A Survey on Models and Query Languages for Temporally Annotated RDF

Anastasia Analyti
Institute of Computer Science, FORTH-ICS,
Heraklion, Greece

Ioannis Pachoulakis
Dept. of Applied Informatics & Multimedia, TEI of Crete,
Heraklion, Greece

Abstract— **In this paper, we provide a survey on the models and query languages for temporally annotated RDF. In most of the works, a temporally annotated RDF ontology is essentially a set of RDF triples associated with temporal constraints, where, in the simplest case, a temporal constraint is a validity temporal interval. However, a temporally annotated RDF ontology may also be a set of triples connecting resources with a specific lifespan, where each of these triples is also associated with a validity temporal interval. Further, a temporal RDF ontology may be a set of triples connecting resources as they stand at specific time points. Several query languages for temporally annotated RDF have been proposed, where most of which extend SPARQL or translate to SPARQL. Some of the works provide experimental results while the rest are purely theoretical.**

Keywords- Temporal RDF; provenance; semantics; query languages.

I. Introduction

RDF ("Resource Description Framework") [1], [2] is a growing semantic web standard for the specification of ontologies. An RDF ontology contains a set of triples (*s*,*p*,*o*), denoting that subject *s* is associated with object *o* by property *p*. However, this information is static meaning that either does not change over time or the whole RDF ontology corresponds to a particular time point. However, the truth of statements often changes with time and Semantic Web applications often need to represent such changes and reason about them. For example, statements regarding airline flights are valid only in certain time intervals. Validity time should also be integrated in the query language allowing to retrieve "flights from London to Paris during Mary's summer vacation". Some additional example temporal queries are the following:

1. Who are the *foaf*:*Persons* whose lifespan overlaps with Einstein's?
2. What is the temperature in Chicago at sunrise of July 20th, 2008?
3. What are the names of the engineers who committed code to a particular software in the first half of 2008?
4. What is the salary of Tom during the interval [2007-01-01, 2009-12-31]?
5. Who was the head of the german government before and after the unification of 1990?
6. Who are the service providers that provide web services for more than 4 consecutive years?

Who are the house members who sponsored a bill after April 2, 2008?

In this paper, we provide a survey on the models and query languages for temporally annotated RDF. In most of the works, a temporally annotated RDF ontology is essentially a set of RDF triples associated with temporal constraints, where, in the simplest case, a temporal constraint is a validity temporal interval. However, a temporally annotated RDF ontology may also be a set of triples connecting resources with a specific lifespan, where each of these triples is also associated with a validity temporal interval. Further, a temporal RDF ontology may be a set of triples connecting resources as they stand at specific time points. Several query languages for temporally annotated RDF have been proposed, where most of which extend SPARQL [3] or translate to SPARQL, the most widely accepted query language for RDF. Some of the works provide experimental results while the rest are purely theoretical.

We divide reviewed works into three main categories: (a) works that they have their own model theory (Section 2), (b) works that they extend RDF simple entailment [2] (Section 3), and (c) works that they extend RDFS entailment [2] (Section 4). Works that extend RDF simple entailment are further divided into works that directly translate into RDF and those that do not. Section 5 concludes the paper and provides a comparison of the presented approaches

II. Works With Their Own Model Theory

In [4], a *temporal RDF* (*tRDF* for short) *database* is a set of triples of the form (*s*, *p*:{*T*}, *o*), (*s*, *p*:<*n*:*T*>, *o*), (*s*, *p*:[*n*:*T*], *o*), and (*p rdfs*:subPropertyOf *p'*), where *s* is a URI reference from a set *U*, *p*,*p'* are URI references from a set *P*, *o* is an entity from $R = U \cup L$, where *L* is a set of literals, *n* is a natural number, and *T* is a temporal interval. Intuitively, the triple (*s*, *p*:{*T*}, *v*) indicates that the association (*s*, *p*, *o*) holds at every time point in *T*, the triple (*s*, *p*:<*n*:*T*>, *o*) indicates that the association (*s*, *p*, *o*) holds at least *n* time points within *T*, and the triple (*s*, *p*:[*n*:*T*], *o*) indicates that the association (*s*, *p*, *o*) holds at most *n* time points within *T*. An *interpretation I* of a *tRDF* database is a function from the set of time points to $U \times P \times R$. Satisfaction of a *tRDF* triple is defined in such a way that intuitive meaning is preserved. Obviously, a *tRDF* database may be inconsistent due to the temporal constraints imposed to RDF triples.

A *tRDF* query over a *tRDF* database *D* is a set of triples of the form (*s*, *p*:{*T*}, *o*), (*s*, *p*:<*n*:*T*>, *o*), (*s*, *p*:[*n*:*T*], *o*), where *s*,*p*,*o*,*T* are possibly variables, with the constraint that each temporal variable appears only once. An answer to a *tRDF* query q is the set of all possible substitutions to the variables in q such that all triples in q after proper substitutions are entailed by D.

To efficiently answer *tRDF* queries, a *tGRIN* index structure is proposed such that temporally closed resources and resources close in the *tRDF* graph are stored in the same index node. Query answering using the *tGRIN* index is shown to outperform query answering using R^+-trees, *SR*-trees, and the *ST*-index, the most promising representatives of valid-time indexing methods, according to [5].

In [6], the authors extend RDF triples with an annotation from a set A which is a partially ordered set. We consider the case that A is the set of all temporal intervals $[t,t']$, where t,t' are natural numbers. The inclusion ordering $\subseteq$ is the partial ordering in this set. An *annotated RDF theory* (*aRDF-theory* for short) is a finite set of triples (s, p:a, o), where s is a resource from a set R, p is a property from a set P, o is a resource in R, and $a \in A$. In addition, an *aRDF* theory contains statements (p, *rdfs:subProperyOf,* p'), where $p,p' \in P$, and statements indicating which properties are *transitive*.

Let O be an *aRDF* theory, let p be a transitive property in O, and let $r,r' \in R$. Then, there is a *p-path* between r,r' if there exists a set of triples $t_1=(r,p_1{:}a_1,r_1),\ldots,\ t_k=(r_{k-1},p_k{:}a_k,r')$ such that for all $i \in [1,k]$, (p_i *rdfs*:*subPropertyOf* * p). A p-path Q is indicated by the set of triples $\{t_1,\ldots,t_k\}$ that form the path.

An *interpretation* I is a mapping from the set of triples (s,p,o), where $s,o \in R$ and $p \in P$, to A. An interpretation I satisfies (s,p:a,o) iff $a \subseteq I(s,p,o)$. I satisfies an *aRDF* theory O iff (i) I satisfies every $(s,p{:}a,o) \in O$ and (ii) for all transitive properties $p \in P$, for all p-paths $Q=\{t_1,\ldots,t_k\}$ in O, where $t_i=(r_i,p_i{:}a_i,r_{i+1})$, and for all $a \in A$ such that $a \subseteq a_i$, it is the case that $a \subseteq I(r_1,p,r_{k+1})$, for all $i \in [1,k]$.

A *simple aRDF query q* has the form (s,p:a,o), where s,p,a,o can be variables. $A_O(q)$ consists of all ground instances of q that are entailed by O. However, $A_O(q)$ may contain redundant triples. For example, if $(a,p{:}[1,100],o) \in A_O(q)$, then there is no point including redundant triples such as $(a,p{:}[1,10],o)$ in it. $Answer_O(q)$ eliminates all redundant triples from $A_O(q)$.

A *conjunctive query Q* is a set of simple *aRDF* queries such that for any simple query $q \in Q$, there is a variable in q that appears in another simple query $q' \in Q$.

The authors present efficient algorithms for simple and conjunctive query answering, showing that the time complexity for answering a conjunctive query is in $O((|R|^2*|P|)^{|Q|})$, where $|Q|$ is the number of simple queries in Q.

The authors also provide experimental results showing the efficiency of their approach.

III. Works That Extend RDF Simple Entailment

A. Approaches that translate to RDF

In [7], instead of having RDF triples associated with their validity temporal interval, named graphs [8] are used both for saving space and for querying the temporal RDF database using standard SPARQL. In particular, each created named graph g is associated with a temporal interval i and all RDF triples whose validity interval is i become members of g (in this process blank nodes are replaced by URIs). The authors introduce through examples a query language, named τ-SPARQL which extends the SPARQL query language for RDF graphs. Each τ-SPARQL query can be translated into a SPARQL query.

A τ-SPARQL query that retrieves all *foaf*:*Persons* whose lifespan overlaps with Einstein's is:

```
SELECT ?s2, ?e2 ?person WHERE {
        [?s1, ?e1] ?einstein foaf:name "Albert Einstein"
        [?s2, ?e2] time:intervalOverlaps [?s1, ?e1]
        [?s2, ?e2] ?person a foaf:Person.}
```

This query is translated into a SPARQL query, as follows:

```
SELECT ?s2, ?e2 ?person WHERE{
        GRAPH ?g1 {?einstein foaf:name "Albert Einstein".}
        ?g2 time:intervalOverlaps ?g1.
        GRAPH ?g2 {?person a foaf:Person.}
        ?g2 time:hasBegining ?s2.
        ?g2 time:hasEnd ?e2.}
```

Temporal relationships between named graphs, such that *time*:*intervalOverlaps* are derived from a temporal reasoning system. Additionally, the authors propose an index structure for time intervals, called *keyTree index*, assuming that triples within named graphs have indices by themselves. The proposed index improves the performance of time point queries over an in-memory ordered list that contains the intervals' start and end times.

Experimental results are provided.

In [9], the *time-annotated RDF* framework is proposed for the representation and management of time-series streaming data. In particular, a TA-RDF graph is a set of triples $<s[t_S], p[t_p], o[t_o]>$, where $<s,p,o>$ is an RDF triple and t_S, t_p, and t_o are time points. In other words, a TA-RDF graph relates streams at certain points in time. To translate a TA-RDF graph into a regular RDF graph, a data stream vocabulary is used, where (i) *dvs*:*belongsTo* is a propery that indicates that a resource is a frame in a stream, (ii) *dvs*:*hasTimestamp* is a property indicating the timestamp of a frame, and (iii) *dvs*:*Nil* is a resource corresponding to the Nil timestamp.

An RDF graph G is the *translation* of a TA-RDF graph G^{TA} iff (B is the set of blank nodes):

$$<s[t_S], p[t_p], o[t_o]> \in G^{TA} \Leftrightarrow \exists\ r_S, r_p, r_o$$

$$[(<r_S, dvs{:}belongsTo, s> \in G \wedge <r_S, dvs{:}hasTimestamp, t_S> \in G \wedge r_s \in B) \vee (t_S = dvs{:}Nil \wedge r_S = s)] \wedge$$
$$[(<r_p, dvs{:}belongsTo, p> \in G \wedge <r_p, dvs{:}hasTimestamp, t_p> \in G \wedge r_p \in B) \vee (t_p = dvs{:}Nil \wedge r_p = p)] \wedge$$
$$[(<r_o, dvs{:}belongsTo, o> \in G \wedge <r_o, dvs{:}hasTimestamp, t_o> \in G \wedge r_o \in B) \vee (t_o = dvs{:}Nil \wedge r_o = o)] \wedge$$
$$< r_S, r_p, r_o> \in G.$$

A query language for the time-annotated RDF, called TA-SPARQL, is proposed which has a formal translation into normal SPARQL. For example, a TA-SPARQL query requesting the temperature in Chicago at sunrise of July 20th, 2008 is:

```
SELECT ?temperature WHERE {
        <urn:OHARE> <urn:hasTemperatureSensor> ?x.
        ?x[“2008-07-20T05:34:00Z”^^xsd:dateTime]
                <urn:hasReading>  ?temperature.}
```

The above TA-SPARQL query is translated into the SPARQL query:

```
SELECT ?temperature WHERE {
        <urn:OHARE> <urn:hasTemperatureSensor> ?x.
        ?F <urn:hasReading> ?temperature.
        ?F dvs:belongsTo ?x.
        ?F dvs:hasTimestamp ?F_T.
        FILTER(?FT= “2008-07-20T05:34:00Z”^^xsd:dateTime)}
```

The system has been implemented on top of the Tupelo[1] semantic middleware. However, no experimental results are provided.

In [10], the authors consider temporal RDF graphs which is a set of triples of the form (*s*,*p*:[*start*,*end*],*o*), where (*s*,*p*,*o*) is an RDF triple and *p*:[*start*,*end*] is a shorthand for a URI that identifies a temporal property which has *base property p*, beginning *start* and ending *end*.

The authors define a *simple temporal interpretation* by extending an RDF simple interpretation as follows:

1. *T* is a subset of the set of resources.
2. *NT* is a value representing no time.
3. The set of properties contains *tb*:*property*, *tb*:*begin*, and *tb*:*end*.
4. *BP* is a subset of resources, called the set of *base properties*.
5. *PT* is a mapping from $BP \times (T \cup \{NT\}) \times (T \cup \{NT\})$ into the set of properties.
6. If $tp=PT(bp,t_1,t_2)$ then (i) $(tp,bp) \in$ IEXT(*tb*:*property*) , (ii) if $t_1 \neq NT$ then $(tp,\ t_1) \in$ IEXT(*tb*:*begin*), otherwise IEXT(*tb*:*begin*) contains no pair (*tp*, *t*), for any *t*, and (iii) if $t_2 \neq NT$ then $(tp,\ t_2) \in$ IEXT(*tb*:*end*), otherwise IEXT(*tb*:*end*) contains no pair (*tp*,*t*), for any *t*.

As temporal RDF graphs are ordinary RDF graphs they can be queried using normal SPARQL. However, it is helpful to the writer of temporal queries to provide some extra syntax to enable queries to be written more compactly and to hide the details of the underline representation.

For example, a query asking for the names of the engineers who committed code to a particular software in the first half of 2008 is the following:

```
SELECT ?name WHERE {
        ?module rdfs:label “module name”.
        ?module f:updatedBy: (?uBegin, ?uEnd) ?person.
        FILTER (tb:intervalsIntersect(?uBegin,?uEnd,
                “2008-01-01”^^xsd:date,
                “20008-7-01”^^xsd:date))
        ?person ex:hasName ?name.}
```

[1] Tupelo. http://tupeloproject.ncsa.uiuc.edu

This query can be expressed in normal SPARQL, as follows:

```
SELECT ?name WHERE {
        ?module rdfs:label “module name”.
        ?updatedBy tb:property f:updatedBy.
        ?updatedBy tb:begin ?uBegin.
        ?updatedBy tb:end ?uEnd.
        FILTER (tb:intervalsIntersect(?uBegin,?uEnd,
                        “2008-01-01”^^xsd:date,
                        “20008-7-01”^^xsd:date))
        ?module f:updatedBy ?person.
        ?person ex:hasName ?name.}
```

Though an implementation of a prototype is mentioned, no experimental results are provided.

B. Other approaches

In [11], an *N*-dimensional time domain has the form: $\mathcal{T}=T_1 \times \ldots \times T_N$, where each T_i is a set of intervals. A *multi-temporal RDF triple* is defined as (*s*,*p*,*o* | *T*), where <*s*,*p*,*o*> is an RDF triple and $T \subseteq \mathcal{T}$. Note that since *T* is a set, some compression is achieved in the storage of multi-temporal RDF triples.

As a query language, the authors propose T-SPARQL, an extension of SPARQL that has many features of TSQL2 [12] (a query language designed for temporal relational databases). As in TQL2, if *T* is a multi-dimensional time element, the expression VALID(*T*) and TRANSACTION(*T*) can be used to express conditions on the valid and transaction components of *T*.

T-SPARQL is demonstrated through examples and a query that requests the salary of Tom during the interval [2007-01-01,2009-12-31] is the following:

```
SELECT ?salary INTERSECT(?t, “ [2007-01-01,2009-12-31]”)
WHERE {
?emp rdf:type ex:employee;
                ex:Name “Tom”;
                ex:Salary ?salary | ?t.
FILTER (VALID(?t) OVERLAPS
“ [2007-01-01,2009-12-31]”^^xs:period)}
```

No implementation of T-SPARQL is provided.

In [13], an *uncertain temporal knowledge base* is a pair *KB* = <*F*, *C*>, where *F* is a set of weighted temporal RDF triples and *C* is a set of first-order temporal consistency constraints. In particular, a fact in *F* has the form: $p(s,o,i)_d$, where $p(s,o)$ is an RDF triple, *i* is a temporal interval, and $d \in [0,1]$ is a confidence degree that $p(s,o)$ is true during interval *i*. Additionally, a temporal consistency constraint in *C* has the form:

$$p_1(?s,?o_1,?i_1) \wedge p_2(?s,?o_2,?i_2) \wedge relA(?o_1,?o_2) \rightarrow rel_T(?i_1,?i_2)$$

or of the form:

$$p_1(?s,?o_1,?i_1) \wedge p_2(?s,?o_2,?i_2) \wedge relA(?o_1,?o_2) \rightarrow false$$

where $?i_1$, and $?i_2$ are temporal interval variables, *relA* is an (optional) arithmetic relation, such as = and ≠, and rel_T is a temporal predicate such as *overlap* and *before* (see Allen's temporal relations among intervals [14]).

For example, the fact that a player can only play for one club at a time is expressed by the query:

$playsForClub(?s,?o_1,?i_1) \wedge playsForClub(?s,?o_2,?i_2) \wedge$
$?o_1 \neq ?o_2 \rightarrow disjoint(?i_1,?i_2)$

A query Q is a conjunction of triples $p(s,o)$, where s and o can be variables. To answer a query Q, all matches from the KB at collected into a set F_Q. Then, all facts possibly conflicting with them are also added to F_Q. To resolve the conflicts, a consistent subset $F_{Q,C}$ of F_Q is selected such that the sum of the weights of the facts in $F_{Q,C}$ is maximized. Then, the matches to Q within $F_{Q,C}$ are returned as answer to the query. The query answering problem is shown to be NP-hard. A scheduling algorithm for query answering is provided, as well as an efficient approximation algorithm with polynomial performance. Experimental results show the efficiency of the proposed approach.

In [15], the authors extend RDF with temporal features and evolution operators. In addition, in contrast to the rest of the reviewed works, they associate concepts with their lifespan. In particular, an *evolution base* Σ is a set of RDF triples and a mapping τ from the set of considered RDF triples and considered resources to the set of temporal intervals. In addition, Σ may contain statements of the form (c, *term*, c'), where *term* is one of the special evolution properties *becomes, join, split, merge,* and *detach.*

The expression (c, *becomes*, c') expresses that the concept c' originates from the concept c and should hold $\tau(c).end < t(c').start$. The expression ($c$, *join*, c') expresses that a part of concept c' born at time t comes from a part of concept c. The expression (c, *spilt*, c') expresses that a part of concept c ending at time t becomes a part of a new concept c'. The expression (c, *merge*, c') indicates that a part of concept c ending at time t becomes part of an existing concept c'. The expression (c, *detach*, c') indicates the new concept c' is formed at time t with at least one part from c.

An evolution base Σ is consistent, if for all $(s,p,o) \in \Sigma$ it holds that $\tau(s,p,o) \subseteq \tau(s)$ and $\tau(s,p,o) \subseteq \tau(o)$. Additionally, if $p \in \{type, subClassOf, subPropertyOf\}$ then it should hold that $\tau(s) \subseteq \tau(o)$.

To support evolution-aware querying, the authors define a navigational query language to traverse temporal and evolution edges in an evolution graph. This language is analogous to nSPARQL [16], a language that extends SPARQL with navigational capabilities based on nested regular expressions. nSPARQL uses four different axes, namely *self* , *next, edge,* and *node,* for navigation on an RDF graph and node label testing. The authors extend the nested regular expressions constructs of nSPARQL with temporal semantics and a set of five evolution axes, namely *join, split, merge, detach,* and *become*s that extend the traversing capabilities of nSPARQL to the evolution edges. The extended query language is formally defined.

An example query is "who was the head of the German government before and after the unification of 1990". The query is expressed as follows:

SELECT ?*Y*, ?*W*
(?*X*, **self**::*Reunified Germany*/**join**$^{-1}$[1990]/
next::*head*[1990], ?*Y*) AND
(?*Z*, **self**::*Reunified Germany*/**next**::*head*[1990], ?*W*)

The first triple finds all the heads of state of the *Reunified Germany* before the unification by following **join**$^{-1}$[1990] and then following **next** :: *head*[1990]. The second triple finds the heads of state of the Reunified Germany after the unification.

No implementation results of this theory are provided.

IV. Works That Extend RDFS Entailment

In [17], a *temporal graph* is a set of temporal triples of the form $(s,p,o)[t]$, where (s,p,o) is an RDF triple and t is a time point. Given a temporal graph G, $G(t)$ denotes the set of RDF triples in G corresponding to time point t.

The authors define *temporal entailment* between two temporal graphs G, G' as follows:

1. For ground temporal RDF graphs G, G', define $G \models_\tau G'$ iff $G(t) \models_{RDFS} G'(t)$, for each t.
2. For general temporal graphs G, G', $G \models_\tau G'$ iff for every ground instance $v(G)$ of G, there exists a ground instance $v'(G')$ of G' such that $v(G) \models_{RDFS} v'(G')$.

It is shown that temporal entailment is NP-complete. To test temporal entailment, the authors define the *slice closure* of G, as follows $scl(G)= \cup_t (cl(G(t)))^t$, where $cl(H)$ is the RDFS closure [18] of an RDF graph H and $H^t=\{(s,p,o)[t] \mid (s,p,o) \in H\}$. In particular, it is proved that $G \models_\tau G'$ iff there is a mapping v such that $v(G')$ is a subgraph of $scl(G)$.

The authors extend their theory to support also anonymous timestamps.

A query is defined as a pair $(H, B \cup A)$, where H and B are temporal RDF graphs without blank nodes and with some elements replaced by variables and A is a set of usual arithmetic built-in predicates over time point variables and time points. All variables appearing in H should also appear in B. For deriving maximal validity intervals a special structure is used. For example a query that asks for the service providers that have web services for more than 4 consecutive years is:

$$(?X, interval, ?t_e\text{-}?t_s) \leftarrow (?Y, provided\ by, ?X) \parallel ?t_s, ?t_e \parallel, ?t_e - ?t_s > 4.$$

No implementation of this theory is provided.

In [19], the authors extend the work in [17] and they define a *temporal graph* as a set of temporal triples of the form $(s,p,o){:}i$, where (s,p,o) is an RDF triple and i is a temporal interval variable or a temporal interval. A *temporal constraint* is an expression of the form $i\ \omega\ i'$, where i, i' are temporal intervals or temporal interval variables and ω is one of the relationships of Allen's temporal interval algebra [14]. A *temporal graph with temporal constrains* (called *c-temporal graph*) is a pair $C = (G, \Sigma)$, where G is a temporal graph and Σ is a set of temporal constraints over the intervals of G.

The authors define entailment between two *c*-temporal graphs *C*, *C'* as follows: $C \models_{\tau(const)} C'$ iff for each time ground instance $v(C)$ of *C*, there is a time ground instance $v'(C')$ of *C'* such that $v(C) \models_{\tau} v(C')$. The authors define the *c-slice closure* of *C*, denoted by *cscl*(*C*), extending the definition of slice closure of [17]. It is proved that $C \models_{\tau(const)} C'$ iff there is an interval map γ from *C'* to *C* and a mapping *v* s.t. $v(\gamma(C'))$ is a subgraph of *cscl*(C). Entailment between two *c*-temporal graphs is shown to be NP-complete. No query language or implementation is provided.

In [20], [21], [22], the authors consider an extension of RDFS with spatial and temporal information. Here, we consider only the extension with temporal information. Assume a set *D* of RDF triples associated with their validity temporal interval *i*. Starting from *D*, the authors apply the inference rules $A{:}?i,\ B{:}?i' \rightarrow C{:}\ ?i \cap ?i'$, where $A, B \rightarrow C$ is an RDFS entailment rule [2] and ?*i*, ?*i'* are temporal interval variables, until a fixpoint is reached. Then, the temporal intervals of the same RDF triple are combined, creating maximal temporal intervals.

Based on these maximal temporal intervals, a formal extension of the SPARQL language is proposed, called SPARQL-ST, supporting however only the AND and FILTER operations. The TEMPORAL FILTER condition is precisely defined supporting all interesting conditions between temporal intervals including Allen's temporal interval relations.

An example SPARQL-ST query that returns all house members who sponsored a bill after April 2, 2008, along with the temporal interval that the bill was sponsored is:

```
SELECT ?p, intersect(#t1, #t2, #t3, #t4) WHERE {
        ?p gov:hasRole ?r #t1.
        ?r gov:forOffice ?o #t2.
        ?o gov:isPartOf gov:congress_house #t3.
        ?p gov:sponsor ?b #t4.
        TEMPORAL FILTER (
        after(intersect(#t1, #t2, #t3, #t4), interval(04:02:2008,
                04:02:2008, MM:DD:YYYY)))}
```

SPARQL-ST has been implemented by extending a commercial relational database system and experimental results are provided.

In [23], [24], the authors extend the RDFS and ter-Horst entailment rules [25] (which extend RDFS with terms from the OWL [26] vocabulary) with temporal information. Four example inference rules are presented, including:

?*s* ?*p* ?*o* ?*b* ?*e*
?*p rdfs*:*domain* ?*dom*
→
?*s rdf*:*type* ?*dom* ?*b* ?*e*

?*p rdf*:*type owl*:*FunctionalProperty*
?*p rdf*:*type owl*:*ObjectProperty*
?*x* ?*p* ?*y* ?*b1* ?*e1*
?*x* ?*p* ?*z* ?*b2* ?*e2*
→
?*y owl*:*sameAs* ?*z*
provided that [?*b1*,?*e1*] and [?*b2*,?*e2*] overlap

?*p rdf*:*type owl*:*FunctionalProperty*
?*p rdf*:*type owl*:*DatatypeProperty*
?*x* ?*p* ?*y* ?*b1* ?*e1*
?*x* ?*p* ?*z* ?*b2* ?*e2*
→
?*x rsd*:*type owl*:*Nothing*
provided that $?y \neq ?z$ and [?*b1*,?*e1*] and [?*b2*,?*e2*] overlap

Note that in the above rules ?*b*, ?*e*, ?*b1*, ?*e1*, ?*b2*, and ?*e2* are time point variables. The last rule indicates that inconsistency is expressed by assigning the bottom type *owl*:*Nothing* to individuals. For checking consistency two additional rules must be added addressing a combination of *owl*:*sameAs* and *owl*:*differentFrom*, as well as *owl*:*disjointWith* together with two *rdf*:*type* statements.

The proposed extension has been implemented using the forward chaining engine *HFC* [27], which supports arbitrary tuples, user defined tests, and actions. Some experimental results are provided. However, no query language is provided.

In [28], a general framework for representing, reasoning, and querying annotated RDFS data is presented. The authors show how their unified reasoning framework can be instantiated for the temporal, fuzzy, and provenance domain. Here, we are concerned with the temporal instantiation. We define $\perp=\{\{\}\}$ and $\top=\{[-\infty,+\infty]\}$. Let $L=\{t \mid t$ is a finite set of disjoint temporal intervals$\} \cup \{\perp, \top\}$. On *L*, the authors define the partial order:

$t \leq t'$ iff for all $i \in t$, there is $i' \in t'$ such that $i \subseteq i'$.

Obviously, $(L, \leq, \perp, \top)$ is a bounded lattice. Between the elements of *L*, the authors define the operations + and × are follows: $t_1 + t_2 = inf(t \mid t_i \leq t,\ i=1,2\}$ and $t_1 \times t_2 = sup(t \mid t \leq t_i,\ i=1,2\}$. For example, {[2,5],[8,12]} + {[4,6],[9,15]} = {[2,6],[8,15]} and {[2,5],[8,12]} × {[4,6],[9,15]} = {[4,5], [9,12]}. An annotated *RDFS graph G* is a set of temporal triples (*s*,*p*,*o*) :*t*, where (*s*,*p*,*o*) is an RDF triple and $t \in L$. The models of *G* are formally defined extending ρRDF semantics, where ρRDF [29] is a subset of RDFS keeping its essential features.

The authors present a set of sound and complete inference rules of the general form:

$(s_1, p_1, o_1) : ?t_1, \ldots,\ (s_n, p_n, o_n) : ?t_n,$
$\{(s_1, p_1, o_1), \ldots, (s_n, p_n, o_n)\} \vdash_{\rho RDF} (s, p, o)$
→
$(s,p,o) : (?t_1 \times \ldots \times ?t_n)$

For example:

$(c_1, rdfs{:}subClassOf, c_2){:}?t_1,\ (c_2, rdfs{:}subClassOf, c_3){:}?t_2$
→
$(c_1, rdfs{:}subClassOf, c_3){:}?t_1 \times ?t_2$

Additionally, the inference rules contain the generalization rule:

$(s,p,o) : ?t,\ (s,p,o) : ?t' \rightarrow (s,p,o) : (?t + ?t')$.

The generalization rule is destructive, meaning that this rule removes its premises as the conclusion is inferred.

An extension of SPARQL is presented for querying an annotated RDF graph. A *basic annotated pattern* is an expression (*s*,*p*,*o*):*t*, where *s, p, o, t* can be variables. Let *P* be a basic annotation pattern and G be a temporal graph. The authors define the evaluation $[P]_G$ as the list of substitutions that are solutions of P, i.e., $[P]_G=\{\theta \mid G \text{ entails } \theta(P)\}$. Based on $[P]_G$ the evaluations $[P \text{ AND } P']_G$, $[P \text{ UNION } P']_G$, $[P \text{ FILTER } R]_G$, $[P \text{ OPTIONAL } P'[R]]_G$ are formally defined, where R is a filter expression.

An example query asking for the employees of eBay during some time period that optionally owned a car at some point during their stay is:

```
SELECT ?p, ?t, ?c WHERE {
        (?p type ebayEmp): ?t
        OPTIONAL {(?p hasCar ?c): ?t'
                FILTER (?t' ⊆ ?t)}}
```

Note that the definition of $[P]_G$ is not based on maximal temporal intervals and, thus all temporal intervals that satisfy the query are returned. Therefore, the authors define an ordering between substitutions: $\theta' \leq \theta$ iff (i) $\theta \neq \theta'$, (ii) $domain(\theta) = domain(\theta')$, (iii) $\theta(x) = \theta'(x)$, for any non-temporal variable x, and (iv) $\theta'(t) \subseteq \theta(t)$, for any temporal variable t. Then, for any $\theta \in [P]_G$, remove any $\theta' \in [P]_G$ such that $\theta' \leq \theta$.

No implementation is provided for this theory.

In [30], a *temporal graph G* is a set of temporal triples $(s,p,o)[t,t']$, where (s,p,o) is an RDF triple and $[t,t']$ is its corresponding validity temporal interval. The semantics of a temporal graph G, assuming an entailment relation X (such as RDF, RDFS, and OWL2 RL/RDF [31] entailment) are formally defined using multi-sorted first order logic.

A *basic graph pattern* (*BGP*) is a set of triples (s,p,o), where s,p,o can be variables. A *temporal group pattern* (*TGP*) is an expression defined inductively, as follows:

B at t_3,	B during $[t_1,t_2]$,	B occurs $[t_1,t_2]$
B maxinterval $[t_1,t_2]$,	B mintime t_3,	B maxtime t_3,
P_1 and P_2,	P_1 union P_2,	P_1 optional P_2,
P_1 filter R,		

where B is a *BGP*, P_1 and P_2 are *TGPs*, R is a built-in expression, and t_1,t_2, and t_3 are either time points or variables. Note that a *TGP* query is an extension of a SPARQL query. For example, a *TGP* query that retrieves all events z in London having at least one time point in common with Oktoberfest is:

```
SELECT ?z WHERE {
{(Munich, hosts, Oktoberfest)} maxint [?x,?y].
(London, host,?z)} occurs [?x,?y].}
```

The evaluation of a *TGP* query w.r.t. a temporal graph G and an entailment relation X is formally defined using multi-sorted first-order logic. Yet, evaluation of a *TGP* using this definition can be inefficient. Therefore, the authors describe an optimization.

Assume that the entailment relation X is characterized by a set of definite rules of the form: $A_1,..,A_n \rightarrow B$.

Then, the rules:

$$A_1[x_1,y_1], \ldots, A_n[x_n,y_n], max(x_1,\ldots,x_n) \leq min(y_1,..,y_n) \rightarrow B[max(x_1,\ldots,x_n), min(y_1,..,y_n)]$$

are applied until a fixpoint is reached, where x_i and y_i are time point variables. Then, based on the result, derived RDF triples are associated with their maximal validity intervals. Now, based on these maximal intervals the evaluation of a *TGP* query is efficiently defined.

Though the authors state that they have implemented their framework using the PostgreSQL database system, no implementation results are provided.

V. Conclusion-Discussion

In this paper, we have reviewed models and query languages of temporally annotated RDF. Below, we compare these models and query languages on various aspects. First, we would like to state that approaches that have their own model theory or extend RDF simple entailment miss important inferences made from the works that extend RDFS entailment. For example, an object o may be an instance of class c during a temporal interval i and the class c may be subclass of a class c' during an interval i'. Only works that extend RDFS entailment are able to derive that o is instance of class c' during the intersection of the intervals i and i'.

From the works that extend RDFS entailment, the approach in [17] seems less efficient since it computes the RDFS closure of RDF triples at each time point. Additionally, [28] considers all temporal intervals that satisfy the query and then selects the maximal ones. In contrast, [22] and [30] achieve query answering using directly maximal temporal intervals achieving a higher performance.

In our opinion, the approach in [28] does not give always the desirable results: For example, assume that an annotated RDFS graph consists of the triples (s,p,o):[1998, 2009] and (s',p',o'):[2008, 2012]. Consider now the query:

```
SELECT ?t, ?t' WHERE {
(s,p,o): ?t.
(s',p',o'): ?t'.
FILTER (before(?t,?t')}}
```

Then, [28] will return the answers (i) ?t=[1998, 2007], ?t'=[2008, 2012], (ii) ?t=[1998, 2008], ?t'=[2009, 2012], and (iii) ?t=[1998,2009], ?t'=[2010, 2012]. In contrast [22], will return no answer to this query since it works with maximal temporal intervals and 2009 > 2008.

In [30], the query:

```
SELECT ?t, ?t' WHERE {
        (s,p,o) during ?t.
        (s',p',o') during ?t',
        FILTER (before(?t,?t')}}
```

will return the answers of [28], as well as the intervals t,t' such that $t \subseteq [1998,2009]$, $t' \subseteq [2008,2012]$, and $t.end < t'.start$. In contrast, in [30], the query:

```
SELECT ?t, ?t' WHERE {
        (s,p,o) maxinterval ?t.
```

(s',p',o') maxinterval ?*t'*,
FILTER (*before*(?*t*,?*t'*)}}

will return no answer. As a criticism, [30] is not able to return maximal intervals within a temporal interval of interest.

Approaches [7], [28], and [11] save some space since they either use name graphs associated with temporal intervals or associate each RDF triple with its set of validity temporal intervals.

Specialized indices for query answering are used only in [4] and [7], while the rest of the approaches use common indexes. As a final remark, we would like to state that [4] can handle some temporal constraints over RDF triples, [17] can handle anonymous timestamps, and [19] can handle anonymous temporal intervals satisfying Allen's temporal interval algebra relations.

Temporal consistency constraints are considered only in [13], which however does not answer temporal queries but only normal queries.

As a criticism to the work in [6], each RDF triple is associated with a single maximal temporal interval while an RDF triple is normally associated with multiple maximal temporal intervals.

Some of the proposed models and query languages have been implemented as stated in the main text of the paper and for some of them experimental results are provided.

In the future, extensions of the proposed temporal RDF query languages with features of SPARQL 1.1 [32], such as subqueries, and negation, will be of great importance. For example, it will be interesting to ask for events that have not occurred simultaneously before a date and their maximal temporal intervals always overlap after that date. Additionally, it will be interesting to ask for companies located in Crete that have exactly one manager at each point in time within a particular temporal interval of interest.

Future work also concerns a survey on spatial, fuzzy, provenance, and contextual RDF. Of course, aspects of contextual RDF can be time, space, trust, and authority.

REFERENCES

[1] G. Klyne and J. J. Carroll, "Resource Description Framework (RDF): Concepts and Abstract Syntax", W3C Recommendation, 10 February 2004, available at http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/.

[2] Patrick Hayes, "RDF Semantics", W3C Recommendation, 10 February 2004, available at http://www.w3.org/TR/2004/REC-rdf-mt-20040210/.

[3] E. Prudhommeaux and A. Seaborne, "SPARQL query language for RDF", W3C Recommendation 15 January 2008, available at http://www.w3.org/TR/rdf-sparql-query/.

[4] A. Pugliese, O. Udrea, and V.S. Subrahmanian. "Scaling RDF with Time", International World Wide Web Conference (WWW), Beijing, China, 2008, pp 605-614.

[5] B. Salzberg and V. J. Tsotras, "Comparison of access methods for time-evolving data", ACM Computing Surveys, 31(2), 1999, pp158–221.

[6] O. Udrea, D. R. Recupero, and V. S. Subrahmanian, "Annotated RDF", ACM Transactions on Computational Logic, 11(2), 2010.

[7] J. Tappolet and A. Bernstein, "Applied Temporal RDF: Efficient Temporal Querying of RDF Data with SPARQL", 6th European Semantic Web Conference (ESWC-2009), 2009, pp. 308-322.

[8] J. J. Carroll, C. Bizer, P. J. Hayes, and P. Stickler, "Named graphs", Journal of Web Semantics, 3(4), 2005, pp. 247-267

[9] A. Rodriguez, R. E. McGrath, Y. Liu, and J. D. Myers, "Semantic Management of Streaming Data", 2nd International Workshop on Semantic Sensor Networks at the International Semantic Web Conference, Washington, 2009.

[10] B. McBride and M. Butler, "Representing and Querying Historical Information in RDF with Application to E-Discovery", HP Laboratories Technical Report, HPL-2009-261, 2009.

[11] F. Grandi, "T-SPARQL: a TSQL2-like Temporal Query Language for RDF", 14th East-European Conference on Advances in Databases and Information Systems (ADBIS-2010) (Local Proceedings), 2010, pp. 21-30.

[12] R.T. Snodgrass (ed.), I. Ahn, G. Ariav, D. Batory, J. Clifford, C.E. Dyreson, R. Elmasri, F. Grandi, C.S. Jensen, W. Kafer, N. Kline, K. Kulkarni, T.Y. Cliff Leung, N. Lorentzos, R. Ramakrishnan, J.F. Roddick, A. Segev, M.D. Soo, and S.M. Sripada, "The TSQL2 Temporal Query Language", Kluwer Academic Publishers, 1995.

[13] M. Dylla, M. Sozio, and M. Theobald, "Resolving Temporal Conflicts in Inconsistent RDF Knowledge Bases", Datenbanksysteme fur Business, Technologie und Web (BTW-2011), 2011, pp. 474-493.

[14] J. Allen, "Maintaining Knowledge about Temporal Intervals", Communications of the ACM, 26(11), 1983, pp. 832-843.

[15] S. Bykau, J. Mylopoulos, F. Rizzolo, and Y. Velegrakis, "On Modeling and Querying Concept Evolution", Journal on Data Semantics, 1, 2012, pp 31-55.

[16] J. Perez, M. Arenas, and C. Gutierrez, "nSPARQL: A navigational language for RDF", Journal of Web Semantics, 8(4), 2010, pp. 255-270.

[17] C. Gutierrez, C. A. Hurtado, and A. A. Vaisman, "Introducing Time into RDF", IEEE Transactions on Knowledge and Data Engineering, 19(2), 2007, 207-218.

[18] C. Gutierrez, C. A. Hurtado, and A.O. Mendelzon, "Foundations of Semantic Web Databases", 23rd Symposium of Principles of Databases Systems (PODS-2004), 2004, pp. 95-196.

[19] C. A. Hurtado and A. Vaisman, "Reasoning with Temporal Constraints in RDF", 4th International Workshop on Principles and Practice of Semantic Web Reasoning (PPSWR-2006), 2006, pp. 164-178.

[20] M. Perry, A. P. Sheth, F. Hakimpour, and P. Jain, "Supporting Complex Thematic, Spatial and Temporal Queries over Semantic Web Data", 2nd International Conference on GeoSpatial Semantics (GeoS-2007), 2007, pp. 228-246.

[21] M. Perry and A. P. Sheth, "A Framework to Support Spatial, Temporal, and Thematic Analytics over Semantic Web Data, Knoesis Center Technical Report, KNOESIS-TR-2008-01, 2008.

[22] M. Perry, P. Jain, and A. P. Sheth, "SPARQL-ST: Extending SPARQL to Support Spatiotemporal Queries", N. Ashish and A.P. Sheth (Eds.) Geospatial Semantics and the Semantic Web - Foundations, Algorithms, and Applications, 2011, pp. 61-86.

[23] H.Krieger, "A Temporal Extension of the Hayes and ter Horst Entailment Rules for RDFS and OWL", AAAI Spring Symposium: Logical Formalizations of Commonsense Reasoning, 2011.

[24] H.Krieger, "A Temporal Extension of the Hayes/ter Horst Entailment Rules and a Detailed Comparison with W3C's N-ary Relations", Deutsches Forschungszentrum fur Kunstliche Intelligenz GmbH Technical Report, RR-11-02, 2011.

[25] H. J. ter Horst, "Completeness, decidability and complexity of entailment for RDF Schema and a semantic extension involving the OWL vocabulary, Journal of Web Semantics, 3(2-3), 2005, pp. 79-115.

[26] G. Antoniou and F. van Harmelen, A semantic web primer, MIT Press, 2004.

[27] H.U. Krieger and G.J.M. Kruijff, "Combining uncertainty and description logic rule-based reasoning in situation-aware robots" Proceedings of the AAAI 2011 Spring Symposium on Logical Formalizations of Commonsense Reasoning, 2011

[28] A. Zimmermann, N. Lopes, A. Polleres, and U. Straccia, "A General Framework for Representing, Reasoning and Querying with Annotated Semantic Web Data", Journal of Web Semantics , 11, 2012, pp. 72–95.

[29] S. Munoz, J. Perez, and C. Gutierrez, "Minimal Deductive Systems for RDF", 4th European Semantic Web Conference (ESWC-2007), 2007, pp. 53-67.

[30] B. Motik, "Representing and Querying Validity Time in RDF and OWL: A Logic-Based Approach", 9th International Semantic Web Conference (ISWC-2010), 2010, pp. 550-565.

[31] B. Motik, B. Cuenca Grau, I. Horrocks, Z. Wu, A. Fokoue, and C. Lutz, "OWL 2 Web Ontology Language Profiles", W3C Recommendation 27 October 2009, available at http://www.w3.org/TR/owl2-profiles/.

[32] Steve Harris and Andy Seaborne, "SPARQL 1.1 Query Language", W3C Working Draft 24 July 2012, available at http://www.w3.org/TR/sparql11-query/

AUTHORS PROFILE

Anastasia Analyti earned a B.Sc. degree in Mathematics from University of Athens, Greece and a M.Sc. and Ph.D. degree in Computer Science from Michigan State University, USA. She worked as a visiting professor at the Department of Computer Science, University of Crete, and at the Department of Electronic and Computer Engineering, Technical University of Crete. Since 1995, she is a principal researcher at the Information Systems Laboratory of the Institute of Computer Science, Foundation for Research and Technology - Hellas (FORTH-ICS). Her current interests include reasoning on the Semantic Web, modular web rule bases, non-monotonic-reasoning, faceted metadata and semantics, conceptual modelling, contextual organization of information, information integration and retrieval systems for the web, interoperability of heterogeneous and distributed information bases, and biomedical information systems. She has participated in several research projects and has published over 55 papers in refereed scientific journals and conferences.

Ioannis Pachoulakis received a B.Sc. in Physics (1988) at the University of Crete, Greece, and a Ph.D. in Astrophysics (1996) and an M.Sc. in Engineering (1998), both from the University of Pennsylvania in the U.S.A. Since 2001, he serves as an Assistant Professor at the Department of Applied Informatics and Multimedia at TEI of Crete with mainstream interests in realistic multimedia applications, virtual reality and multimedia applications for science.