A fuzzy extension of the semantic Building Information Model

J. Gómez-Romero\textsuperscript{a,b,*}, F. Bobillo\textsuperscript{c}, M. Ros\textsuperscript{a,b}, M. Molina-Solana\textsuperscript{a,b}, M.D. Ruiz\textsuperscript{a,b}, M.J. Martín-Bautista\textsuperscript{a,b}

\textsuperscript{a}Department of Computer Science and Artificial Intelligence, University of Granada (Spain)
\textsuperscript{b}Research Centre for Information and Communications Technologies (CITIC-UGR) (Spain)
\textsuperscript{c}Department of Computer Science and Systems Engineering, University of Zaragoza (Spain)

Abstract

The Building Information Model (BIM) has become a key tool to achieve communication during the whole building life-cycle. Open standards, such as the Industry Foundation Classes (IFC), have contributed to expand its adoption, but they have limited capabilities for cross-domain information integration and query. To address these challenges, the Linked Building Data initiative promotes the use of ontologies and Semantic Web technologies in order to create more formal and interoperable BIMs. In this paper, we present a fuzzy logic-based extension of such semantic BIMs that provides support for imprecise knowledge representation and retrieval. We propose an expressive fuzzy ontology language, and describe how to use a fuzzy reasoning engine in a BIM context with selected examples. The resulting fuzzy semantic BIM enables new functionalities in the project design and analysis stages –namely, soft integration of cross-domain knowledge, flexible BIM query, and imprecise parametric modeling.

Keywords: Building Information Model, Ontologies, Fuzzy Ontologies

1. Introduction

The use of Building Information Models (BIMs) in the architecture, engineering and construction industry has evolved from the early three-dimensional blueprints of the building geometry developed in the 70s to the complex representations of volumes, materials, and equipment that are nowadays more and more frequent. BIMs have proved very effective to increase building quality while reducing design and construction costs by enabling better interoperation between stakeholders [1].
According to the US National Building Information Model Standard Project Committee [2], “a Building Information Model (BIM) is a digital representation of physical and functional characteristics of a facility.” A remarkable feature of this definition is that it highlights the relevance of the BIM as a “shared knowledge resource for information about a facility” that provides support for decision-making “during its life-cycle”, thus expanding its utilization beyond the design stage. Nevertheless, full-fledged BIM applications covering the whole building life-cycle are still scarce, because it implies interfacing with heterogeneous users who have different background, objectives, and priorities.

For this reason, in the last years there is an increasing interest in the development of knowledge representations able to capture the semantics of BIM data models, but also more flexible and with enhanced query capabilities in order to express different perspectives on building information, to facilitate information retrieval, and to integrate the BIM with other information resources. Given its open and neutral character, the Industry Foundation Classes (IFC) standard, proposed by the buildingSMART alliance [3], has been typically used as the basis for these extended representations. The IFC specification defines an object-based data model written in the EXPRESS data definition language, and an accompanying text-based file interchange format based on STEP. It allows creating readable models and data validation rules, but it lacks a mathematical characterization of the semantics of its representation primitives. Consequently, querying the model is essentially an informal procedure supported by ad hoc implementations.

Not surprisingly, Semantic Web technologies have been proposed to address the challenges of the next generation BIMs [4], since they offer a complete framework for the management of the knowledge published in the Web, arguably the most heterogeneous information environment of our days. The envisioned Linked Building Data cloud, based on the Semantic Web technology stack [5], increases interoperability during the building life-cycle by connecting distributed pieces of BIM data [6, 7] and cross-domain data [8]. At the core of the Linked Building Data cloud, ontologies encoded in OWL 2 (Ontology Web Language) [9] are used to define a formal conceptual schema for BIM constituents, and the RDF (Resource Description Framework) language [10] is used to encode BIM instances. We call semantic BIMs to these BIMs represented in OWL/RDF. The semantic BIM leverages classical BIM query capabilities by enabling automatic reasoning to retrieve information and to infer implicit knowledge.

The theoretical underpinnings of Semantic Web ontologies are strongly based on Description Logics (DLs), a subset of first order logic especially suitable for representing structured knowledge [11]. However, DLs cannot directly manage imprecise knowledge, which is inherent to several real-world problems [12]. This is the case of the semantic BIM, in which we may like for instance to represent that a building element is quite big, two elements are quite similar, there are several elements inside a space, and so forth. It would be also convenient to allow querying the system in these same terms; for example, to retrieve all the elements with size around a dimension value, or those that have been built with similar materials.
Fuzzy logic and fuzzy set theory are appropriate formalisms to handle imprecise knowledge. Hence, several proposals of fuzzy Description Logics to support fuzzy ontologies have emerged [13]. Generally speaking, in fuzzy ontologies concepts denote fuzzy sets, relations denote fuzzy relations, and axioms and facts are not in general either true or false, but they may hold to some degree of truth. Fuzzy ontologies are represented by using fuzzy ontology languages, and can be queried by using fuzzy ontology reasoners, such as DeLorean [14]. Although fuzzy ontologies have been used in different Information Science research areas—e.g., information retrieval [15], knowledge merge and summarization [16–19], recommender systems [20, 21], and decision-making [22]—to the best of our knowledge they have not been yet applied to solve industrial problems in practice.

The overarching objective of this paper is to present the fundamental characteristics and the applications of fuzzy ontologies that can be of interest to the BIM users. Rather than focusing on the formal description of the mathematical foundations of fuzzy DLs, we provide examples that show the representation and reasoning capabilities of such formalisms. To do so, we extend the ifcOWL and ifcRDF models obtained by the IFC-to-RDF conversion tool[1] with fuzzy information. In addition, we describe how they can be exploited in different use cases that illustrate common problems in the building design and analysis stages.

Accordingly, the main contributions of the paper are the following:

- We provide a description of the main features of fuzzy ontologies in the context of the Linked Building Data initiative, avoiding the cumbersome details of the underlying theoretical framework. For the interested reader, we also provide a selection of pointers to related works that elaborate on these topics.

- We present illustrative examples of the main representation primitives of a typical fuzzy Description Logic, and how they can be applied in different use scenarios. We also explain how to use a fuzzy ontology reasoner to query the resulting fuzzy semantic BIM for practical purposes.

- We discuss the advantages of using fuzzy ontologies over non-fuzzy (crisp) representations in the scope of the Linked Building Data research area, considering the current state of the art and the level of maturity of the existing tools, as well as their interrelation with the ifcOWL and ifcRDF technologies.

The remainder of the paper is structured as follows. First, we describe the materials, methods and tools used in this research work; namely: (i) ontologies as representation formalisms for the BIM; (ii) fuzzy logic and fuzzy DLs for the creation of fuzzy ontologies; (iii) reasoning with fuzzy ontologies. Next, we
proceed to describe some relevant representation primitives of the selected fuzzy ontology language. We explain the meaning of each primitive with examples of their use to represent imprecise building data and to make fuzzy queries. We elaborate afterwards on the added value of such extensions in three specific use cases: soft integration of cross-domain knowledge, flexible BIM query and imprecise parametric modeling. We discuss the limitations of fuzzy ontologies and their implementation feasibility in a BIM context, especially from a performance perspective. Finally, the paper finishes with a summary of the most important conclusions achieved and some directions for future work.

2. Materials, methods and tools

2.1. Ontologies and the BIM

Ontologies are typically used for representing knowledge in scenarios that require interoperation between heterogeneous agents. As mentioned, this is the case of the BIM, where several individuals with different expertise are usually involved. Essentially, an ontology is developed from the following primitive elements: (i) classes (or concepts), which determine sets that classify domain objects; (ii) instances (or individuals), which are concrete occurrences of concepts; (iii) properties (also named relations or roles), which represent binary connections between individuals, or individuals and typed values (integers, strings, etc.); and (iv) axioms, which establish restrictions over classes, instances and properties that characterize their features.

Descriptions Logics (DLs) provide a formal substratum to ontology representation primitives by mathematically defining the constructors that can be used to form complex classes, properties, and axioms, as well as their semantics. In particular, the OWL 2 language, the current standard for Semantic Web ontologies [9], is based on the DL named SROIQ(D) (each letter corresponds to a constructor or set of constructors). A detailed description of DLs is out of the scope of this paper, but the interested reader can found a concise summary in [23].

Beetz et al. proposed in [24] a mapping from the IFC data model to OWL that generates an ifcOWL ontology. Later works have implemented procedures to convert a given BIM in STEP format to RDF instances in order to obtain a specific semantic BIM [25]. The IFC-to-RDF conversion software is a publicly available tool that performs both tasks (see Footnote 1). In this section, we describe some aspects of the models obtained by the IFC-to-RDF tool that are relevant for our fuzzy extension. Notice that, as mentioned by the authors, the translation of a model is not unique, since different conversion strategies can be applied depending on the user needs. We will focus on an slightly modified version of the ‘OWL 2 EL – RDF List’ ontology. To increase readability, we will use the OWL Manchester syntax in the following examples [26].

Footnote 1: [http://ugritlab.ugr.es/r/ifc/schema-EL-RDFList.owl]
In the conversion, IFC EXPRESS classes are mapped into OWL classes, and subtype and supertype relations are represented with class inclusion axioms. For example, the IfcWindow entity is represented as follows:

```
Class: ifc:IfcWindow
    SubClassOf: ifc:IfcBuildingElement,
```

Analogously, attributes are translated into OWL properties. Due to some particular features of EXPRESS, such as the rich data type system and the capability to define attributes local to classes, the conversion of properties is not straightforward. Among the possible alternatives, the authors of IFC-to-RDF have successfully used property reification, wrapper classes for data types, and variant names for local properties.

The snippet below represents the overallHeight_of_IfcWindow attribute, which translates into a functional DataProperty property with defined domain and range. In OWL 2, it would be possible to define a range restriction based on a facet to delimit the values allowed for the attribute:

```
DataProperty: ifc:overallHeight_of_IfcWindow
    SubPropertyOf: ifc:overallHeight
    Characteristics: Functional
    Domain: ifc:IfcWindow
    Range: xsd:float [> 0.0f] # strictly positive values
```

Class definitions can include additional restrictions based on properties besides class inclusion axioms. For instance, following the previous example, the class definition below states that any IfcWindow is related with the property overallHeight_of_IfcWindow only to float values, and at most to one of them:

```
Class: ifc:IfcWindow
    SubClassOf: ifc:IfcBuildingElement,
            ifc:overallHeight_of_IfcWindow
    only xsd:float,
            ifc:overallHeight_of_IfcWindow
    max 1 xsd:float,
```

Instances are defined by asserting its type and the property values. This example shows the definition of a window instance named window1:

```
Individual: window_1
    Types: ifc:IfcWindow
    Facts:
        ifc:overallWidth_of_IfcWindow 1.0f,
```

The use of OWL and RDF to represent IFCs allows circumventing the lack of mathematical formality of EXPRESS. This has two direct consequences. First, the RDF model of the building information can be shared and linked on the Web of data. We can use the SPARQL language \cite{27} to formulate flexible and distributed queries that, in general, cannot be easily solved within the IFC standard. Likewise, the availability of RDF storage and manipulation tools notably reduces the effort required to implement new products and to support unpredicted users’ needs.

Second, existing OWL inference engines can be used to reason with the semantic BIM models. Reasoning within an ontology is an automatic procedure that infers new knowledge that has not been explicitly included in the ontology but is a logical consequence of the represented axioms. For instance, a valid inference is that “\texttt{window\_1} is an instance of \texttt{IfcBuildingElement}”, because \texttt{window\_1} is an instance of \texttt{IfcWindow}, and \texttt{IfcWindow} is a subclass of \texttt{IfcBuildingElement}. It goes without saying that these inferences become more complicated when complex concept expressions are used in large knowledge bases. The reasoning algorithms are implemented by reasoning engines, such as HermiT \cite{28}.

As a matter of example, we can define a complex class and retrieve all the (explicit and implicit) instances of that class. The following expression denotes all the building elements built of concrete:

\[
\text{Class: } \text{BuildingElementsMadeOfConcrete} \\
\text{EquivalentTo:} \\
\text{ifc:IfcBuildingElement} \\
\text{and} \\
\text{inverse ifc:relatedObjects_of_IfcRelAssociates} \\
\text{some (ifc:relatingMaterial} \\
\text{some (ifc:IfcMaterial} \\
\text{and (ifc:name value "CONCRETE")))}
\]

Reasoning can be also useful to facilitate the detection of inconsistencies in the representation, and therefore, to support complex modeling. Furthermore, it is possible to extend the model with additional functionalities based on other Semantic Web technologies. This is the case of \cite{29}, where logic-based rules for building performance checking are defined with notable advantages over more traditional approaches.

2.2. Fuzzy ontologies

2.2.1. Fuzzy Logic

Our extension to manage imprecise knowledge in the semantic BIM is based on fuzzy Description Logics, a fuzzy logic-based extension of Description Logics. Fuzzy Logic and Fuzzy Sets were proposed by L. Zadeh \cite{30} to manage imprecise and vague knowledge. While in classical set theory elements either belong to a set or not, in fuzzy set theory elements can belong to a set to a certain degree.
More formally, let $X$ be a set of elements called the reference set. A fuzzy subset $A$ of $X$ is defined by a membership function $\mu_A(x)$, or simply $A(x)$, which assigns any $x \in X$ to a value in the interval of real numbers between 0 and 1. As in the classical (or crisp) case, 0 means no membership and 1 full membership, but now a value between 0 and 1 represents the extent to which $x$ can be considered an element of $X$. Some membership functions commonly used to define fuzzy sets are the trapezoidal and the triangular. Classical examples of concepts that can be described using fuzzy sets are Tall or Young.

In fuzzy logic, it is common to restrict to finite chains of degrees of truth, instead of using the real interval $[0,1]$. In our case, we will work with the finite chain of $p + 1$ elements: $\mathcal{N} = \{0 = \gamma_0 < \gamma_1 < \ldots < \gamma_p = 1\}$, where $p \geq 1$. Such $\mathcal{N}$ can be understood as a set of linguistic terms or labels; for example, $\{0,0.2,0.4,0.6,0.8,1\} \to \{\text{false, closeToFalse, slightlyFalse, slightlyTrue, closeToTrue, true}\}$. From a practical point of view, a small $p$ is sufficient in many applications, and preferred against a large $p$ to facilitate the interpretability of the semantics to the users. Figure 1.a shows an example of a discretized trapezoidal membership function defined over the set of real numbers, so $q_1, q_2, q_3, q_4 \in \mathbb{R}$ and $\gamma_1, \ldots, \gamma_p \in \mathcal{N}$.

Fuzzy set theory extends all classical set operations to fuzzy sets. The intersection, union, complement and implication set operations are performed by corresponding functions; respectively, a t-norm, a t-conorm, a negation, and an implication. The combination of them is called a fuzzy logic, and there are several of them depending on the selected functions. We will consider the fuzzy connectives originally proposed by Zadeh, namely the Gödel conjunction and disjunction, Łukasiewicz negation, and Kleene-Dienes implication (Table 1). Other typical fuzzy logics are Łukasiewicz, Gödel, and Product, which have different properties [32].

Relations can also be extended to the fuzzy case. A (binary) fuzzy relation $R$ over two countable sets $X$ and $Y$ is a function $R : X \times Y \to \mathcal{N}$. Several properties of the relations (such as reflexive, irreflexive, symmetric, asymmetric, transitive,
Table 1: Specification of the Zadeh fuzzy logic

<table>
<thead>
<tr>
<th>Notation</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>t-norm</td>
<td>$\alpha \otimes \beta$</td>
</tr>
<tr>
<td>t-conorm</td>
<td>$\alpha \oplus \beta$</td>
</tr>
<tr>
<td>negation</td>
<td>$\ominus \alpha$</td>
</tr>
<tr>
<td>implication</td>
<td>$\alpha \Rightarrow \beta$</td>
</tr>
</tbody>
</table>

or disjointness) and operations (inverse, composition) can be trivially extended to the fuzzy case.

Fuzzy modifiers (or hedges) apply to fuzzy sets to change their membership function. For instance, given a fuzzy set Tall we may want to define the fuzzy set of very Tall people by using an appropriate modifier. Formally, a modifier is a function $f_m: \mathcal{N} \rightarrow \mathcal{N}$. Examples of modifiers are very, more or less, and slightly. They can be defined by using different types of membership functions; e.g., triangular functions (Figure 1.b).

Eventually, changing the usual true/false convention leads to a new type of logical propositions, called fuzzy propositions. Each fuzzy proposition may have a degree of truth in $\mathcal{N}$, denoting the compatibility of the fuzzy proposition with a given state of facts. For example, $x$ is a big window $\geq 0.8$ says that we have a fairly big window (the degree of truth of $x$ being a big window is at least 0.8).

2.2.2. Fuzzy Description Logics

In this work we have considered a fuzzy extension of the SROIQ(D) DL, which in turn corresponds to a fuzzy extension of the OWL 2 language. This fuzzy DL, presented in [33] and [34], is a subset of the logic supported by the fuzzy reasoning engine DeLorean used in the next section.

We summarize in Table 2 the main concept and role constructors of the fuzzy SROIQ(D). It can be seen that they have a direct correspondence with their crisp counterparts. The syntax is equivalent to the language accepted by DeLorean, which in turn is a variation of the Manchester syntax for OWL 2 in a more functional-programming style. We purposely do not include the formal semantics of these expressions, which can be found in the previously mentioned works. The notation used in Table 2 is the following:

- $a, b$ are individuals
- $A$ is an atomic (simple) concept and $C, C_1, \ldots$ are (possible complex) concepts
- $R$ is a fuzzy role
- $M$ is a strictly positive natural number and $N$ is a natural number

---

3Strictly speaking, there are certain syntactic restrictions in some expressions regarding the use of roles, but we omit here the details.
<table>
<thead>
<tr>
<th>Notation</th>
<th>Constructor</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Concept constructors</strong></td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>Atomic concept</td>
</tr>
<tr>
<td>top</td>
<td>Universal concept</td>
</tr>
<tr>
<td>bottom</td>
<td>Empty concept</td>
</tr>
<tr>
<td>(and C1 ... Cn)</td>
<td>Concept conjunction</td>
</tr>
<tr>
<td>(or C1 ... Cn)</td>
<td>Concept disjunction</td>
</tr>
<tr>
<td>(not C)</td>
<td>Concept negation</td>
</tr>
<tr>
<td>(some R {C</td>
<td>d})</td>
</tr>
<tr>
<td>(all R {C</td>
<td>d})</td>
</tr>
<tr>
<td>(one-of a [D])</td>
<td>Nominal concept</td>
</tr>
<tr>
<td>(self R)</td>
<td>Local reflexivity</td>
</tr>
<tr>
<td>(at-least M R {C</td>
<td>d})</td>
</tr>
<tr>
<td>(at-most N R {C</td>
<td>d})</td>
</tr>
<tr>
<td>(triangular D1 D2 D3 C)</td>
<td>Fuzzy modified concept</td>
</tr>
<tr>
<td><strong>Role constructors</strong></td>
<td></td>
</tr>
<tr>
<td>R</td>
<td>Atomic role</td>
</tr>
<tr>
<td>inv</td>
<td>Inverse role</td>
</tr>
<tr>
<td>top-role</td>
<td>Universal role</td>
</tr>
</tbody>
</table>

- d is a fuzzy datatype, defined with the syntax \((\text{trapezoidal } q_1 q_2 q_3 q_4)\) (q1, q2, ... are real values, as depicted in Figure 1)
- D, D1, ... are degrees in the finite chain \(\mathcal{N}\)
  - In fuzzy nominals, D is in \(\{\gamma_1, \ldots, \gamma_p\}\). If omitted, \(\gamma_p\) is assumed.
  - In axioms, D is in \(\{\gamma_1, \ldots, \gamma_p\}\) if preceded by \(\geq\), or in \(\{\gamma_0, \ldots, \gamma_{p-1}\}\) if preceded by \(\leq\).
- | denotes alternative sub-expressions
- [ ] denotes optional sub-expressions
- { } denotes grouping to make precedence explicit

A fuzzy Knowledge Base \((f\mathcal{K})\), or just a fuzzy ontology, is therefore composed by a finite set of fuzzy axioms. The axioms are grouped into a fuzzy ABox describing individuals, a fuzzy TBox describing concepts, and a fuzzy RBox describing roles. Table 3 shows the axiom constructors of the fuzzy DL \(\mathcal{SROIQ}(D)\). Note that the bounds for the degrees are optional, so if \(\geq D\) is omitted, \(\geq \gamma_p\) is assumed.

Examples of the use of the most relevant fuzzy representation primitives are offered in Section 3. We do not use all the primitives listed in Table 2 but they are included for the sake of completeness. Particularly, all, one-of and self are not considered. These would be some typical (simplified) examples of their use:
Table 3: Axioms in fuzzy \( SROIQ(D) \)

<table>
<thead>
<tr>
<th>Notation</th>
<th>Axiom</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Instance axioms (ABox)</strong></td>
<td></td>
</tr>
<tr>
<td>A1</td>
<td>(instance a C ([&gt;= D</td>
</tr>
<tr>
<td>A2</td>
<td>(related a b R ([&gt;= D</td>
</tr>
<tr>
<td>A3</td>
<td>(not-related a b R ([&gt;= D</td>
</tr>
<tr>
<td>A4</td>
<td>(same-as a b) a and b represent the same individual</td>
</tr>
<tr>
<td>A5</td>
<td>(different-to a b) a and b should be interpreted as different individuals</td>
</tr>
<tr>
<td><strong>Concept axioms (TBox)</strong></td>
<td></td>
</tr>
<tr>
<td>C1</td>
<td>(implies-concept C1 C2 ([&gt;= D])) C1 is a subclass of C2 with a degree (&gt;= D)</td>
</tr>
<tr>
<td>C2</td>
<td>(equivalent-concepts C1 C2) C1 is equivalent to C2</td>
</tr>
<tr>
<td>C3</td>
<td>(disjoint-concepts C1 .... Cn) C1 ... Cn are mutually disjoint</td>
</tr>
<tr>
<td>C4</td>
<td>(crisp-concept C) C is crisp</td>
</tr>
<tr>
<td><strong>Role axioms (RBox)</strong></td>
<td></td>
</tr>
<tr>
<td>R1</td>
<td>(implies-role R1 R2 ... Rn R ([&gt;= D])) (R) subsumes the role chain (R1 ... Rn) with a degree (&gt;= D)</td>
</tr>
<tr>
<td>R2</td>
<td>(equivalent-roles R1 R2) (R1) is equivalent to (R2)</td>
</tr>
<tr>
<td>R3</td>
<td>(inverse R1 R2) (R1) is inverse to (R2)</td>
</tr>
<tr>
<td>R4</td>
<td>(domain R C) Equivalent to the axiom (implies-concept (some R top) C)</td>
</tr>
<tr>
<td>R5</td>
<td>(range R C) Equivalent to the axiom (implies-concept top (all R C))</td>
</tr>
<tr>
<td>R6</td>
<td>(functional R) (R) is functional</td>
</tr>
<tr>
<td>R7</td>
<td>(inverse-functional R) The inverse of (R) is functional</td>
</tr>
<tr>
<td>R8</td>
<td>(transitive R) (R) is transitive</td>
</tr>
<tr>
<td>R9</td>
<td>(symmetric R) (R) is symmetric</td>
</tr>
<tr>
<td>R10</td>
<td>(asymmetric R) (R) is asymmetric</td>
</tr>
<tr>
<td>R11</td>
<td>(reflexive R) (R) is reflexive</td>
</tr>
<tr>
<td>R12</td>
<td>(irreflexive R) (R) is irreflexive</td>
</tr>
<tr>
<td>R13</td>
<td>(disjoint-roles R1 .... Rn) Roles (R1 ... Rn) are mutually disjoint</td>
</tr>
<tr>
<td>R14</td>
<td>(crisp-role R) (R) is crisp</td>
</tr>
</tbody>
</table>

- \((\text{and Wall (all include Window)})\) denotes walls only including windows
- \(\text{(at-least 1 hasRelatedMaterial (one-of concrete paper mortar))}\) denotes building elements having at least one material of the set \{concrete paper mortar\}
- \(\text{(self powers)}\) denotes auto-powered building equipment (devices that provide power to themselves)

### 2.2.3. Reasoning with fuzzy ontologies

In fuzzy DLs there are many reasoning tasks involving the axioms of the knowledge base. Some of them are extensions of the reasoning tasks of DLs, whilst those concerning degrees are specific of fuzzy DLs. Usually, reasoning tasks can be reduced to \(fK\) consistency [33]. The most interesting ones are informally described below. Notice that in all cases, the elements that are used as input to the reasoning task may not be explicitly included in the ontology.
• Fuzzy knowledge base consistency (or satisfiability): check if all the axioms in the knowledge can be satisfied; i.e., they do not contradict.

• Concept satisfiability: check if a given concept does not correspond to an empty set of instances.

• Entailment: check if a given axiom is entailed by the explicit axioms of the knowledge base.

• Concept subsumption: check if there is a subsumption relation between two given concepts.

• Instance retrieval: retrieve all the instances of a given concept (optionally, with a minimum degree).

• Best degree bound (bdb): get the maximum degree to which an axiom holds.

• Maximal concept satisfiability degree: get the maximum degree to which an individual can belong to a given concept.

As in the crisp case, reasoning with fuzzy ontologies is performed with reasoning engines. The two most prominent fuzzy engines are fuzzyDL\(^4\) and the previously mentioned DeLorean (DEscription LOgic REasoner with vAgueNESS\(^5\)\(^6\)). Both can be freely used and support expressive fuzzy ontology languages. The main difference between them is that they apply distinct strategies to carry out the reasoning process. fuzzyDL implements a mixture of tableau algorithms and a MILP optimization problem, whereas DeLorean implements a reduction procedure that transform a fuzzy ontology into a crisp ontology that can be processed by any non-fuzzy DL reasoning engine. In terms of efficiency, fuzzyDL includes several optimizations to reduce the time required for the most common reasoning tasks\(^7\), whereas DeLorean exploits the cases in which recomputing the reduction of the fuzzy ontology is not necessary, and the capability of using different crisp reasoners under the hood. Given our scenario, in which a non-fuzzy ontology is already available, using DeLorean for fuzzy ontology reasoning is an appropriate choice.

3. Results and discussion

3.1. Fuzzy ontology constructors and axioms in the BIM

To illustrate the use of the representation primitives introduced in Tables\(^2\) and \(^3\) in the construction domain, in this section we describe some examples based on the model files provided with the IFC-to-RDF tool. To facilitate the reproduction of the examples while maintaining simplicity, we have used

\(^4\)http://webdiis.unizar.es/~fbobillo/fuzzyDL

\(^5\)http://webdiis.unizar.es/~fbobillo/delorean
a sample IFC file that describes a model of a wall with an opening section corresponding to a window. Notice however that the presented examples are valid in any model regardless of its size and complexity, since constructors and axioms are applied exactly in the same way. Detailed use cases in a wider BIM context are described in Section 3.2.

The ifcOWL and the ifcRDF files resulting from the conversion with the ‘OWL 2 EL – RDF List’ options have been slightly adapted to make the explanations more readable. The sample model, which is depicted in Figure 2, can be also obtained from the authors’ web page. The fuzzy extension has been created with the Protégé ontology editor and the Fuzzy OWL 2 language plugin, which supports exporting to the native format of the DeLorean reasoner. The complete process is shown in Figure 3. The interested reader can find more details on the practical use of DeLorean in [14].

In the remainder of the section, we will use the DeLorean syntax. We will note the terms defined in the original ifcOWL ontology with the prefix ifc, and our additions with an empty prefix (:). The examples are built incrementally from the initial model in such a way that, if not explicitly stated, the additions in former examples apply in later examples. The final file with the fuzzy ontology

---

6 [http://ugritlab.ugr.es/r/ifc/sample-model-EL-RDFList.ttl](http://ugritlab.ugr.es/r/ifc/sample-model-EL-RDFList.ttl)
7 [http://protege.stanford.edu](http://protege.stanford.edu)
in *DeLorean* format, including all the examples, can be also found at the authors’ web page[9].

### 3.1.1. Fuzzy concept assertions

As already mentioned, in a crisp ontology, instances either belong to a class or not. In a fuzzy ontology, it is possible to assert that an instance belongs to a class to a degree. This degree can be an unrestricted real value in \([0, 1]\), or as in our case, a value in a finite chain \(\mathcal{N}\). This kind of axiom is called a fuzzy concept assertion, and is noted with *instance*, as shown in Table 3 (A1). We will assume a finite chain \(\mathcal{N} = \{0, 0.2, 0.4, 0.6, 0.8, 1\}.

**Example 1.** If we consider the class *IfcMaterial* as a fuzzy concept, we can add a new instance representing “paper” that can be only partially considered a material. (Note that *IfcMaterial* already has an individual, *material_1*, as depicted in Figure 2.)

\[
\begin{align*}
(\text{instance} : \text{material}_2 \text{ ifc:IfcMaterial} & \geq 0.8) \\
(\text{related} : \text{material}_2 \text{ "PAPER" ifc:name_of_IfcMaterial})
\end{align*}
\]

From this point on, we can retrieve those instances that can be considered materials at least to a degree. Following the example, we can ask the reasoner to query all the instances of *IfcMaterial* with degree \(\geq 0.6\), which will trivially include *material_1* (degree 1) and *material_2* (degree 0.8). Many other fuzzy concept assertions could be added to denote elements that, for any reason, we do not want to fully belong to a type, or we do not know exactly: additional materials, spaces, equipment, etc.

Furthermore, we can use more complex concept expressions. For example, we can ask the reasoner to retrieve all the instances of the intersection class *IfcMaterial* and *IfcMaterialLayer*. Let us suppose that we have also asserted:

\[
\begin{align*}
(\text{instance} : \text{material}_2 \text{ ifc:IfcMaterialLayer} & \geq 0.6)
\end{align*}
\]

Therefore, only \texttt{material\_2} would be inferred as an instance of the intersection class. The degree of this result would be the (Gödel) conjunction of the degrees of belonging to \texttt{IfcMaterial} (0.8) and to \texttt{IfcMaterialLayer} (0.6); i.e., \( \min(0.8, 0.6) = 0.6 \).

\subsection{3.1.2. Fuzzy role assertions}

Fuzzy roles denote fuzzy relations between ontology instances. A fuzzy role assertion materializes an association between two instances that holds to a degree. To create a fuzzy role assertion, we use the axiom A2. Similarly, we can use axiom A3 to denote that two instances are not related to a degree. Same-as and different-to are special relations represented with the axioms A4 and A5, respectively. Typical fuzzy DLs, and particularly our fuzzy SROIQ\((D)\), do not fuzzify these two relations. Therefore, if we want to model that two individuals are similar, usually a new similarity relation is defined.

\textbf{Example 2.} A new fuzzy role has been defined in the ontology to relate the similarity degree between two building materials, namely the \texttt{similar\_to\_IfcMaterial} object property. This property can be defined as symmetric (R9), because it holds in both directions (with the same degree), and transitive (R8). By extension, it would be possible to define other features of the property with the axioms R3-R14: reflexive, irreflexive, functional, etc. Let us also suppose that we have in the fuzzy ontology additional instances of \texttt{IfcMaterial} representing ‘mortar’ and ‘ecologic mortar’ materials. We can now assert that ‘concrete’ is quite similar to ‘mortar’, but ‘mortar’ is only moderately similar to ‘ecologic mortar’.

\begin{verbatim}
( instance   :material\_3 ifc:IfcMaterial)
( related :material\_3 "MORTAR"

   ifc:name_of_IfcMaterial)

( instance   :material\_4 ifc:IfcMaterial)
( related :material\_4 "ECOLOGIC MORTAR"

   ifc:name_of_IfcMaterial)

( symmetric :similar\_to\_IfcMaterial)
( transitive :similar\_to\_IfcMaterial)

( related :material\_1 :material\_3

   similar\_to\_IfcMaterial >= 0.8)

( related :material\_3 :material\_4

   similar\_to\_IfcMaterial >= 0.6)

A possible query would be to retrieve the materials that are quite similar to “concrete”; i.e., those belonging to the following class with degree \( \geq 0.6 \):

\begin{verbatim}
(some :similar\_to\_IfcMaterial

   (value ifc:name "CONCRETE"))
\end{verbatim}

Since the \texttt{similar\_to\_IfcMaterial} property is symmetric and transitive, the query returns the results \texttt{material\_1 (1), material\_3 (0.8) and material\_4 (0.6 =}
min(0.8, 0.6), due to the propagation of the transitivity with the Gödel conjunction operation.)

An extension of this query would be to obtain all the building elements that are built with materials somehow similar to ‘mortar’. These individuals would be the instances of the following complex class filtered by degree $\geq 0.2$:

$$(\text{and}$$

$$(\text{ifc:IfcBuildingElement}$$

$$(\text{some inv ifc:relatedObjects_of_IfcRelAssociates}$$

$$(\text{some ifc:relatingMaterial}$$

$$(\text{and}$$

$$(\text{ifc:IfcMaterial}$$

$$(\text{some :similar_to_IfcMaterial}$$

$$(\text{value ifc:name "MORTAR"})))))$$

The result of the query would be wall_1 (0.8), but we can imagine that in a larger model it would give less evident outputs.

3.1.3. Fuzzy general concept inclusions

General concept inclusion axioms (GCIs) in crisp DLs, noted $C_1 \sqsubseteq C_2$, represent the notion that any instance of the class $C_1$ is also an instance of class $C_2$. GCIs are often read as $C_2$ subsumes $C_1$. A fuzzy GCI, represented with the $\text{implies-concept}$ axiom (C1), states that $C_1$ is a subconcept of $C_2$ to degree at least $D$. As expected, $C_1$ and $C_2$ may be complex concepts; i.e., concepts denoted by complex expressions built with the constructors enumerated in Table 2. Based on the idea of GCI, the logic also allows other restrictions such as concept equivalency (C2) and disjointness (C3). The corresponding primitive of GCIs for roles are role subsumption axioms (R1), which can use chains of roles in the right part of the subsumption.

Example 3. Let us suppose that a taxonomy of materials is added to the model. This taxonomy includes several material types divided in two main categories, artificial and natural, which establish a non-strict partition of IfcMaterial.

$$\text{(equivalent-concepts}$$

$$\text{(or :ArtificialIfcMaterial}$$

$$\text{:NaturalIfcMaterial})$$

$$\text{ifc:IfcMaterial})$$

$$\text{(implies-concept :CementBasedIfcMaterial}$$

$$\text{:IfcArtificialMaterial})$$

$$\text{(implies-concept :ConcreteIfcMaterial}$$

$$\text{:CementBasedIfcMaterial})$$

$$\text{(implies-concept :MortarIfcMaterial}$$

$$\text{:CementBasedIfcMaterial})$$

$$\text{(implies-concept :VegetalIfcMaterial}$$

$$\text{:NaturalIfcMaterial})$$
However, a large amount of building materials are part natural, because their raw components, and part artificial, because they are processed, refined, or mixed with other components. This context can modeled with an imprecise taxonomy created by using fuzzy GCIs. For example, we can consider that ‘plywood’ is, to some extent, a vegetal material, because it is mainly made of wood, but also an artificial material, because it is glued with chemical products. Similar fuzzy GCIs can be used to define materials like ‘glass’ or ‘reinforced concrete’:

\[
(\text{implies-concept} :\text{PlywoodIfcMaterial} :\text{ArtificialMaterial} \geq 0.8) \\
(\text{implies-concept} :\text{PlywoodIfcMaterial} :\text{VegetalMaterial} \geq 0.6) \\
(\text{implies-concept} :\text{GlassBasedIfcMaterial} :\text{ArtificialMaterial} \geq 0.6) \\
(\text{implies-concept} :\text{GlassBasedIfcMaterial} :\text{MineralMaterial} \geq 0.8) \\
(\text{implies-concept} :\text{ReinforcedConcreteIfcMaterial} :\text{ConcreteIfcMaterial} \geq 0.8)
\]

Based on this fuzzy taxonomy, we can assign our material instances to classes by using (fuzzy) concept assertions (replacing those defined in the original model and Examples 1-2):

\[
(\text{instance material}_1 \text{ ConcreteIfcMaterial} \geq 1.0) \\
(\text{instance material}_2 \text{ PaperIfcMaterial} \geq 1.0) \\
(\text{instance material}_3 \text{ MortarIfcMaterial} \geq 1.0) \\
(\text{instance material}_4 \text{ MortarIfcMaterial} \geq 1.0) \\
(\text{instance material}_5 \text{ GlassBasedIfcMaterial} \geq 0.8)
\]

Afterwards, we can query the model to retrieve \text{ArtificialMaterials} with degree $\geq 1$. The results would be: \text{material}_1, \text{material}_3, \text{and} \text{material}_4 (degree propagation is calculated with the Kleene-Dienes implication function). Note that \text{material}_5 is not in the result list because it belongs to \text{ArtificialMaterial} only with degree 0.8.

More interestingly, we can obtain all the elements of the building built with artificial materials by extending the query concept; in our case, \text{wall}_1:
Moreover, we can query the elements built with materials similar to artificial materials (in our case, wall_1) by reusing the fuzzy similarity property defined in Section 3.1.2:

(\text{and} \text{IfcElement} \text{some inv IfcRelatedObjects_of_IfcRelAssociates} \text{some IfcRelatingMaterial} \text{ ArtificialMaterial}))

Trivially, the queries can be adapted to other material types, and by extension, to any other fuzzy taxonomy integrated into the model. It is worth to highlight than fuzzy taxonomies can coexist with crisp taxonomies, thus providing high representation flexibility while avoiding the user to change the whole model.

3.1.4. Fuzzy datatypes

Fuzzy datatypes are the natural extension of crisp datatypes, since they allow imprecise statements over a concrete domain. The truth value of a datatype predicate is given by a (discretized) function, which is typically a trapezoidal function like the one depicted in Figure 1a. Fuzzy data types can be used in several concept expressions, as noted in Table 2 with d.

Example 4. Let us define two new classes based on fuzzy datatypes: a “high height window” and a “wide width window”. A high height window is a window that has some “high” height value. The fuzzy notion of “high” is characterized by a trapezoidal function, which will calculate the “degree of being high” for any given real value. Similarly, we use another trapezoidal function to characterize the degree of being wide for any given real value:

(\Rightarrow-rate-concept :High_IfcWindow
 (\text{and} \text{IfcWindow} \text{some IfcOverallHeight_of_IfcWindow} \text{trapezoidal 1.2 1.7 10 10}) )

(\Rightarrow-rate-concept :Wide_IfcWindow
 (\text{and} \text{IfcWindow} \text{some IfcOverallWidthof_IfcWindow} \text{trapezoidal 0.8 1.3 10 10}) )

Now, it is possible to query all the big windows of the model, being them those instances of the class (\text{and} High_IfcWindow Wide_IfcWindow). Since in the
sample file the window size is \((width \times height) = (1.0 \times 1.5)\), issuing such query to the reasoner would give as a result \(\text{window}_1\) (with degree \(0.4 = \min\{0.4, 0.6\}\)). To obtain this result, we use the Gödel conjunction to apply the and operator over the membership values calculated with the trapezoidal fuzzy datatype (Figure 4). If necessary, these values would be discretized to fit the ones in \(N\).

We can easily modify the query to work with building elements other than windows, or to add more conditions to restrict ourselves to elements located inside a certain level, having a relation to other elements, with a given property, etc.

### 3.1.5. Fuzzy modifiers

Fuzzy modifiers are used to change the meaning of a fuzzy concept by modulating its membership function. Roughly, a fuzzy modified concept is therefore a variation of the original concept that assigns slightly different membership degrees to its instances. Our fuzzy SROIQ(D) allows the use of the fuzzy modifier triangular, defined by a function like the one presented in Figure 1.b.

**Example 5.** Based on the class \(\text{High\_IfcWindow}\) created in Example 4, we can define the class of windows with very high height with the expression:

\[
(\text{implies-concept} : \text{Very\_High\_IfcWindow} \\
(\text{triangular} 0.4 1 1 : \text{High\_IfcWindow}))
\]

In consequence, \(\text{window}_1\) is a \(\text{Very\_High\_IfcWindow}\) with degree 0.2, resulting from the modification and discretization to \(N\) value of the \(\text{High\_Ifc}\) membership value with the triangular function (Figure 5). Naturally, it is also possible to define more complex concepts; e.g., to select building spaces with very high windows.

### 3.2. Use cases for the fuzzy semantic BIM

The previous section describes the basic building blocks of a fuzzy ontology, and explains how they can be used for imprecise building information modeling.
and retrieval. The fuzzy ontology framework precisely defines the semantics of these components and how they can be combined, letting the user to use them at his/her convenience. In this section, we elaborate on different use cases that show how the fuzzy semantic BIM can address common problems appearing during the building life-cycle that cannot be solved with a non-fuzzy BIM.

3.2.1. Cross-domain knowledge linking

Fuzzy general concept inclusions (Section 3.1.3) are the backbone of a fuzzy ontology. Similarly to the crisp case, fuzzy GCIs trigger the most interesting inferences, because we can use complex concept constructors in the left and the right part of these axioms (e.g. Example 5). Essentially, fuzzy GCIs allow us to define imprecise concept inclusions, in such a way that we can quantify the degree of overlap between two classes of individuals, and then conveniently operate with this degree within the Fuzzy Logic framework. As in the crisp case, fuzzy GCIs can be multi-dimensional, in the sense that multiple hierarchies can be defined relating the same concepts.

One direct application of fuzzy GCIs is the integration of heterogeneous building entity taxonomies. Let us suppose that we have two different catalogues of materials supplied by two different contractors. Usually, we cannot expect that both catalogues will use the same names and codes for the material types. Moreover, it may happen that the material types in the catalogues do not directly correspond to the ones used in the BIM. This situation requires developing a mapping between the taxonomies to establish a semantic link between related concepts. The mapping would support to automatize calculations such as the construction cost of a part of the modeled building using materials from a selected contractor –this can be implemented based on an ontology query that returns the price of the materials matching the ones in the BIM, similar to the one shown at the end of Section 2.1, and then multiplies them by their area of use. In the crisp setup, the mappings between material types are binary (yes/no), whereas with the fuzzy BIM, it is possible to establish a degree of similarity between material types (as in Example 3). Therefore, with our proposal, the query can be expanded to material types similar to the ones actually selected in the BIM at least to a degree, or even implement a cost
calculation that increases the price of a piece of material depending on this similarity value. Naturally, this approach can be extended to map other building element classifications; for example, those defined in the International Building Code [39].

From a broader perspective, the fuzzy semantic BIM increases the knowledge integration features provided by the Semantic Web technologies, thus improving interoperation, a typical problem in the architecture, engineering and construction industry [40]. In general, cross-domain and cross-cultural knowledge can be more flexibly incorporated and managed in the building knowledge base. Following the previous example, we can see that compound domain-specific fuzzy concepts can be now formally defined and linked. One application scenario of this is creating a vocabulary to facilitate the exchange of information between the stakeholders involved in the construction process (designers, constructors, facility managers, construction workers, etc.) Some of these actors are not expected to use the IFC conventions, and therefore, it can be very helpful to formally define their daily-use concepts in standard terms. The fuzzy approach allows us to work with imprecise definitions, which are more appropriate in several cases. For instance, a concept like room occupancy, which is important for energy facility managers, is better expressed with fuzzy values (high, normal, etc.), similar to the ones described in Section 3.1.4. From this representation, it is possible to retrieve information from the BIM and perform additional calculations by aggregating data; e.g., to calculate the approximated expected energy consumption from the specification of individual room equipment, load predictions, and occupancy profiles. Interestingly enough, all the parameters of this kind of what if scenarios do not need to be exactly assigned. Other scenario is the classification of buildings or building components according to their features. For example, we can first define imprecise end-user concepts from IFC elements (a breezeway, a dining room), and then create a flexible building description (a dogtrot house typically has one story, a breezeway, and at least two rooms about 20 feet wide) readily-available for BIM data retrieval and exchange.

3.2.2. Imprecise BIM queries

In the explanation above, we have implicitly assumed that the creation of concept definitions makes it possible to retrieve the instances of these new concepts. That means that, for example, if we define the (fuzzy) concept ‘dogtrot house’, we can query a semantic BIM by using a (fuzzy) reasoning engine to retrieve the instances of this concept; i.e., if there is only one building represented in the model, the building instance itself (with a fulfillment degree) or an empty set. In this section, we study in more detail the query capabilities of the fuzzy semantic BIM, which provides an open, formal language for the definition of imprecise sophisticated queries, relaxed constraint checks, and partial model filters.

In the literature, we can find some proposals aimed at the creation of non-proprietary BIM query languages. ifcRDF and ifcOWL are contributions in that regard, since the resulting models can be loaded in a triplestore or a reasoning engine, and handled with standard SPARQL and OWL queries. Simi-
larly, BIMQL is a language for generic querying of IFC-based BIM models [41],
supporting free-variable queries and model updates, among other features. Our
fuzzy ontology model extends the capabilities of these approaches by allowing
imprecise knowledge retrieval. A straightforward example of a fuzzy query has
been presented in Example 4 in which we define a fuzzy concept based on a
fuzzy datatype value, and then retrieve the model instances belonging to this
concept, as well as their membership degree. In general, fuzzy role assertions
and fuzzy datatypes enable storing imprecise information (e.g. a fuzzy dimen-
sion value), and formulating imprecise queries over precise and imprecise data
(e.g. retrieve elements with dimension values within an imprecise range). This
kind of queries can be used to obtain a filtered view of specific building ele-
ments, with the advantage that the condition is more flexible and the results
are ordered by the degree of fulfillment of the condition.

Fuzzy role assertions can be also applied to create imprecise geometric prop-
erties. For instance, it may be interesting to define a fuzzy property representing
how close two arbitrary building elements are, similar to the one in Example 2.
The degree of such relation can be automatically computed with a distance mea-
sure from the building geometry, and then stored in the model as fuzzy data
value. Notice that this would require to calculate pair-wise distances between
all building elements, which may be computationally expensive and require ad-
ditional optimizations. Other interesting improvement can be the combination
with RCC (Region Connection Calculus) predicates, which allow symbolic rep-
resentation and reasoning with topological relations [42]. RCC is not directly
supported in crisp OWL, but still, it is possible to create axioms to model
common relations, like tangent, overlap, disjoint, etc., and instantiate them by
relying on an external module [43]. By extension, fuzzy ontology axioms can
be used to formally define soft positioning constraints; e.g., small overlapping,
approximately tangent, etc. If basic fuzzy axioms are combined to other fuzzy
property and concept constructors, we can represent for instance that there are
several objects in the surroundings of a big element. If these axioms are con-
tradictory to others in the BIM, the reasoning engine will infer that the model
is inconsistent, which can be useful for the detection of clashes. In contrast to
the crisp approach, in the fuzzy case we can impose a threshold degree, and
therefore relax these restrictions at convenience. Last but not least, specific
extensions of fuzzy DLs could be exploited to natively support fuzzy geometric
reasoning in the BIM [44].

3.2.3. Fuzzy parametric modeling

The notion of soft positioning constraint presented before is related to that
of parametric modeling in normal BIMs. Parametric models are based on para-
metric components, which are virtual BIM elements that have associated a
range of possible values to properties (a numeric interval for a dimension, a set
of colors for a solid element) rather than a fixed value. Accordingly, paramet-
ric components are usually provided as templates, sometimes by third-parties,
to be instantiated in the BIM. This allows the designer to reuse components,
and even to place parametric components in the model and let the software
to optimize the parameter values according to the requirements imposed by
the non-parametric components. The latter requires solving an optimization
problem expressed as a system of linear inequalities.

As introduced in the previous section, some kinds of fuzzy axioms express
semantics similar to parametric constraints. In particular, a fuzzy role is not
very different to a parametric property: it also defines a plausible value range,
although all the values may not totally satisfy the property. In this regard, we
can rely on concepts similar to the ones in Example 4 to define fuzzy axioms
establishing that a building story must include a big room and two small rooms,
and the small rooms must be very close. Given an instantiation of these con-
cepts and relations, via crisp or fuzzy position and dimension values, we can test
whether the constraints are satisfied or not by testing if the model is consistent.

Similarly, finding a model that maximizes the degree of fulfillment of the fuzzy
axioms representing the constraints roughly corresponds to finding the best de-
gree bound of these axioms. The fuzzy approach has important advantages
compared to the crisp approach, because relaxing the constraint to a degree
lesser than 1 allows reducing the problem of conflicting parameters in overcon-
strained models. Unfortunately, only a limited set of constraints can be modeled
with fuzzy axioms, and in the best case, they require intensive geometric cal-
culations that are not directly performed within the ontology. For instance, in
the previous example we cannot find the best room arrangement among all the
possible positioning alternatives by only using the fuzzy reasoner. Therefore,
we consider that the fuzzy BIM can be helpful to encode imprecise constraints,
but solving the optimization problem would require the implementation of an
external fuzzy constraint satisfaction algorithm [45]. Nevertheless, it is worth
to mention that the fuzzyDL reasoner provides limited support for systems of
fuzzy inequalities. Exploring these capabilities is an interesting direction for
future work.

3.3. Discussion

The previous examples and use cases show that the proposed fuzzy extension
notably increases the capabilities of the semantic BIM, and provides support for
much more expressive representations. Queries retrieve not only data explicitly
asserted in the model, but also information automatically inferred by a logic-
based reasoning process. Fuzzy DLs offer a sound theoretical framework for
such fuzzy ontologies, and the availability of languages and tools facilitates the
implementation of this kind of solutions. We have used a fuzzy version of the
\textit{SROIQ(D)} DL, but it would be possible to change the underlying logic without
any loss of generality. Actually, other even more expressive logics, including
supplementary representation primitives targeted at particular needs of BIM
users could be considered. Some of them have been mentioned through the
paper: spatial reasoning, alternative families of fuzzy logic operators, constraint
checking, etc.

However, adding additional representation layers to the semantic BIM comes
with a cost. Expressive ontologies and fuzzy ontologies are known by their
inferior performance, which can be a serious drawback when working with larger
models. To test the scalability of our proposal, we have executed the same example queries of Section 3.1 with the duplex apartment model used in the 2009 COBie Challenge[^10]. These tests have proved that efficiency quickly degrades with such realistic models, even to the point of making them unusable because of the high time needed to solve relatively simple queries (over 1 minute for examples 4 and 5). We have to mention, however, that no specific optimization strategies were applied to the fuzzy inference procedure.

One possible solution to this issue would be to restrict the semantic BIM to a tractable OWL profile. As a matter of fact, this was the purpose of using the EL strategy of the IFC-to-RDF tool, which produces a slightly less expressive but more efficient model. Due to its internal design, if the fuzzy ontology used in DeLorean is based on OWL 2 EL, a DL engine optimized for this subset of OWL can be used in the last reasoning step (e.g., ELK[^46]). Nevertheless, the output produced by IFC-to-RDF is not a strict OWL 2 EL ontology (e.g., class unions are generated). This also indicates that a more precise characterization of the conversion procedure could be useful to make better use of the features of available techniques and tools.

Other solutions specifically aimed at improving the performance of the reasoner itself would be to pre-calculate some common queries when a model is firstly created, such as the concept hierarchy or the membership values of selected fuzzy datatypes. Since we may expect several queries concerning the geometric properties of building elements, the latter may have a notable impact in the efficiency. In addition, DeLorean can pre-calculate the reduction from the fuzzy ontology to the crisp ontology, which remains unchanged if no new instances are added (as it may be expected after the initial model creation stage). This is consistent with the foreseen use of our proposal. As it can be concluded from the examples, the fuzzy extension is likely to be more useful in the analysis and decision stages of a project.

### 4. Conclusions and future work

In this paper, we have presented a fuzzy logic-based extension of the semantic BIM proposed within the Building Linked Data research initiative. Our proposal allows imprecise data linking and querying to ontological building models within the well-defined theoretical framework of fuzzy Description Logics. The resulting models support very expressive imprecise queries, which offer new ways to retrieve information not available in the current systems. Unfortunately, fuzzy inference engines are experimental tools, and therefore, they present some drawbacks in terms of efficiency (high execution time) and ease of use (the only supporting tool to facilitate the creation of a fuzzy model is the Fuzzy OWL Protégé plugin and API[^14], which is not suitable for medium-scale ontologies). However, the initial results are promising and useful in different scenarios, and several optimizations can be applied in the near future.

Besides these improvements, a prospective research direction is to continue the development of more complex use cases with the collaboration of BIM users. This will help to determine the most appropriate fuzzy DL to be used, and to clarify the role of the fuzzy queries within the users’ workflow. Necessarily, this will require the implementation of appropriate interfaces to facilitate the interaction with the system to users without previous experience in ontological modeling.

Acknowledgements

This research work has been partially funded by the Spanish Ministry of Economy and Competitiveness under project TIN2012-30939, and by the European Union in the 7th Framework Programme under project Energy IN TIME EeB.NMP.2013-4 (grant agreement no. 608981). The authors have used the IFC-to-RDF tool created by Seppo Törnä and Nam Vu Hoang, and the Protégé resource, which is supported by grant GM10331601 from the National Institute of General Medical Sciences of the United States National Institutes of Health.

References


