Multidimensional RDF[⋆]

Manolis Gergatsoulis and Pantelis Lilis

Department of Archive and Library Sciences, Ionian University,
Palea Anaktora, Plateia Eleftherias, 49100 Corfu, Greece.
emails: mgerg@otenet.gr, manolis@ionio.gr, pantelis@ionio.gr.gr

Abstract. RDF has been proposed by W3C as a metadata model and language for representing information about resources in WWW. In this paper we introduce *Multidimensional RDF* (or *MRDF*), as an extension of RDF, suitable for representing context-dependent RDF data. In MRDF we have a set of dimensions whose values define different contexts (or worlds) under which different parts of an RDF graph may hold. We define the semantics of MRDF in terms of the semantics of RDF. Additionally, we propose appropriate extensions, suitable for representing MRDF graphs in triples notation and RDF/XML syntax. Finally, we demonstrate how an MRDF graph, embodying a single time dimension, can be used to model the history of a conventional RDF graph.
Keywords: RDF databases, RDF model, Semantic Web, versioning.

1 Introduction

The success of the Internet and the Web during the last few years led to an explosion of the amount of data available. Managing and processing such a huge collection of interconnected data proved to be difficult due to the fact that the Web lacks semantic information. The Semantic Web is a proposal to build an infrastructure of machine-readable metadata (expressing semantic information) for the data on the Web.

In 1998, W3C proposed *Resource Description Framework* (RDF) [5], a metadata model and language which can serve as the basis for such infrastructure. Apart from being a metadata model and language, RDF can also express semantics, empowering the vision of semantic Web. However, RDF falls short when it comes to represent multidimensional information; that is, information that presents different facets under different contexts. Actually, there are many cases where variants of the same information do exist. As a simple example imagine a report that needs to be represented at various degrees of detail and in various languages. A solution would be to create a different document for every possible combination. Such an approach is certainly not practical, since it involves excessive duplication of information. What is more, the different variants are not associated as being parts of the same entity. A similar problem arises when we

[⋆] This research was partially co-funded by the European Social Fund (75%) and National Resources (25%) - Operational Program for Educational and Vocational Training (EPEAEK II) and particularly by the Research Program "PYTHAGORAS II".

want to represent the history of a changing RDF graph. In this case, we want to retain information about the past states of the graph as well as about the sequence of changes applied to this graph. Querying on and reasoning about context dependent information is also important.

To the best of our knowledge there is a limited number of papers [19, 14] on this subject. Quite recently, in [14], a temporal extension of RDF, based on the idea of assigning timestamps to RDF triples, is proposed. The same idea is employed in [19] where a technique to track changes of RDF repositories is presented.

In this paper we demonstrate how RDF can be extended so as to be suitable for expressing context-dependent information. The extension that we propose, called *Multidimensional RDF* (or *MRDF* in short), is capable of representing in a compact way multiple facets of information related to the same RDF triple. MRDF employs a set of parameters, called *dimensions*, that are used to determine specific environments, called *worlds*, under which specific pieces of information hold. Conventional RDF graphs, holding under specific worlds, called *snapshots*, can be easily extracted from MRDF graphs. As we demonstrate, MRDF is suitable for representing the history of conventional RDF graphs. Our approach is general enough to be used with more rich formalisms such as RDFS or OWL based ontologies which may also change over time.

The work presented in this paper is based on previous results on providing multidimensional extensions to OEM and XML [20, 11, 21]. The main contributions of this paper can be summarized as follows:

1. A multidimensional extension of RDF, called MRDF, is proposed and its semantics is defined in terms of the semantics of RDF.
2. The notions of *snapshots*, *canonical forms*, and *projections* of MRDF-graphs are defined and some useful properties are discussed.
3. Reification is extended so as to apply to MRDF statements.
4. It is demonstrated that MRDF graphs can be used to represent the history of (conventional) RDF graphs. Basic change operations on RDF graphs are proposed, and it is shown how their effect on RDF triples, can be represented in MRDF-graphs.
5. Extensions of the Triples Notation and the XML/RDF syntax suitable for representing MRDF graphs are proposed.

The rest of the paper is organized as follows: In Section 2, some preliminaries are given. In Section 3, Multidimensional RDF is introduced and some of its properties are discussed. In Section 4, the notions of snapshots, projections and canonical forms of MRDF graphs are introduced. In Section 5, it is illustrated how reification can be extended to MRDF statements. In Section 6, the semantics of MRDF are discussed. In Section 7, extensions of Triples Notation and XML/RDF syntax suitable for representing MRDF graphs are proposed. In Section 8, it is demonstrated how an MRDF graph with a single time dimension can be used to represent the history of conventional RDF graphs. In Section 9, it is shown how history graphs can be stored in RDBMS. In Section 10, related work is discussed. Finally, in Section 11 some hints for future work are given.

2 Preliminaries

Resource Description Framework (RDF) [1, 5] has been proposed by W3C as a language for representing information about resources in the World Wide Web. It is particularly intended for representing metadata about Web resources. RDF is based on the idea of identifying resources using Web identifiers (called *Uniform Resource Identifiers*, or URIs), and describing them in terms of simple properties and property values. This enables RDF to represent simple statements about resources as a *graph* of nodes and arcs representing the resources, with their properties and values.

2.1 RDF graph

In this subsection we give an abstract representation of the RDF model as graphs, based on some definitions borrowed from [13]. This model consists of an infinite set U (called the *RDF URI references*); an infinite set $B = \{N_j : j \in \mathcal{N}\}$ (called the *blank nodes*); and an infinite set L (called the *RDF literals*). A triple $(v_1, v_2, v_3) \in (U \cup B) \times U \times (U \cup B \cup L)$ is called an *RDF triple*. In such a triple, v_1 is called the *subject*, v_2 the *predicate* (also called *property*), and v_3 the *object*. Each triple represents a *statement* of a relationship between the things denoted by the nodes that it links.

Definition 1. *An* RDF graph *is a set of RDF triples. An* RDF-subgraph *is a subset of an RDF-graph. The* universe *of an RDF-graph G, $universe(G)$, is the set of elements of $(U \cup B \cup L)$ that occur in the triples of G. The* vocabulary *of G is the set $universe(G) \cap (U \cup L)$. An RDF-graph is* ground *if it has no blank nodes.*

Graphically, RDF graphs are represented as follows: each triple (a, b, c) is represented by $a \xrightarrow{b} c$. Note that the set of arc labels may have non-empty intersection with the set of node labels. The direction of the arc is significant: it always points toward the object.

A *map* is a function $\mu : (U \cup B \cup L) \rightarrow (U \cup B \cup L)$ such that $\mu(u) = u$ and $\mu(l) = l$ for all $u \in U$ and $l \in L$. If G is a graph then $\mu(G)$ is defined as follows: $\mu(G) = \{(\mu(s), \mu(p), \mu(o)) \mid (s, p, o) \in G\}$. A map μ is *consistent* with an RDF-graph G if $\mu(G)$ is also an RDF-graph, i.e. if $(s, p, o) \in G$ then $\mu(s) \in (U \cup B)$, and $\mu(p) \in U$. In this case we say that $\mu(G)$ is an *instance* of G. An instance $\mu(G)$ is a *proper instance* of G if $\mu(G)$ has less blank nodes that G.

Two graphs G_1, G_2 are said to be *isomorphic*, denoted by $G_1 \cong G_2$, if there are maps μ_1, μ_2 such that $\mu_1(G_1) = G_2$, and $\mu_2(G_2) = G_1$.

Let G_1, G_2 be RDF-graphs. Then, the *union* of G_1, G_2, denoted by $G_1 \cup G_2$, is the set theoretical union of their sets of triples. The *merge* of G_1, G_2, denoted by $G_1 + G_2$, is the union $G_1 \cup G_2'$, where G_2' is an isomorphic copy of G_2 whose set of blank nodes is disjoint with the set of blank nodes of G_1.

The assertion of an RDF triple says that some relationship, indicated by the predicate, holds between the things denoted by the subject and the object of the triple. The assertion of an RDF graph amounts to asserting all the triples in it, so the meaning of an RDF graph is the conjunction (logical AND) of the statements corresponding to all the triples it contains. A formal account of the meaning of RDF graphs is given in [2].

Example 1. A fragment of an RDF-graph representing information about a book is shown in Figure 1.

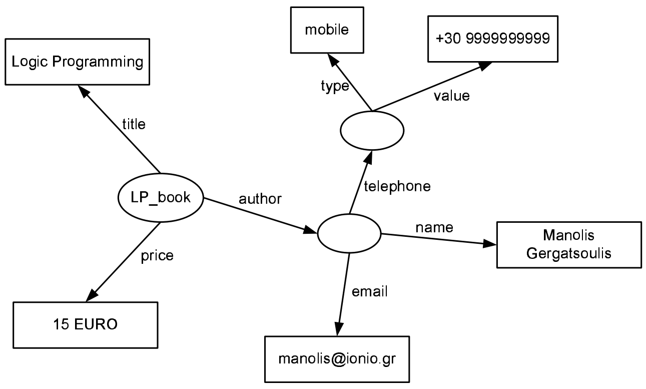

Fig. 1. An RDF-graph.

2.2 RDF triples notation

Sometimes it is convenient instead of drawing RDF graphs, to have an alternative way of writing down their statements. In the *triples notation*, each statement in the graph is written as a simple triple of the order, subject, predicate, and object.

Example 2. The statements in the RDF-graph of Figure 1 would be written in the triples notation as:

```
LP_book title      "Logic Programming"
LP_book price      "15 EURO"
LP_book author     _:abc
_:abc   email      "manolis@ionio.gr"
_:abc   name       "Manolis Gergatsoulis"
_:abc   telephone  _:def
_:def   type       "mobile"
_:def   value      "+30 9999999999"
```

2.3 RDF/XML Syntax

RDF also provides an XML-based syntax (called RDF/XML) for encoding and exchanging RDF graphs [1, 4].

Example 3. The RDF-graph of Figure 1 is written in RDF/XML as follows:

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
    <rdf:Description rdf:about="http://www.ionio.gr/LP_book">
       <title>Logic Programming</title>
       <price>15 EURO</price>
       <author rdf:nodeID="abc"/>
    </rdf:Description>
    <rdf:Description rdf:nodeID="abc"/>
       <email>manolis@ionio.gr</email>
       <name>Manolis Gergatsoulis</name>
       <telephone rdf:nodeID="def"/>
    </rdf:Description>
    <rdf:Description rdf:nodeID="def"/>
       <type>mobile</type>
       <value>+30 9999999999</value>
    </rdf:Description>
</rdf:RDF>
```

3 Adding Dimensions to RDF-graphs

In this section we introduce the notion of *Multidimensional RDF graph* (*MRDF graph* in short). In a Multidimensional RDF graph, *contexts* may be used to determine the circumstances under which RDF triples are present or not in a graph. RDF triples whose existence depends on a number of dimensions are called *multidimensional RDF triples.*

3.1 Dimensions and Worlds

The notion of *world* is fundamental in our approach (see also [11, 20]). A world represents an environment under which RDF data obtain a substance. A world is determined by assigning values to a set $\mathcal{S}$ of *dimensions.*

Definition 2. *Let $\mathcal{S}$ be a set of* dimension names *and for each $d \in \mathcal{S}$, let $\mathcal{D}_d$, with $\mathcal{D}_d \neq \emptyset$, be the* domain *of d. A* world *w is a set of pairs (d, u), where $d \in \mathcal{S}$ and $u \in \mathcal{D}_d$ such that for every dimension name in $\mathcal{S}$ there is exactly one element in w. The set of all possible worlds is denoted by $\mathcal{U}$.*

In Multidimensional RDF, a triple may have different objects under different worlds. To specify sets of worlds under which a specific node plays the role of the object of a triple we use syntactic constructs called *context specifiers.* Context specifiers qualify the so called *context arcs* which determine the resource that plays the role of the object of a specific property and the worlds under which this holds. The different resources that may be objects of the same property under different worlds are called the *facets* of that object. In the rest of this paper, by $\mathcal{W}(c)$ we denote the set of worlds specified by a context specifier c.

Two context specifies c_1 and c_2 are said to be *equivalent* if they represent the same set of worlds i.e. $\mathcal{W}(c_1) = \mathcal{W}(c_2)$. Two context specifiers c_1 and c_2 are said to be *mutually exclusive* if and only if $\mathcal{W}(c_1) \cap \mathcal{W}(c_2) = \emptyset$.

In this paper we consider the following syntax for context specifiers: A context specifier consists of one or more *context specifier clauses* separated by "|". Each context specifier clause consists of one or more *dimension specifiers* separated by comma. Thus a context specifier clause is of the form:

```
dimension_1_specifier, ..., dimension_m_specifier
```

where `dimension_i_specifier`, $1 \leq$ `i` $\leq$ `m`, is a *dimension specifier* of the form:

```
dimension_name  specifier_operator  dimension_value_expression
```

A *specifier_operator* is one of $=$, $!=$, `in`, `not in`. If the *specifier_operator* is either $=$ or $!=$, the *dimension_value_expression* consists of a single dimension value. Otherwise, if the *specifier_operator* is either `in` or `not in`, the *dimension value expression* is a set of values of the form $\{\mathtt{value_1}, \ldots, \mathtt{value_k}\}$. For linear and discrete domains we will also use the notation $\{a..b\}$ for dimension value expressions to denote the set of all values x such that $a \leq x \leq b$.

Example 4. The following are context specifiers:

```
1) [time=7:45]
2) [lang=Greek, detail in {low,medium} | lang=French, detail=high]
3) [currency in {EURO,USD}, customer_type = student]
4) [season != summer, weekday not in {Saturday,Sunday}]
```

Notice that the set of worlds represented by the second context specifier is $\{(\mathtt{lang} = \mathtt{greek}, \mathtt{detail} = \mathtt{low}), (\mathtt{lang} = \mathtt{greek}, \mathtt{detail} = \mathtt{medium}),$ $(\mathtt{lang} = \mathtt{French}, \mathtt{detail} = \mathtt{high})\}$, while the set of worlds represented by the third context specifier is: $\{(\mathtt{currency} = \mathtt{EURO}, \mathtt{customer\_type} = \mathtt{student}),$ $(\mathtt{currency} = \mathtt{USD}, \mathtt{customer\_type} = \mathtt{student})\}$. Concerning the fourth context specifier we have to take into account the domains of the dimensions `season` and `weekday` in order to find the set of worlds that it represents.

Context specifiers impose constraints that restrict the set of worlds under which an entity holds. In this sense, if a context specifier does not contain a dimension specifier for a specific dimension then no constraint is imposed on the values of this dimension. Thus, the context specifier [ ] represents the set of all possible worlds $\mathcal{U}$.

3.2 Multidimensional RDF-graphs

In Multidimensional RDF-graphs we will use the following sets of symbols: a set U of RDF URI references, a set B of blank nodes, a set L of RDF literals, a set M of *multidimensional nodes*, and a set C of context specifiers.

A triple $(v_1, v_2, v_3) \in (U \cup B) \times U \times (M \cup U \cup B \cup L)$ is called *statement triple*, while a triple $(v_1, c, v_3) \in M \times C \times (U \cup B \cup L)$ is called a *context triple.*

Definition 3. *A* Multidimensional RDF-graph *(*MRDF-graph*) is a set $\mathcal{S}_t \cup \mathcal{C}_t$, where $\mathcal{S}_t$ is a set of statement triples and $\mathcal{C}_t$ is a set of context triples, such that:*

1. *For every context triple $(v_3, c, v_4) \in \mathcal{C}_t$ there exist a node $v_1 \in (U \cup B)$ and a node $v_2 \in U$ such that $(v_1, v_2, v_3) \in \mathcal{S}_t$.*
2. *For every statement triple $(v_1, v_2, v_3) \in \mathcal{S}_t$ for which v_3 is a multidimensional node, there exist a context specifier $c \in C$ and a node $v_4 \in (U \cup B \cup L)$ such that $(v_3, c, v_4) \in \mathcal{C}_t$.*

Example 5. A Multidimensional RDF-graph is shown in Figure 2. This graph is

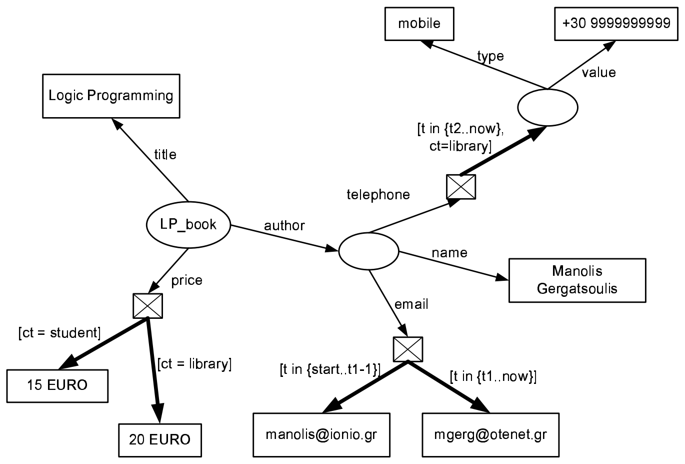

Fig. 2. A Multidimensional RDF-graph.

a variant of the RDF graph in Figure 1. Some parts of this graph are context-dependent. In particular, we have two dimensions; the dimension `time` (abbreviated as `t`) and the dimension `customer_type` (abbreviated as `ct`). The value of the property `price` of the book is `15 EURO` for all worlds in which the value of the dimension `ct` is `student` while the value of the same property is `20 EURO` for all worlds in which the value `ct` is `library`. The value of the property `email` is time-dependent. This value is `manolis@ionio.gr` for the time points in the interval {`start..t1-1`}, where the reserved words `start` represents the beginning of time, while the value of the property is `mgerg@otenet.gr` for the time points in the interval {`t1..now`}, where `now` represents the current time. Finally, notice that the property `telephone` has meaning only under the worlds in which the value of the dimension `ct` is `library` and the value of the dimension `t` is in the interval {`t2..now`}. This means that from the time point `t2` the mobile telephone of the author is available only for customers that are libraries.

Notice that in an MRDF-graph statement triples are represented by thin lines while context triples are represented by thick lines.

An MRDF graph G is said to be *deterministic*, if for every multidimensional node m in G, the context specifiers of all context triples departing from m are mutually exclusive each other. Although non-deterministic MRDF-graphs may have interesting properties, in this paper we focuss mainly on deterministic MRDF-graphs.

4 Snapshots, Projections and Canonical forms of MRDF graphs

4.1 RDF snapshots of MRDF-graphs

A Multidimensional RDF-graph G can be seen as a compact representation of a set of conventional RDF-graphs called the *snapshots* of G:

Definition 4. *Let $G = (\mathcal{S}_t \cup \mathcal{C}_t)$ be an MRDF-graph and w be a world. We define the* snapshot of G under w*, denoted by $Snap(G, w)$, as follows: $Snap(G, w) = \{(r1, p, r2) \mid r2 \in U \cup B \cup L$ and $(r1, p, r2) \in \mathcal{S}_t\} \cup \{(r1, p, r2) \mid \exists m \in M, \exists c \in C$ such that $(r1, p, m) \in \mathcal{S}_t$ and $(m, c, r2) \in \mathcal{C}_t$ and $w \in \mathcal{W}(c)\}$.*

Notice that to each multidimensional node of a deterministic MRDF-graph corresponds at most one RDF triple at each world. In non-deterministic MRDF-graphs, multiple RDF triples may correspond to every multidimensional node.

According to the above definition, a multidimensional RDF-graph G can be seen as a compact representation of the set $Snap(G,\mathcal{U}) = \{Snap(G, w) \mid w \in \mathcal{U}\}$, where $\mathcal{U}$ is the set of all possible worlds, that is, $Snap(G,\mathcal{U})$ represents the set of all (conventional) RDF-graphs each of them holding under a specific world.

Example 6. Consider the world $w = \{$`t = t2`$^+$`, ct = library`$\}$, where `t2`$^+$ is a time point such that `t1 < t2 < t2`$^+$. Then the snapshot of the MRDF-graph in Figure 2 under the world w, is the RDF-graph shown in Figure 3.

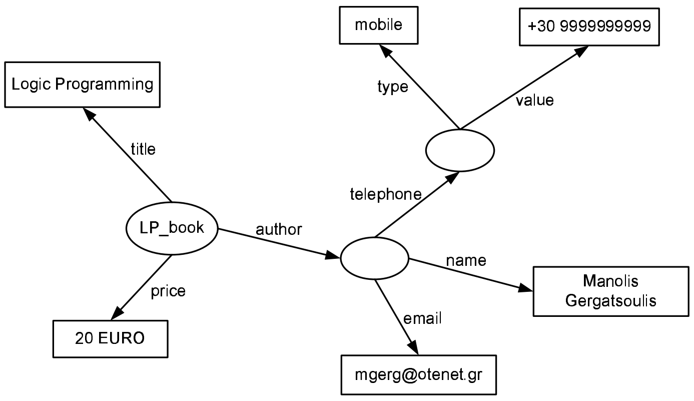

Fig. 3. A snapshot of the Multidimensional RDF-graph in Figure 5.

Based on the notion of snapshots we define equivalence of MRDF-graphs:

Definition 5. *Let G_1 and G_2 be MRDF graphs. We say that G_1* is equivalent *with G_2 if and only if for every world w, $Snap(G_1, w) \cong Snap(G_2, w)$.*

4.2 Projections of MRDF graphs

Another useful operation on MRDF-graphs is *projection* with respect to a set of dimensions. Let S be a set of dimensions and c a context specifier. Then the *projection of a context specifier* c with respect to S is a context specifier c' obtained from c by eliminating all dimension specifiers of c whose dimension name does not belong to S.

Definition 6. *Let* $G = (\mathcal{S}_t \cup \mathcal{C}_t)$ *be a multidimensional RDF-graph and* S *be a set of dimensions. Then the* projection of G with respect to S, *denoted by* $Proj(G, S)$, *is defined as follows:*

$$Proj(G, S) = \mathcal{S}_t \cup \{(m, c', r2) \mid (m, c, r2) \in \mathcal{C}_t \text{ and } c' \text{ is the projection of } c \text{ with respect to } S\}.$$

Projecting an MRDF-graph means that we remove all constraints concerning the values of the dimensions not belonging to S. Notice that the projection of a deterministic MRDF-graph G, may be a non-deterministic MRDF-graph.

4.3 Canonical form of MRDF graphs

In this section we define the notion of *canonical form* of an MRDF graph. In a canonical MRDF graph all statement arcs point to multidimensional nodes.

Definition 7. *An MRDF-graph* $G = (\mathcal{S}_t \cup \mathcal{C}_t)$ *is said to be in* canonical form *if for every statement triple* $(v_1, v_2, v_3) \in \mathcal{S}_t$, v_3 *is a multidimensional node.*

Definition 8. *Let* $G = (\mathcal{S}_t \cup \mathcal{C}_t)$ *be an MRDF-graph. The* canonical representation $Can(G)$ *of* G *is an MRDF-graph obtained from* G *by replacing each statement triple* $(v_1, v_2, v_3) \in \mathcal{S}_t$ *for which* $v_3 \notin M$, *by a pair of a statement triple and a context triple of the form* (v_1, v_2, m) *and* $(m, [\,], v_3)$ *respectively, where* m *is a fresh multidimensional node in* M.

This construction of the canonical representation of an MRDF-graph according to the above definition is shown in Figure 4. Recall that the context specifier [ ] represents the set of all possible worlds $\mathcal{U}$.

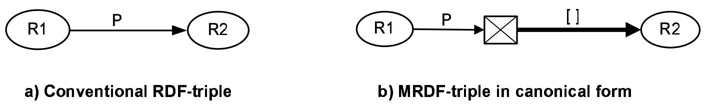

Fig. 4. Transforming an MRDF-graph in canonical form.

Example 7. The canonical representation of the graph in Figure 2 is shown in Figure 5. This graph is obtained by replacing each statement triple in the graph of Figure 2 leading to a non-multidimensional node, by a pair of a statement triple followed by a context triple as described above (and is depicted in Figure 4).

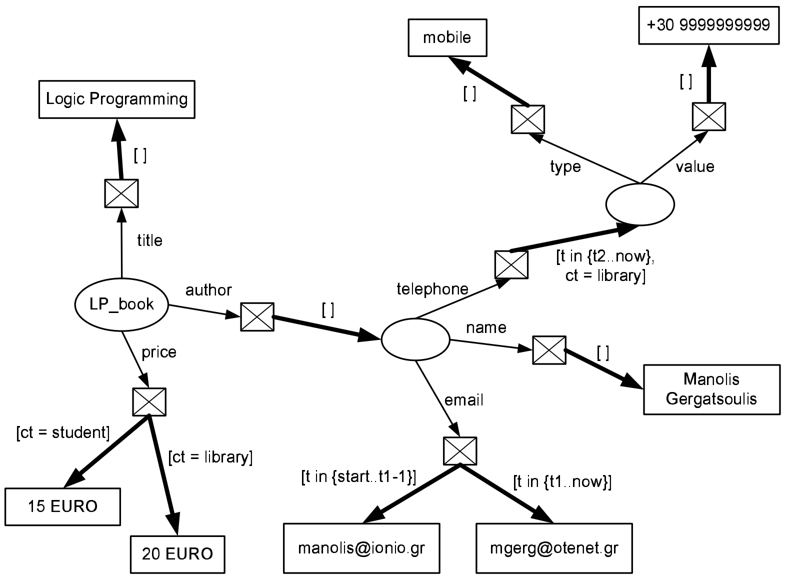

Fig. 5. The canonical representation of the MRDF-graph in Figure 2.

The following lemma demonstrates that an MRDF-graph and its canonical representation are equivalent graphs.

Lemma 1. *Let G be an MRDF-graph and $Can(G)$ be its canonical representation. Then G and $Can(G)$ are equivalent.*

Proof. It is easy to prove that for every world w, $Snap(G, w) \cong Snap(Can(G), w)$.

5 Reification in Multidimensional RDF-graphs

RDF applications often need to describe other RDF statements or, in general, to express statements about other statements. It is useful, for instance, to record in RDF, information about when statements were made, or who made them. RDF embodies [1] a built-in vocabulary intended for describing RDF statements. A statement description using this vocabulary is called a *reification* of the statement. The RDF reification vocabulary consists of the type `rdf:Statement`, and the properties `rdf:subject`, `rdf:predicate`, and `rdf:object`. The use of this vocabulary to describe RDF statements is shown in Figure 6 where the triple $(R1, P, R2)$ shown in Figure 6(a) is represented by the RDF graph in Figure 6(b). This graph says that the anonymous resource is an RDF statement, the subject of the statement is $R1$, its predicate is P, and its object is $R2$.

For MRDF an extension of the RDF reification mechanism is needed. Such an extension is shown in Figure 7. Since in MRDF we may have multiple facets of an object of a triple, corresponding to different sets of worlds, the property

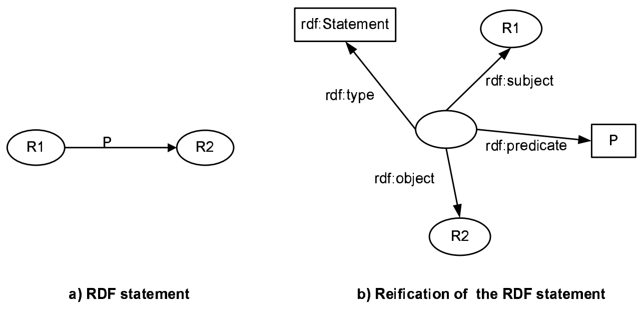

Fig. 6. Reification in RDF.

`rdf:object` is now represented by a statement triple whose arc points to a multidimensional node. From this multidimensional node depart different context triples each of them leading to a different facet of the statement's object. Notice also that we now use a type `mrdf:Statement` which states that the reified statement is not a conventional but a multidimensional one.

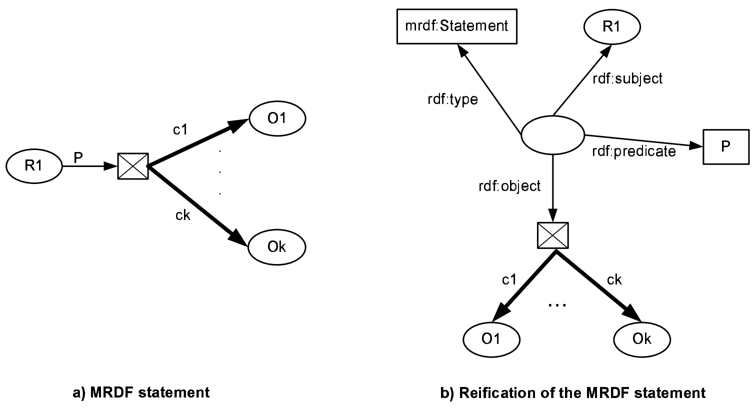

Fig. 7. Reification in Multidimensional RDF.

It is important to note that the definition of snapshots has to be slightly modified so as to take into account the presence of reification. However, we will not discuss it further here due to space limits.

6 Semantics of MRDF

In this section we define the semantics of MRDF in terms of the semantics of RDF.

Definition 9. *Let F be an MRDF-graph, G be an RDF-graph, w be a world, and W be a set of worlds. We say that F* entails G in the world w*, denoted by $F \models_w G$, if and only if $Snap(F, w) \models G$. We say that F* entails G in a set of worlds W*, denoted by $F \models_W G$, if and only if for every world $w \in W$, $Snap(F, w) \models G$.*

The following lemma can be easily proved.

Lemma 2. *Let F be an MRDF-graph and $Can(F)$ be its canonical form. Then for every RDF graph G and every world w, $F \models_w G$ if and only if $Can(F) \models_w G$.*

7 Triples Notation and RDF/XML syntax for MRDF

Syntactic constructs suitable for representing MRDF-graphs are presented in this section.

7.1 Extended Triple Notation

In order to express MRDF graphs in triples notation it is necessary to extent this notation in order to be capable of representing context triples. The extension that we propose retains the basic structure of the triples but allows their third component to be a list, called *object list*. Object list is used when the object of a statement triple is a multidimensional node. In this case the third component contains pairs of values. The first value of each pair is a context specifier while the second value is the object (resource/literal) corresponding to that specifier.

Example 8. The representation in the extended triple notation of the MRDF graph in Figure 2 is as follows:

```
LP_book title "Logic Programming"
LP_book price [([ct = student],"15 EURO"), ([ct = library],"20 EURO")]
LP_book author _:abc
_:abc email [([t in {start..t1-1})],"manolis@ionio.gr"),
             ([t in {t1..now}],"mgerg@otenet.gr")]
_:abc name "Manolis Gergatsoulis"
_:abc telephone [([t in {t2..now},ct = library],_:def)]
_:def type "mobile"
_:def value "+30 9999999999"
```

7.2 RDF/XML syntax for MRDF

In order to express MRDF in RDF/XML syntax we need an extension of XML capable of expressing contexts. Such an extension of XML is *Multidimensional XML* (or MXML), which has been proposed in [11]. In Example 9 below we illustrate how MRDF can be expressed in MXML/RDF syntax.

Example 9. The following RDF/MXML document represents the MRDF graph in Figure 2:

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
    <rdf:Description rdf:about="http://www.ionio.gr/LP_book">
       <title>Logic Programming</title>
       <@price>
         [ct=student]<price>15 EURO</price>[/]
         [ct=library]<price>20 EURO</price>[/]
       </@price>
       <author rdf:nodeID="abc"/>
    </rdf:Description>
    <rdf:Description rdf:nodeID="abc"/>
       <@email>
         [t in {start..t1-1}]<email>manolis@ionio.gr</email>[/]
         [t in {t1..now}]<email>mgerg@otenet.gr</email>[/]
       </@email>
       <name>Manolis Gergatsoulis</name>
       <@telephone>
         [t in {t2..now},ct=library]<telephone rdf:nodeID="def"/>[/]
       </@telephone>
    </rdf:Description>
    <rdf:Description rdf:nodeID="def"/>
       <type>mobile</type>
       <value>+30 9999999999</value>
    </rdf:Description>
</rdf:RDF>
```

It should be noted that instead of using MXML we can use conventional (but more verbose) XML syntax for representing MRDF-graphs (see [10]).

8 Representing changes in RDF graphs using MRDF

As described previously, MRDF is a general formalism and powerful enough to represent context-dependent RDF data that may occur in real world RDF applications. In this section we demonstrate that MRDF graphs can be used as a formalism for tracking the history of conventional RDF-graphs. We assume the following scenario: the user manipulates an RDF graph and applies changes to it, at specific time points. The changes are described through specific primitives called *basic change operations*. In order to keep track of the sequence of changes and in particular of the sequence of the (conventional) RDF graphs obtained by applying these changes, the system keeps a Multidimensional RDF graph, called the *History Graph*, which encodes all these changes applied by the user to the conventional RDF graph. The History Graph employs a single dimension `t` representing time. For this dimension, we assume a time domain T which is linear and discrete. We also assume a reserved value `start`, such that `start` $<$ `t` for every `t` $\in T$, representing the *beginning of time*, and a reserved value `now`, such that `t` $<$ `now` for every `t` $\in T$, representing the *current time*. Notice that, in our scenario, the user is only capable to apply change operations on a conventional RDF graph being unaware of the History Graph that lies beneath.

8.1 Basic Change Operations on RDF triples

We consider three primitive change operations on RDF graphs, namely *update*, *delete*, and *insert*, and demonstrate how their effect on RDF-graph can be mapped into changes on the underlying History Graph:
a) **Update:** Update operation can be used to change the value of a property (i.e. the object of a triple). Updating a triple can be seen (at the level of the RDF graph being updated) as the replacement of the triple with another triple which has the same subject and predicate but different object. The way that update operation affects the underlying History Graph is depicted in Figure 8(a). The value of the property P in the triple on the left part of the figure is updated at time `t1` from `R2` to the new value `R3`. The MRDF representation of this operation is shown on the right side of the figure. The multidimensional node has now two facets. The first one is valid in all time points of the interval {`start..t1-1`}, while the second is valid for all time points in the interval {`t1..now`}.

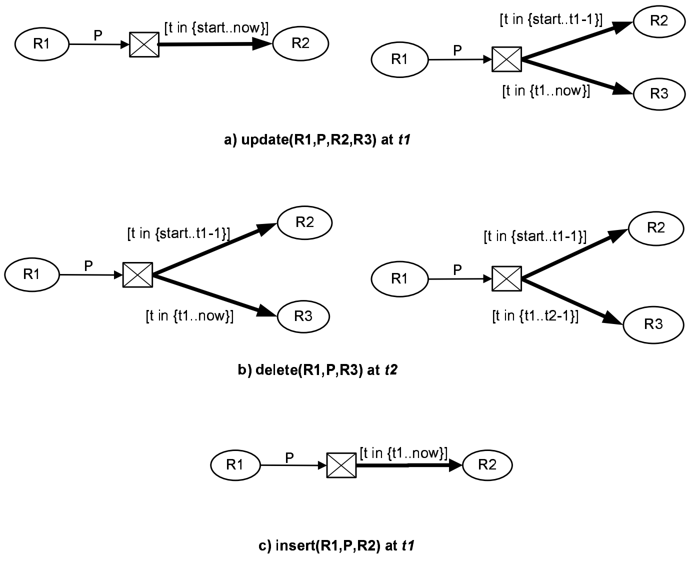

Fig. 8. Representation of the basic change operations in the History Graph.

Note that a subsequent update of the same statement at a time point `t2` will be represented in the History Graph as follows: a) by simply adding a new context triple departing from the same multidimensional node and holding in the interval {`t2..now`} and b) by changing the value of the time dimension `t` of the most recent context triple from {`t1..now`} to {`t1..t2-1`}. Note also that the resource $R3$ may be a new resource which is introduced by the update operation or it may be a resource that already exists in the graph. In both cases the update operation results in adding a context arc from the multidimensional node to $R3$.

b) **Delete:** The deletion of a triple $(R1, P, R3)$ from the RDF graph at time `t2`, is represented in the History Graph by simply changing the end time point of the most recent interval from `now` to `t2-1`, as shown in Figure 8(b). Note that, if the deleted triple is a conventional triple in the History Graph, then deletion is modeled by first obtaining the canonical form of the triple and then applying the process described above to the caninical triple as described above.
c) **Insert:** As depicted in Figure 8(c), the new triple $(R1, P, R2)$ inserted at the time point `t1`, is modeled in the History Graph by adding a new statement triple followed by a single context triple holding during the interval {`t1..now`}.

Notice that the triple being inserted on the RDF graph may refer either to resources (subject and/or object) that already exist in the RDF-graph or to new resources that are added by the insert operation.

9 Storing the History Graphs in RDBMS

A simple relational database schema can be easily designed for storing the History Graphs using an RDBMS. For this we assume that the graph is in its canonical form. Such a graph can be stored in a database which has two relations, namely `statement` and `context` to store the statement and the context triples respectively. The schema of these relations is as follows:
`statement(Subject,Predicate,MultidimensionalNode)`: where `Subject` is the subject, `Predicate` is the predicate and `MultidimensionalNode` is the multidimensional node identifier.
`context(MultidimensionalNode,Object,S,E)`: where `MultidimensionalNode` is the multidimensional node identifier from which the arc departs, `Object` is the object, `S` is the start time point and `E` is the end time point of the time interval.

Notice that this schema is appropriate only for MRDF which has only one dimension which takes as values time intervals.

Concerning the RDF/XML syntax it is important to note that in the case of the History Graph, where we have only a single time dimension, we could use (instead of MXML), a temporal extension of XML. Such a temporal extension of XML can be found in [22, 23], where two extra attributes are added to XML elements, namely `vstart` and `vend`, representing the end points of the time interval in which this elements version is valid.

10 Related Work

The Multidimensional extension to RDF proposed in this paper is based on similar ideas to that on which Multidimensional OEM [20] and the Multidimensional XML [11] are based. However, to the best of our knowledge, there is no other research work in the direction of incorporating dimensions in RDF. Quite recently [14], Temporal RDF, a transaction time extension of RDF has been proposed. However, our approach is more general than Temporal RDF which may be considered as a special case of Multidimensional RDF, since we allow multiple dimensions (and even multiple time dimensions).

To the best of our knowledge, only a few papers refer to the problem of representing changes in RDF databases. One approach to this problem has been proposed in [19]. Their model is based on the admission that an RDF statement is the smallest manageable piece of knowledge in an RDF repository. They propose only two basic operations, addition and removal, since they argue that an RDF statement cannot be changed, it can only be added or removed. In their approach, they used versions as the labeled states of the repository. However, our approach is more general and flexible than the approach in [19] as besides addition and deletion we also introduce an update operation. In this way different resources that are in fact different versions of an object of a property are grouped together and its relationship is recorded. Besides, our representation formalism is more general as it allows multiple dimensions (which might be multiple times such as *valid time* or *transaction time*). Consequently, in our approach one can encode multiple versioning information not only with respect to time but also to other context parameters such as language, degree of detail, geographic region etc.

Some related research to the problem of RDF versioning has also been done in the field of ontology versioning. In [16, 15], Ontoview, a web-based management system for evolving ontologies in the Web, is used. Ontoview has the ability to compare ontologies at a structural level. It finds changes in ontologies and it visualizes them, helping the user to specify the conceptual implication of the differences. In [17] a component-based framework for ontology evolution is proposed. This framework integrates a description of different representations about change information. Thus, they present an ontology of change operations, which is the kernel of their framework. The problem of managing (storing, retrieving and querying) multiple versions of XML documents is also examined in [8, 9]. Recently, an approach of representing XML document versions was proposed [22, 23]. The basic idea is to add two extra attributes, namely `vstart` and `vend`, that represent the time interval in which the corresponding version of the element is valid. Temporal extensions of XML have also been proposed in [6, 12].

The problem of representing and querying changes in semistructured data has also been studied in [7], where *Delta OEM* (DOEM in short), a graph model that extends OEM with annotations containing temporal information, was proposed. Four basic change operations on OEM graphs, namely $creNode$, $updNode$, $addArc$, and $remArc$ are considered. Those operations are mapped to annotations, which are tags attached to nodes or edges, containing information that encodes the history of changes for these nodes or edges. Recently [21], Multidimensional OEM [20], has been proposed as a formalism for representing the history of time-evolving semistructured data. Finally, in [10], the authors propose the use of Multidimensional XML [11] for the representation of the history of XML documents.

11 Discussion and Future Work

Investigation of other real application domains to demonstrate usefulness of multidimensional RDF is between our plans for future work. An attempt to explore such applications is described in [18], where the problem of representing and manipulating time-dependent information in collection-level cultural metadata is investigated. In that paper, MRDF employing two independent time dimensions is used as a formalism to enriches a metadata application profile for the collection-level description of cultural collections, with the ability of time representation and manipulation.

Investigation of query languages and inference systems for MRDF repositories when RDFS vocabulary is used [3], are important problems for future work. The study of the semantics of non-deterministic MRDF-graphs and their applications are also interesting problems.

References

1. RDF Primer (W3C Recommendation, 10 February 2004). http://www.w3.org/TR/2004/REC-rdf-primer-20040210/, 2004.
2. RDF Semantics (W3C Recommendation, 10 February 2004). http://www.w3.org/TR/2004/REC-rdf-mt-20040210/, 2004.
3. RDF Vocabulary Description Language 1.0: RDF Schema (W3C Recommendation, 10 February 2004). http://www.w3.org/TR/2004/REC-rdf-schema-20040210, 2004.
4. RDF/XML Syntax Specification (Revised), (W3C Recommendation 10 February 2004). http://www.w3.org/TR/2004/REC-rdf-syntax-grammar-20040210/, 2004.
5. Resource Description Framework (RDF): Concepts and Abstract Syntax (W3C Recommendation, 10 February 2004). http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/, 2004.
6. T. Amagasa, M. Yoshikawa, and S. Uemura. A Data Model for Temporal XML Documents. In M. T. Ibrahim, J. Kung, and N. Revell, editors, *Database and Expert Systems Applications, 11th International Conference (DEXA 2000), London, UK, Sept. 4-8, Proceedings*, Lecture Notes in Computer Science (LNCS) 1873, pages 334–344. Springer-Verlag, Sept. 2000.
7. S. S. Chawathe, S. Abiteboul, and J. Widom. Managing Historical Semistructured Data. *Theory and Practice of Object Systems*, 24(4):1–20, 1999.
8. S.-Y. Chien, V. Tsotras, and C. Zaniolo. Efficient Schemes for Managing Multiversion XML Documents. *The VLDB Journal*, 11(4):332–353, 2002.
9. S.-Y. Chien, V. J. Tsotras, C. Zaniolo, and D. Zhang. Efficient Complex Query Support for Multiversion XML Documents. In *Advances in Database Technology - EDBT 2002, Proceedings of the 8th Conference on Extending Database Technology*, Lecture Notes in Computer Science (LNCS) Vol 2287, pages 161–178. Springer-Verlag, 2002.
10. M. Gergatsoulis and Y. Stavrakas. Representing Changes in XML Documents using Dimensions. In Z. Bellahsene, A. B. Chaudhri, E. Rahm, M. Rys, and R. Unland, editors, Database and XML Technologies, 1st International XML Database Symposium, XSym' 03, Berlin, Germany, September 2003, Proceedings, Lecture Notes in Computer Science (LNCS), Vol. 2824, pages 208–222. Springer-Verlag, 2003.
11. M. Gergatsoulis, Y. Stavrakas, and D. Karteris. Incorporating Dimensions to XML and DTD. In H. C. Mayr, J. Lanzanski, G. Quirchmayr, and P. Vogel, editors, Database and Expert Systems Applications (DEXA' 01), Munich, Germany, September 2001, Proceedings, Lecture Notes in Computer Science (LNCS), Vol. 2113, pages 646–656. Springer-Verlag, 2001.

12. F. Grandi and F. Mandreoli. The Valid Web: an XML/XSL Infrastructure for Temporal Management of Web Documents. In T. M. Yakhno, editor, *Advances in Information Systems. First International Conference (ADVIS'02), Izmir, Turkey, 25-27 October*, pages 294–303, 2000.
13. C. Gutiérrez, C. Hurtado, and A. O. Mendelzon. Foundations of Semantic Web Databases. In *Proceedings of the 23rd ACM SIGACT-SIGMOD-SIGART Symposium on Principles of DataBase Systems*, pages 95–106. ACM Press, 2004.
14. C. Gutiérrez, C. Hurtado, and A. Vaisman. Temporal RDF. In Asunción Gómez-Pérez and Jérôme Euzenat, editors, *The Semantic Web: Research and Applications, Second European Semantic Web Conference, ESWC 2005, Heraklion, Crete, Greece, May 29 - June 1, 2005, Proceedings,*, volume 3532 of *Lecture Notes in Computer Science*, pages 93–107. Springer, 2005.
15. M. Klein, D. Fensel, A. Kiryakov, and D. Ognyanov. Finding and characterizing changes in ontologies. In *Proceedings of the 21st International Conference on Conceptual Modelling (ER'02)*, volume 2503 of *Lecture Notes in Computer Science*, pages 79–89. Springer-Verlag, Heidelberg, 2002.
16. M. Klein, D. Fensel, A. Kiryakov, and D. Ognyanov. Ontology versioning and change detection on the web. In *Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web (EKAW'02)*, volume 2473 of *Lecture Notes in Computer Science*, pages 197–212. Springer-Verlag, Heidelberg, 2002.
17. M. Klein and N.F. Noy. A component-based framework for ontology evolution. Technical Report IR-504, Department of Computer Science, Vrije Universiteit, Amsterdam, March 2003.
18. P. Lilis, E. Lourdi, C. Papatheodorou, and M. Gergatsoulis. A metadata model for representing time-dependent information in cultural collections. Submitted for publication, 2005.
19. D. Ognyanov and A. Kiryakov. Tracking Changes in RDF(S) Repositories. In A. Gomez-Perez and V. Richard Benjamins, editors, *Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web, 13th International Conference, EKAW 2002, Siguenza, Spain, October 1-4, 2002, Proceedings.* Lecture Notes in Computer Science (LNCS) 2473, pages 373–378. Springer-Verlag, 2002.
20. Y. Stavrakas and M. Gergatsoulis. Multidimensional Semistructured Data: Representing Context-Dependent Information on the Web. In A. B. Pidduck, J. Mylopoulos, C. Woo, and T. Oszu, editors, *Advanced Information Systems Engineering, 14th International Conference (CAiSE'02), Toronto, Ontario, Canada, May 2002. Proceedings.*, Lecture Notes in Computer Science (LNCS), Vol. 2348, pages 183–199. Springer-Verlag, 2002.
21. Y. Stavrakas, M. Gergatsoulis, C. Doulkeridis, and V. Zafeiris. Representing and Querying Histories of Semistructured Databases Using Multidimensional OEM. *Information Systems*, 29(6):461–482, 2004.
22. F. Wang and C. Zaniolo. Temporal Queries in XML Document Archives and Web Warehouses. In *Peoceedings of the 10th International Symposium on Temporal Representation and Reasoning / 4th International Conference on Temporal Logic (TIME-ICTL 2003), 8-10 July 2003, Cairns, Queensland, Australia*, pages 47–55. IEEE Computer Society, 2003.
23. Fusheng Wang. *Efficient Support for Queries and Revisions in XML Document Archives.* PhD thesis, Computer Science Department, University of California, Los Angeles (UCLA), August 2004.