Reasoning About Temporal Constraints in RDF

Carlos Hurtado[1] and Alejandro Vaisman[2]

[1] Universidad de Chile
churtado@dcc.uchile.cl
[2] Universidad de Buenos Aires
avaisman@dc.uba.ar

Abstract. Time management is a key feature needed in any query language for web and semistructured data. However, only recently this has been addressed by the Semantic Web community, through the study of temporal extensions to RDF (*Resource Description Framework*). In this paper we show that the ability of the RDF data model of handling unknown resources by means of blank nodes, naturally yields a rich framework for temporal reasoning in RDF. That is, even without knowing the interval of validity of some statements we can still entail useful knowledge from temporal RDF databases. To take advantage of this, we incorporate a class of temporal constraints over anonymous timestamps. We show that testing entailment in temporal graphs with constraints reduces to closure computation and mapping discovery, that is, an extended form of the standard approach for testing entailment in non-temporal RDF graphs.

1 Introduction

The *Resource Description Framework (RDF)* [19] is a metadata model and language recommended by the W3C in order to create an infrastructure that will allow to build the so-called *Semantic Web*. In the RDF model, the universe to be modeled is a set of *resources*, essentially anything that can have a *universal resource identifier*, URI. The language to describe them is a set of binary predicates denoted *properties*. Descriptions are *statements* of the form subject-predicate-object. Both subject and object can be anonymous objects, known as *blank nodes*. In addition, the RDF specification includes a built-in vocabulary with a normative semantics (RDFS) [6]. This vocabulary deals with inheritance of classes and properties, as well as typing, among other features that allow describing the concepts and relationships that may exist in a community of people and software agents. The RDF specification can be seen as a graph where each subject-predicate-object triple is represented as a node-edge-node structure.

Time is present in almost any Web application. Thus, there is a clear need of applying temporal database concepts to RDF in order to be able to represent temporal knowledge. We illustrate this claim with the following motivating example, where RDF data is used to describe a collection of *web services*. Web services are software applications that interact using web standards. The Semantic

Web has been proposed as a tool for making applications able to automatically discover or invoke web services. In this way, *ontologies of services* could be used by service-seeking agents for representing a *service profile* (a mechanism for describing services offered by a web site). Our example is based in the web service ontology introduced by Antoniou *et al* [5] for a non-temporal RDF model. In order to keep track of the changes that can occur throughout the life cycle of the web service we introduce temporal features to a standard RDF graph representing the ontology, according to [15]. Figure 1 shows an example of an RDF representation of an evolving ontology for a web service denoted *Sport News*, first offered by the sports network ESPN, and later by another network, Fox Sports. The web site delivers up-to-date articles about sports. As input, the service receives a sports category and the customer's credit card number; it returns the requested articles. The arcs in the graph are labeled with their interval of validity. [3] The interval [0,2] over the edge between 'Sport News' and 'ESPN' means that the triple (Sports News, provided by, ESPN) is valid from time instant "0" to time instant "2". Analogously, the interval [3,Now] over the edge between 'Sport News' and 'ESPN' means that the triple (Sports News, provided by, Fox Sports) is valid from time instant "3" to the current time. For the sake of clarity, no temporal labels over an edge means that triple is valid in the interval [0,Now]. There is also an anonymous node (of type "service provider"), created at time "6". The impact of blank nodes in a temporal setting deserved an in-depth study in [15, 14].

1.1 Temporal RDF

In former work [14, 15] we studied the problem of adding the time dimension to RDF documents, and we discussed the main problems and possibilities that arise when we address the problem of keeping track of the changes occurring over an RDF graph. We denoted this problem *Temporal RDF*. The work was based on the theoretical framework provided by Gutierrez *et al* [13]

In a nutshell, a Temporal RDF graph is a set of temporal triples labeled with their interval of validity. These triples are of the form $(a, b, c)[t_1, t_2]$. Figure 1 explained above is an example of a Temporal RDF graph. We showed that temporal RDF can be implemented within the RDF specification, making use of a simple additional vocabulary. We also defined constructs that allow moving between point-based and interval-based representations of the time dimension. An RDF graph can be regarded as a knowledge base from which new knowledge, i.e., other graphs, may be entailed. In temporal RDF, entailment is slightly more involved. We studied this problem, and called it *temporal entailment*. The main issue here is the treatment of blank nodes. Defining the semantics of temporal RDF in the presence of blank nodes turns out to be non-trivial, because we cannot consider the temporal database as the union of all its snapshots (a *snapshot*

[3] Note that the standard graph(ical) representation of an RDF graph is not the most faithful to convey the idea of statements(triples) being labeled by a temporal element. Technically, temporal labels should be attached to a whole subgraph $u \xrightarrow{p} v$, and not only to an arc.

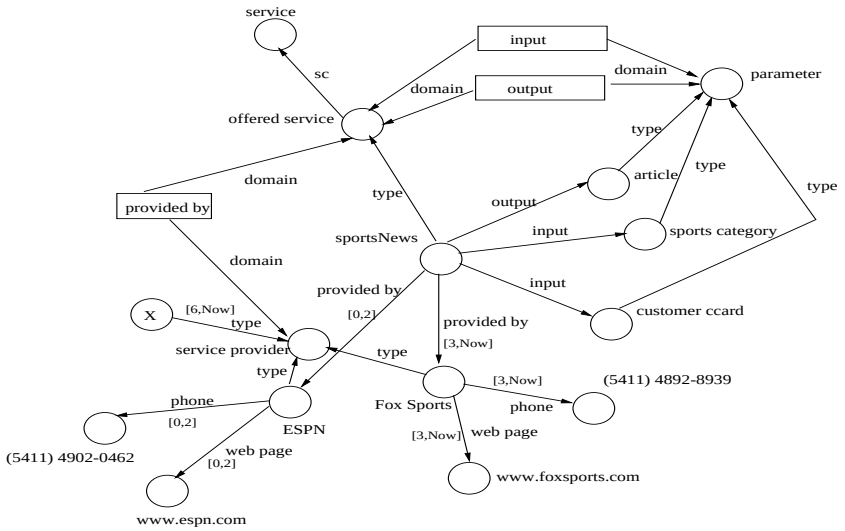

Fig. 1. An RDF graph for web services profiling of Sports networks.

at time t of a temporal RDF graph G is the corresponding subgraph formed by triples labeled by an instant t). This means that even though two temporal graphs G_1 and G_2 are such that all snapshots of G_1 entail a snapshot of G_2, we cannot say that G_1 entails G_2. We defined the notion of temporal entailment based on the *slice closure* of a temporal graph G ($\texttt{scl}(G)$). This concept of closure extends the one defined in [13]. The slice closure of G, in short, is the union of the temporalization of the closures of all the snapshots of G. We proved [14] that $G_1 \models_\tau G_2$ if and only if there is map from G_2 to $\texttt{scl}(G_1)$. Thus, testing temporal entailment reduces to computing the slice closure and find a mapping between two temporal graphs. We also proved that deciding temporal entailment is NP-complete.

1.2 Problem Statement

The work described above [14] includes a first study of the problem of *anonymous time* in temporal RDF graphs, i.e., graphs containing temporal triples labeled with blanks. In this setting, we admit triples of the form $(a, b, c)[X]$, where X is an anonymous timestamp stating that the triple (a, b, c) is valid in some unknown time, as shown in Figure 1 (in [14] we called these graphs *general temporal graphs* to differentiate them from temporal graphs without blank timestamps).

Temporal blanks considerably extend the capabilities of the temporal RDF model. First, they allow defining temporal constraints over the model. In this way, a richer treatment of time, along the lines of *constraint databases* [8] is possible (in relational constraint databases, the time of validity of a tuple can be defined by a formula Φ). There has been a substantial amount of work from

the Artificial Intelligence community on temporal reasoning systems that use constraint propagation. Thus, adding constraints to temporal RDF allows reasoning about RDF graphs in order to infer useful knowledge. However, as Allen points out [2, 3], the point-based representation of time cannot naturally capture some interval relationships used in reasoning about constraints. Thus, we also include intervals with anonymous starting and/or ending points in our model for representing anonymous time in temporal RDF graphs.

Example 1. From the following extended temporal graph:

$$(a, \texttt{sc}, b)[T_1], (b, \texttt{sc}, c)[t_1, t_2], t_1 \leq t_2, t_1 \leq T_1, T_1 \leq t_2.$$

our approach allows inferring the graph $(a, \texttt{sc}, c)[T_2]$.

Thus, even though our approach is close to temporal logics and constraint databases, temporal reasoning about RDF and RDFS ontologies introduces additional difficulties not present in the other settings.

A second property of anonymous time is that it allows representing incomplete temporal information [18]. This information can be represented by *indefinite timestamps* that can be implemented as global conditions of the form $2004 \leq k$, where k is a kind of *null* value that may appear in some constraint. We will denote a temporal graph with constraints a *c-temporal graph*.

The goal of this paper is to extend the temporal RDF model presented in previous work [14, 15] with intervals and temporal constraints. We propose and study in detail *temporal graphs with constraints*, extending our previous results to these kinds of graphs. Temporal RDF graphs with constraints (and intervals) expand the expressive power of the temporal RDF data model, allowing to represent information about events occurring within some unknown intervals. Without this capability, this information could not be represented in a natural way, as the following example shows.

Example 2. Let us suppose that, in the example depicted in Figure 1, we are not certain about the time when 'Sport News' was transferred from ESPN to Fox Sports. A *c-temporal graph* for representing this situation is shown in Figure 2. The triple (Sports News, provided by, ESPN) is now labeled [0,T1] (instead of [0,2]) , where T1 is a time variable. Analogously, (Sports News, provided by, Fox Sports) is labeled [T2,Now] (instead of [3,Now]). There is also a constraint stating that T2 is greater than T1. There are also other variables indicated in the figure, and the constraints are specified in the *Constraint* box, at the bottom right of the picture. We have also labeled with temporal blanks the triple (Sports News, `type`, offered service). Also note the triple (X,`type`, service provider) has been labeled with temporal blanks.

Although the addition of constraints enriches the model, it introduces some problems that we study in the paper. The results obtained in [14] do not work any more in the presence of temporal blank nodes and constraints. For example, the notion of slice closure must be modified. Consequently, testing temporal entailment must be modified accordingly, as well as the proofs that were obtained

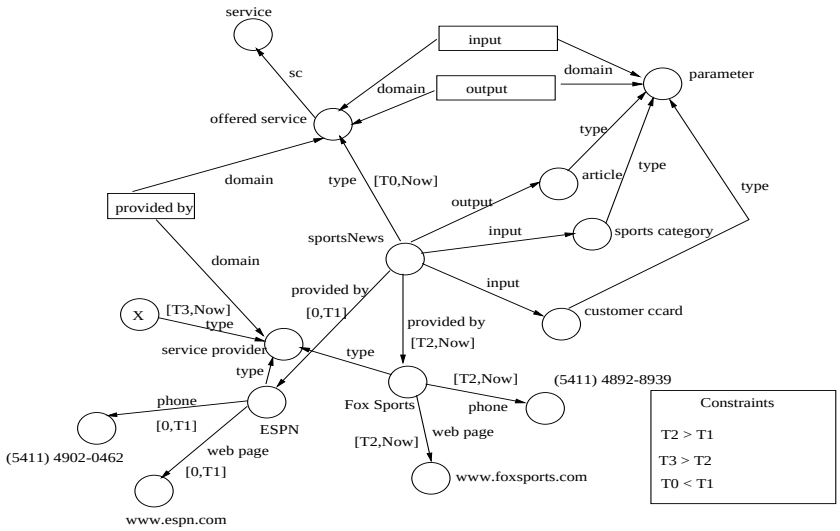

Fig. 2. The RDF graph of Figure 1 with Anonymous Time.

under the assumption that the temporal labels were only concrete time instants. The question that arises is whether testing temporal entailment in this new setting reduces to computing an extended notion of slice closure and find a mapping between two temporal graphs.

1.3 Related Work

The RDF model was introduced in 1998 by the World Wide Web Consortium (W3C) [19]. Formal work includes the study of formal aspects of RDF data and query languages [12, 13, 25], considering RDF features like entailment, the impact of blank nodes, reification, premises in queries, and the RDFS vocabulary with predefined semantics. Several query languages for RDF have been proposed and implemented. Some of them along the lines of traditional database query languages (e.g. SQL, OQL), others based on logic and rule languages. Good surveys are [16, 20]. Temporal database management has been extensively studied, including data models, mostly based on the relational model and query languages [23], leading to the TSQL2 language [22]. Chomicki [8] provides a comprehensive survey of temporal query languages. Beyond the relational model, several works proposed temporal extensions for non-temporal models, like the semistructured data model and XML [7, 4, 10, 11, 21].

Regarding temporal extensions to RDF, Visser *et al* [24] proposed a temporal reasoning framework for the Semantic Web, which has been applied in BUSTER, an ontology-based prototype developed at the University of Bremen, supporting the so-called *concept@location in time* type of query. To the best of our knowledge, our previous work [15, 14] constitutes the first formal study of temporality

issues in RDF graphs and RDF query languages. However, these previous works do not include temporal constraints. I addition, although they consider intervals, intervals are only finite sets of timestamps, and therefore intervals are syntax inside a point-based modeling of time.

1.4 Contributions

In this paper we incorporate temporal constraints and intervals (with unknown starting and/or ending time instants) to temporal RDF graphs, and denote the resulting graphs *c-temporal graphs*.

We extend temporal graphs in a stepwise manner. First, we include intervals and study the inference problem for temporal graphs with intervals. Then, we generalize the former framework incorporating temporal constraints. We formalize c-temporal graphs, allowing modeling anonymous timestamps, anonymous intervals, and constraints over them. We define and study a notion of entailment for c-temporal graphs. Further, a new notion of closure is proposed for c-temporal graphs, and temporal entailment is characterized in terms of this notion of closure. In particular, we show that testing entailment for temporal graphs with constraints, reduces to closure computation and mapping discovery, that is, an extended form of the standard approach for testing entailment in non-temporal RDF graphs. This yields a procedure for testing entailment and leads to complexity bounds for the problem of testing entailment of c-temporal graphs. Finally, we provide an RDF vocabulary to represent c-temporal graphs using standard RDF data, and sketch a syntax for this.

1.5 Outline

The remainder of the article is organized as follows. Section 2 presents preliminary notation related to RDF and RDFS and temporal RDF graphs from previous work [13, 15, 14]. Section 3 studies temporal graphs with intervals. Section 4 introduces constraints to temporal graphs and their semantics, presents the notion closure, and characterizes entailment in terms of them. Finally, in Section 5 we conclude and outline some prospects for future work.

2 Preliminaries

2.1 RDF Notation

The following is an excerpt of notation introduced in [6, 13, 17] that will be used subsequently in this paper.

In this paper we work with RDF graphs whith RDFS vocabulary. An RDF graph is a set of triples $(v_1, v_2, v_3) \in (U \cup B) \times U \times (U \cup B \cup L)$, where U is a set of URIs, B is a set of blank nodes, and L is a set of litereals (the sets are pairwise disjoint). An *RDF term* is a URI, a blank, or a literal. We consider RDF graphs that can mention the RDFS vocabulary. The RDFS vocabulary defines *Classes*

as sets of resources. Elements of a class are known as *instances* of that class. To state that a resource is an instance of a class, the property rdf:type may be used. The following are the most important classes (in brackets the name we will use in this paper) rdfs:Resource [`res`], rdfs:Class [`class`], rdfs:Literal [`literal`], rdfs:Datatype [`datatype`], rdf:XMLLiteral [`xmlLit`], rdf:Property [`property`]. *Properties* are binary relations between subject resources and object resources. The built-in properties are: rdfs:range [`range`], rdfs:domain [`dom`], rdf:type [`type`], rdfs: subClassOf [`sc`], rdfs:subPropertyOf [`sp`].

In this paper, we work with a characterization of entailment of RDF graphs in term of the notions of map and closure.

A *map* is a function $\mu : (U \cup B \cup L) \rightarrow (U \cup B \cup L)$ preserving URIs and literals, i.e., $\mu(u) = u$ and $\mu(l) = l$ for all $u \in U$ and $l \in L$. Given a graph G, we define $\mu(G)$ as the set of all $(\mu(s), \mu(p), \mu(o))$ such that $(s, p, o) \in G$. We will overload the meaning of map and speak of a *map* $\mu : G_1 \rightarrow G_2$ if there is a map μ such that $\mu(G_1)$ is a subgraph of G_2. A map μ is *consistent* with G if $\mu(G)$ is an RDF graph, *i.e.*, if s is the subject of a triple, then $\mu(s) \in U \cup B$, and if p is the predicate of a triple, then $\mu(p) \in U$. In this case, we say that the graph $\mu(G)$ is an *instance* of the graph G. An instance of G is *ground* if $\mu(G)$ does not mention blanks.

A *closure* of a graph G, denoted $\mathrm{cl}(G)$, is a maximal set of triples G' over universe(G) plus the RDFS vocabulary such that $G \subseteq G'$ and $G \models G'$.

Theorem 1 (cf. [17, 13]). *$G_1 \models G_2$ if and only if there is a map from G_2 to a* closure *of G_1.*

2.2 Temporal Graphs

In this section, we present in a compressed form relevant notation and results for temporal graph from previous work [15, 14].

A *temporal triple* is an RDF triple (a, b, c) with a temporal timestamp t, which is a positive real number. We will use the notation $(a, b, c)[t]$. Usually for a temporal graph G we will apply the same notions used for standard RDF graphs, for example, we will say "G is ground" meaning that $u(G)$ is ground, write $\mu(G)$ for $\{(\mu(a), \mu(b), \mu(c))[t] : (a, b, c)[t] \in G\}$, and so on.

Definition 1 (Entailment (c.f. [14])). *Let G_1, G_2 be RDF temporal graphs. (1) For ground temporal RDF graphs G_1, G_2 define $G_1 \models_\tau G_2$ if and only if $G_1(t) \models G_2(t)$ for each t; (2) For temporal RDF graphs, define $G_1 \models_\tau G_2$ if and only if for every ground instance $\mu_1(G_1)$ there exists a ground instance $\mu_2(G_2)$ such that $\mu_1(G_1) \models_\tau \mu_2(G_2)$.*

Temporal entailment can be characterized in terms of a notion of clousure of temporal graphs, denoted *slice closure*. Given a temporal graph G and a timestamp t: the *snapshot* of G at t, is defined as the graph $G[t] = \{(a, b, c) \mid (a, b, c)[t] \in G\}$. For an RDF graph H and a time stamp t, define H^t as the temporalization of all its triples by a temporal mark t, that is, $H^t = \{(a, b, c)[t] \mid (a, b, c) \in H\}$. The *slice closure* of G, denoted $\mathtt{scl}(G)$, is a temporal graph defined by the expression $\bigcup_t (\mathrm{cl}(G[t]))^t$, where $\mathrm{cl}(G[t])$ is any closure of the RDF graph $G[t]$.

Theorem 2 (c.f. [15]). *Let G_1, G_2 be temporal RDF graphs. Then $G_1 \models_\tau G_2$ if and only if there is a map from G_2 to* $\texttt{scl}(G_1)$.

This result yields an algorithm for testing temporal entailment. Indeed, the slice closure can be obtained by computing the closures of the snapshots of the temporal graph.

3 Temporal Graphs with Time Intervals

In this section we extend temporal graphs introduced in Section 2.2 to model time intervals defined by timestamps, that is, intervals whose extremes are positive real numbers.

3.1 Basic Definitions

We extend *temporal triples* to triples of the form $(a, b, c)[t_i, t_f]$, where t_i, t_f are timestamps (positive real numbers) such that $t_i \leq t_f$, yielding *temporal graphs with intervals*. For the case where $t_i = t_f$ a triple $(a, b, c)[t_i, t_f]$ is equivalent to temporal triple $(a, b, c)[t_i]$, as defined in Section 2.2, therefore temporal graphs with intervals subsume temporal graphs. Given a temporal graph with intervals G, we denote by $I(G)$ the intervals mentioned in G, and denote by $T(G)$ the set of timestamps mentioned in G. Given an interval $[t_i, t_f] \in I(G)$, we denote by $G[l_i, l_j]$, the set containing RDF triples (a, b, c) such that $(a, b, c)[t_i, t_f] \in G$.

A temporal graph with intervals represents a (possibly infinite) temporal graph, that is, each triple $p[t_i, t_j]$ represents the set of temporal triples $\{p[t] \ : \ t_i \leq t \leq t_j\}$. Given a temporal graph with intervals G, we denote by G^+ the temporal graph that represent G, and call it the underlying temporal graph of G.

3.2 Reasoning

The notion of entailment from Definition 1 can be naturally extended to temporal graph with intervals. Formally, we write $G \models_\tau H$ iff $G^+ \models_\tau H^+$. Therefore, Theorem 2 also characterizes entailment for temporal graphs with intervals (just consider the underlying temporal graphs involved). However, the theorem has no practical application, since underlying graphs (and therefore mappings) may be infinite. In this section, we propose a characterization of entailment that yields a procedure for the testing entailment of temporal graphs with intervals.

We will use a basic relationship between intervals (see Allen's well-known work on interval operations [2]). Given two intervals $[t_i, t_f], [t_u, t_v]$, we write that $[t_i, t_f] \sqsubseteq [t_u, t_v]$ iff $t_u \leq t_i$ and $t_f \leq t_v$. Given an interval $[t_i, t_f]$ and a set of intervals S, we denote by $\texttt{ContainSet}([t_i, t_f], S)$ the set of intervals $[t_u, t_v] \in S$ such that $[t_u, t_v] \sqsubseteq [t_i, t_f]$. As notation, we usually write $t_j \in [t_i, t_f]$, meaning that $t_i \leq t_j \leq t_f$.

The following definition extends the notion of slice closure (Section 2.2) to temporal graphs with intervals.

Definition 2. *Let G be a temporal graph with intervals. The slice closure of G, denoted* $\texttt{iscl}(G)$*, is a temporal graph with intervals H defined as follows:*

1. *Let H' be the following temporal graph with intervals. For each pair of timestamps $t_i, t_f \in T(G)$, $H'[t_i, t_f] = \text{cl}(\bigcup_{[t_u,t_v] \in \texttt{ContainSet}([t_i,t_f],I(G))} G[t_u, t_v])$.*
2. *Then, for each set of intervals $S = \{[t_1, t_2], [t_2, t_3], \ldots, [t_{n-1}, t_n]\} \subseteq I(H')$, we have $H[t_1, t_n] = \bigcap_{[t_i,t_{i+1}] \in S} H'[t_i, t_{i+1}]$.*

Example 3. Figure 3 shows a the temporal graph with constraints G and its slice closure $\texttt{iscl}(G)$. First, we illustrate condition 1 of Definition 2. As an example, consider the pair of timestamps $2, 3 \in I(G)$. Then $\texttt{ContainSet}([1, 2], I(G)) = \{[1, 3], [2, 3]\}$. Therefore, $H'[2, 3] = \text{cl}((a, \texttt{sc}, b), (b, \texttt{sc}, c)\})$, which is $\{(a, \texttt{sc}, b)$, $(b, \texttt{sc}, c)$, $(a, \texttt{sc}, c)\}$. Now, in order to explain condition 1 of Definition 2, consider the set of intervals $\{[2, 4], [4, 5]\}$. We have that $H[2, 5] = H'[2, 4] \cap H'[4, 5]$, which is $\{(a, \texttt{sc}, c)\}$. Only the triples in the last file of the table of Figure 3 are derived from condition 2 of Definition 2.

Original Graph (G)	**Slice Closure** ($\texttt{iscl}(G)$)
$(a, \texttt{sc}, b)[1, 3]$, $(b, \texttt{sc}, c)[2, 4]$, $(a, \texttt{sc}, c)[3, 5]$	$(a, \texttt{sc}, b)[1, 3]$, $(b, \texttt{sc}, c)[2, 4]$, $(a, \texttt{sc}, c)[3, 5]$
	$(a, \texttt{sc}, b)[2, 3]$, $(b, \texttt{sc}, c)[2, 3]$, $(a, \texttt{sc}, c)[2, 3]$
	$(b, \texttt{sc}, c)[3, 4]$, $(a, \texttt{sc}, c)[3, 4]$, $(a, \texttt{sc}, c)[4, 5]$
	$(a, \texttt{sc}, c)[2, 4]$, $(a, \texttt{sc}, c)[2, 5]$

Fig. 3. Slice closure of a temporal graph with intervals.

For simplicity, the previous example and others presented in this paper use only the subclass property (`sc`). The examples could be easily turned much more complex if we include in the graphs RDFS built-in-properties, such as rdfs: range [`range`], rdfs:domain [`dom`], rdf:type [`type`], rdfs: subClassOf [`sc`], rdfs:sub-PropertyOf [`sp`]. We refer the reader to [13] for examples and details on RDFS entailment.

Observe that $G \subseteq \texttt{iscl}(G)$. The following lemma states other important properties of the slice closure.

Lemma 1. *Let G be a temporal graph with intervals.*
(1) $\texttt{scl}(G^+) = (\texttt{iscl}(G))^+$.
(2) $G \equiv_\tau \texttt{iscl}(G)$.
(3) Given a timestamp $t \in [t_i, t_f]$, and a triple $(a, b, c) \in \texttt{scl}(G^+)[t]$, then there is an interval $[t_u, t_v] \in I(\texttt{iscl}(G))$ such that $[t_i, t_f] \sqsubseteq [t_u, t_v]$ and $(a, b, c) \in \texttt{iscl}(G)[t_u, t_v]$.

Proof. (1) Follows from Condition 1 of Definition 2. Let $H = \texttt{iscl}(G)^+$. Consider a timestamp t, let S be the set containing intervals $[t_o, t_p] \in I(G)$ such

that $t \in [t_o, t_p]$. We have that $G^+[t] = \bigcup_{[t_u,t_v]\in S} G[t_u, t_v]$. Hence, $\texttt{scl}(G^+)[t] = \text{cl}(\bigcup_{[t_u,t_v]\in S} G[t_u, t_v])$. Now, let $[t_i, t_f]$ be the smallest interval (among all the intervals defined with timestamps in $T(G)$) that contains t. It is easily verified that $S = \texttt{ContainSet}([t_i, t_f], I(G))$. Then, we have that $\texttt{scl}(G^+)[t] = \text{cl}(\bigcup_{[t_i,t_f]\in \texttt{ContainSet}([t_i,t_f],I(G))} G[t_i, t_f])$, and hence $\texttt{scl}(G^+) = H$.
(2) We have that $G \equiv_\tau \texttt{iscl}(G)$ iff $G^+ \equiv_\tau \texttt{iscl}(G)^+$. But $G^+ \equiv_\tau \texttt{scl}(G^+)$, and from (1) we derive the lemma.
(3) Then there should be timestamps $t_1, t_2, \ldots, t_{n-1}, t_n \in T(G)$ such that $t_1 \leq t_i, t_f \leq t_n$, $t_1 \leq t_2 \leq \ldots \leq t_n$, and for each interval $[t_i, t_{i+1}]$, $(a, b, c) \in \text{cl}(\bigcup_{[t_u,t_v]\in \texttt{ContainSet}([t_i,t_{i+1}],I(G))} G[t_u, t_v])$. Therefore, from condition 1 of Definition 2, it follows that $(a, b, c) \in H'[t_i, t_{i+1}]$. And by condition 2 of Definition 2, $(a, b, c) \in \texttt{iscl}(G)[t_1, t_n]$.

We define *interval mappings* as follows. Given two sets of intervals S, S' an interval mapping is a function $\gamma : S \to S'$, such that for each interval $[t_i, t_f] \in S$, $[t_u, t_v] = \gamma([t_i, t_f])$ satisfies $t_u \leq t_i, t_f \leq t_v$. When we apply an interval mapping to a temporal graph with intervals G, we obtain the temporal graph with interval G' containing the triples $(a, b, c)\gamma([t_i, t_j])$ such that $(a, b, c)[t_i, t_f] \in G$. In addition, we extend maps between temporal graphs to maps between temporal graphs with intervals.

Theorem 3. *Let G, H be temporal RDF graphs with intervals. Then $G \models_\tau H$ if and only if there is an interval mapping $\gamma : I(H) \to I(G)$, and a mapping μ from $\gamma(H)$ to $\texttt{iscl}(G)$.*

Proof. We will prove that (*) there is a map μ from H^+ to $\texttt{scl}(G^+)$ iff there is an interval map $\gamma : I(H) \to I(G)$, such that μ is a map from $\gamma(H)$ to $\texttt{iscl}(G)$. The theorem follows from (*) and Theorem 2. It remains to prove (*). (If) It is easy to show that μ is a map from H^+ to $(\texttt{iscl}(G))^+$. But from Lemma 1, $\texttt{scl}(G^+) = (\texttt{iscl}(G))^+$. Therefore, μ is a map from H^+ to $\texttt{scl}(G^+)$. (Only If) Consider an interval $[t_i, t_j] \in I(H)$, and let $P = H[t_i, t_j]$. We have that for all $t \in [t_i, t_f]$, $P \in H^+[t]$. Therefore, $\mu(P) \in \texttt{scl}(G^+)[t]$ for all $t \in [t_i, t_f]$. Then, by Lemma 1 (3), we have that there should be an interval $[t_u, t_v] \in I(\texttt{iscl}(G))$ such that $[t_i, t_f] \sqsubseteq [t_u, t_v]$ and $\mu(P) \in \texttt{iscl}(G)[t_u, t_v]$. We let $\gamma([t_i, t_j]) = [t_u, t_v]$. Therefore, we have $\mu(\gamma(H[t_i, t_j])) \in \texttt{iscl}(G)$. By the same procedure we define $\gamma(r)$ for each interval $r \in I(H)$, and we have $\mu(\gamma(H)) \subseteq \texttt{iscl}(G)$.

Theorem 3 yields a two-steps procedure for testing implication for temporal graphs with intervals, which requires to first compute a slice closure and then an interval mapping. In Section 4.4, we study the complexity of testing entailment.

4 Temporal Graphs with Temporal Constraints

In this section, we define *temporal graphs with temporal constraints* (c-temporal graphs in short), which generalize temporal graphs with intervals introduced in Section 3.

4.1 Temporal Constraints

In this paper, we focus on a class of temporal constraints, which is common in the literature on temporal logic (see e.g., [8]). These constraints are also equivalent to a basic class of inequality constraints studied in databases [1].

Temporal constraints will be used to state relationships between *time labels*. Time labels may be timestamps (positive real numbers) or anonymous timestamps, which are temporal variables. As notation, we use $T_1, T_2, \ldots$ for anonymous timestamps, $t_1, t_2, \ldots$ for timestamps, and $l_1, l_2, \ldots$ for temporal labels. In our model RDF terms and temporal labels belong to different frameworks: time labels and triples, and are therefore disjoint.

Temporal labels are interpreted as points in the temporal domain. We model time as a temporal domain $(R, <, \leq, >, \geq, =, \neq)$, where R is the set of positive real numbers, and $<, \leq, >, \geq, =$, and $\neq$ are the standard arithmetic comparison predicates. So we assume a dense temporal domain isomorphic to the real numbers. By a *temporal constraint* we refer to an expression of the form $l_i \omega l_j$, where ω is an arithmetic comparison predicate. Given a set of temporal constraints Σ we denote $L(\Sigma)$ the temporal labels mentioned in Σ. A map for a set of temporal constraints Σ is a function $\gamma : L(\Sigma) \rightarrow R$ preserving timestamps. Given a set of temporal constraints Σ and a map γ we denote by $\gamma(\Sigma)$ the set of constraints resulting from Σ by replacing each time label l by $\gamma(l)$.

An *instance* for a set of temporal constraints $\Sigma = \{\alpha_1, \ldots, \alpha_n\}$ is a map μ such that $\mu(\Sigma)$ is ground and each $\gamma(\alpha_i)$ holds in the temporal domain. If Σ is empty the emptyset is its unique ground instance. Σ is *consistent* iff it has at least one instance. Notice that an empty set of constraints is consistent. Given two sets of temporal constraints Σ_1, Σ_2, define $\Sigma_1 \models_{constr} \Sigma_2$ if and only if for each instance γ of Σ_1, there is also and instance of $\gamma(\Sigma_2)$.

Testing entailment and consistency for temporal constraints can be done in polynomial time. An algorithm to efficiently implement the test can be found in [1]. We next show the central ideas of it. Following standard results in inequality constraints (e.g. [3, 1]), we can "close" a set of temporal constraints Σ as follows. We build the *inequality graph* for the constraints (which is a particular case of a temporal constraint network [9]), that is, a directed graph with a node for each temporal label, and an edge (l_i, l_j) for each constraint $l_i \omega l_j \in \Sigma$, were $\omega \in \{\leq, <, \leq\}$ (we can transform the constraints to have them expressed in these predicates). The edge (l_i, l_j) is labeled with the predicate ω. The graph has also edges (t_i, t_j) that capture the natural ordering between timestamps. As an example, if the timestamps 3 and 7 are mentioned in the constraints, we place an edge associated to the constraint $3 < 7$. We assume a set K containing real numbers mentioned in the constraints. The constraints can be propagated in the graph by simple transitive closure computation. Finally, the closure of Σ, denoted $\texttt{ccl}(\Sigma, K)$ are the constraints associated to the edges in the resulting graph. The closure of the inequality graph can be used for testing entailment and consistency of a set of constraints.

The following are standard results. Let Σ be a set of constraints. Σ is consistent iff its inequality graph does not have cycles. Let α be another set of

constraints such that $L(\alpha) \subseteq L(\Sigma)$, then $\Sigma \models_{constr} \alpha$ iff $\alpha \in \texttt{ccl}(\Sigma, K)$, where K is the set of timestamps mentioned in both Σ and α. For the case where $L(\alpha) \not\subseteq L(\Sigma)$, we need to perform basic quantifier elimination, that is, we need to delete the variables in $L(\alpha) \setminus L(\Sigma)$ from $\texttt{ccl}(\alpha, K)$, and test whether the resulting set of constraints is contained in $\texttt{ccl}(\Sigma, K)$.

The inequality graph can be transformed by deleting equal nodes, yielding a *minimal inequality graph* [1]. That is, if the constraints imply $Y = X$, we just rewrite the constraints by substituting X by Y. However, in order to prevent information lost, we preserve timestamps in the substitutions. In this form, we can transform a c-temporal graph into another equivalent c-temporal graph whose constraint set is *free of equalities*. In the sequel, without loss of generality, we assume that constraint sets are free of equalities.

4.2 Basic Definitions

We extend the notion of temporal graph to handle anonymous labels in timestamps and interval. We extend intervals to be pairs of temporal labels $[l_1, l_2]$. So we consider a temporal triple to be an element of the form $p[l_1, l_2]$, where p is an RDF triple and l_1, l_2 are temporal labels. The temporal graphs we next define can represent two main situations relating anonymous time: (i) a triple holds in an interval but we do not know it (this is the case where $l_1 = l_2$); and (ii) the triple holds in an interval, but one or both of the start or the ending time instants of the interval are unknown. When, $l_1 = l_2$ we denote $p[l_1, l_2]$ simply as $p[l_1]$.

Definition 3. *A* temporal graph with temporal constraints *(subsequently called a c-temporal graph) is a pair $C = (G, \Sigma)$, where G is a graph with temporal triples of the form $p[l_1, l_2]$, p is an RDF triple, and l_1, l_2 temporal labels, and Σ is a set of temporal constraints over time labels in G. For each temporal triple $p[l_1, l_2] \in G$, Σ should contain the constraint $l_1 \leq l_2$.*

For simplicity, we sometimes write the temporal constraints and the temporal triples in a single set. Given a c-temporal graph $C = (G, \Sigma)$, we denote $I(C)$ the intervals that appear in the triples in G.

We extend an interval map γ to consider intervals defined with temporal labels. An interval mapping is a function from a set of intervals to a set of intervals. If we apply γ to a c-temporal graph C, we obtain another c-temporal graph, denoted $\mu(C)$, by renaming each interval r with $\mu(r)$. A time-ground instance of a c-temporal graph $C = (G, H)$ is a temporal graph with intervals $\mu(C)$ (i.e., μ maps each interval to an interval defined by timestamps) such that $\mu(\Sigma)$ is consistent.

Definition 4 (Entailment). *Let $C_1 = (G_1, \Sigma_1)$ and $C_2 = (G_2, \Sigma_2)$ be c-temporal graphs. Define $C_1 \models_{\tau(constr)} C_2$ if and only if for each time-ground instance $\nu_1(C_1)$ of C_1 there is a time ground instance $\nu_2(C_2)$ of C_2 such that $\nu_1(C_1) \models_\tau \nu_2(C_2)$.*

Example 4. Let C_1 be the c-temporal graph

$$\{(a, \mathtt{sc}, b)[T_1], (b, \mathtt{sc}, c)[t_1, t_2], t_1 \leq t_2, t_1 < T_1, T_1 < t_2\}.$$

In this graph, $t_1 \leq t_2$ is the constraint associated to the interval $[t_1, t_2]$. The following entailment holds: $C_1 \models_{\tau(constr)} \{(a, \mathtt{sc}, c)[T_2]\}$.

Example 5. Let C_1 be the c-temporal graph

$$\{(a, \mathtt{sc}, b)[T_1, T_2], (b, \mathtt{sc}, c)[T_3, T_4], T_1 \leq T_2, T_3 \leq T_4, T_1 \leq T_3 \leq T_2\}.$$

The first two constraints $T_1 \leq T_2, T_3 \leq T_4$ are the constraints associated to the intervals $[T_1, T_2], [T_3, T_4]$, respectively. The last constraint shows that an "overlap" between the two intervals can be entailed. From this overlap and the transitivity of the subclass relationship, we obtain the entailment $C_1 \models_{\tau(constr)} \{(a, \mathtt{sc}, c)[T_5]\}$.

The following lemma can be easily verified.

Lemma 2. *Let $C_1 = (G_1, \Sigma_1)$ and $C_2 = (G_2, \Sigma_2)$ be c-temporal graphs. If $C_1 \models_{\tau(constr)} C_2$, then $C_1 \models_{\tau(constr)} (G_2, \emptyset)$.*

4.3 Reasoning

First, we extend the interval containment relationship of Section 3.2 to intervals over anonymous timestamps restricted by constraints. Given two intervals $[l_1, l_2], [l_3, l_4]$ and a set of temporal constraints Σ, we write that $[l_1, l_2] \sqsubseteq_\Sigma [l_3, l_4]$ iff $\Sigma \models l_3 \leq l_1, l_2 \leq l_4$. Given an interval $[l_i, l_f]$ and a set of intervals S, we denote by $\mathtt{ContainSet}_\Sigma([l_i, l_f], S)$ the set of intervals $[l_u, l_v] \in S$ such that $[l_u, l_v] \sqsubseteq_\Sigma [l_i, l_f]$.

The following definition extends the notion of slice closure (Definition 2) to c-temporal graphs.

Definition 5. *Let $C = (E, \Sigma)$ be a c-temporal graph. The slice closure of C, denoted $\mathtt{cscl}(C)$, is a c-temporal graph (F, Σ), where F is defined as follows:*

1. *Let F' be the following c-temporal graph. For each pair of labels $l_i, l_f \in L(C)$, $F'[l_i, l_f] = \mathrm{cl}(\bigcup_{[l_u, l_v] \in \mathtt{ContainSet}_\Sigma([l_i, l_f], I(C))} E[l_u, l_v])$.*
2. *Then, for each set of intervals $S = \{[l_1, l_2], [l_2, l_3], \ldots, [l_{n-1}, l_n]\} \subseteq I(F')$, we have $F[l_1, l_n] = \bigcap_{[l_i, l_{i+1}] \in S} F'[l_i, l_{i+1}]$.*

Given a c-temporal graph $C = (G, \Sigma)$ we define a *representative instance* of C, as the ground-temporal instance $\nu(C)$ such that: (i) each time label $l \in L(\Sigma)$ goes to a unique fresh timestamp $\nu(l)$; and for all labels $l_1, l_2, l_3, l_4 \in L(\Sigma)$ if $[l_1, l_2] \not\sqsubseteq_\Sigma [l_3, l_4]$, then $[\nu(l_1), \nu(l_2)] \not\sqsubseteq [\nu(l_3), \nu(l_4)]$. It can be verified that consistent c-temporal graph always has a representative instance; just properly instantiate the inequality graph to satisfies properties (i) and (ii). In particular, property (i) can be accomplished because C is assumed to be free of equalities.

Lemma 3. *Let $C = (G, \Sigma)$ be a c-temporal graph.*
(1) For each time-ground instance $\gamma(C)$ of C, $\gamma(\texttt{cscl}(C)) \subseteq \texttt{iscl}(\gamma(C))$.
(2) $\texttt{cscl}(C) \equiv_{\tau(constr)} C$.
(3) For a representative instance $\nu(C)$ of C, $\nu(\texttt{cscl}(C)) = \texttt{iscl}(\nu(C))$

Proof. (1) Consider H', H and $G = \gamma(C)$ in Definition 2. Let $r \in I(C)$. observe that if $r \in \texttt{ContainSet}_\Sigma(r, I(C))$, then $\gamma(r) \in \texttt{ContainSet}(\gamma(r), I(G))$. Therefore, $\gamma(\texttt{ContainSet}_\Sigma(r, I(C))) \subseteq \texttt{ContainSet}(\gamma(r), I(G))$ (we assume that in the left-hand part of the expression we apply the mapping γ to a set of intervals obtaining a set of ground intervals). Because of this, we have $\gamma(F')[\gamma(r)] \subseteq H'[\gamma(r)]$ for all $r \in I(C)$, and hence $\gamma(F') \subseteq H'$. By a similar argument, we prove that $\gamma(F) \subseteq H$, and hence $\gamma(\texttt{cscl}(C)) \subseteq \texttt{iscl}(\gamma(C))$.
(2) Because $C \subseteq \texttt{cscl}(C)$, it is direct that $\texttt{cscl}(C) \models_{\tau(constr)} C$. Now, we prove that $C \models_{\tau(constr)} \texttt{cscl}(C)$. Consider an time-ground instance $\gamma(C)$ of C. We have that $\gamma(\texttt{cscl}(C))$ is also a time-ground instance of $\texttt{cscl}(C)$. But from (1), it follows that $\gamma(\texttt{cscl}(C)) \subseteq \texttt{iscl}(\gamma(C))$. Therefore, $\gamma(C) \models_\tau \gamma(\texttt{cscl}(C))$. Therefore, $C \models_{\tau(constr)} \texttt{cscl}(C)$.
(3) Consider H', H and $G = \gamma(C)$ in Definition 2. Let $r \in I(C)$. In this case we have $\gamma(\texttt{ContainSet}_\Sigma(r, I(C))) = \texttt{ContainSet}(\gamma(r), I(G))$. From this we can verified that $H' = F'$ and $H = F$.

A c-temporal graph is consistent if it has at least one temporal-ground instance. Since we can entail anything from a inconsistent c-temporal graph, we will study entailment from consistent graphs. In order to simplify the presentation, we subsequently assume that c-temporal graphs $C = (G, \Sigma)$ are consistent.

We define interval mappings between c-temporal graphs. Let $C_1 = (G_1, \Sigma_1)$ and $C_2 = (G_2, \Sigma_2)$ be two independent c-temporal graphs. An interval mapping from C_2 to C_2 is a function $\mu : I(C_2) \to I(C_1)$, which satisfies $\Sigma_1 \models_{constr} (\Sigma_2 \cup \Sigma_u)$, where Σ_u has the constraints $\{l_3 \leq l_1, l_2 \leq l_4 \ : \ \mu([l_1, l_2]) = [l_3, l_4]\}$.

Theorem 4. *Let $C_1 = (G_1, \Sigma_1), C_2 = (G_2, \Sigma_2)$ be c-temporal RDF graphs. Then $G_1 \models_{\tau(constr)} G_2$ if and only if there exist an interval map γ from C_2 to C_1 and a map μ from $\gamma(C_2)$ to $\texttt{cscl}(C_1)$.*

Proof. **(IF)** Consider a time-ground instance $\gamma_1(C_1)$ of C_1. Let $\gamma_2 = \gamma \circ \gamma_1$. Now, because $\gamma_1(C_1)$ is a time-ground instance, γ_2 maps intervals to ground intervals. Because $\Sigma_1 \models_{constr} (\Sigma_2 \cup \Sigma_u)$, $\gamma_2(\Sigma_2)$ is consistent. Therefore $\gamma_2(C_2)$ is a time ground instance of C_2. It remains to prove that $\gamma_1(C_1) \models_\tau \gamma_2(C_2)$. But we have $\mu(\gamma(C_2)) \subseteq \texttt{cscl}(C_1)$. We apply γ_1 to both sides and obtain $\mu(\gamma_2(C_2)) \subseteq \mu(\gamma_1(\texttt{cscl}(C_1))$. But from Lemma 3 (1), it follows that $\mu(\gamma_1(\texttt{cscl}(C_1)) \subseteq \mu(\texttt{iscl}(\gamma_1(C_1)))$. Hence $\mu(\gamma_2(C_2)) \subseteq \mu(\texttt{iscl}(\gamma_1(C_1)))$. Then, from Theorem 3, we obtain $\gamma_1(C_1) \models_\tau \gamma_2(C_2)$.
(Only If) Let $\gamma_1(C_1)$ be the representative instance of C_1. by Lemma 3 (3), we have that $\gamma_1(\texttt{cscl}(C_1)) = \texttt{iscl}(\gamma_1(C_2))$. From Definition 4 it follows that there exists a temporal-ground instance $\gamma_2(C_2)$ of C_2 such that $\gamma_1(C_1) \models_\tau \gamma_2(C_2)$. Then, from Theorem 3, there is an interval mapping γ' and a mapping ν from $\gamma'((\gamma_2(C_2)))$ to $\texttt{iscl}(\gamma_1(C_1))$. Then ν is a mapping from $\gamma'((\gamma_2(C_2)))$ to $\gamma_1(\texttt{cscl}(C_1))$. Let $\gamma = \gamma_2 \circ \gamma' \circ \gamma_1^{-1}$ (notice that γ_1 is 1-1 because it is a representative instance). Then we have that Then ν is a mapping from $\gamma(C_2)$ to $\texttt{cscl}(C_1)$.

4.4 Complexity

A standard result regarding RDFS entailment is that the closure $\mathrm{cl}(G)$ of an RDF G graph is of polynomial size in $|G|$; computing the closure also takes polynomial time (an upper bound for both is $O(n^3)$, where n is the number of RDF terms mentioned in G). We consider a polynomial $p(|G|)$ that bounds the size of the closure and the time it takes to compute it. We also consider a polynomial $q(|\Sigma|)$ that bounds the time of computing an implication of temporal constraints.

Lemma 4. *Let $C = (G, \Sigma)$ be a temporal graph with intervals and let $(E, \Sigma) = \texttt{cscl}(C)$. (1) The graph E is of size $O(N^2 p(|G|))$, where $N = |L(C)|$. (2) The slice closure can be computed in time $O(N^4(q(|\Sigma| + p(|G|)))$.*

Proof. (1) Notice in Definition 5 that H' has at most N^2 intervals, and for each interval r of them the size of $H'[r]$ is in $O(p(|G|))$. Therefore, H' is of size $O(N^2 p(|G|))$. In the second condition of Definition 5, the number of triples added are at most $O(N^2 p(|G|))$, because there are at most N^2 sequences of valid intervals $[l_1, l_2], [l_2, l_3], \ldots [l_{k-1}, l_k]$. Hence, the size of the slice closure is in $O(N^2 p(|G|))$.
(2) For an interval $[l_i, l_j]$, the computation of $H'[l_i, l_j]$ takes $O(N^2 q(|\Sigma|) + p(|G|))$ steps, hence the computation of H' takes $O(N^4 q(|\Sigma|) + p(|G|))$ steps. The computation involved in condition (2) of Definition 5 takes $O(N^3 p(|G|))$ steps. We can do it by finding paths in the inequality graphs and performing intersection operations for each of them. There are at most N^2 paths and for each of them the intersection operations takes $O(Np(|G|))$ steps. Hence, the overall computation is dominated by the computation of H'.

Better complexity bounds for computing the slice closure could be certainly obtained by developing more efficient algorithms, an issue we do not address in this paper. We next show that the decision problem of entailment for c-temporal graphs is NP-complete, thus maintaining the complexity of temporal graphs (and also of the non-temporal case).

Theorem 5. *(1) Given two temporal c-temporal graphs C_1, C_2, the problem of deciding whether $C_1 \models_{\tau(constr)} C_2$ is NP-complete. (1) Given two temporal graphs with intervals G_1, G_2, the problem of deciding whether $G_1 \models_{\tau} G_2$ is NP-complete.*

Proof. (1) Membership in NP-hard follows from the fact that testing the implication $H_1 \models_{\tau} H_2$, for two temporal graphs H_1, H_2, is equivalent to testing the implication for the temporal graphs with intervals resulting from H_1, H_2 by replacing each triple $(a, b, c)[t]$ by $(a, b, c)[t, t]$. Proof of membership in NP is as follows. A witness for $G_1 \models_{\tau} G_2$ is a pair of mappings that satisfies condition of Theorem 4, which can be tested in polytime since $\texttt{cscl}(G_1)$ is of polysize and can be computed in polytime (Lemma 4).
(2) It is direct by a similar argument.

5 Conclusion

In this paper we have extended temporal RDF graphs with temporal constraints and intervals. In this way, temporal reasoning about these constructs is enabled. First, taking advantage of the support of blank nodes in RDF, we introduced intervals such that one and/or both boundaries are anonymous timestamps. We developed a notion of closure for temporal RDF graphs with intervals. Then, we introduced c-temporal graphs (temporal graphs with constraints and the intervals previously defined), and gave a notion of closure for these temporal graphs. We also proved that entailment from such graphs reduces to finding mappings to the "closed" version of the graphs. These results also show that query processing for temporal graphs with constraints also reduces to computing a matching between the query and the closed graphs.

Open problems are the modeling of more expressive classes of temporal constraints [8] in temporal graphs. As an example, an interesting generalization would be to model constraints expressed via simple temporal networks [9], which allow to state upper an lower bounds for the distance of pairs of time labels. We are also beginning to work on an implementation of the theoretical framework presented here, which comprises the design and implementation of algorithms and heuristics for computing the slice closure and testing entailment.

References

1. F. Afrati, C. Li, and P. Mitra. On containment of conjunctive queries with arithmetic comparisons. In *UCIISC Technical Report*, 2003.
2. J. Allen. Maintaining knowledge about temporal intervals. *Communications of the ACM 26(11)*, pages 832–843, 1983.
3. J. Allen. Time and time again: The many ways to represent time. *International Journal of Intelligent Systems, 6(4)*, pages 341–355, 1990.
4. T. Amagasa, M. Yoshikawa, and S. Uemura. A temporal data model for XML documents. In *Proceedings of DEXA Conference*, pages 334–344, 2000.
5. G. Antoniou and F. van Harme. *A Semantic Web Primer*. MIT Press, London, England, 2004.
6. D. Brickley and R.V.(Eds.) Guha. RDF vocabulary description language 1.0: RDF schema. *W3C Working Draft 23 January 2003*.
7. S. Chawathe, S. Abiteboul, and J. Widom. Managing historical semistructured data. In *Theory and Practice of Object Systems, Vol 5(3)*, pages 143–162, 1999.
8. J. Chomicki. Temporal query languages: a survey. In *Proceedings of First International Conference on Temporal Logic. Lecture Notes in Artificial Intelligence 827, Springer-Verlag*, Bonn, Germany, 1994.
9. R. Dechter, I. Meiri, and J. Pearl. Temporal constraint networks. In *Artificial Intelligence 40:61*, pages 49: 61–95, 1991.
10. C.E. Dyreson. Observing transaction-time semantics with TTXPath. In *Proceedings of WISE 2001*, pages 193–202, 2001.
11. C. Gao and R. Snodgrass. Temporal slicing in the evaluation of XML queries. In *Proceedings of the 29th International Conference on Very Large Data Bases*, pages 632–643, Berlin, Germany, 2003.

12. C. Gutierrez, C. Hurtado, and A.O. Mendelzon. Formal aspects of querying RDF databases. In *Proceedings of SWDB*, pages 293–307, 2003.
13. C. Gutierrez, C. Hurtado, and A.O. Mendelzon. Foundations of semantic web databases. In *23rd. Symposium on Principles of Database Systems (PODS'04)*, pages 95–106, 2004.
14. C. Gutierrez, C. Hurtado, and A. Vaisman. Incorporing time into RDF. In *Internal Technical Report, Department of Computer Science, Universidad de Chile (submitted for journal review)*, 2005.
15. C. Gutierrez, C. Hurtado, and A. Vaisman. Temporal RDF. In *European Conference on the Semantic Web (ECSW'05) (Best paper award)*, pages 93–107, 2005.
16. P. Haase, J. Broekstra, A. Eberhart, and R. Volz. A comparison of RDF query languages. In *International Semantic Web Conference*, 2004.
17. Patrick Hayes(Ed.). RDF semantics. *W3C Working Draft, October 1st., 2003*.
18. M. Koubarakis. Temporal query languages: a survey. *Information Systems, 19(2):141-174*, 1993.
19. O. Lassila and R.(Eds.) Swick. Resource description framework (RDF) model and syntax specification. *W3C Working Draft, 1998*.
20. A. Magkanaraki, G. Karvounarakis, T.T. Anh, V. Christophides, and D. Plexousakis. Ontology storage and querying. *Technical Report No. 308 Foundation for Research and Technology Hellas, Institute of Computer Science, Information System Laboratory*, 2002.
21. F. Rizzolo, A.O. Mendelzon, and A. Vaisman. Indexing temporal XML documents. In *Proceedings of the 30th International Conference on Very Large Databases*, pages 216–227, Toronto, Canada, 2004.
22. Richard Snodgrass. *The TSQL2 Temporal Query Language*. Kluwer Academic Publishers, 1995.
23. A. Tansel, J. Clifford, and S. Gadia (eds.). *Temporal Databases: Theory, Design and Implementation*. Benjamin/Cummings, 1993.
24. U. Visser. Intelligent information integration for the semantic web. *Lecture Notes in Artificial Intelligence (3159)*, 2004.
25. G. Yang and M. Kifer. On the semantics of anonymous identity and reification. In *Proceedings of the First International Conference on Ontologies, Databases and Applications of Semantics (ODBASE)*, pages 1047–1066, 2002.