Storing and Querying Fuzzy Knowledge in the Semantic Web using FiRE

Nikolaos Simou, Giorgos Stoilos, and Giorgos Stamou

Department of Electrical and Computer Engineering,
National Technical University of Athens,
Zographou 15780, Greece
{nsimou,gstoil,gstam}@image.ntua.gr

Abstract. An important problem for the success of ontology-based applications is how to provide persistent storage and querying. For that purpose, many RDF tools capable of storing and querying over a knowledge base, have been proposed. Recently, fuzzy extensions to ontology languages have gained considerable attention especially due to their ability to handle vague information. In this paper we investigate on the issue of using classical RDF storing systems in order to provide persistent storing and querying over large scale fuzzy information. To accomplish this we propose a novel way for serializing fuzzy information into RDF triples, thus classical storing systems can be used without any extensions. Additionally, we extend the existing query languages of RDF stores in order to support expressive fuzzy querying services over the stored data. All our extensions have been implemented in FiRE—an expressive fuzzy DL reasoner that supports the language fuzzy-$\mathcal{SHIN}$. Finally, the proposed architecture is evaluated using an industrial application scenario about casting for TV commercials and spots.

1 Introduction

Despite the great success of the World Wide Web during the last decade, it is still not uncommon that information is extremely difficult to find. Such cases include searching for popular names, multi-language words with different definitions in different languages and specific information that requires more sophisticated queries. The Semantic Web—an extension of the current Web—aims at alleviating these issues by adding semantics to the content that exists on the Web. Information in the (Semantic) Web would be linked forming a distributed knowledge base and documents would be semantically annotated.

Ontologies, through the OWL language [14], are expected to play a significant role in the Semantic Web. OWL is mainly based on Description Logics (DLs) [1], a popular family of knowledge representation languages. Their well-defined semantics, together with their decidable reasoning algorithms, have made them popular to a variety of applications [1]. However, despite their rich expressiveness, they are not capable of dealing with vague and uncertain information, which is commonly found in many real-world applications such as multimedia content, medical informatics and more. For this purpose fuzzy extensions to Description Logics have been proposed [11, 20, 18, 2].

Fuzzy ontologies have also gained considerable attention in many research applications. Similar to crisp ontologies, they can serve as basic semantic infrastructure, providing shared understanding of certain domains across different applications. Furthermore, the need for handling fuzzy and uncertain information is crucial to the Web. This is because information and data along the Web may often be uncertain or imperfect. Such cases are domains that may be described using concepts, like “near” for modeling distance or concept “tall” for modeling people and buildings height. The paradox that arises in the latter case is that there is no distinction between a person’s and a building’s height.

Clearly the information represented by those kinds of concepts is very important though imperfect. Therefore sophisticated uncertainty representation and reasoning are necessary for the alignment and integration of Web data from different sources. Recently, also fuzzy DL reasoners such as fuzzyDL [3] and FiRE [17] that can provide practical reasoning over imprecise information have been implemented. Despite these implementations little work has been done towards the persistent storage and querying of information. Closely related works are those on fuzzy DL-Lite [22, 13], where the reasoning algorithms assume that the data have been stored previously in a relational database.

In this paper we follow a different paradigm and study the use of an RDF triple-store for the storage of fuzzy information. More precisely, we propose an architecture that can be used to store such information in an off-the-shelf triple-store and then show how the stored information can be queried using a fuzzy DL reasoner. The main contributions of this paper are the following:

1. It presents a novel framework for persistent storage and querying of expressive fuzzy knowledge bases,
2. It presents the first ever integration of fuzzy DL reasoners with RDF triple-stores, and
3. It provides experimental evaluation of the proposed architecture using a real-world industrial strength use-case scenario.

The rest of the paper is organized as follows. Firstly, in Section 2 the theoretical description of the fuzzy DL f-$\mathcal{SHIN}$ [18] is given. After that in Section 3 the fuzzy reasoning engine FiRE which supports the fuzzy DL f-$\mathcal{SHIN}$ and was used in our approach is presented. In the following Section (4) the proposed triples syntax accommodating the fuzzy element used for storing a fuzzy knowledge base in RDF-Stores, is presented. Additionally, the syntax and the semantics of expressive queries that have been proposed in the literature [13] to exploit fuzziness are briefly presented. In the last Section (5) the applicability of the proposed architecture is demonstrated, presenting a use case based on a database of human models. This database was used by a production company for the purposes of casting for TV commercials and spots. Some entries of the database were first fuzzified and then using an expressive knowledge base, abundant implicit knowledge was extracted. The extracted knowledge was stored to an RDF

Store, and various expressive queries were performed in order to benchmark the proposed architecture.

2 Preliminaries

2.1 The Fuzzy DL $\mathbf{f}_{KD}$-$\mathcal{SHIN}$

In this section we briefly present the notation of DL f-$\mathcal{SHIN}$ which is a fuzzy extension of DL $\mathcal{SHIN}$ [8]. Similar to crisp description logic languages, a fuzzy description logic language consist of an alphabet of distinct concepts names ($\mathbf{C}$), role names ($\mathbf{R}$) and individual names ($\mathbf{I}$), together with a set of constructors to construct concept and role descriptions. If R is a role then R^- is also a role, namely the inverse of R. f-$\mathcal{SHIN}$-concepts are inductively defined as follows:

1. If $C \in \mathbf{C}$, then C is a f-$\mathcal{SHIN}$-concept,
2. If C and D are concepts, R is a role, S is a simple role and $n \in \mathbb{N}$, then $(\neg C)$, $(C \sqcup D)$, $(C \sqcap D)$, $(\forall R.C)$, $(\exists R.C)$, $(\geq nS)$ and $(\leq nS)$ are also f-$\mathcal{SHIN}$-concepts.

In contrast to crisp DLs, the semantics of fuzzy DLs are provided by a *fuzzy interpretation* [20]. A fuzzy interpretation is a pair $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$ where $\Delta^{\mathcal{I}}$ is a non-empty set of objects and $\cdot^{\mathcal{I}}$ is a fuzzy interpretation function, which maps an individual name a to elements of $\mathsf{a}^{\mathcal{I}} \in \Delta^{\mathcal{I}}$ and a concept name A (role name R) to a membership function $\mathsf{A}^{\mathcal{I}} : \Delta^{\mathcal{I}} \to [0,1]$ ($R^{\mathcal{I}} : \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \to [0,1]$).

By using fuzzy set theoretic operations the fuzzy interpretation function can be extended to give semantics to complex concepts, roles and axioms [9]. FiRE uses the standard fuzzy operators of $1 - x$, where x is a degree, for fuzzy negation (c), $\max(x, y)$, for fuzzy union (u), $\min(x, y)$ for fuzzy intersection (t) and $\max(1-x, y)$ for fuzzy implication ($\mathcal{J}$). For example, if d_1 is the degree of membership of an object o to the set $A^{\mathcal{I}}$ and d_2 the degree of membership to $B^{\mathcal{I}}$, then the membership degree of o to $(A \sqcap B)^{\mathcal{I}}$ is given by $\min(d_1, d_2)$, while the membership degree to $(A \sqcup B)^{\mathcal{I}}$ is given by $\max(d_1, d_2)$. The complete set of semantics is depicted in Table 1.

A f-$\mathcal{SHIN}$ knowledge base Σ is a triple $\langle \mathcal{T}, \mathcal{R}, \mathcal{A} \rangle$, where $\mathcal{T}$ is a fuzzy $TBox$, $\mathcal{R}$ is a fuzzy $RBox$ and $\mathcal{A}$ is a fuzzy $ABox$. $TBox$ is a finite set of fuzzy concept axioms which are of the form $C \sqsubseteq D$ called fuzzy concept inclusion axioms and $C \equiv D$ called fuzzy concept equivalence axioms, where C, D are concepts, saying that C is a sub-concept or C is equivalent of D, respectively. In cases where C is allowed to be a complex concept we have a general concept inclusion axiom (GCI). At this point it is important to note that in our approach f_{KD}-$\mathcal{SHIN}$ without GCIs is considered. The interested reader is referred to [19, 10] that present how GCIs are handled in fuzzy DLs and also to fuzzy DLs that have been proposed in the literature [7, 18] and support GCIs. $RBox$ is a finite set of fuzzy role axioms of the form $\mathsf{Trans}(R)$ called fuzzy transitive role axioms and $R \sqsubseteq S$ called fuzzy role inclusion axioms saying that R is transitive and R is a sub-role of S respectively. Finally, $ABox$ is a finite set of fuzzy assertions of the

Constructor	Syntax	Semantics
top	$\top$	$\top^{\mathcal{I}}(a) = 1$
bottom	$\bot$	$\bot^{\mathcal{I}}(a) = 0$
general negation	$\neg C$	$(\neg C)^{\mathcal{I}}(a) = c(C^{\mathcal{I}}(a))$
conjunction	$C \sqcap D$	$(C \sqcap D)^{\mathcal{I}}(a) = t(C^{\mathcal{I}}(a), D^{\mathcal{I}}(a))$
disjunction	$C \sqcup D$	$(C \sqcup D)^{\mathcal{I}}(a) = u(C^{\mathcal{I}}(a), D^{\mathcal{I}}(a))$
exists restriction	$\exists R.C$	$(\exists R.C)^{\mathcal{I}}(a) = \sup_{b \in \Delta^{\mathcal{I}}}\{t(R^{\mathcal{I}}(a,b), C^{\mathcal{I}}(b))\}$
value restriction	$\forall R.C$	$(\forall R.C)^{\mathcal{I}}(a) = \inf_{b \in \Delta^{\mathcal{I}}}\{\mathcal{J}(R^{\mathcal{I}}(a,b), C^{\mathcal{I}}(b))\}$
at-most	$\leq pR$	$\inf_{b_1,\ldots,b_{p+1} \in \Delta^{\mathcal{I}}} \mathcal{J}(t_{i=1}^{p+1} R^{\mathcal{I}}(a,b_i), u_{i<j}\{b_i = b_j\})$
at-least	$\geq pR$	$\sup_{b_1,\ldots,b_p \in \Delta^{\mathcal{I}}} t(t_{i=1}^{p} R^{\mathcal{I}}(a,b_i), t_{i<j}\{b_i \neq b_j\})$
inverse role	R^-	$(R^-)^{\mathcal{I}}(b,a) = R^{\mathcal{I}}(a,b)$
equivalence	$C \equiv D$	$\forall a \in \Delta^{\mathcal{I}}.C^{\mathcal{I}}(a) = D^{\mathcal{I}}(a)$
sub-concept	$C \sqsubseteq D$	$\forall a \in \Delta^{\mathcal{I}}.C^{\mathcal{I}}(a) \leq D^{\mathcal{I}}(a)$
transitive role	$\mathsf{Trans}(R)$	$\forall a,b \in \Delta^{\mathcal{I}}.R^{\mathcal{I}}(a,b) \geq \sup_{c \in \Delta^{\mathcal{I}}}\{t(R^{\mathcal{I}}(a,c), R^{\mathcal{I}}(c,b))\}$
sub-role	$R \sqsubseteq S$	$\forall a,b \in \Delta^{\mathcal{I}}.R^{\mathcal{I}}(a,b) \leq S^{\mathcal{I}}(a,b)$
concept assertions	$\langle a : C \bowtie n\rangle$	$C^{\mathcal{I}}(a^{\mathcal{I}}) \bowtie n$
role assertions	$\langle\langle a,b\rangle : R \bowtie n\rangle$	$R^{\mathcal{I}}(a^{\mathcal{I}}, b^{\mathcal{I}}) \bowtie n$

Table 1. Semantics of concepts and roles

form $\langle a : C \bowtie n\rangle$, $\langle(a,b) : R \bowtie n\rangle$, where $\bowtie$ stands for $\geq, >, \leq$ or $<$, or $a \not\doteq b$, for $a, b \in \mathbf{I}$. Intuitively, a fuzzy assertion of the form $\langle a : C \geq n\rangle$ means that the membership degree of a to the concept C is at least equal to n.

Example 1. An example of a fuzzy knowledge base Σ is shown below.
$\mathcal{T} = \{\mathsf{MiddleAged} \equiv \mathsf{40s} \sqcup \mathsf{50s}, \mathsf{TallChild} \equiv \mathsf{Child} \sqcap (\mathsf{Short} \sqcup \mathsf{Normal\_Height}),\}$
$\mathcal{R} = \{\mathsf{isFriendOf}^- = \mathsf{isFriendOf}\}$ and
$\mathcal{A} = \{\langle michalis1539 : \mathsf{Male}\rangle, \langle michalis1539 : (\mathsf{Tall} \sqcap \mathsf{GoodLooking}) \geq 0.8\rangle,$ $\langle(michalis1539, maria1343) : \mathsf{isFriendOf} \geq 0.7\rangle\}$

3 Fuzzy Reasoning Engine FiRE

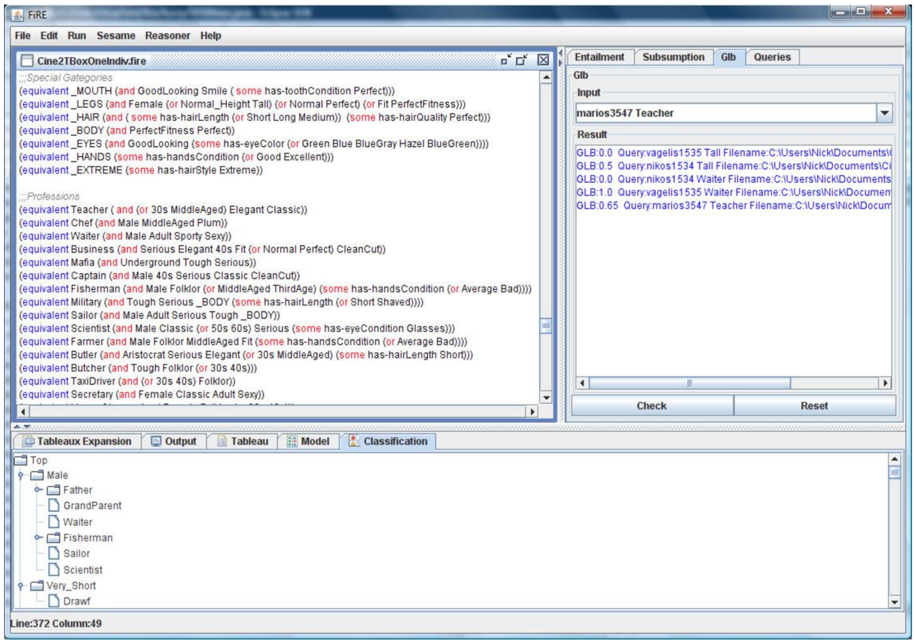
In this section we present the graphical user interface, the syntax and the inference services of FiRE—an expressive fuzzy DL reasoner.

3.1 FiRE interface

FiRE[1] is a Java based fuzzy reasoning engine. FiRE implements the tableau reasoning algorithm for fuzzy-$\mathcal{SHIN}$ presented in [18]. It can be used either as an API by another application or through its graphical user interface. The graphical user interface of FiRE consists of the editor panel, the inference services panel and the output panel (Figure 1). Hence the user has the ability to create or edit an existing fuzzy knowledge base using the editor panel, and to use the inference services panel to make different kinds of queries to the fuzzy knowledge

[1] `http://www.image.ece.ntua.gr/~nsimou/FiRE/`

base. Finally, the output panel consists of four different tabs, each one displaying feedback depending on the user operation.



```
;;;Special Gategories
(equivalent _MOUTH (and GoodLooking Smile ( some has-toothCondition Perfect)))
(equivalent _LEGS (and Female (or Normal_Height Tall) (or Normal Perfect) (or Fit PerfectFitness)))
(equivalent _HAIR (and ( some has-hairLength (or Short Long Medium))  (some has-hairQuality Perfect)))
(equivalent _BODY (and PerfectFitness Perfect))
(equivalent _EYES (and GoodLooking (some has-eyeColor (or Green Blue BlueGray Hazel BlueGreen))))
(equivalent _HANDS (some has-handsCondition (or Good Excellent)))
(equivalent _EXTREME (some has-hairStyle Extreme))

;;;Professions
(equivalent Teacher ( and (or 30s MiddleAged) Elegant Classic))
(equivalent Chef (and Male MiddleAged Plum))
(equivalent Waiter (and Male Adult Sporty Sexy))
(equivalent Business (and Serious Elegant 40s Fit (or Normal Perfect) CleanCut))
(equivalent Mafia (and Underground Tough Serious))
(equivalent Captain (and Male 40s Serious Classic CleanCut))
(equivalent Fisherman (and Male Folklor (or MiddleAged ThirdAge) (some has-handsCondition (or Average Bad))))
(equivalent Military (and Tough Serious _BODY (some has-hairLength (or Short Shaved))))
(equivalent Sailor (and Male Adult Serious Tough _BODY))
(equivalent Scientist (and Male Classic (or 50s 60s) Serious (some has-eyeCondition Glasses)))
(equivalent Farmer (and Male Folklor MiddleAged Fit (some has-handsCondition (or Average Bad))))
(equivalent Butler (and Aristocrat Serious Elegant (or 30s MiddleAged) (some has-hairLength Short)))
(equivalent Butcher (and Tough Folklor (or 30s 40s)))
(equivalent TaxiDriver (and (or 30s 40s) Folklor))
(equivalent Secretary (and Female Classic Adult Sexy))
```

Input: marios3547 Teacher

```
GLB:0.0  Query:vagelis1535 Tall Filename:C:\Users\Nick\Documents\
GLB:0.5  Query:nikos1534 Tall Filename:C:\Users\Nick\Documents\Ci
GLB:0.0  Query:nikos1534 Waiter Filename:C:\Users\Nick\Documents
GLB:1.0  Query:vagelis1535 Waiter Filename:C:\Users\Nick\Documen
GLB:0.65  Query:marios3547 Teacher Filename:C:\Users\Nick\Docum
```

```
Top
  Male
    Father
    GrandParent
    Waiter
    Fisherman
    Sailor
    Scientist
  Very_Short
    Drawf
```

Fig. 1. The FiRE user interface consists of the editor panel (upper left), the inference services panel (upper right) and the output panel (bottom)

3.2 FiRE syntax

As previously mentioned, a fuzzy knowledge base consists of three components $TBox$, $RBox$ and $ABox$. The $TBox$ and the $RBox$ are defined using the Knowledge Representation System Specification [16] proposal since they do not include uncertainty. So, transitive roles or the sub-role of another role can be defined by using the keywords **transitive** and **parent** respectively and concept axioms by the keywords **implies** and **equivalent**. (Please refer to [16] for a full specification.)

On the contrary, since the assertions are extended in order to represent imperfect knowledge, the ABox specified differently. Instances in FiRE are defined using the keyword **instance** followed by the individual, the concept in which the individual participates, the inequality type (one of $<, <=, >, >=$) and the degree of confidence $degree \in [0, 1]$. Similarly, role assertions are defined by using the keyword **related** followed by subject and object individuals, the inequality type and the degree of confidence. In both cases the inequality type and the degree of confidence are required only for fuzzy assertions, if these are not mentioned then the assertions are assumed as crisp (i.e >= 1).

Example 2. The syntax of the assertions defined in example 1 are shown below in FiRE syntax.

```
(instance michalis1539 Male)
(instance michalis1539 (and Tall GoodLooking) >= 0.8)
(related michalis1539 maria1343 isFriendOf >= 0.7)
```

3.3 FiRE reasoning services

Description Logics offer a large range of inference services, which the user can query over a knowledge base. The main reasoning services offered are satisfiability checking, subsumption and entailment of concepts and axioms with respect to an ontology. For example, one is capable of asking queries like "Can the concept C have any instances in models of the ontology T?" (satisfiability of C), or "Is the concept D more general than the concept C in models of the ontology T?" (subsumption $C \sqsubseteq D$), or "Does axiom Ψ logically follow from the ontology?" (entailment of Ψ).

In addition to these reasoning services, fuzzy DLs also provide with *greatest lower bound queries* (GLB). In the case of fuzzy DL, satisfiability questions become of the form "Can the concept C have any instances with degree of participation $\bowtie n$ in models of the ontology T?". Furthermore, the incorporation of degrees in assertions makes the evaluation of the best lower and upper truth-value bounds of a fuzzy assertion vital. The term of *greatest lower bound* of a fuzzy assertion with respect to Σ was defined in [20]. Informally, greatest lower bound queries are queries like "What is the greatest degree n that our ontology entails an individual a to participate in a concept C?".

FiRE uses the tableau algorithm of f-$\mathcal{SHIN}$ presented in [18], in order to decide the key inference problems of a fuzzy ontology. Hence entailment queries that ask whether our knowledge base logically entails the membership of an individual to a specific concept and to a certain degree, are specified in the *Entailment* inference tab (see Figure 1). Their syntax is the same as the one used for the definition of a fuzzy instance. For example a statement of the form:

```
(instance michalis1539 (and Tall GoodLooking) > 0.8)
```

asks whether michalis1539 is Tall and GoodLooking to a degree greater than or equal to 0.8. If there are assertions in the *ABox* of our Σ that satisfy this query (i.e. there is a model for our ontology) then FiRE will return true.

On the other hand subsumption queries that are specified in the *Subsumption* inference tab evaluate whether a concept is more general than another concept. Their syntax is of the following form:

```
(concept1) (concept2)
```

where `concept1` and `concept2` are f-$\mathcal{SHIN}$ concepts. Let us assume that the first concept is Father while the second concept is Male. Subsequently, assume that Father has been defined in the *TBox* using an equivalence axiom as follows: $\mathsf{Father} \equiv \mathsf{Male} \sqcap \mathsf{MiddleAged}$. Then, the following subsumption query will always return true since Father will always be (in all models) a sub-concept of Male.

```
(Father) (Male)
```

Additionally, the user can perform a global concept classification procedure presenting the concept hierarchy tree in the *Classification* tab of the output panel.

Finally, FiRE supports GLB queries, which are evaluated by FiRE by performing entailment queries. During this procedure a set of entailment queries is constructed consisting of an entailment query for every degree contained in the ABox, using the individual and the concept of interest. These queries are performed using the binary search algorithm to reduce the degrees search space, resulting the GLB. The syntax of GLB queries is of the form:

```
individual (concept)
```

where `concept` can be any f-$\mathcal{SHIN}$-concept. In order to illustrate the operation of the GLB service we will present a trivial example using an atomic concept. Let the following *ABox* (in FiRE syntax)

```
(instance michalis1539 Tall > 0.8)
(instance maria231 GoodLooking > 0.6)
(instance nikos Male > 1)
(instance nikos Tall > 0.9)
```

We want to evaluate the GLB of individual `michalis1539` in the concept Tall. In FiRE this is specified by issuing the query `michalis1539` Tall. Then, the system proceeds as follows: Firstly, all the degrees that appear in *ABox* are sorted. FiRE then performs entailment queries for the individual `michalis1539` with the concept Tall, using the binary search algorithm. This procedure is repeated until the entailment query is unsatisfiable. The greatest degree found before unsatisfiability is the greatest lower bound. In this example the following entailment queries are performed with the indicated results in order to evaluate that the greatest lower bound of `michalis1539` participating in concept Tall is 0.8.

```
(instance michalis1539 Tall > 0.5) TRUE
(instance michalis1539 Tall > 0.8) TRUE
(instance michalis1539 Tall > 1)   FALSE
```

Finally, FiRE offers the possibility to perform a global GLB for the whole fuzzy knowledge base. Global GLB evaluates the greatest lower bound degree of all combinations of individuals and concepts in Σ.

4 Storing and Querying a Fuzzy Knowledge Base

In the current section we will present how FiRE stores and queries fuzzy knowledge. More precisely in the proposed architecture a triple-store is used as a back end for storing and querying RDF triples in a sufficient and convenient way, while the reasoner is the front end, which the user can use in order to store and query a fuzzy knowledge base. In that way, a user is able to access data from a repository, apply any of the available reasoning services on this data and then store back in the repository the implicit knowledge extracted from them.

Many triple-store systems have been developed in the context of the Semantic Web, like Sesame[2], Kowari[3], Jena[4] and more. These provide persistent storage of ontology data as well as querying services. In our approach FiRE was integrated with Sesame (Sesame 2 beta 6), an open source Java framework for storing and querying RDF/RDFS data. Sesame supports two ways of storing RDF data (called RDF Repositories). The first is the in-memory RDF Repository, which stores all the RDF triples in the main memory, while the second one is the native RDF Repository, which stores the RDF triples on the hard disk and uses B-Trees to index and access them.

4.1 Storage of a Fuzzy Knowledge Base

In order to use a triple-store for storing a fuzzy knowledge without enforcing any extensions on it, we needed to find a way to serialize fuzzy knowledge into RDF triples. For that purpose a fuzzy-OWL to RDF mapping was required, similar to the one provided in the OWL abstract syntax and semantics document [14]. In previous works Mazzieri and Dragoni [12] used RDF *reification* in order to store the membership degrees. However it is well-known that reification has weak, ill-defined model theoretic semantics and its support by RDF tools is limited. In another approach, Vaneková et al. [23] suggest the use of datatypes, but we argue that the use of a concrete feature like datatypes to represent abstract information such as fuzzy assertions is not appropriate.

Consequently, we were lead to propose a new way of mapping fuzzy-OWL into RDF triples. The technique makes use of RDF's blank nodes. First, we define three new entities, namely `frdf:membership`, `frdf:degree` and `frdf:ineqType` as types (i.e. `rdf:type`) of `rdf:Property`.

Using these new properties together with blank nodes we can represent fuzzy instances. Let's assume for example that we want to represent the assertion $\langle(michalis1539 : \mathsf{Tall}) \geq 0.8\rangle$. The RDF triples representing this information are the following:

[2] `http://www.openrdf.org/`
[3] `http://www.kowari.org/`
[4] `http://jena.sourceforge.net/`

```
region1                   frdf:membership   _:michalis1539membTall .
_:michalis1539membTall    rdf:type          Tall .
_:michalis1539membTall    frdf:degree       “0.8^^xsd:float” .
_:michalis1539membTall    frdf:ineqType     “=” .
```

where `_:michalis1539membTall` is a blank node used to represent the fuzzy assertion of `michalis1539` with the concept Tall.

Mapping fuzzy role assertions, however, is more tricky since RDF does not allow for blank nodes in the predicate position. To overcome this issue we define new properties for each assertion; for example, the fuzzy role assertion $\langle(michalis1539, maria1343) : \mathsf{isFriendOf} \geq 0.7\rangle$ is mapped to

```
michalis1539          frdf:p1p2isFriendOf   maria1343 .
frdf:p1p2isFriendOf   rdf:type              isFriendOf .
frdf:p1p2isFriendOf   frdf:degree           “0.7^^xsd:float” .
frdf:p1p2isFriendOf   frdf:ineqType         “=” .
```

It is worth mentioning at that point that recently a fuzzy ontology representation has been proposed by Bobillo et. al [4] that it could be also used for the RDF serialization.

4.2 Fuzzy queries

One of the main advantages of persistent storage systems, like relational databases and RDF storing systems, is their ability to support *conjunctive queries*. Conjunctive queries generalize the classical inference problem of *realization* of Description Logics [1], i.e. “get me all individuals of a given concept C”, by allowing for the combination (conjunction) of concepts and roles. Formally, a conjunctive query is of the following form:

$$q(X) \leftarrow \exists Y.conj(X, Y) \tag{1}$$

or simply $q(X) \leftarrow conj(X, Y)$, where $q(X)$ is called the head, $conj(X, Y)$ is called the body, X are the *distinguished variables*, Y are the existentially quantified variables called *non-distinguished variables*, and $conj(X, Y)$ is a conjunction of atoms of the form $\mathsf{A}(v)$, $R(v_1, v_2)$, where A, R are concept and role names, respectively, and v, v_1 and v_2 are *individual* variables in X and Y or individuals from the ontology.

Since in our case we extend classical assertions to fuzzy assertions, new methods of querying fuzzy information are possible. More precisely, in [13] the authors extend ordinary conjunctive queries to a family of significantly more expressive query languages, which are borrowed from the fields of fuzzy information retrieval [6]. These languages exploit the membership degrees of fuzzy assertions by introducing weights or thresholds in query atoms. In particular, the authors first define *conjunctive threshold queries* (CTQs) as:

$$q(X) \leftarrow \exists Y. \bigwedge_{i=1}^{n} (atom_i(X,Y) \geq k_i) \tag{2}$$

where $k_i \in [0,1]$, $atom_i(X,Y)$ represents either a fuzzy-DL concept or role and all $k_i \in (0,1]$ are thresholds. As it is obvious those answers of CTQs are a matter of true or false, in other words an evaluation either is or is not a solution to a query.

The authors also propose *General Fuzzy Conjunctive Queries* (GFCQs) that further exploit fuzziness and support degrees in query results. The syntax of a GFCQ is the following:

$$q(X) \leftarrow \exists Y. \bigwedge_{i=1}^{n} (atom_i(X,Y) : k_i) \tag{3}$$

where $atom_i(X,Y)$ is as above while k_i is the degree associated weight. As shown in [13], this syntax is general enough to allow various choices of semantics, which emerge by interpreting differently the degree of each fuzzy-DL atom $(atom_i(X,Y))$ with the associated weight (k_i). In what follows, we give some example of the semantic functions for conjunctions and degree-associated atoms.

1. **Fuzzy threshold queries:** As we mentioned earlier in a conjunctive threshold query an evaluation either satisfies the query entailment or not, thus providing only crisp answers. A straightforward extension will be instead of using crisp threshold we can use fuzzy ones with the aid of fuzzy R-implications. Hence, let a t-norm (t) as the semantic function for conjunctions and R-implications (ω_t) as the semantic function for degree-associated atoms, we get fuzzy threshold queries, in which the degree of truth of q_F under $\mathcal{I}$ is

$$d = \sup_{S' \in \Delta^{\mathcal{I}} \times \ldots \times \Delta^{\mathcal{I}}} \{t_{i=1}^{n}\ \omega_t(k_i, atom_i^{\mathcal{I}}(\bar{v})_{[X \mapsto S, Y \mapsto S']})\}.$$

   Given some S', if for all atoms we have $atom_i^{\mathcal{I}}(\bar{v})_{[X \mapsto S, Y \mapsto S']} \geq k_i$, since $\omega_t(x,y) = 1$ when $y \geq x$ [9], we have $d = 1$; this corresponds to threshold queries introduced earlier.
2. **Traditional conjunctive queries [21]:** These are the traditional conjunctive queries of the form 1, but instead of classical (boolean) conjunction between the atoms of the query we use the semantics of a t-norm. Then, the degree of truth of q_F under $\mathcal{I}$ is:

$$d = \sup_{S' \in \Delta^{\mathcal{I}} \times \ldots \times \Delta^{\mathcal{I}}} \{t_{i=1}^{n}\ atom_i^{\mathcal{I}}(\bar{v})_{[X \mapsto S, Y \mapsto S']}\}.$$

   It is worth noting that this query language is a special case of the fuzzy threshold query language, where all $k_i = 1$ and since $\omega_t(1,y) = y$ [9].
3. **Fuzzy aggregation queries:** Another example of semantics for GFCTs would be to use fuzzy aggregation functions [9]. For example, let $G(x) =$

$\sum_{i=1}^{n} x_i$, as a function for conjunctions and $a(k_i, y) = \frac{k_i}{\sum_{i=1}^{n} k_i} * y$ as the semantic function for degree-associated atoms. Then we get an instance of fuzzy aggregation queries, in which the degree of truth of q_F under $\mathcal{I}$ is

$$d = \sup_{S' \in \Delta^{\mathcal{I}} \times \ldots \times \Delta^{\mathcal{I}}} \frac{\sum_{i=1}^{n} k_i * atom_i^{\mathcal{I}}(\bar{v})_{[X \mapsto S, Y \mapsto S']}}{\sum_{i=1}^{n} k_i}.$$

4. **Fuzzy weighted queries:** If we use generalised weighted t-norms [5] as the semantic function for conjunction, we get fuzzy weighted queries, in which the degree of truth of q_F under $\mathcal{I}$ is

$$d = \sup_{S' \in \Delta^{\mathcal{I}} \times \ldots \times \Delta^{\mathcal{I}}} \{\min_{i=1}^{n} u(\overline{k} - k_i, t(\overline{k}, atom_i^{\mathcal{I}}(\bar{v})_{[X \mapsto S, Y \mapsto S']}))\},$$

   where $\overline{k} = \max_{i=1}^{n} k_i$ and u is a t-conorm (fuzzy union), such as $u(a, b) = \max(a, b)$. The main idea of this type of queries is that they provide an aggregation type of operation, on the other hand an entry with a low value for a low-weighted criterion should not be critically penalized. Moreover, lowering the weight of a criterion in the query should not lead to a decrease of the relevance score, which should mainly be determined by the high-weighted criteria. For more details see [5].

4.3 Fuzzy queries using FiRE

All of the above mentioned queries have been implemented in FiRE by using the SPARQL [15] query language for RDF. The queries are translated into SPARQL and are then issued on Sesame. A user can specify such queries using the *Queries* inference tab, and in the case of generalized fuzzy conjunctive queries a choice of the above mentioned semantics is possible.

Example 3. The following is a threshold query:

```
x,y <- Father(x) >= 0.8 ^ isFriendOf(x,y) >= 1.0
                                  ^ Teacher(y) >= 0.7
```

Queries consist of two parts: the first one specifies the variables for which their bindings with individuals from the ABox will be returned as an answer, while the second one states the condition that has to be fulfilled for the individuals. The above query asks for all individuals that can be mapped to x and y, such that the individual mapped to x will be a Father to a degree at least 0.8, and then, also participate in the relation `isFriendOf` (to a degree 1.0) with another individual that y is mapped to, which also belongs to the concept Teacher to a degree at least 0.7.

The query is firstly converted from the FiRE conjunctive query syntax to the SPARQL query language. Based on the fuzzy OWL syntax in triples that we have defined previously, the query of Example 3 is as follows in SPARQL. (The query results are evaluated by the Sesame engine and visualized by FiRE.)

```
SELECT ?x WHERE {
    ?x frdf:membership ?Node1 .
    ?Node1 rdf:type ?Concept1 .
    ?Node1 frdf:ineqType ?IneqType1 .
    ?Node1 frdf:degree ?Degree1 .
    FILTER regex (?Concept1 , "CONCEPTS#Father")
    FILTER regex (?IneqType1 ,">")
    FILTER (?Degree1 >= "0.8^^xsd:float")

    ?BlankRole2 frdf:ineqType ?IneqType2 .
    ?BlankRole2 frdf:degree ?Degree2 .
    ?BlankRole2 rdf:type ?Role2 .
    ?x BlankRole2 ?y .
    FILTER regex (?Role2 , "ROLES#isFriendOf")
    FILTER regex (?IneqType1 ,">")
    FILTER (?Degree2 >= "1^^xsd:float")

    ?y frdf:membership ?Node3,
    ?Node3 rdf:type ?Concept3 .
    ?Node3 frdf:ineqType ?IneqType3 .
    ?Node3 frdf:degree ?Degree3 .

    FILTER regex (?Concept3 , "CONCEPTS#Short")
    FILTER regex (?IneqType1 ,">")
    FILTER (?Degree1 >= "0.7^^xsd:float")
}
```

Example 4. GFCQ are specified using the symbol ":" followed by the importance of participation for each condition statement. For example, we can ask for female persons having LongLegs and BeautifulEyes and rank higher those with larger degree for the latter:

```
x <- Female(x):1 ^ LongLegs(x) : 0.6  BeautifulEyes(x) : 0.8
```

In the case of GGCQs the operation is different. The SPARQL query is constructed in a way that retrieves the participation degrees of every role or concept used in the atoms criteria, for the results that satisfy all of the atoms. The participation degrees retrieved for each query atom together with its weight are then used by FiRE for the ranking procedure of the results based on the selected semantics. An excerpt of the SPARQL query for Example 4 follows.

```
SELECT ?x ?Degree1...
WHERE {
    ?x frdf:membership ?Node1 .
    ?Node1 rdf:type ?Concept1 .
    ?Node1 frdf:ineqType ?IneqType1 .
    ?Node1 frdf:degree ?Degree1 .
```

```
        FILTER regex (?Concept1 , "CONCEPTS#Female")
        FILTER regex (?IneqType1 ,">")
        FILTER (?Degree1 >= "0.0^^xsd:float")
        ...
    }
```

Concluding this section, it should be noted that the proposed architecture (clearly) does not provide a complete query answering procedure for f-$\mathcal{SHIN}$, since queries are directly evaluated using the RDF triple-store and no f-$\mathcal{SHIN}$ reasoning is involved. However, prior to storing the information in the triple-store, FiRE applies a global GLB on the given ontology to materialize as much implied information as possible. This makes the proposed architecture complete for ground conjunctive queries, i.e., those queries that do not contain any non-distinguished variables, and that additionally do not allow for complex roles in the query body, i.e., roles that are transitive or inverse. On the one hand, this is a common practice for many proposed practical systems such as Jena,[5] OWLim,[6] and DLEJena[7], which first materialize implied knowledge and then store it into a triple-store, aiming at sacrificing some completeness in favour of performance. On the other hand, how to efficiently answer general conjunctive queries over expressive DLs is also still an open question and most DL reasoners, such as KAON2[8] and RacerPro[9] mainly only support ground conjunctive queries.

5 Evaluation

In the current section we present our use case scenario and the methods used to provide support through the previously presented architecture. First, we describe the use case and we show how we have fuzzified a certain number of database fields, in order to extract rich semantic information from numerical fields. Then we describe the models' knowledge base presenting the definitions of some new interesting concepts, which can be used to assist the process of casting.

5.1 Models use case

The data were taken from a production company database containing 2140 human models. The database contained information on each model regarding their height, age, body type, fitness type, tooth condition, eye-condition and color, hair quality, style, color and length, ending with the hands' condition. Apart from the above, there were some additional, special-appearance characteristics for certain models such as good-looking, sexy, smile, sporty, etc., introduced by the casting producer. Finally for a minority of models, a casting-video was stored

[5] `http://jena.sourceforge.net/`
[6] `http://www.ontotext.com/owlim`
[7] `http://lpis.csd.auth.gr/systems/DLEJena/`
[8] `http://kaon2.semanticweb.org/`
[9] `http://www.racer-systems.com/`

in the database. The main objective of the production company was to pick a model, based on the above features, who would be suitable for a certain commercial spot. Furthermore, depending on the spot type, inquiries about models with some profession-like characteristics (like teacher, chef, mafia etc.) were also of interest.

Despite the fact that the database information on each model was relatively rich, there was great difficulty in querying models of appropriate characteristics and the usual casting process involved almost browsing over a large part of the database to find out for people matching the desired characteristics. One major issue is that information in the database is not semantically organised. Moreover, the organisation of the information in the various tables made searching for combined characteristics impossible. Additionally, the crisp approach of querying over databases makes it difficult to perform queries over fields such as age, height and weight. If a person does not match exactly the specified criteria it will not be returned by the database. Hence, casting people usually set those criteria much lower than the wanted in order to avoid having a low recall and then browsed through the returned set to exclude false positive answers. Our goal in the current use case was to semantically enrich the database in order to make this process more easy. Moreover, we aimed at using the fuzzy technologies to alleviate the problem of precise matching of query criteria.

The process of constructing the fuzzy knowledge base was split into two parts. The first part involved the generation of the fuzzy *ABox*. The characteristics given by numerical values in the database, like height and age, were fuzzified giving rise to new (fuzzy) concepts, while remaining characteristics were used as crisp assertions. For example, the fuzzification process of the field age was performed by setting fuzzy partitions depending on age and by defining the concepts Kid, Teen, Baby, $\mathsf{20s}$, $\mathsf{30s}$, $\mathsf{40s}$, $\mathsf{50s}$, $\mathsf{60s}$ and Old. As can be observed from the age fuzzification graph, a model who is 29 years old participates in both concepts $\mathsf{20s}$ and $\mathsf{30s}$ with degrees 0.35 and 0.65 respectively. Similarly for the fuzzification of field height, the concepts $\mathsf{Very\_Short}$, Short, $\mathsf{Normal\_Height}$, Tall and $\mathsf{Very\_Tall}$ were defined. In the case of the height, the fuzzy partition used for female models was different from the one used for males, since the average height of females is lower than that of males. The fuzzification graphs of age and men's height are shown in Figure 2. An example of the produced assertions is shown in Example 5.

Example 5. An excerpt of the ABox for the model $michalis1539$:

$$\begin{aligned}
&\langle michalis1539 : \mathsf{20s} \geq 0.66\rangle, \langle michalis1539 : \mathsf{30s} \geq 0.33\rangle,\\
&\langle michalis1539 : \mathsf{Normal\_Height} \geq 0.5\rangle,\\
&\langle michalis1539 : \mathsf{Tall} \geq 0.5\rangle,\ \langle michalis1539 : \mathsf{GoodLooking} \geq 1\rangle,\\
&\langle (michalis1539, good) : \mathsf{has-toothCondition} \geq 1\rangle,\\
&\langle good : \mathsf{Good} \geq 1\rangle
\end{aligned}$$

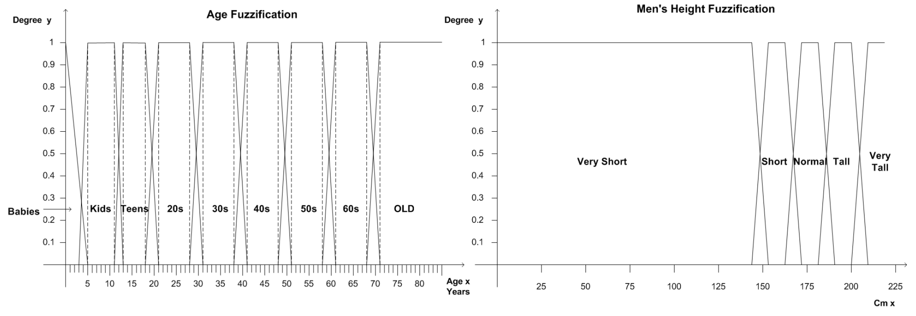


Fig. 2. Fuzzification graphs

5.2 The fuzzy knowledge base

In order to permit knowledge-based retrieval of human models we have implemented an expressive terminology for a fuzzy knowledge base. The alphabet of concepts used for the fuzzy knowledge base consists of the features described above while some characteristics like hair length, hair condition etc. were represented by the use of roles. The most important of roles and concepts are shown below. In the concept set the concept features are highlighted in bold.

The set of individuals consist of the models name along with an ID.

The effective extraction of implicit knowledge from the explicit one requires an expressive terminology capable of defining higher concepts. In our case the higher domain concepts defined for human models lie into five categories: age, height, family, some special categories and the professions. Hence, the profession Scientist has been defined as male, between their 50s or 60s, with classic appearance who also wears glasses. In a similar way we have defined 33 domain concepts; an excerpt of the *TBox* can be found in Table 3.

5.3 Results

All the experiments were conducted under Windows XP on a Pentium 2.40 GHz computer with 2. GB of RAM.

The described fuzzy knowledge base was used in the evaluation of our approach. Implicit knowledge was extracted using the greatest lower bound service of FiRE, asking for the degree of participation of all individuals, in all the defined domain concepts. The average number of explicit assertions per individual was 13 while the defined concepts were 33, that together with the 2140 individuals (i.e entries of the database) resulted to 29460 explicit assertions and the extraction of 2430 implicit. These results, together with concept and role axioms, were stored to a Sesame repository using the proposed fuzzy OWL triples syntax to form a repository of 529.926 triples.

The average time for the GLB reasoning process and the conversion of explicit and implicit knowledge to fuzzy OWL syntax in triples was 1112 milliseconds.

Concepts	
Gender:	Male, Female
Height:	Very_Short, Short, Normal_Height, Tall, Very_Tall
Age:	Baby, Kid, Teen, 20s,30s, 40s, 50s, 60s, Old
Body Type:	Fat, Normal, Perfect, Plum, Slim
Fitness:	Fit, Unfit, PerfectFitness
Hair Length:	Long, Medium, Shaved, Tonsure, Bold
Hair Color:	Black, BrownLight, Brown, BlondLight, Grey, BlondDark, BlondRed, BrownRed, Blond, Red, White, Brown − Grey, 2 − Colour, Platinum, Black − Grey
Hair Style:	Wavy, Straight, Curly, Extreme, Rasta, Frizy
Hair Quality:	Natural, Dyed, Highlights
Eyes Color:	Green, Blue, BlueGray, Hazel, BlueGreen
Eyes Condition:	Healthy, NotHealthy, Glasses, ContactLenses
Tooth Condition:	MissingTooth, Brace, Good, Bad, MissingTeeth
Hands Condition:	Excellent, Average
Special Characteristics:	Sexy, GoodLooking, Smile, Sporty, Ugly, Serious, Funny, Tough, Aristocrat, Artistic, Folklor, Romantic, Elegant, Classic, CleanCut, Underground
Roles	
has − hairLength, has − hairColour, has − eyeColor, has − eyeCondition, has − handsCondition	

Table 2. Concepts and Roles defined for the various characteristics (fields in the database)

The time required for uploading the knowledge to a Sesame repository depends on the type of repository (Memory or Native) and also on repository's size. Based on our experiments, we have observed that the upload time is polynomial to the size of the repository but without significant differences. Therefore, the average minimum upload time to an almost empty repository (0-10.000 triples) is 213 milliseconds while the average maximum upload time to a full repository (over 500.000 triples) is 700 milliseconds.

FiRE and Sesame were also tested on expressive fuzzy queries. We have used the following set of test queries:

$$q_1 : \ x \leftarrow \mathsf{Scientist}(x)$$
$$q_2 : \ x \leftarrow \mathsf{Father}(x) \geq 1 \wedge \mathsf{Teacher}(x) \geq 0.8$$
$$q_3 : \ x \leftarrow \mathsf{Legs}(x) \geq 1 \wedge \mathsf{Eyes}(x) \geq 0.8 \wedge \mathsf{20s}(x) \geq 0.5$$
$$q_4 : \ x \leftarrow \mathsf{Scientist}(x) : 0.8$$
$$q_5 : \ x \leftarrow \mathsf{Father}(x) : 0.6 \wedge \mathsf{Teacher}(x) : 0.7$$
$$q_6 : \ x \leftarrow \mathsf{Legs}(x) : 0.8 \wedge \mathsf{Eyes}(x) : 1 \wedge \mathsf{20s}(x) : 0.6$$

The performance in this case mainly depended on the complexity of the query but also on the type and size of the repository. Queries using role names in combination with large repositories can dramatically slow down the response. Table 4 illustrates the response times in milliseconds using both types of repositories and different repository sizes. Repository sizes was set by adjusting the number

$$
\begin{aligned}
\mathcal{T} = \{\mathsf{MiddleAged} &\equiv \mathsf{40s} \sqcup \mathsf{50s},\\
\mathsf{TallChild} &\equiv \mathsf{Child} \sqcap (\mathsf{Short} \sqcup \mathsf{Normal\_Height}),\\
\mathsf{Father} &\equiv \mathsf{Male} \sqcap (\mathsf{30s} \sqcup \mathsf{MiddleAged}),\\
\mathsf{Legs} &\equiv \mathsf{Female} \sqcap (\mathsf{Normal\_Height} \sqcup \mathsf{Tall})\\
&\qquad\quad \sqcap(\mathsf{Normal} \sqcup \mathsf{Perfect}) \sqcap (\mathsf{Fit} \sqcup \mathsf{PerfectFitness}),\\
\mathsf{Teacher} &\equiv (\mathsf{30s} \sqcup \mathsf{MiddleAged}) \sqcap \mathsf{Elegant} \sqcap \mathsf{Classic},\\
\mathsf{Fisherman} &\equiv \mathsf{Male} \sqcap \mathsf{Folklor} \sqcap (\mathsf{MiddleAged} \sqcup \mathsf{ThirdAge})\\
&\qquad\qquad\quad \sqcap \exists \mathsf{has-handsCondition}.(\mathsf{Average} \sqcup \mathsf{Bad}),\\
\mathsf{Military} &\equiv \mathsf{Male} \sqcap \mathsf{Tough} \sqcap \mathsf{Serious} \sqcap \mathsf{\_BODY}\\
&\qquad\qquad\quad \sqcap \exists \mathsf{has-hairLength}.(\mathsf{Short} \sqcap \mathsf{Shaved}),\\
\mathsf{Scientist} &\equiv \mathsf{Male} \sqcap \mathsf{Classic} \sqcap (\mathsf{50s} \sqcup \mathsf{60s})\\
&\qquad\qquad \sqcap \mathsf{Serious} \sqcap \exists \mathsf{has-eyeCondition}.\mathsf{Glasses}\ \}\\
\mathsf{Butler} &\equiv \mathsf{Male} \sqcap \mathsf{Aristocrat} \sqcap \mathsf{Serious} \sqcap \mathsf{Elegant}\\
&\qquad\qquad \sqcap(\mathsf{30s} \sqcup \mathsf{MiddleAged}) \sqcap \exists \mathsf{has-hairLength}.\mathsf{Short}\ ,\\
\mathsf{Butcher} &\equiv \mathsf{Male} \sqcap \mathsf{Tough} \sqcap \mathsf{Folklor} \sqcap (\mathsf{30s} \sqcup \mathsf{40s})\ ,\\
\mathsf{TaxiDriver} &\equiv \mathsf{Folklor} \sqcap (\mathsf{30s} \sqcup \mathsf{40s})\ ,\\
\mathsf{Secretary} &\equiv \mathsf{Female} \sqcap \mathsf{Classic} \sqcap \mathsf{Adult} \sqcap \mathsf{Sexy}\ ,\\
\mathsf{HouseCleaner} &\equiv \mathsf{Female} \sqcap \mathsf{Folklor} \sqcap (\mathsf{30s} \sqcup \mathsf{40s})\ \}
\end{aligned}
$$

Table 3. An excerpt of the Knowledge Base ($TBox$).

Query	**Native**			**Memory**		
	100.000	250.000	500.000	100.000	250.000	500.000
q_1	1042	2461	3335	894	2364	3332
q_2	1068	2694	3935	994	2524	3732
q_3	3667	8752	21348	4267	7348	9893
q_4	2562	4173	5235	3042	4543	6027
q_5	4318	6694	8935	4341	7896	9306
q_6	9906	29831	66251	12164	16421	20489

Table 4. Evaluation of the fuzzy queries (time in ms)

of assertions. As it can be observed, very expressive queries seeking for young female models with beautiful legs and eyes as well as long hair, a popular demand in commercial spots, can be easily performed. It is worth mentioning that these queries consist of higher domain concepts defined in our fuzzy knowledge base.

Since our system is not a sound and complete query answering system for f-$\mathcal{SHIN}$, the GLB service performed before uploading the triples is employed in order to use as much of the expressivity of the language as possible producing new implied assertions.

Furthermore, the results regarding query answering time are also very encouraging, at least for the specific application. Although, compared to crisp querying, over crisp knowledge bases, our method might require several more seconds to be answered (mainly due to post processing steps for GFCQs or due to very lengthy SPARQL queries for CTQs) this time is significantly less, compared to the time spent by producers on casting (usually counted in days), since they usually have to browse through a very large number of videos and images before they decide.

6 Conclusions

Due to the fact that imperfect information is inherent in many web-based applications, the effective management of imperfect knowledge is very important for the realisation of the Semantic Web. In this paper, we have proposed an architecture that can be used for storing and querying fuzzy knowledge bases for the Semantic Web. Our proposal which is based on the DL f-$\mathcal{SHIN}$, integrates the Sesame RDF triple-store (through a proposed serialisation of f-OWL to RDF triples), the fuzzy reasoning engine FiRE and implements very expressive fuzzy queries on top of them.

The proposed architecture was evaluated using an industrial application scenario about casting for TV commercials and spots. The obtained results are very promising from the querying perspective. From the initial 29460 explicit assertions made by database instances for models, 2430 new implicit assertions where extracted and both uploaded in the Sesame repository. In this way expressive semantic queries like "Find me young female models with beautiful legs and eyes as well as long hair", that might have proved very difficult or even impossible using the producing company's database, are applicable through FiRE. This reveals both the strength of knowledge-based applications, and technologies for managing fuzzy knowledge, since a wealth of the information of the databases, like age, height, as well as many high level concepts of the specific application, like "beautiful eyes", "perfect fitness" and "scientist look" are inherently fuzzy.

As far as future directions are concerned, we intend to further investigate on different ways of performing queries using expressive fuzzy description logics. Finally, it would be of great interest to attempt a comparison between the proposed architecture and approaches using fuzzy DL-lite ontologies and approximation.

References

1. F. Baader, D. McGuinness, D. Nardi, and P.F. Patel-Schneider. *The Description Logic Handbook: Theory, implementation and applications.* Cambridge University Press, 2002.
2. F. Bobillo, M. Delgado, and J. Gómez-Romero. Crisp representations and reasoning for fuzzy ontologies. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 17(4):501–530, 2009.
3. F. Bobillo and U. Straccia. fuzzydl: An expressive fuzzy description logic reasoner. In *In Proceedings of the 17th IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2008)*, pages 923–930, 2008.
4. F. Bobillo and U. Straccia. Fuzzy ontology representation using owl 2. international journal of approximate reasoning. *International Journal of Approximate Reasoning*, 52(7):1073–1094, 2011.
5. A. Chortaras, Giorgos Stamou, and Andreas Stafylopatis. Adaptation of weighted fuzzy programs. In *Proc. of the International Conference on Artificial Neural Networks (ICANN 2006)*, pages 45–54. Springer, 2006.

6. V. Cross. Fuzzy information retrieval. *Journal of Intelligent Information Systems*, 3:29–56, 1994.
7. J. Gmez-Romero F. Bobillo, M. Delgado. A crisp representation for fuzzy shoin with fuzzy nominals and general concept inclusions. In *In Proceedings of the 2nd Workshop on Uncertainty Reasoning for the Semantic Web (URSW 2006)*, 2006.
8. I. Horrocks, U. Sattler, and S. Tobies. Reasoning with Individuals for the Description Logic $\mathcal{SHIQ}$. In David MacAllester, editor, *CADE-2000*, number 1831 in LNAI, pages 482–496. Springer-Verlag, 2000.
9. G. J. Klir and B. Yuan. *Fuzzy Sets and Fuzzy Logic: Theory and Applications*. Prentice-Hall, 1995.
10. Y. Li, B. Xu, J. Lu, and D. Kang. Discrete tableau algorithms for $\mathcal{FSHI}$. In *Proceedings of the International Workshop on Description Logics (DL 2006), Lake District, UK*, 2006.
11. Thomas Lukasiewicz and Umberto Straccia. Managing uncertainty and vagueness in description logics for the semantic web. *Web Semant.*, 6(4):291–308, November 2008.
12. M.Mazzieri and A.F.Dragoni. A fuzzy semantics for semantic web languages. In *ISWC-URSW*, pages 12–22, 2005.
13. J.Z. Pan, G. Stamou, G. Stoilos, and E. Thomas. Expressive querying over fuzzy DL-Lite ontologies. In *Proceedings of the International Workshop on Description Logics (DL 2007)*, 2007.
14. P. F. Patel-Schneider, P. Hayes, and I. Horrocks. OWL Web Ontology Language Semantics and Abstract Syntax. Technical report, W3C, Feb. 2004. W3C Recommendation, URL http://www.w3.org/TR/2004/REC-owl-semantics-20040210/.
15. E. Prud'hommeaux and A. Seaborne. SPARQL query language for RDF, 2006. W3C Working Draft, http://www.w3.org/TR/rdf-sparql-query/.
16. Peter P. Schneider and Bill Swartout. Description-logic knowledge representation system specification from the KRSS group of the ARPA knowledge sharing effort. urlhttp://www.bell-labs.com/user/pfps/papers/krss-spec.ps, 1993.
17. N. Simou and S. Kollias. Fire : A fuzzy reasoning engine for impecise knowledge. K-Space PhD Students Workshop, Berlin, Germany, 14 September 2007, 2007.
18. G. Stoilos, G. Stamou, V. Tzouvaras, J. Z. Pan, and I. Horrocks. Reasoning with very expressive fuzzy description logics. *Journal of Artificial Intelligence Research*, 30(5):273–320, 2007.
19. G. Stoilos, U. Straccia, G. Stamou, and Jeff Z. Pan. General concept inclusions in fuzzy description logics. 17th European Conference on Artificial Intelligence (ECAI 06), Riva del Garda, Italy, 2006, 2006.
20. U. Straccia. Reasoning within fuzzy description logics. *Journal of Artificial Intelligence Research*, 14:137–166, 2001.
21. U. Straccia. Answering vague queries in fuzzy DL-Lite. In *Proceedings of the 11th International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems, (IPMU-06)*, pages 2238–2245, 2006.
22. U.Straccia and G.Visco. DLMedia: an ontology mediated multimedia information retrieval system. In *Proceeedings of the International Workshop on Description Logics (DL 07)*, volume 250, Insbruck, Austria, 2007. CEUR.
23. V. Vaneková, J. Bella, P. Gurský, and T. Horváth. Fuzzy RDF in the semantic web: Deduction and induction. In *Proceedings of Workshop on Data Analysis (WDA 2005)*, pages 16–29, 2005.