rules with different confidence levels, and **HEVA** was therefore run in its different modes to generate the four kinds of timeline. As runtime results across the different modes were broadly similar, we only report results for the lung cancer study in the `repair` mode (consistent timelines)[7]. Table 1 provides statistics on runtime and the number of stable models (corresponding to consistent timelines) for each of the 322 patients in the lung cancer use case.

The number of input observation facts varied significantly across patients, reflecting the heterogeneity of patient histories, which also led to variability in the number of ground rules and the execution times. This meant that while execution time remained low for most patients, typically completing in less than half a second, higher runtimes (up to 5 seconds) were observed for the few 'outlier' patients. The vast majority of cases (295 out of 322) triggered **HEVA**'s repair mechanism to resolve temporal constraint violations, which led to an average of 3.4 (up to a maximum of 24) stable models ($\sim$ consistent timelines) per patient.

5.3 Qualitative Study

To gain insights into the quality of the inferred events, we compared the events from **HEVA**'s *consistent timelines* against the individual annotations generated by four experts. We randomly selected 30 patients from the lung cancer cohort from those with a number of observations between the median and third quartile. Each expert manually examined 15 patient records (giving two sub-cohorts: Annotators 1 & 2, Annotators 3 & 4) to identify the target clinical events and to indicate their start date and, when applicable, their end date. An inter-annotator agreement[8] score was computed using a component-based weighted scheme that ac-

[7]Results for the other modes are provided in the appendix.

[8]Details on the agreement score computation and a breakdown of the results are provided in the appendix of the long version.

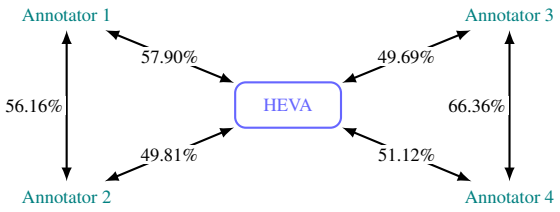


Figure 1: Mean agreement between HEVA (consistent timelines) and annotators, with inter-annotator agreement per sub-cohort.

counts for event structure, with higher weights assigned to events requiring finer-grained annotation. Figure 1 summarizes the inter-annotator agreement between the two experts handling the same sub-cohort, as well as the mean agreement ratios between **HEVA** and each annotator.

Overall inter-annotator agreement among medical experts averaged 61%, reflecting substantial variability, particularly for temporal boundaries (start/end dates), highlighting the intrinsic difficulty of this task even for human experts. Comparing the expert annotations, we found that agreement was consistently high for event presence across all categories, while start dates showed the lowest concordance, highlighting the inherent ambiguity of temporal information in EHRs. Primary lung cancer episodes and lung cancer disease achieved the highest agreement, whereas secondary cancer episodes and ALK mutations showed lower consistency, likely due to clinical complexity and data sparsity.

We compared **HEVA**'s inferred events against expert annotations using the same scoring scheme. As each consistent timeline gives a plausible interpretation of the data, for the evaluation, we retained the consistent timeline that achieved the highest agreement with the corresponding expert annotations. Agreement between **HEVA** and experts ranged from 49% to 58% per annotator, rising to 60% when merging annotator pairs. This is close to the inter-annotator agreement, indicating that **HEVA**'s outputs are roughly as consistent with expert judgments as experts amongst themselves.

6 Related Work

We briefly review approaches to temporal reasoning that are closest to our own in terms of motivations or methods.

The original event calculus (EC), introduced for reasoning about actions and their effects (Kowalski and Sergot 1986), considers instantaneous events (akin to our observations) that initiate or terminate fluents (properties whose value may change over time), which persist by default through inertia until terminated. Domain modeling in EC is done via rules for specifying initiation or termination of fluents where rule heads use special predicates `initiatesAt` and `terminatesAt`, and rule bodies speak of which other events/actions and fluents (do not) hold at the considered timepoint. Our modeling of persistent simple events is broadly similar to the handling of fluents in EC (but using existence rules rather than initiation rules to determine interval start times and with different restrictions on rule body syntax). By contrast, our formalization of non-persistent simple events via existence, termination, and window rules, equipped with a "group close existence points together" semantics, has no direct analog in any EC dialects. Another common point with the EC is the use of special predicates which avoids the need for temporal logic operators in rule bodies, arguably leading to simpler and more intuitive specifications for domain experts.

Several extensions of the EC, in particular the runtime event calculus (RTEC), have been subsequently developed for *complex (aka composite) event recognition (CER)* (Artikis, Sergot, and Paliouras 2015; Mantenoglou, Pitsikalis, and Artikis 2025), with an emphasis on performance to enable real-time processing of streaming data. Some such EC dialects (like RTEC) adopt an interval-based semantics for fluents (akin to our event predicates) and additionally allow for rules to define complex events in terms of other complex events using interval manipulation (intersecting or unioning event intervals) or Allen relations (Mantenoglou, Kelesis, and Artikis 2023). Our meta-event rules likewise support hierarchical and compositional modeling of complex events.

The need to handle various kinds of uncertainty in CER is widely recognized, cf. survey by Alevizos et al. (2017), motivating the development of probabilistic CER frameworks, including EC-based ones, for handling uncertain data, where probabilities are attached to the timestamped facts. Computing the probability of a complex event is computationally challenging and has been recently tackled using linear algebraic methods (Tsilionis, Artikis, and Paliouras 2025). Uncertainty in event inference (called pattern uncertainty by Alevizos et al.) is less explored. Moreover, to the best of our knowledge, qualitative approaches to handling uncertainty and inconsistency (like our confidence levels and repair-based timeline semantics) have not yet been considered for CER, nor is it evident how they could be simulated using existing probabilistic CER methods.

Several other rule-based formalisms have been proposed for reasoning over temporal data. DatalogMTL (Brandt et al. 2018; Walega 2025) extends Datalog with metric temporal operators to define complex temporal queries. Due to its expressivity, reasoning is highly intractable (PSPACE data complexity even for the integer timeline), though relevant fragments with lower complexity have been identified and implemented (Walega et al. 2020b; Walega et al. 2020a; Wang et al. 2022), with recent support for streaming data (Walega et al. 2023). LARS is an expressive rule-based language specifically designed for reasoning over streaming data (Beck, Dao-Tran, and Eiter 2018), with a dedicated window operator to restrict to recent timepoints or atoms, and a recent extension to support ontological reasoning (Urbani, Krötzsch, and Eiter 2022). Extensions of ASP with linear and metric temporal operators have also been explored, implemented in the ASP tool telingo (Cabalar et al. 2019; Cabalar 2022). Differently from these works, our existence and termination rule bodies do not utilize temporal operators but only conditions close to relational queries with which medical informatics practitioners are typically familiar.

In the medical domain, the need for presenting temporal information at different levels of abstraction by combining timestamped observations has been long acknowledged (Shahar 1997; Shahar and Musen 1996). Rule-based

approaches are desirable as they support easy integration of domain knowledge and explainability. The advantages of adopting declarative approaches were highlighted in the works of Falcionelli *et al.* (2019) who employ an EC-based CER approach to monitor chronic conditions from sensor data, and Dwyer *et al.* (2023) who employ the Vadalog rule language to extract and analyze patient pathways from EHRs (neither work considers consistency handling). Temporal rules have also been successfully used to implement domain-specific algorithms, as exemplified by the work of Lyu *et al.* (2022) on gestational age detection from EHR data.

7 Conclusion and Future Work

In this paper, we introduced an expressive logical framework for inferring temporally extended events. At the heart of our approach is a novel method for specifying simple events through the use of existence and termination conditions and temporal windows, without the need to write rules with temporal logic operators. Another distinguishing feature is the use of confidence levels, constraints, and a repair mechanism to define different kinds of (preferred) timelines, accounting for the inherent uncertainty in the event detection process (which is a difficult task even for human experts). This unique combination of language features required us to conduct a new complexity analysis, leading to the identification of relevant special cases with more favorable computational properties, and it also meant that we could not straightforwardly implement our framework on top of existing temporal reasoning systems. We therefore developed an ASP-based prototype **HEVA**, which showed promising results on a lung cancer use case.

Our decision to use ASP for the implementation was motivated not only by its ease of use for prototyping, but also by its high expressive power, which will be useful when extending **HEVA** to handle arbitrary TESs and new reasoning tasks (e.g. temporal query answering over the generated timelines). Indeed, in order to handle repairs w.r.t. arbitrary constraints, we hope to leverage existing work on ASP-based repair techniques (Eiter et al. 2008; Manna, Ricca, and Terracina 2013; Bienvenu et al. 2026). It would also be relevant to conduct a detailed expressivity study in order to understand precisely which fragments of our language can be captured by existing temporal formalisms (we expect for instance that simple event inference could be reduced to reasoning in suitably chosen EC and DatalogMTL dialects). Such expressivity results may suggest new ideas for optimizing timeline computation or for adapting our framework to handle streaming data.

While our logical framework is application-independent, it was developed with clinical event detection in mind. To facilitate its use by medical practitioners, we plan to develop a domain-specific language offering users a simplified syntax and templates covering common medical event types. We also wish to explore how system-generated explanations could help domain experts better understand, validate, and potentially revise their own annotations and/or support iterative refinement of the rules. The (semi-)automatic generation of domain constraints and atemporal facts from medical ontologies is another interesting direction.

Acknowledgements

This work was partially supported by the ANR AI Chair INTENDED (ANR-19-CHIA-0014), JST CREST Grant Number JPMJCR22D3, and the RIKEN TRIP initiative (AGIS). The authors would also like to acknowledge Frantz Thiessard, Antoine Lanusse, Guillaume Verdy, Léodoric Ahouanse and Arslane Tedlaouti for their valuable contributions and support.

AI Declaration

During the preparation of this paper, the authors used AI-based writing assistants (Grammarly, ChatGPT) solely for: Grammar and spelling check, translation, rewording. The final text has been reviewed and edited as needed by the authors, who take full responsibility for the paper's content.

References

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

Alevizos, E.; Skarlatidis, A.; Artikis, A.; and Paliouras, G. 2017. Probabilistic complex event recognition: A survey. *ACM Comput. Surv.* 50(5):71:1–71:31.

Allen, J. F. 1983. Maintaining knowledge about temporal intervals. *Commun. ACM* 26(11):832–843.

Apt, K. R.; Blair, H. A.; and Walker, A. 1988. Towards a theory of declarative knowledge. In Minker, J., ed., *Foundations of Deductive Databases and Logic Programming*. Morgan Kaufmann. 89–148.

Artikis, A.; Sergot, M.; and Paliouras, G. 2015. An event calculus for event recognition. *IEEE Trans. Knowl. Data Eng.* 27(4):895–908.

Augusto, J. C. 2005. Temporal reasoning for decision support in medicine. *Artif. Intell. Med.* 33(1):1–24.

Awuklu, Y. K.; Mougin, F.; Griffier, R.; Bienvenu, M.; and Jouhet, V. 2025. Ontology-driven identification of inconsistencies in clinical data: A case study in lung cancer phenotyping. *J. Biomed. Inform.* 165:104808.

Awuklu, Y.; Bienvenu, M.; Inoue, K.; Jouhet, V.; and Mougin, F. 2026. Inferring high-level events from timestamped data: Complexity and medical applications. Extended version with appendix available at: https://arxiv.org/abs/2604.21793.

Beck, H.; Dao-Tran, M.; and Eiter, T. 2018. LARS: A logic-based framework for analytic reasoning over streams. *Artif. Intell.* 261:16–70.

Bertossi, L. E. 2011. *Database Repairing and Consistent Query Answering*. Synthesis Lectures on Data Management. Morgan & Claypool Publishers.

Bienvenu, M., and Bourgaux, C. 2016. Inconsistency-tolerant querying of description logic knowledge bases. In *Reasoning Web Tutorial Lectures*, volume LNCS 4126, 156–202.

Bienvenu, M.; Bourgaux, C.; Jean, R.; and Mazzotta, G. 2026. Using ASP(Q) to handle inconsistent prioritized data. In *Proc. of KR*.

Bienvenu, M.; Bourgaux, C.; and Goasdoué, F. 2014. Querying inconsistent description logic knowledge bases under preferred repair semantics. In *Proc. of AAAI*, 996–1002.

Brandt, S.; Kalayci, E. G.; Ryzhikov, V.; Xiao, G.; and Zakharyaschev, M. 2018. Querying log data with metric temporal logic. *J. Artif. Intell. Res.* 62:829–877.

Brewka, G.; Eiter, T.; and Truszczynski, M. 2011. Answer set programming at a glance. *Commun. ACM* 54(12):92–103.

Cabalar, P.; Kaminski, R.; Morkisch, P.; and Schaub, T. 2019. telingo = ASP + time. In *Proc. of LPNMR*, 256–269.

Cabalar, P. 2022. Temporal ASP: From logical foundations to practical use with telingo. In *Reasoning Web Tutorial Lectures*, volume LNCS 13100. 94–114.

Dantsin, E.; Eiter, T.; Gottlob, G.; and Voronkov, A. 2001. Complexity and expressive power of logic programming. *ACM Comput. Surv.* 33(3):374–425.

Dwyer, O. P.; Baldazzi, T.; Davies, J.; Sallinger, E.; and Vlad, A. 2023. Reasoning over health records with Vadalog: A rule-based approach to patient pathways. *Proc. of Int. Rule Challenge @ RuleML+RR* 3485:15.

Eiter, T.; Fink, M.; Greco, G.; and Lembo, D. 2008. Repair localization for query answering from inconsistent databases. *ACM Trans. Database Syst.* 33(2):10:1–10:51.

Falcionelli, N.; Sernani, P.; Brugués, A.; Mekuria, D. N.; Calvaresi, D.; Schumacher, M.; Dragoni, A. F.; and Bromuri, S. 2019. Indexing the event calculus: Towards practical human-readable personal health systems. *Artif. Intell. Med.* 96:154–166.

Gebser, M.; Kaminski, R.; Kaufmann, B.; and Schaub, T. 2012. *Answer Set Solving in Practice*. Morgan & Claypool Publishers.

Gebser, M.; Kaminski, R.; Kaufmann, B.; and Schaub, T. 2019. Multi-shot ASP solving with clingo. *Theory Pract. Log. Program.* 19(1):27–82.

Hripcsak, G., and Albers, D. J. 2013. Next-generation phenotyping of electronic health records. *J Am Med Inform Assoc* 20(1):117–121.

Kowalski, R. A., and Sergot, M. J. 1986. A logic-based calculus of events. *New Gener. Comput.* 4(1):67–95.

Li, F.; Du, J.; He, Y.; Song, H.-Y.; Madkour, M.; Rao, G.; Xiang, Y.; Luo, Y.; Chen, H. W.; Liu, S.; Wang, L.; Liu, H.; Xu, H.; and Tao, C. 2020. Time event ontology (TEO): To support semantic representation and reasoning of complex temporal relations of clinical events. *J Am Med Inform Assoc* 27(7):1046–1056.

Lifschitz, V. 2019. *Answer Set Programming*. Springer.

Lyu, T.; Liang, C.; Liu, J.; Campbell, B.; Hung, P.; Shih, Y.-W.; Ghumman, N.; and Li, X. 2022. Temporal Events Detector for Pregnancy Care (TED-PC): A rule-based algorithm to infer gestational age and delivery date from electronic health records of pregnant women with and without COVID-19. *PLoS One* 17(10):e0276923.

Manna, M.; Ricca, F.; and Terracina, G. 2013. Consistent query answering via ASP from different perspectives: Theory and practice. *Theory Pract. Log. Program.* 13(2):227–252.

Mantenoglou, P.; Kelesis, D.; and Artikis, A. 2023. Complex event recognition with Allen relations. In *Proc. of KR*, 502–511.

Mantenoglou, P.; Pitsikalis, M.; and Artikis, A. 2025. Reasoning over streams of events with delayed effects. *J. Artif. Intell. Res.* 84:1–37.

Rance, B.; Canuel, V.; Countouris, H.; Laurent-Puig, P.; and Burgun, A. 2016. Integrating heterogeneous biomedical data for cancer research: the CARPEM infrastructure. *Appl Clin Inform* 7(2):260–274.

Reiter, R. 2001. *Knowledge in Action: Logical Foundations for Specifying and Implementing Dynamical Systems*. MIT Press.

Shahar, Y., and Musen, M. A. 1996. Knowledge-based temporal abstraction in clinical domains. *Artif. Intell. Med.* 8(3):267–298.

Shahar, Y. 1997. A framework for knowledge-based temporal abstraction. *Artif. Intell.* 90(1-2):79–133.

Shanahan, M. 1997. *Solving the Frame Problem - A Mathematical Investigation of the Common Sense Law of Inertia*. MIT Press.

Tsai, C. H.; Eghdam, A.; Davoody, N.; Wright, G.; Flowerday, S.; and Koch, S. 2020. Effects of electronic health record implementation and barriers to adoption and use: A scoping review and qualitative analysis of the content. *Life (Basel)* 10(12):327.

Tsilionis, E.; Artikis, A.; and Paliouras, G. 2025. A tensor-based probabilistic event calculus. In *Proc. of KR*, 643–652.

Urbani, J.; Krötzsch, M.; and Eiter, T. 2022. Chasing streams with existential rules. In *Proc. of KR*, 415–419.

Walega, P. A.; Grau, B. C.; Kaminski, M.; and Kostylev, E. V. 2020a. DatalogMTL over the integer timeline. In *Proc. of KR*, 768–777.

Walega, P. A.; Grau, B. C.; Kaminski, M.; and Kostylev, E. V. 2020b. Tractable fragments of datalog with metric temporal operators. In *Proc. of IJCAI*, 1919–1925.

Walega, P. A.; Kaminski, M.; Wang, D.; and Grau, B. C. 2023. Stream reasoning with DatalogMTL. *J. Web Semant.* 76:100776.

Walega, P. A. 2025. Reasoning about time in DatalogMTL: Course notes. In *Proc. of Reasoning Web Summer School*, OASIcs, 9:1–9:23.

Wang, D.; Hu, P.; Walega, P. A.; and Grau, B. C. 2022. MeTeoR: Practical reasoning in datalog with metric temporal operators. In *Proc. of AAAI*, 5906–5913.

Zhou, L., and Hripcsak, G. 2007. Temporal reasoning with medical data – a review with emphasis on medical natural language processing. *J. Biomed. Inform.* 40(2):183–202.