Reasoning with 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 based on Allen's interval algebra. We show that testing entailment in temporal graphs with the constraints proposed 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)* [22] 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*.

* This research was supported by Millennium Nucleus, Center for Web Research (P01-029-F), Mideplan, Chile. C. Hurtado was supported by FONDECYT 1030810, Chile.

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 on 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 [17].

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. [1] The interval [0,3] 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 "3". 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". Anonymous (or blank) nodes are needed in an RDF graph when we do not know the global name for a node (or there is no name for it, no matter the reason why), and we need to write statements about this node. The impact of blank nodes in a temporal setting was given an in-depth study in [17,16].

1.1 Problem Statement

In former work 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* [17,16]. The work was based on the theoretical framework provided by Gutierrez *et al* [15].

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 $(a, b, c) : [t_1, t_2]$. The graph in Figure 1 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 in a discrete time dimension. An RDF graph can be regarded as a knowledge base from which new knowledge,

[1] 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.

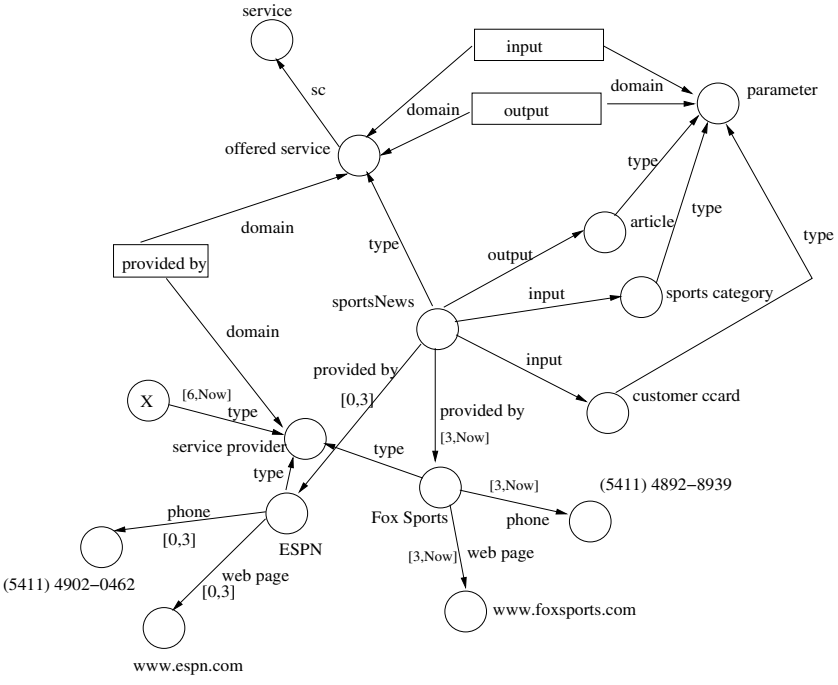

Fig. 1. An RDF graph for web services profiling of Sports networks

i.e., other graphs, may be entailed. In temporal RDF, entailment is slightly more involved. We studied this problem, and called it *temporal entailment*.

An important 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* 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.

The work in [16] also 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 time we do not exactly know yet (in [16] we called these graphs *general temporal graphs* to differentiate them from temporal graphs without blank timestamps). In our model, the sets of anonymous timestamps and blank nodes are disjoint, as we will explain later in the paper (actually they belong to different frameworks, namely time labels and triples, respectively).

Temporal blanks considerably extend the capabilities of the temporal RDF model by allowing representing incomplete temporal information [20]. In this

paper we show that they also allow defining temporal constraints over the model. In this way, a richer treatment of time, along the lines of *constraint databases* [10] 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 (anonymous intervals) in in temporal RDF graphs.

Example 1. Consider the following extended temporal graph:

$$\{(a, \texttt{sc}, b) : i_1, (b, \texttt{sc}, c) : i_2, i_1 \texttt{ during } i_2\}.$$

here i_1 and i_2 are intervals whose endpoints are unknown. The temporal triple $(a, \texttt{sc}, b) : i_1$ states that $(a, \texttt{sc}, b)$ holds in all the timestamps inside the interval (which are infinite), and the constraint i_1 `during` i_2 states that the i_1 is inside i_2. Then, our approach allows inferring the graph $\{(a, \texttt{sc}, c) : i_3\}$. Intuitively, this means that, given the original temporal graph, we can infer that in some unknown interval i_3, a was a subclass of c.

Relation	**Meaning**
$[l_1, l_2]$ `before` $[l_3, l_4]$	$l_2 < l_3$
$[l_1, l_2]$ `meets` $[l_3, l_4]$	$l_2 = l_3$
$[l_1, l_2]$ `overlaps` $[l_3, l_4]$	$l_3 < l_2 < l_4$ and $l_1 < l_3 < l_2$
$[l_1, l_2]$ `starts` $[l_3, l_4]$	$l_1 = l_3$ and $l_2 < l_4$
$[l_1, l_2]$ `during` $[l_3, l_4]$	$l_3 < l_1$ and $l_2 < l_4$
$[l_1, l_2]$ `ends` $[l_3, l_4]$	$l_1 > l_3$ and $l_2 = l_4$
$[l_1, l_2]$ `equals` $[l_3, l_4]$	$l_1 = l_3$ and $l_2 = l_4$

Fig. 2. Basic Interval Relations

The temporal RDF graphs with constraints and anonymous intervals we introduce in this paper (denoted *c-temporal graphs*), 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 3. The triple (Sports News, provided by, ESPN) is now labeled with an anonymous interval i_1. (instead of [0,2]). Analogously, (Sports News, provided by, Fox Sports) is labeled with an anonymous interval i_3 (instead of [3,Now]). We have

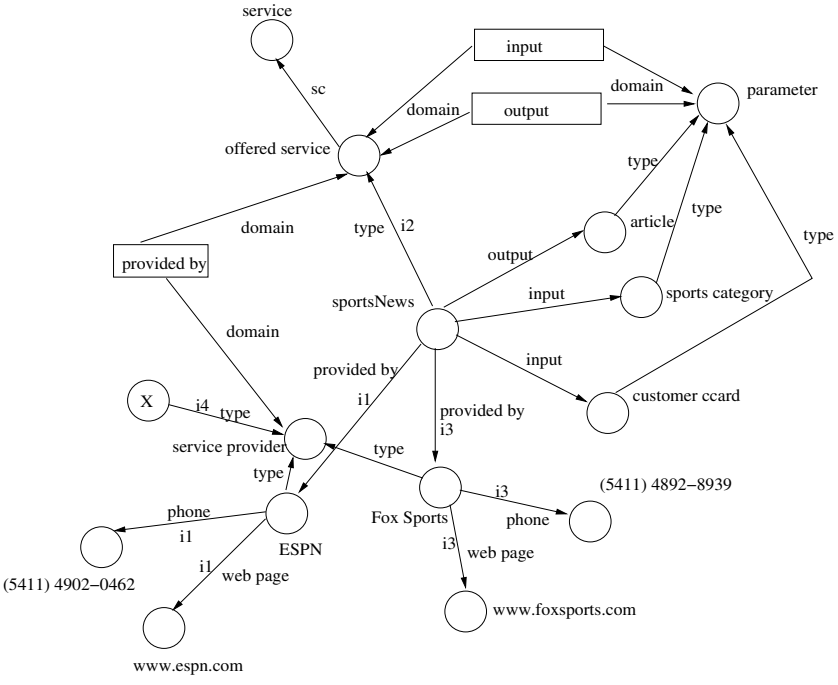

Fig. 3. The RDF graph of Figure 1 with Anonymous Time

also labeled with anonymous intervals the triples (Sports News, `type`, offered service), and (X,`type`, service provider) has been labeled with temporal blanks.

Now, we can use the basic Allen's interval relations [2] depicted in Figure 2 to place constraints over the anonymous intervals. As an example, we can use the constraint i_1 `meets` i_3 to state that Fox Sports started offering 'Sport News' immediately after ESPN stops offering it. We can also state that ESPN started offering the service at time 0 with the constraint i_1 `starts` $[0, Now]$. We can model that during interval i_1 the service was of type 'offered service' using the constraint i_1 `during` i_2.

Although the addition of anonymous time enriches the model, it introduces some problems that we study in the paper. Many of the results obtained in [16] 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 under the assumption that the temporal labels were only concrete time instants.

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. In this paper we study in detail *temporal graphs with constraints*, extending our previous results to these kinds of graphs.

1.2 Contributions and Outline

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. We consider intervals over a dense time domain, which allows a full treatment of intervals in temporal RDF.

Then, we generalize the former framework incorporating a fragment of Allen's interval algebra [2] for 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 the fragment of constraints studied, reduces to closure computation and mapping discovery, that is, an extended form of the standard approach for testing entailment in non-temporal RDF graphs. We also provide an algorithm for computing the slice closure of c-temporal graphs.

The remainder of the article is organized as follows. Section 2 reviews related work. Section 3 presents preliminary notation related to RDF and RDFS and temporal RDF graphs from previous work [15,17,16]. Section 4 studies temporal graphs with intervals. Section 5 introduces constraints to temporal graphs and their semantics, presents the notion closure, and characterizes entailment in terms of them. Finally, in Section 6 we conclude and outline some prospects for future work.

2 Related Work

The RDF model was introduced in 1998 by the World Wide Web Consortium (W3C) [22]. Formal work includes the study of formal aspects of RDF data and query languages [14,15,28], 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 [18,21]. Temporal database management has been extensively studied, including data models, mostly based on the relational model and query languages [26], leading to the TSQL2 language [25]. Chomicki [10] 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 [9,4,12,13,24].

Regarding temporal extensions to RDF, Visser *et al* [27] 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. Bry *et al* [8,7], in the context of the REWERSE project [23], have stated the need of providing query languages and models for the web with temporal reasoning capabilities.

To the best of our knowledge, our previous work [17,16] constitutes the first formal study of temporality issues in RDF graphs and RDF query languages. In the present paper we continue this line of research with the study of Temporal RDF graphs with constraints and anonymous time.

3 Preliminaries

3.1 RDF Notation

The following is an excerpt of notation introduced in [6,15,19] 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 literals (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.

In this paper, we use a working characterization of the standard notion of entailment between RDF graphs (cf. [19]), which will be denoted by $\models$. We use the known notion of *closure* of a RDFS graph G, denoted $\mathrm{cl}(G)$, which is the maximal graph that can be derived from the set of inference rules given in [19,15].

Theorem 1 (cf. [19,15]). *$G_1 \models G_2$ if and only if there is a map from G_2 to the* closure *of G_1.*

3.2 Temporal Graphs

In this section, we present in a compressed form relevant notation and results for temporal graph from previous work [17,16].

A *temporal triple* is an RDF triple (a, b, c) with a temporal timestamp t, which is a positive rational number. We will use the notation $(a, b, c) : [t]$. The *snapshot* of a temporal graph G at t, is defined as the graph $G(t) = \{(a, b, c) \mid (a, b, c) : [t] \in G\}$. 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 $\{(\mu(a), \mu(b), \mu(c)) : [t] \mid (a, b, c) : [t] \in G\}$, and so on.

Definition 1 (Entailment (c.f. [16])). *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*.

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 $\texttt{scl}(G)$, is a temporal graph defined by the expression $\bigcup_t (\text{cl}(G(t)))^t$, where $\text{cl}(G(t))$ is any closure of the RDF graph $G(t)$.

Theorem 2 (c.f. [17]). *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.

4 Temporal Graphs with Time Intervals

In this section we extend temporal graphs introduced in Section 3.2 to model time intervals defined by timestamps, that is, intervals whose extremes are positive rational numbers.

4.1 Basic Definitions

We extend *temporal triples* to triples of the form $(a, b, c) : i$, where $i = [t_1, t_2]$ is an interval defined by the timestamps (positive rational numbers) t_1, t_2, $t_1 \leq t_2$, yielding *temporal graphs with intervals*. For the case where $t_1 = t_2$ a triple $(a, b, c) : [t_1, t_2]$ is equivalent to temporal triple $(a, b, c) : [t_1]$, as defined in Section 3.2, 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 that appear as bounds in the intervals in $I(G)$. Two timestamps $t_1, t_2 \in T(G)$ are *consecutive* if there is

no timestamp $t' \in T(G)$, such that $t_1 < t' < t_2$. Given an interval $i \in I(G)$, we denote by $G(i)$, the set containing RDF triples (a, b, c) such that $(a, b, c) : i \in G$.

A temporal graph with intervals represents a (possibly infinite) temporal graph, that is, each triple $p : [t_1, t_2]$ represents the set of temporal triples $\{p : [t] \mid t_1 \leq t \leq t_2\}$. Given a temporal graph with intervals G, we denote by G^+ the temporal graph that represents G. In this form, the notion of entailment from Definition 1 can be naturally extended to temporal graph with intervals. Formally, we write $G \models_\tau H$ iff $G^+ \models_\tau H^+$.

4.2 Reasoning

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 give a characterization of entailment that yields a procedure for the testing entailment of temporal graphs with intervals.

Given two intervals $[t_1, t_2], [t_3, t_4]$, we write that $[t_1, t_2]$ `contains` $[t_3, t_4]$ iff $t_1 \leq t_3$ and $t_4 \leq t_2$. Given an interval i and a set of intervals S, we denote by $\texttt{CoverSet}(i, S)$ the set containing intervals $i' \in S$ such that i' `contains` i.

The following definition extends the notion of slice closure (Section 3.2) to temporal graphs with intervals.

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

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

Example 3. Consider the temporal graph with constraints $G = \{(a, \texttt{sc}, b) : [1, 3], (b, \texttt{sc}, c) : [2, 4], (a, \texttt{sc}, c) : [3, 5]\}$. First, we illustrate condition 1 of Definition 2. As an example, consider the two timestamps $2, 3 \in T(G)$. Then $\texttt{CoverSet}([2, 3], I(G)) = \{[1, 3], [2, 4]\}$. Therefore, $H'([2, 3]) = \mathrm{cl}((a, \texttt{sc}, b), (b, \texttt{sc}, c)\})$, which is $\{(a, \texttt{sc}, b), (b, \texttt{sc}, c), (a, \texttt{sc}, c)\}$. Now, in order to explain condition 2 of Definition 2, consider the set of consecutive timestamps $2, 3, 4, 5$ in $T(G)$. Then, $H([2, 5]) = H'([2, 3]) \cap H'([3, 4]) \cap H'([4, 5])$, which is $\{(a, \texttt{sc}, c)\}$.

For simplicity, the previous example use only the subclass property (`sc`). The example could be easily turned much more complex if we include in the graphs other RDFS built-in-properties.

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) If there is a triple $(a, b, c) \in \texttt{scl}(G^+)(t)$ *for all timestamps* t *in some arbitrary interval* i*, then there is an interval* $i' \in I(\texttt{iscl}(G))$ *such that* i' `contains` i *and* $(a, b, c) : i' \in \texttt{iscl}(G)$.

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 $i \in S$, $i' = \gamma(i)$ should satisfy i' `contains` i. When we apply an interval mapping to a temporal graph with intervals G, we obtain the temporal graph with interval G' containing the triples $(a, b, c) : \gamma(i)$ such that $(a, b, c) : i \in G$. In addition, we extend maps between temporal graphs (see Section 3.2) 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* $\texttt{iscl}(G)$.

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 5.4, we study the complexity of testing entailment.

5 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 4.

5.1 Temporal Constraints

In this paper, we focus on a basic fragment of the known Allen's interval algebra [2]. The temporal primitive here is an interval $[l_i, l_f]$ which is an ordered pair of *time labels* l_i, l_f. Time labels may be timestamps (positive rational numbers) or anonymous timestamps, which are temporal variables. 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, which is the set of positive rational numbers. So we assume a dense temporal domain.

The algebra considers one of the seven relationships depicted in Figure 2 to state relationships between intervals. By a *temporal constraint* we refer to an expression of the form $l_i \ \ \omega \ \ l_j$, where ω is one of the seven relationships of Allen's algebra.

Given a set of temporal constraints Σ we denote $I(\Sigma)$ the intervals in Σ and by $L(\Sigma)$ the temporal labels that appears in intervals in $I(\Sigma)$. A map for a set of temporal constraints Σ is a function γ from $I(\Sigma)$ to ground intervals (intervals whose limits are timestamps) preserving timestamps. We denote by $\gamma(\Sigma)$ the set of constraints resulting from Σ by replacing each interval i by $\gamma(i)$.

An *instance* for a set of temporal constraints $\Sigma = \{\alpha_1, \ldots, \alpha_n\}$ is a map μ such that $\mu(\Sigma)$ is ground (i.e., mentions only timestamps) and each $\gamma(\alpha_i)$ holds in the temporal domain. If Σ is empty the empty set 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 an instance of $\gamma(\Sigma_2)$.

Testing entailment and consistency for the class of temporal constraints considered can be done in polynomial time. Following standard results in inequality constraints (e.g. [3,1]), we can represent the fragment we presented in a point based algebra, by building the *inequality graph* for the labels in the constraints (which is a particular case of a temporal constraint network [11]), that is, a directed graph with a node for each temporal label in $L(\Sigma)$, and edges (l_i, l_j) labeled with the arithmetic comparisons $=, <, \leq$, that models Allen's relationships in Σ. As an example, the constraint $[l_1, l_2]$ `during` $[l_3, l_4]$ yields $l_1 \leq l_2, l_3 \leq l_4, l_3 < l_1$, and $l_2 < l_4$. The graph has also edges that capture the natural ordering between timestamps mentioned in the constraints and between each pair of time labels that mark the bound of an interval. The arithmetic constraint in the inequality graph can be propagated by simple transitive closure computation, yielding the closed inequality graph, which can be used for implementing an efficient testing of entailment and consistency of a set of constraints. We refer the reader to e.g., [3,1,11] for further details.

In this paper we consider constraints Σ whose inequality graph is totally ordered (modulo renaming time labels that are entailed to be equal). Therefore, even though the intervals themselves may be unknown, the relationship between any two of them is fully determined by the constraints, that is for all $i, i' \in I(\Sigma)$ we have $\Sigma \models_{constr} i \ \omega \ i'$ (or the inverse $i' \ \omega \ i$) for some interval relation ω .

5.2 Basic Definitions

We extend the notion of temporal graph to handle anonymous labels in timestamps and interval. So we consider a temporal triple to be an element of the form $p : i$, where p is an RDF triple and i is an interval.

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 and Σ is a set of temporal constraints over the intervals of G.*

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 by $I(C)$ and $L(C)$ the intervals and time labels that appear in the triples in G.

Interval maps defined in Section 4 can be naturally extended to consider intervals defined with temporal labels. If we apply an interval map ν to a c-temporal graph C, we obtain another c-temporal graph, denoted $\nu(C)$, by renaming each interval r with $\nu(r)$. A time-ground instance of a c-temporal graph $C = (G, H)$ is a temporal graph with intervals $\nu(C)$ (i.e., ν maps each interval to an interval defined by timestamps) such that $\nu(\Sigma)$ 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

$$\{(a, \mathtt{sc}, b) : i_1, (b, \mathtt{sc}, c) : i_2, i_1 \ \mathtt{during}\ i_2, i_1 \ \mathtt{starts}\ [3, now]\}.$$

The following entailment holds: $C_1 \models_{\tau(constr)} \{(a, \mathtt{sc}, c) : i_3, i_3 \ \mathtt{starts}\ [3, now]\}$.

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 $C_1 \models_{\tau(constr)} (G_2, \emptyset)$.*

5.3 Reasoning

First, we extend the interval containment relationship of Section 4.2 to intervals over anonymous timestamps restricted by constraints. Given a set of intervals S, and an interval i, we denote by $\mathtt{CoverSet}_{\Sigma}(i, S)$ the set of intervals $i' \in S$ that can be entailed from Σ to contain i.

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 $H = \mathtt{cscl}(C)$, 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 $l_1, l_2 \in L(C)$, $F'([l_1, l_2]) = \mathrm{cl}(\bigcup_{i \in \mathtt{CoverSet}_{\Sigma}([l_1, l_2], I(C))} C(i))$.*
2. *Then, for each set of consecutive labels $l_1, l_2, l_3, \ldots, l_{n-1}, l_n$ in $L(C)$, we have $F([l_1, l_n]) = \bigcap_{[l_j, l_{j+1}] \in S} F'[l_j, l_{j+1}]$.*

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

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_1 is a function $\mu : I(C_2) \rightarrow I(C_1)$, which satisfies $\Sigma_1 \models_{constr} (\Sigma_2 \cup \Sigma_u)$, where Σ_u is the following set of 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 $C_1 \models_{\tau(constr)} C_2$ 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 $\mathtt{cscl}(C_1)$.*

5.4 Algorithm and Complexity

Theorem 4 yields an algorithm for testing the entailment $C_1 \models_{\tau(constr)} C_2$, which consists of the following two steps: (i) compute the slice closure $\texttt{cscl}(C_1)$ by applying rules (1) and (2) of Definition 5; and (ii) find an interval map γ from C_2 to C_1 and a map μ from $\gamma(C_2)$ to $\texttt{cscl}(C_1)$. Step (ii) is similar to finding a mapping between non-temporal graphs [15]. In the remaining of this section we study the complexity of the two steps of the algorithm.

A standard result regarding RDFS entailment is that the closure $\text{cl}(G)$ 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 $\texttt{cscl}(C)$ can be computed in time $O(N^4(q(|\Sigma| + p(|G|))))$.*

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.*

As stated previously, for testing whether $C_1 \models_{\tau(constr)} C_2$, Theorem 4 requires the inequality graph of Σ_1 to yield a total ordering of time labels. However, if this is not the case, the condition can be adapted to be required by each topological ordering of the inequality graph. So, testing entailment for graphs with few topological orderings still does not add extra complexity to RDF entailment. Further techniques can be used to make this case of entailment more efficient. As an example, if the graph has connected components it is enough to consider combinations of topological orderings inside each component, while keeping a fixed ordering for the components themselves, thus reducing significantly the processing. We left this problem for future work.

6 Conclusion

In this paper we have extended temporal RDF graphs with a class of temporal constraints over 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 boundaries may be 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 show that query processing for temporal graphs with constraints also reduces to computing a matching between the query and the closed graphs.

We left as future work the study of entailment for more expressive classes of constraints based either in Allen's interval algebra or point algebras [10]. In particular, we plan to study entailment for the case where the constraints do not entail a total ordering of anonymous timestamps. We are also beginning to work on an implementation of the theoretical framework presented here.

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 Recommendation, 10 February 2004*.
7. F. Bry, B. Lorenz, H.J. Ohlbach, and S. Spranger. On reasoning on time and location on the web. In *Proceedings of ICLP03*, Mumbai, India, 2003.
8. F. Bry and S. Spranger. Temporal constructs for a web language. In *Proceedings of the 4 Workshop on Interval Temporal Logics and Duration Calculi, ESSLLI'03*, Austria, 2003.
9. 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.
10. 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.
11. R. Dechter, I. Meiri, and J. Pearl. Temporal constraint networks. In *Artificial Intelligence 40:61*, pages 49: 61–95, 1991.
12. C.E. Dyreson. Observing transaction-time semantics with TTXPath. In *Proceedings of WISE 2001*, pages 193–202, 2001.
13. 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.
14. C. Gutierrez, C. Hurtado, and A.O. Mendelzon. Formal aspects of querying RDF databases. In *Proceedings of SWDB*, pages 293–307, 2003.

15. 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.
16. C. Gutierrez, C. Hurtado, and A. Vaisman. Introducing time into RDF. In *(to appear) IEEE Transactions on Knowledge and Data Engineering, Special Issue on Knowledge and Data Engineering in the Semantic Web Era, 2007*.
17. 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.
18. P. Haase, J. Broekstra, A. Eberhart, and R. Volz. A comparison of RDF query languages. In *International Semantic Web Conference*, 2004.
19. Patrick Hayes(Ed.). RDF semantics. *W3C Recommendation, 10 February 2004*.
20. M. Koubarakis. Temporal query languages: a survey. *Information Systems, 19(2):141-174*, 1993.
21. 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.
22. F. Mannola and E.(Eds.) Miller. Rdf primer. *W3C Recommendation, 10 February 2004*.
23. The REWERSE Project. http://www.rewerse.net.
24. 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.
25. Richard Snodgrass. *The TSQL2 Temporal Query Language*. Kluwer Academic Publishers, 1995.
26. A. Tansel, J. Clifford, and S. Gadia (eds.). *Temporal Databases: Theory, Design and Implementation*. Benjamin/Cummings, 1993.
27. U. Visser. Intelligent information integration for the semantic web. *Lecture Notes in Artificial Intelligence (3159)*, 2004.
28. 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.