Empirically Evaluating Three Proposals for Representing Changes in OWL2

J.-R. Bourguet a, G. Guizzardi a,b A. Botti Benevides a and V. Zamborlini c

a Ontology & Conceptual Modeling Research Group (NEMO)
Federal University of Espírito Santo (UFES), Vitória – Brazil
b Research Centre for Knowledge and Data (KRDB)
Free University of Bozen-Bolzano (UNIBZ), Bozen-Bolzano – Italy
c Institute for Logic, Language and Computation (ILLC)
Universiteit van Amsterdam (UvA), Amsterdam – The Netherlands

Abstract. In almost all domains in practice, it is fundamental to properly represent entities amenable to changes. For instance, in business analytics, we must be able to reason with large amounts of time-changing KPI (key performance indicators) data. For this reason, general-purpose practical knowledge representation frameworks must be able to support the representation of temporally changing information and in a way that affords decidable automated reasoning. In this paper, we address the issue of representing entities amenable to intrinsic or extrinsic changes in OWL2. These sources of change are illustrated in a simplified model of the scholar domain. We then propose three strategies to represent entities amenable to changes as well as their changes. In particular, we do that by employing strategies that are based on a philosophical stance called perdurantism, which sees all individuals as 4D entities, i.e., as individuals that unfold in time as well as in space. Finally, we compare these three alternatives by generating synthetic instances and performing an empirical evaluation of reasoning tasks.

Keywords. Perdurantist representation, Temporally changing information, Formal Ontology, OWL, Empirical Assessment of Ontology Codification Alternatives

1. Introduction

It is of crucial importance for the Knowledge Representation community to provide means for the modeler to explicitly represent information in a declarative form that is suitable for performing reasoning tasks. For instance, there are attempts to apply time changing information-based models in a range of domains, including enterprise contracts [1], environmental data integrations [2], stock analysis (buy-hold-sell) [3] or court proceedings, as litigations [4]. A classical problem is the trade-off between expressivity and time/space computational complexity of reasoning tasks. The OWL2 ontology representation language [5] is underpinned by restricted DL fragments, the majority of which were designed to preserve decidability and provide tractability. Nevertheless, OWL2 has been designed focusing on the representation of scenarios with immutable truth-values and unchangeable information about the world. Different approaches to cope with this issue have been propose in the literature. These include concrete domains, refi-
cations, annotations, versioning, named graphs and perdurantism-based representations (see [6] and [7] for overviews). The perdurantist view claims that all entities have temporal parts and can be intuitively represented as four dimensional space-time worms whose temporal parts are time slices of the worm. This representation is relevant in business analytics for example where it is frequent to reason with large amounts of historical time-changing data. A series of works adopt an OWL-based perdurantist representation of the world. Welty and Fikes [8] introduced the idea of 4D fluents that provide temporal parts to each instance (extended toward ND fluents in [9]). Krieger et al [10,11] proposed a total perdurantist view by introducing the class of time slices as a superclass of both contingent and mandatory classes. Finally in [12,13], the authors present a proposal in relation with the ways properties and relations can evolve in time (e.g., (im)mutability). In this paper, we propose three alternatives based on the approach presented in [12,13] and perform an empirical evaluation to compare them. Our evaluation is based on a model representing the scholar domain, which illustrates some mutable aspects. Thus, we formalized this model in the three alternatives and generated synthetic instances in order to perform an empirical evaluation of some reasoning tasks. The remainder of this paper is structured as follows. In Section 2, we introduce the notions of mandatoriness, contingency, (im)mutability, and dependence, illustrated in a purely illustrative UML-like diagram of the scholar domain. Section 3 describes our three proposals for mapping UML-like diagram to TBoxes in OWL2. In Section 4, we present the results of our empirical evaluation of the aforementioned alternatives. Section 5 presents some related works. Finally, Section 6 presents some final considerations.

2. Modeling dynamic phenomena

Most Description Logics (DLs) (except the temporal DLs [14] for example) have been designed to represent immutable and unchangeable information capturing snapshots of the world. However, some applications require modelers to represent and reason over different kinds of information; for instance, decision support systems like business analytics usually require keeping track of historical changes in order to compute crucial indicators. We illustrate some kinds of mutable information by means of a model of scholar activities represented in Figure 1(a). In this domain, Authors (which are Persons) write Publications, which can be classified into Papers, Articles, Chapter or Books. A Publication can be cited by another Publication, and Authors have also an hindex. Since 2012, orcid (Open researcher and contributor id) can provide a persistent digital identifier to academic Researchers. orcid was created as a response to several problems: authors may be called by different names through time (e.g. a marriage can append a name for an author); cultural differences can exist in naming people, in the ordering of names and surnames; and names can be written in different alphabets. Going back to our example, a Researcher can be remunerated by Scholarships provided by one Organisation in such a way that an Organisation can provide several Scholarships, and a Scholarship remunerates one Researcher. Authors and Organisations can be associated. We specify two kinds of Organisations: a Team and a University. A Team is part of one or more Universities, and a University can be ranked with regard to arwu (academic ranking of world universities), also known as Shanghai ranking, an an-


Annual publication of University rank. This model includes time changing and obsolete information that should be properly represented. For example, once a name form for a Person is used in the head of a Publication, this occurrence will always refer to the specific Person; this situation is different for an email address, which can be replaced or removed. Moreover, indicators such as hindex and arwu are typically volatile information and also seems pertinent to keep track of their historical changes to support reasoning on issues such as causality or correlation. Also, hindex is an attribute that is always available (0 by default and monotonic afterwards), while arwu is non mandatory for a university (there can be universities not ranked by this metric), and a University can change its name or place. Finally, an association between an Author and an Organisation can also cease to hold. One can notice three sources of changes in Figure 1(a): attributes, relations and class instantiations.

Concerning attributes and relations, we highlight two characteristics: (i) **mandatory-ness vs. contingency**, and (ii) **mutability vs. immutability**. (i) is usually represented in a class diagram by means of cardinality constraints, where a cardinality greater or equal to one ensures that the attribute or relation is mandatory, otherwise it is contingent (optional). For example, if an Author exists, her name and hindex are mandatory, while her orcid is optional. Also, it is mandatory for a Scholarship to remunerate one Researcher, while it is contingent for a Researcher to be remunerated by a Scholarship. (ii) can be represented in a class diagram by placing or not \{readOnly\} close to the corresponding immutable attribute or association end. If an attribute is immutable, once the value of the attribute is set, it cannot change; and given a relation $R$ from a class $A$ to a class $B$ and s.t. the association end near $B$ is tagged as immutable, once an instance $x$ of $A$ starts to relate to $y_1, \ldots, y_n$ via $R$, then $x$ cannot start an $R$-relationship with any other $y_i$, and no $xRy_1, \ldots, xRy_n$ can cease to hold until $x$ ceases to exist. For example, if an Author exists, orcid is immutable, while hindex is mutable. Similarly, a Scholarship cannot change its isRemuneratedBy relation from a Researcher to another (it must always remunerate the same Researcher), while a Researcher can cease to be remunerated by a Scholarship without ceasing himself to exist.

One can ground such notions on Formal Ontology (see for example [15]). A **generic dependence** holds between an individual $x$ via relation $R$ to a type $T$ when, in order to exist, an individual $x$ has to be $R$-related with an instance of $T$. A **specific depen-
A relationship \( xRy \) holds, if \( x \) and \( y \) must exist, specific dependence is entailed by means of attributes/relations that are tagged as \{readOnly\} and having a cardinality greater or equal to one, i.e., mandatory immutable attributes/relations. Moreover, some class instantiations must always hold for its individuals, while others do not necessarily have to hold. For example, an instance of Person cannot cease to instantiate it without ceasing to exist, while a Researcher can cease to instantiate Researcher without ceasing to exist. This property of the class Person is called rigidity [15, Ch. 4]. The property of the class Researcher is called anti-rigidity\(^1\) as it requires an entity that instantiates Researcher at a time \( t \) to not instantiate this class at a different time \( t' \). We highlight here that the ontological choices made at this example are intuitive, but arguable. Our aim was to illustrate ontological notions, not to propose an ontological analysis of the scholar domain.

3. Representing changes in OWL

In this section, we present some alternatives to represent changes in OWL. First, in order to clarify the choices in the following sections, we show a static-world mapping that is incapable of dealing with changes. We introduce here a UML interpretation for SROIQ\(^D\), the fragment of DL underpinning OWL2. We denote \( \mathcal{C} \) a set of concepts or classes, \( \mathcal{R} \) a set of object properties or relations, \( \mathcal{R}_T \) a set of data properties or attributes, \( \mathcal{F}_T \) a set of datatypes or attributes types, \( \mathcal{S} \) a set of symbols from the alphabet of DLs (see [16] for their interpretations in first order logic), \( \Omega \) a set of UML cardinalities and a function \( \ell(\Omega \rightarrow S) \) such that \( \ell(*) \rightarrow \forall, \ell((n) \rightarrow \forall_{=n}, \ell((0..n)) \rightarrow \forall_{\leq n} \) and \( \ell(n..*) \rightarrow \forall_{\geq n} \) (with \( n > 0 \)).\(^2\)

**Definition 1.** Let \( \{ \mathcal{C}, D_1, \ldots, D_n, E \} \subseteq \mathcal{C}, \{ r_1, \ldots, r_n \} \subseteq \mathcal{R}, \{ t_1, \ldots, t_m \} \subseteq \mathcal{R}_T, \{ \phi_1, \ldots, \phi_n \} \subseteq \mathcal{F}_T \) and \( \{ \mu_1, \ldots, \mu_n, \sigma_1, \ldots, \sigma_n, \psi_1, \ldots, \psi_m \} \subseteq \Omega \), an UML interpretation \( U \) is defined below:

\[
\begin{array}{c|c|c|c|c}
D_1 & r_1 & C & D_n & r_n \\
\hline
\end{array}
\]

\[
\left\{ (C \sqsubseteq E \cap \prod_{i=1}^n \ell(\mu_i)r_i, D_i \cap \prod_{j=1}^m \ell(\psi_j)t_j, \phi_j) \right\} ^U
\]

\[
\Leftrightarrow
\left\{ (D_i \equiv \ell(\sigma_i)r_i^{-1}C) ^U \right\}
\]

This mapping cannot represent any change, thus assuming a world in which everything is static and immutable, i.e., attributes and relations cannot change, including the relation of instantiation between individuals and classes. The fundamental dichotomy Endurant vs. Perdurant, between types of individuals, appears in the systems of categories of Foundational Ontologies like UFO [17], BFO [18] and DOLCE [19] and have already been employed to address the issue of changing information in DL (e.g. Descriptions and Situations ontology [20]). Figure 1(b) shows the representation of a 3D en-

\(^1\) Anti-rigidity is stronger than non-rigidity, the logical negation of rigidity.

\(^2\) We denote: \( \forall_{\geq n}\mathcal{C} \equiv \forall_{\leq n}\mathcal{C} \cap =_{nr}\mathcal{C}, \forall_{\geq n}\mathcal{C} \equiv \forall_{\leq n}\mathcal{C} \cap \geq_{nr}\mathcal{C} \) and \( \forall_{\leq n}\mathcal{C} \equiv \forall_{\geq n}\mathcal{C} \).
durant with a fourth temporal dimension, where an individual named John is represented as a 4D object—also called a space-time worm—whose slices are snapshots of John’s orcid 0000-0003-0634-3277 during his life as an author. An endurant, such as a person, is fundamentally different from what is called a perdurant (or process), which has temporal parts unfolding in time, e.g., a flight, a conference, or a PhD defense. Intuitively, endurants exist at times, while perdurants happen at times.

Contrariwise to endurantism, the perdurantist approach removes the distinction between endurants and perdurants by defending that “objects are composed of so-called temporal parts. When we see an object here and now, we are seeing the parts of it that are now — but there are other parts of it at other times that we might have encountered or might yet encounter.” [21]. While objects are seen as 3D endurants through the endurantist approach, they appear otherwise as perdurantist worms, i.e., four dimensional “space-time worms” whose temporal parts are slices (snapshots) of the worms. In the following, we present the temporally changing information frameworks proposed in [12,13]. We illustrate this framework in our domain by snapshots of Mary’s life during her existence as an author.

First introduced by Leibniz [22], the notion of “individual concept” allows the mapping of an individual to all its snapshots (or time slices), whenever it exists, by referring to a single characteristic (or set of characteristics). These characteristics, said essential (i.e., necessary and immutable), define the identity of an individual [15] (e.g., the proper name of an individual in the Kripkean sense [23]). In [12,13], the authors use the UML diagram pattern depicted in the Figure 2(a) as a framework to capture temporally changing information. The main idea is to partition the domain in two levels: the static level (IC level), regarding individual concepts; and the dynamic level (TS level), concerning changeable parts of individual’s snapshots. The timeSliceOf relation connects both levels such that each instance of IndividualConcept maps to one or more instances of TimeSlice, while an instance of TimeSlice refers to exactly one instance of IndividualConcept. Indeed, the life-time of an instance of IndividualConcept can be determined by the initial instant (the value of the startSAt dataproperty) of its first TimeSlice and the final instant of its last TimeSlice (the value of the endsSAt dataproperty). Every instance of IndividualConcept must have at least one time slice for representing its life-time.

![Figure 2. 4D representation](image-url)
Figure 2(b) illustrates the 4D approach by presenting a situation in which Mary is temporally associated with a University. The value of the Mary’s hindex and the value of the rank of the University evolves through time. The ellipses at the top represent the individual concepts, which are instances of some classes of the IC level (Person and University). Inside the TS level, a cylinder represents the temporal projection of the individual concept to which it is connected. Each division in the cylinder is a new (contiguous) TimeSlice of the connected individual concept, and thus the inter-level vertical arrows (at the top) represent instantiations of the tSliceOf property. The temporal extension of each time slice goes until the next one (or until the end of the cylinder for the last division). The horizontal arrows represent the instantiation of the isAssociatedWith property. The darker ellipses represent that some change occur w.r.t. the previous time slice. Thereafter, we present some alternatives (namely, A0, A1 and A2) to design a TBox based on the framework introduced in the Figure 2(a), in order to capture the perdurantist and endurantist notions together.

3.1. The mapping alternative A0

In the mapping alternative A0 (Figure 3(a), exemplified in Figure 3(b)), the IC level comprises rigid classes, while the TS level concerns all the others classes, relations and attributes.

3.2. The mapping alternative A1

In the mapping alternative A1, while the IC level level comprises rigid classes, simultaneously mandatory and immutable attributes, and relations determining mutual existential dependencies; the TS level concerns the non-rigid classes, properties and relations that do not configure mutual existential dependencies (see Figure 4(a)). Figure 4(b) exemplifies the alternative A1. The main difference in using the alternative A1 w.r.t. A0 is the decreasing of redundancy (by using A0, all the attributes, including immutable and mandatory, are duplicated).
3.3. The mapping alternative A2

In the mapping alternative A2, the **IC level** comprises rigid classes, simultaneously mandatory and immutable attributes, while the **TS level** concerns non-rigid classes, contingent and mutable properties (see Figure 5(a)). Figure 5(b) exemplifies the alternative A2. The main difference in using the alternative A2 instead of the alternative A1 is the decrease in the proliferation of time slices. Differently from the alternative A2, by using the alternative A1, every time slice in a chain of connected instances is duplicated when a new time slice is created. Note that the relations implying unilateral existential dependence are represented in the **IC level** and interpreted as valid through the whole lifetime of the dependent entity.

The generic dependences require relaxing the maximum cardinality at the side of the dependent entity, as the relationship is changeable w.r.t. the independent individual.
For example, an instance of Scholarship can participate in a relationship isRemuneratedBy with different instances of Researcher during its existence (it can be the case that one researcher is hired to complete the scholarship of another researcher). The maximum cardinality constraint should be relaxed in order to allow this kind of change.

In next section, we report on an experimental comparison of A0, A1 and A2.

4. Empirical comparison

We evaluated the three alternatives by performing reasoning tasks. Firstly, we developed a TBox populator that generates random consistent ABoxes for the three alternatives.3 The populator was developed in Java and is supported by the OWLAPI 4. The population starts with initial simple assertions: John and Mary are co-authors of the book Commitments when each one was associated with the organisation City Hall, after what the random process of axioms creation begins revolving around these assertions about John and Mary. We included a parameter $k$ for the total number of ABoxes axioms (assertions of person, publications and affiliations changes and inherent roles). We made 5 populations $P_k$ for each alternatives and $k \in \{10,100,5000,10000, 150000,200000, 300000,350000,500000\}$. Table 1 presents the median w.r.t. the 5 populations of the total number of time slices created in function of the alternatives and the parameter $k$. Note that a “-” in the table does not mean that the population could not be performed, but that the reasoning task on the ABox was impossible due to a heap space limit as explained thereafter.

<table>
<thead>
<tr>
<th>$k$</th>
<th>$A_0$</th>
<th>$A_1$</th>
<th>$A_2$</th>
<th>$k$</th>
<th>$A_0$</th>
<th>$A_1$</th>
<th>$A_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>17</td>
<td>14</td>
<td>6</td>
<td>25000</td>
<td>149278</td>
<td>149207</td>
<td>129989</td>
</tr>
<tr>
<td>100</td>
<td>409</td>
<td>362</td>
<td>262</td>
<td>30000</td>
<td>180300</td>
<td>178723</td>
<td>156201</td>
</tr>
<tr>
<td>1000</td>
<td>5526</td>
<td>5404</td>
<td>4755</td>
<td>35000</td>
<td>210436</td>
<td>209279</td>
<td>182756</td>
</tr>
<tr>
<td>5000</td>
<td>30167</td>
<td>28822</td>
<td>25491</td>
<td>40000</td>
<td>-</td>
<td>217732</td>
<td>209315</td>
</tr>
<tr>
<td>10000</td>
<td>59350</td>
<td>59090</td>
<td>52117</td>
<td>45000</td>
<td>-</td>
<td>268551</td>
<td>234595</td>
</tr>
<tr>
<td>15000</td>
<td>89735</td>
<td>89664</td>
<td>77115</td>
<td>50000</td>
<td>-</td>
<td>-</td>
<td>260493</td>
</tr>
<tr>
<td>20000</td>
<td>119221</td>
<td>119576</td>
<td>104165</td>
<td>55000</td>
<td>-</td>
<td>-</td>
<td>287296</td>
</tr>
</tbody>
</table>

Table 1. Number of time slices created in the ABoxes

Once the population was performed, we launched some queries listed in Figure 6 over the generated KBoxes4 (TBoxes + ABoxes) to represent the evolution of some key indicators. We designed the SPARQL Query 1 in order to retrieve the evolution of the hindex for each Author present in the ABoxes, while the Query 2 was designed to retrieve the evolution of the arwu index (based on the average of hindex in our simulation) of all the Organisations present in the ABoxes for the alternatives $A_0$, $A_1$ and $A_2$. The Query 3 was designed to retrieve the number of citations for each Publication present in an ABox. Query 3 was designed for $A_2$, while Query 3’ is the same query adapted for $A_0$ and $A_1$. The Query 4 was designed to output all the Mary’s co-Authors that are present in an ABox. Query 4 was designed for $A_2$, while Query 4’ is the same query adapted for $A_0$ and $A_1$. The Query 5 was designed to output all the

3We consider here an ABox as a finite set of concept and role (abstract and concrete assertions).
4Some samples are available at https://ontohub.org/repositories/linkedun
Query 1.
```
SELECT ?α ?β ?γ
WHERE {
  {?
  α a lu:Person .
  ?α lu:timeSlice ?δ .
  ?α lu:startsAt ?γ .}
UNION
{?
  α a lu:Person .
  ?α lu:timeSlice ?δ .
  ?δ lu:endsAt ?γ .}
}
```

Query 2.
```
SELECT ?α ?β ?γ
WHERE {
  {?
  α a lu:Organisation .
  ?α lu:timeSlice ?δ .
  ?δ lu:startsAt ?γ .}
UNION
{?
  α a lu:Organisation .
  ?α lu:timeSlice ?δ .
  ?δ lu:endsAt ?γ .}
}
```

Query 3.
```
SELECT ?α ?β ?γ ?ϵ
WHERE {
  {?
  α a lu:Author .
  ?α lu:timeSlice ?δ .
  ?δ lu:startsAt ?γ .
  ?δ lu:citations ?ϵ .}
}
```

Query 3'.
```
SELECT ?α ?β ?γ ?ϵ
WHERE {
  {?
  α a lu:Author .
  ?α lu:timeSlice ?δ .
  ?δ lu:startsAt ?γ .
  ?δ lu:citations ?ϵ .}
}
```

Query 4.
```
SELECT DISTINCT ?α
WHERE {
  ?.α lu:writes ?β .
  ?.β lu:isWrittenBy :Mary .
} FILTER(?α != :Mary)
```

Query 4'.
```
SELECT DISTINCT ?α
WHERE {
  ?.α lu:writes ?β .
  ?.β lu:isWrittenBy :Mary .
} FILTER(?α != :Mary)
```

Query 5.
```
SELECT ?α
WHERE {
  ?.α lu:cites :Commitments .
}
```

Query 5'.
```
SELECT DISTINCT ?α
WHERE {
  ?.α lu:timeSlice ?δ .
  ?.δ lu:cites ?γ .
  ?.γ lu:writes ?ζ .
  ?.ζ lu:tSOf :John .
}
```

Query 6.
```
SELECT DISTINCT ?β
WHERE {
  ?.α lu:isPartOf ?β .
  ?.α lu:cites ?γ .
  ?.γ lu:writes ?ζ .
  ?.ζ lu:writes ?ξ .
  ?.ξ lu:isWrittenBy :Mary .
  ?.ξ lu:tSOf :John .
}
```

Query 6'.
```
SELECT DISTINCT ?β
WHERE {
  ?.α lu:isPartOf ?β .
  ?.α lu:cites ?γ .
  ?.γ lu:writes ?ζ .
  ?.ζ lu:writes ?ξ .
  ?.ξ lu:isWrittenBy :Mary .
  ?.ξ lu:tSOf :John .
}
```

Query 6''.
```
SELECT DISTINCT ?β
WHERE {
  ?.α lu:isPartOf ?β .
  ?.α lu:cites ?γ .
  ?.γ lu:writes ?ζ .
  ?.ζ lu:writes ?ξ .
  ?.ξ lu:isWrittenBy :Mary .
  ?.ξ lu:tSOf :John .
}
```

Figure 6. Different SPARQL queries designed for A0, A1 and A2
Publications, present in an ABox, and that cite the book of John and Mary. Query 5 was designed for \( A_2 \), while Query 5’ is the same query adapted for \( A_0 \) and \( A_1 \). Finally, the Query 6 was designed to output all the Books present in an ABox having a chapter that cites a Publication of John. Query 6 was designed for \( A_2 \), while Query 6’ is the same query adapted for \( A_1 \), and Query 6” for \( A_0 \).

![Figure 7](image_url)

Figure 7. Experimental comparison between the alternatives.

We performed an empirical analysis on a machine equipped with an Intel Core at 3.30GHz and Ubuntu 15.04. We ran the Java-based reasoner Pellet with Sun Java 1.8, and we set the maximum heap space to 7.5 GB. Figure 7 shows a comparison performed by launching queries with Pellet and measuring the elapsed CPU times. For each query and each alternative, we performed 5 query answering tasks on the ABoxes corresponding to each population \( P_k \), after what we retained the median value of the CPU times.

We make the following observations from Figure 7. For all the queries, for a number of instances \( \geq 5000 \), the alternative A2 is always the fastest model w.r.t. CPU time. A2
succeeds to compute the results until \( \sim 50000 \) instances for all the queries, after what a heap space limit occurs and precludes the computation. Note that this heap space limit occurs earlier w.r.t. the number of instances for the alternatives A1 (\( \geq 40000 \) instances, except for the Query 3’) and again earlier for A0 (\( \geq 30000 \) instances, except for the Query 3’). Generally, the Queries 3 / 3’ are the queries for which the heap space limit occurs earlier (for A0 and A1) and require more CPU time to finish (around twice more). The Query 4’ is the query requiring the maximum amount of time to output the results for a large number of instances (\( \geq 35000 \) instances for A1, and \( \geq 25000 \) instances for A0). The proliferated time slices fill the memory space, eventually reaching the heap space limit and precluding the query task. We noticed that for the alternative A1 and close to the heap space limit, the task takes a longer time to finish. We hypothesize that, due to the non proliferation of the immutable attributes (e.g. name), the remaining memory enables the computation for a higher number of instances; while the proliferation of some dependent relations (those that are not mutually dependent) among the time slices makes the higher density of the graph of instances to slow down the computation.

On the Queries 1 / 2, starting with all the instances of an individual concept (e.g., Person), the reasoner explores all their time slice to output the evolution in the time of the mutable attributes (e.g., hindex or arwu). For these queries (and also for their alternatives) the node length amplitude of the matching graph pattern is 2.

On the Queries 3 / 3’ (and their alternatives) the node length amplitude of the matching graph pattern is 3. Nevertheless, the Query 3’ designed for the alternatives A0 and A1 differs from the Query 3 designed for A2. The latter launches a task where the matching graph pattern has parts in both the static and dynamic levels, while the matching graph pattern of A0 and A1 is only in the dynamic level. For the alternative A2, the computation is 1.5 times slower than the times spent with the other queries for the same alternative, what suggests that dealing with a mix of static and dynamic entities increased the computational times. For A0 and A1, the computation is also slower than the times spent with the other queries for the same alternatives, what suggests that dealing with both object properties and data properties (i.e. writes, citations and startsAt) among the time slices can be also much more greedy.

The Queries 4 / 4’ and 5 / 5’ confront the speed of exploring (i) only in the static level (A2), and (ii) only in the dynamic level (A0, A1). For the Queries 4 / 4’, the node length amplitude of the matching graph pattern is: 4 for the alternatives A0 and A1, and 2 for A2. For the Queries 5 / 5’, the node length amplitude of the matching graph pattern is: 3 for the alternatives A0 and A1, and 1 for A2. We encoded in our populating algorithm a random draw for the authors of publications (1~5) and the citations of publications (1~15). Thus, the lower number of instances involving the relation writes could explain the performance of the reasoner for the Query 4 being better than for the Query 5. Nevertheless, for A0 and A1, it seems that due to the node length amplitude of the matching graph pattern, the reasoner performed the Query 5’ in a shorter time.

The Queries 6 / 6’ / 6” deal with mutual dependencies (e.g., partOf). The Query 6 (A2) only explores the static level, and the node length amplitude of the matching graph pattern is 3. The Query 6’ (A1) explores both the static and the dynamic levels, and the node length amplitude of the matching graph pattern is 5. The Query 6” (A0) explores only the dynamic level, the node length amplitude of the matching graph pattern is 5.

To summarize, the relative similitude between the behaviors of the reasoner confronted to the same alternatives for the Query 1, the Query 2, the Queries 5 / 5’ and the
Queries 6 / 6’ / 6” suggests little difference between exploring only in the dynamic level: (i) compared to an exploration in both the static and the dynamic levels for A0 and A1; or (ii) compared to an exploration only in static level for A2. The notable difference occurs when the reasoner performs an exploration both in the static and the dynamic levels for the alternative A2 (corresponding to the Query 3), or an exploration tackling object properties and data properties among the time slices (i.e., writes, citations and startsAt) for the alternatives A0 and A1 (corresponding to the Query 3’). Finally, if it could be in one sense unsurprising that these different queries have different performances in their executions, note that the comparison touched upon the performance of different mapping frameworks to retrieve the same kind of domain information.

5. Related Works

Four-dimensionalism is a significant school of thought, particularly in the field of Formal Ontology (see [24]). Krieger et al [10,11] proposed to reinterpret the 4D view by introducing the class of time slices as a superclass of both contingent and mandatory classes. In a sense, this proposal is represented by our alternative A0 where all the class, relations and attributes are encoded in a dynamic level.

As we mentioned in the introduction, the first attempt to deal with a four dimensional approach in OWL was proposed by Welty and Fikes [8], who introduced the idea of using 4D fluents to deal with relationships that change over time. Nevertheless, the nature of the relations or attributes were not considered, and the dependence between individuals was not evoked in their model. That is why in [12,13], the authors introduced a static level in which immutable properties could be encoded in order to optimize the memory requirements during the reasoning task. This proposal corresponds to the alternative A1. A 4D-based analysis of “Roles” (a specific kind of non-rigid classes) has also been performed in [25]. Later, in [26], some foundational ontologies were compared also considering the scholar domain (with a particular perdurantist view of the behavior of the role student).

Concerning the experimental analysis on temporally changing information-based models, such validations or comparisons have been attempted in very few cases. In [27,28], the authors proposed alternatives (based on a combination of qualitative and quantitative representation for interval and point relations) to represent in OWL and/or in SWRL the so called Allen’s temporal relations. Note that in [29], the authors point out the potential usage of such alternatives to express relations between time intervals of 4D-fluents in OWL. Thus, the authors performed an experimental comparison of the alternatives w.r.t. consistency tests using the reasoners Hermit and Pellet on a data-set with a relative small amounts of instances (100 to 1000 intervals generated randomly). The authors claimed it was the first such experimental evaluation of both qualitative and quantitative Semantic Web temporal representations.

Finally, Gutierrez et al. [30] were the first to propose a formal extension of the RDF data model to integrate a consideration of time validity. Thus, they introduced graphs containing quads of the form (s,p,o)[t] where t is a timestamp during which the triple (s,p,o) is valid. In [31], the authors implemented a solution to query such quads stored and experimentally demonstrated that their implementation (based on the system Strabon) outperforms all other existing implementations (e.g. AnQL, AllegroGraph).
6. Conclusion

It is very important to properly represent entities amenable to changes in terms of a knowledge representation language that could support decidable automated reasoning. In this paper, after introducing some notions that can describe the qualities of attributes and relations subject to change, we presented three strategies to map a stereotyped UML-like class diagram into TBoxes. We also performed an experimental comparison to observe in real reasoning tasks how the aforementioned alternatives would behave. The comparison showed that the alternative A2 had the best performance for all the queries. The empirical studies reported here serve the purpose of stress testing these mapping frameworks as practical alternatives to represent large instance datasets. In a sense, the observed differences between these frameworks (some of which are analogous to well-know proposals in the literature [12,9]) could be expected from an analytical study of how each of these frameworks structures information. However, the study reported here is in a much better position to analyze and quantify these differences in terms of the performance of execution of representative queries. A future work would be to perform an experimental comparison between the alternative A2 and a reification-based model [12,13], in order to assess in which situation one alternative is more suitable than the other.

References


