

SWRL-F - A Fuzzy Logic Extension of the Semantic Web Rule Language

Tomasz Wiktor Włodarczyk, Chunming Rong
University of Stavanger, Norway
tomasz.w.wlodarczyk@uis.no

Marting O'Connor, Mark Musen
Stanford University, USA

ABSTRACT

Enhancing Semantic Web technologies with ability to express uncertainty and imprecision is widely discussed topic. While SWRL can provide additional expressivity to OWL-based ontologies, it does not provide any way to handle uncertainty or imprecision. There is a pressing need to provide a standard-based, simple and functioning solution. We describe an extension of SWRL called SWRL-F that we believe can provide such a solution. SWRL-F is based on SWRL rule language and uses SWRL's strong semantic foundation as its formal underpinning. We extend it with a SWRL-F ontology to enable fuzzy reasoning in the rule base basing on the fuzzy control systems approach. The resulting language provides small but powerful set of fuzzy operations that do not introduce inconsistencies in the host ontology. We present it basing on the example of risk assessment in oil and gas industry which was a driving use case for this project.

Categories and Subject Descriptors

I.2.4 [Computing Methodologies]: Knowledge Representation Formalisms and Methods – *Representations (procedural and rule-based)*.

General Terms

Algorithms, Documentation, Design, Experimentation, Standardization, Languages.

Keywords

SWRL, SWRL-F, fuzzy logic, fuzzy rules, fuzzy, rule language, risk.

1. INTRODUCTION

This is an extended version of a paper with the same title published as a position paper at 6th International Workshop on Uncertainty Reasoning for the Semantic Web (URSW), held as part of the 9th International Semantic Web Conference (ISWC) in October of 2010 in Shanghai, China.

Fuzzy Logic (FL) has many applications in engineering and other domains. It provides a way to express imprecise information and helps in simplifying knowledge representation. For these reasons it is considered in Semantic Web research.

Our motivating scenario was risk monitoring in oil and gas (O&G) operations. Risk was supposed to be estimated using set of rules that describe dependencies between incoming data from underwater sensors. In recent years, Semantic Web (SW) started to penetrate the O&G industry, in particular in terms of data provisioning e.g. ISO15926 [1], WITSML [2]. Therefore, Semantic Web Rule Language (SWRL) was chosen as a basis to benefit from already existing SW-based developments. However, in our work we have discovered that constructing engineering rules in SWRL was an inefficient process. It resulted in a large set of long and difficult to understand rules. The scenario that lead us to adopting fuzzy control systems approach might have to some extent influences the generality of proposed solution. We aimed to minimize this influence, however, SWRL-F does not provide exactly the same capabilities as generic efforts of supplementing DL with FL. This difference is analyzed in further sections.

In this paper we present SWRL-F, a Fuzzy Logic extension of the Semantic Web Rule Language. It allows expressing imprecise information and helps in simplifying knowledge representation in SWRL. It consists of two parts. SWRL-F ontology that allows representing FL knowledge in the ontology and SWRL rule base, and execution engine that integrates with Protégé [3]. We present it in this paper basing on an example from our original study in O&G. It has been simplified to focus on SWRL-F description and discussion. Any presented values and dependencies cannot be seen as actual values from the rig.

Related Work. The need to supplement SW technologies with Fuzzy Logic has been already recognized. Thomas and Sheth in [4], in the volume Fuzzy Logic and Semantic Web, state the need and provide RDF-based examples for storing fuzzy information. The work on FL in SW has been focused on two fields: Description Logic and Rule Languages. Sometimes those two fields were approach collectively. We present the first field in brief and mostly focus on the latter one.

Straccia [5-8] is presenting various extensions of DL in the direction of handling uncertainty and FL. Holldobler, et al. [9] present fuzzy description logic ALCFH, where primitive concepts are modified by means of hedges. Hajek [10] present ACL-like fuzzy DL, powerful but still decidable. Stoilos and Samou [11] define fuzzy-SROIQ DL and an abstract, XML syntax and semantics of a fuzzy extension to OWL 1.1. These papers focus mostly on theoretical aspect of DL and FL, and no implementation efforts are mentioned or presented by authors.

Loia and Senatore [12] use fuzzy clustering in their web service-based, RDF-oriented classification. Li et al. [13] present a type-2 fuzzy version of ALC with syntax, semantics, reasoning algorithm and implementation. Pan et al. in [14] present a prototype implementation for querying f-DL-Lite ontologies. The implementation is done using extended version of SPARQL that can handle fuzzy queries. Neither of these implementations mentioned is publicly available.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

WIMS'11, May 25-27, 2011 Sogndal, Norway.

Copyright © 2011 ACM 978-1-4503-0148-0/11/05... \$10.00.

Lukasiewicz and Straccia [15] describe an infrastructure for handling uncertainty and vagueness in the Rules, Logic and Proof layers of the Semantic Web. Lukasiewicz in [16] presents an approach to fuzzy description logic programs under the answer set semantics for the Semantic Web, which is an integration of description logics with nonmonotonic logic programs. He mentions two approaches to Rules and Ontology layer integration. First allows for building rules on top of ontologies (rules use vocabulary from ontology knowledge bases). Second, builds ontologies on top of rules (ontological definitions are supplemented by rules or imported from rules).

Wang et al. [17] proposed new rule language: fuzzy Nonmonotonic Semantic Web Rule Language (f-NSWRL), which they claim suffices to represent and reason nonmonotonic fuzzy knowledge. In [18] they combine if-then rules with OWL DL in the framework of fuzzy sets and possibility distribution. They propose fuzzy Semantic Web if-then Rule Language (f-SW-if-the-RL). Agarwal and Hitzler [19] show how fuzzy membership functions and fuzzy rules can be modeled by means of appropriate description logic and how this can be employed for query answering. Damasio et al. [20] propose general extension to RuleML to represent uncertainty. Wulff and Sottaa [21] present fuzzy reasoning with a Rete-OO Rule Engine. They mention importance of connecting Fuzzy Logic with Semantic Web, but do not consider it in their implementation. Stoilos et al. [22] report on uncertainty and RuleML rulebases. They mention that despite the expressivity of OWL and rules, this connection still fails to accurately represent knowledge or our world. Pan et al. [23] propose f-SWRL, a fuzzy extension to SWRL. It includes fuzzy assertions and fuzzy rules. However, their approach is criticized in Agarwal and Hitzler [24], who explain that syntax and semantics of f-SWRL actually offers no fuzziness in f-SWRL rules. No implementation efforts are mentioned in these papers apart from [21]. However, this implementation does not in fact take into consideration any of Semantic Web technologies.

Bobillo et al. [25] present a semantic fuzzy expert system for a fuzzy balanced scorecard. They use OWL ontology to represent knowledge about variables. They also provide and interface to FuzzyJess to execute fuzzy rules. Protege is used as a development platform; however, implementation focuses only on balanced scorecard and rules are not based on SWRL. A need for more generic approach is mentioned in conclusions.

Stoilos et al. [26] discuss Fuzzy OWL and uncertainty representation with rules. They present a fuzzy reasoning engine that implements a reasoning algorithm for a fuzzy DL language fKD-SHIN. It handles most of OWL features. However, the implementation is proprietary and does not directly connect with established Semantic Web technologies or tools like OWL, SWRL or Protege.

Contributions. Despite the existing research work the problem of supplementing SW with FL remains without implemented, generic, publicly available, standards-based and widely used solution. In SWRL-F we aim to provide an FL extension to SWRL, which is based on standard OWL DL and SWRL. SWRL-F ontology enables description of FL knowledge and its application in SWRL rules. We also implemented a test execution engine and development environment that is publically available¹.

Organization of the Paper. After the Introduction, in Section 2 we provide a background on Fuzzy Logic, Description Logic, Web Ontology Language and Semantic Web Rule Language. In Section 3 we explain our design choices for SWRL-F in term of their influence on semantics of rules and logical soundness of the ontology. In Section 4 we describe and explain how SWRL-F is used to define fuzzy elements in OWL. Further, in Section 5, we describe the way of constructing and understanding fuzzy rules in SWRL-F. In Section 6 we give insight into parsing process for the SWRL-F rules. Section 7 discusses some benefits and drawbacks of SWRL-F. We conclude in Section 8

2. BACKGROUND

In this section we provide short summary of Description Logic, OWL, SWRL, Protege and Fuzzy Logic.

Description Logic (DL) and Web Ontology Language (OWL).

Description Logic is formal knowledge representation language. In terms of expressiveness and efficiency in decision problems it lays between propositional logic and first-order predicate logic [27]. Web Ontology Language is intended to provide a language that can be used to describe the classes and relations between them that are inherent in Web documents and applications. OWL DL is a sublanguage of OWL that supports maximum expressiveness without losing computational completeness [28].

Semantic Web Rule Language (SWRL). SWRL is a combination of the OWL DL and OWL Lite sublanguages with the Unary/Binary Datalog RuleML sublanguages of the Rule Markup Language. SWRL includes a high-level abstract syntax for Horn-like rules in both the OWL DL and OWL Lite sublanguages of OWL [29].

Protégé. It is a free, open source ontology editor and knowledge-base framework. The Protégé platform allows to model ontologies via the Protégé-OWL editor [3].

Fuzzy logic (FL). FL is a form of multi-valued logic, which is derived from the fuzzy set theory introduced by Zadeh [30]. Fuzzy set extends binary set by adding a degree of membership of an element to a set. If we take for example a variable describing age. We can define several sets describing this variable e.g. young, middle aged, old. In binary set theory a particular value would either belong to one or more of such sets or not. Such information has a limited value. In fuzzy set theory a particular value could belong to each set with different degree of membership. This can provide more valuable information. FL allows using non-numeric linguistic variables. They can facilitate expression of knowledge and rules, as they make them easier to understand. This way one can formulate rules in a form: IF person is old THEN risk of cancer is high.

One of the areas where fuzzy logic found significant application are control systems. The usual approach to fuzzy control systems is following: collect crisp inputs, fuzzify inputs, perform fuzzy inference, defuzzify inputs, apply crisp outputs [31].

3. DESIGN CHOICES

In this section we explain most significant design choices in SWRL-F, related to standards, scope of fuzzy inference, fuzzy assertions representation and rule engine.

¹ <http://protege.cim3.net/cgi-bin/wiki.pl?SWRLF>

As one can see from an analysis of related work a connection between FL and SW technologies is a non-trivial problem. Our leading idea was to follow fuzzy control systems scheme: fuzzification, inference, defuzzification. In addition the goal was to make SWRL-F rules as simple as possible for the final user. It was important as this way our solution was based on well established standards in fuzzy logic and at the same time provided user with solution that they can actually use even without significant technical knowledge. Below we detail four specific design choices that came from these principles.

First, SWRL-F should be standard based. It includes anchoring in the well established fuzzy logic scheme like fuzzy control systems. Moreover, SWRL-F can be fully expressed using OWL and SWRL, by importing SWRL-F ontology that we created. This ontology is purely OWL-based and it is described in the Section 4.

Second, fuzzy inference in SWRL-F is limited to the rules only basing on the fuzzy control systems approach. This way we can avoid inconsistencies in the ontology. Ontology is used to describe fuzzy knowledge base, however, for reasoned (DL or other) it appears like any other knowledge. It can be interpreted in limited, non-fuzzy way and used to create standard SWRL rules. Until we connect fuzzy rule reasoner knowledge based on SWRL-F ontology has limited use, but it does not create any inconsistencies with standard SW technologies.

Third, representation of fuzzy assertions in SWRL is using object property. We were taking into consideration two options. One option was based on writing a new built-in [32]. Second option included adding a standard object property to SWRL-F ontology, which would have special meaning when interpreted by a fuzzy rule reasoner. We decided on the second option as it provided the most natural way of expression and can be interpreted (though not in a fuzzy way) by a non-fuzzy rule reasoner.

Fourth, SWRL-F utilizes already existing fuzzy rule engine. We decided to reuse already existing reasoner, namely FuzzyJess [33]. There are several advantages of such choice. We could implement our solution faster and be at the same time sure that it will be stable and reasonably efficient. As FuzzyJess is a superset of Jess we could automatically provide compatibility with existing extensions and built-ins available for SWRL and SWRLJESSTab [34]. There is, though, one notable limitation of such approach: not all the OWL constructs can be represented, which follows the limitations as described in [35].

4. SWRL-F ONTOLOGY

In this section we present and explain all main elements of SWRL-F ontology in the context of the risk assessment example from O&G operations.

In order to express necessary fuzzy knowledge, namely fuzzy: sets, terms, variables and values, we have created SWRL-F ontology. We present the ontology and an example use case using Manchester syntax [36] and screenshot from Protégé.

```
Class: FuzzyVariable
Class: FuzzyTerm
Class: FuzzyValue
```

```
Class: FuzzySet
Class: LFuzzySet
  SubClassOf: FuzzySet
Class: LRFuzzySet
  SubClassOf: FuzzySet
Class: RFuzzySet
  SubClassOf: FuzzySet
Class: RightLinearFuzzySet
  SubClassOf: RFuzzySet
DataProperty: LeftEdge
  Domain: SFuzzySet, LeftLinearFuzzySet,
  RightLinearFuzzySet, ZFuzzySet
  Range: float
ObjectProperty: hasFuzzySet
  Characteristics: Functional
  Domain: FuzzyTerm, FuzzyValue
  Range: FuzzySet
  InverseOf: belongsToFuzzyTermOrValue
ObjectProperty: hasFuzzyTerm
  Domain: FuzzyVariable
  Range: FuzzyTerm
  InverseOf: belongsToFuzzyVariable
ObjectProperty: hasFuzzyValue
  Characteristics: InverseFunctional
  Domain: FuzzyVariable
  Range: FuzzyValue
  InverseOf: hasFuzzyVariable
ObjectProperty: hasFuzzyVariable
  Characteristics: Functional
  Domain: FuzzyValue
  Range: FuzzyVariable
  InverseOf: hasFuzzyValue
DataProperty: hasCaption
  Domain: FuzzyVariable
  Range: string
DataProperty: hasUnit
  Domain: FuzzyVariable
  Range: string
DataProperty: hasRangeFinish
  Domain: FuzzyVariable
  Range: float
DataProperty: hasRangeStart
  Domain: FuzzyVariable
  Range: float
```

Definition process starts with FuzzyVariable. One of variables in our use case is CurveDelayDuration that describes how long is the delay observed. FuzzyVariable has four datatype properties: hasCaption, hasUnit, hasRangeStart and hasRangeFinish. Last two are obligatory and describe the value range of the variable.

```
Individual: CurveDelayDuration
  Types: FuzzyVariable
  Facts: hasCaption "period in which delay
  observed", hasUnit "minutes", hasRangeFinish
  60.0, hasRangeStart 0.0, hasFuzzyTerm
  LongCurveDelayDuration, hasFuzzyTerm
  MediumCurveDelayDuration, ...
```

INDIVIDUAL EDITOR for CurveDelayDuration (Instance of FuzzyVariable)

For Individual: <http://www.owl-ontologies.com/FuzzySWRL.owl#CurveDelayDuration>

Property	Value	Lang
rdfs:comment		

hasCaption

hasUnit

hasRangeFinish

hasRangeStart

hasFuzzyTerm

- LongCurveDelayDuration
- MediumCurveDelayDuration
- ShortCurveDelayDuration
- VeryLongCurveDelayDuration
- VeryShortCurveDelayDuration

Figure 1. FuzzyVariable definition in Protégé.

Property hasFuzzyTerm adds linguistic terms that are later used in the SWRL-F rules and defines sets that describe FuzzyVariable. One of FuzzyTerms is LongCurveDelayDuration.

Individual: LongCurveDelayDuration
Types: FuzzyTerm
Facts: hasFuzzySet TrapezoidFuzzySet_16

FuzzyTerm is defined using FuzzySet. FuzzySet class contains subclasses that define different types of fuzzy sets that can be used. Definition of all FuzzySets follows that presented in FuzzyJ package [33].

Individual: TrapezoidFuzzySet_16
Types: TrapezoidFuzzySet
Facts: oneLeftX 12.0, oneRightX 30.0, zeroLeftX 10.0, zeroRightX 31.0

INDIVIDUAL EDITOR for LongCurveDelayDuration (Instance of FuzzyTerm)

For Individual: <http://www.owl-ontologies.com/FuzzySWRL.owl#LongCurveDelayDuration>

Property	Value	Lang
rdfs:comment		

hasFuzzySet

- TrapezoidFuzzySet_16

Figure 2. FuzzyTerm definition in Protégé.

INDIVIDUAL EDITOR for TrapezoidFuzzySet_16 (Instance of TrapezoidFuzzySet) ⊕ ⊖ F T

For Individual: http://www.owl-ontologies.com/FuzzySWRL.owl#TrapezoidFuzzySet_16

Annotations

Property	Value	Lang
rdfs:comment		

oneLeftX

oneRightX

zeroLeftX

zeroRightX

Figure 3. FuzzySet definition in Protégé.

The only class that is left to be defined is FuzzyValue. Let us create an OWL class Curve. Curve is one of the names used for dataset describing a sensor-based variable in O&G industry. In our project we have preprocessed that data using DataStorm [37] and Hadoop [38] to extract the information we required. CurveDelay represents delay of curve under analysis with respect to the reference curve. CurveDelayDuration describes for how long the delay is observed. CurveValueDeviation describes the deviation of the value on the curve from the reference value. Finally we can define the FuzzyValue for Risk that we want to calculate using SWRL-F.

```
Individual: TestCurve3
  Types: Curve
  Facts: hasCurveDelay CurveDelay3,
hasCurveValueDeviation CurveValueDeviation3,
hasCurveDelayDuration CurveDelayDuration3,
hasRisk CurveRisk3
```

The value of hasCurveDelayDuration is a FuzzyValue. Each FuzzyValue has four parameters. Their use depends on whether it is an input or output value. hasFuzzyVariable is the main

parameter used both for input and output. It lets the rule engine know how to interpret this value. hasFuzzySet gives the input value of the FuzzyValue. If the input is in fact crisp one can use SingletonFuzzySet to describe it. Using other types of fuzzy sets allows for fuzzification of the input. hasDefuzzificationMethod (not shown below) describes what method should be used for defuzzification and it is used both for input and output value, however, in a different way as described further. Finally, hasCrispValue (not shown below), is automatically filled in by the rule engine. If the particular FuzzyValue is an input it comes from a defuzzification of input defined by hasFuzzySet. If the FuzzyValue is an output it comes from a defuzzification of all the assertions made to this value in SWRL-F rules. Whether FuzzyValue is an input or output is automatically determined by the engine.

```
Individual: CurveDelayDuration3
  Types: FuzzyValue
  Facts: hasFuzzyVariable
CurveDelayDuration, hasFuzzySet
CurveDelayDurationFuzzySet3
```

INDIVIDUAL EDITOR for TestCurve3 (Instance of Curve) ⊕ − F T

For Individual: <http://www.owl-ontologies.com/FuzzySWRL.owl#TestCurve3>

⊕ ⊖ ⊗ ⊗ Annotations

Property	Value	Lang
<code>rdfs:comment</code>		

hasCurveDelay ⊕ ⊖ ⊗ ⊗

CurveDelay3

hasCurveValueDeviation ⊕ ⊖ ⊗ ⊗

CurveValueDeviation3

hasCurveDelayDuration ⊕ ⊖ ⊗ ⊗

CurveDelayDuration3

hasRisk ⊕ ⊖ ⊗ ⊗

CurveRisk3

Figure 4. Example class definition in Protégé containing four related FuzzyValues.

INDIVIDUAL EDITOR for CurveDelayDuration3 (Instance of FuzzyValue) ⊕ − F T

For Individual: <http://www.owl-ontologies.com/FuzzySWRL.owl#CurveDelayDuration3>

⊕ ⊖ ⊗ ⊗ Annotations

Property	Value	Lang
<code>rdfs:comment</code>		

hasDefuzzificationMethod ⊗

weightedAverageDefuzzify

hasFuzzyVariable ⊕ ⊖ ⊗ ⊗

CurveDelayDuration

hasCrispValue ⊗

fuzzymatch ⊕ ⊖ ⊗ ⊗

hasFuzzySet ⊕ ⊖ ⊗ ⊗

CurveDelayDurationFuzzySet3

Figure 5. FuzzyValue definition in Protégé.

5. SWRL-F RULES

In this section we demonstrate syntax and functioning of SWRL-F rules.

Having FuzzyValues and FuzzyTerms described one can construct rules in SWRL-F. To do so we use modified SWRLJessTab. SWRL-F rules are normal SWRL rules. However, they make use of fuzzymatch object property from SWRL-F ontology. If executed using standard rule engine like Jess this property acts as any other object property. However, if run using modified version of SWRLJessTab (that we provide) together with FuzzyJ and FuzzyJess packages, fuzzymatch property allows constructing fuzzy rules.

```
ObjectProperty: fuzzymatch
  Domain: FuzzyValue
  Range: FuzzyTerm
```

Let us analyze a simple example:

```
Curve(?c) ^ hasCurveDelayDuration(?c, ?d) ^
fuzzymatch(?d, ShortCurveDelayDuration) ^
hasRisk(?c, ?r) → fuzzymatch(?r, LowRisk)
```

The fuzzymatch property is used to calculate degree of membership of CurveDelayDuration ?d in the ShortCurveDelayDuration term and set. Second use of fuzzymatch allows to bind the new value LowRisk to the Risk ?r FuzzyValue.

There can be many rules that assign new values to the same FuzzyValue. In contrast with standard SWRL where such assertions would not carry any additional semantics, in SWRL-F they constitute a very important tool. They allow creating several rules that can influence the result at the same time. The values that each rule assigns are then grouped together and collectively defuzzified into one final crisp result. This can be used to separate possibly very complicated rules into a set of simpler and more straight forward rules. Apart from simplifying management and creation of rules, this allows to create rules in a more natural way.

6. REPRESENTATION OF SWRL-F IN FUZZYJESS

In this section we present the process of representing SWRL-F ontology and rules in FuzzyJess. This should be useful for somebody interested in further extending SWRL-F or working on similar extension. This section assumes familiarity with Jess and FuzzyJess.

A large number of rule engines work well with Java, and many are available as open source software. Some of the most popular engines include Jess, Algernon and SweetRules. Some of the enable use of fuzzy logic for rules e.g. FuzzyJess, FuzzyDrools. We chose FuzzyJess as the first integration candidate for the SWRL-F because it works seamlessly with Java, has an extensive user base, is well documented, and is very easy to use and configure. Moreover, we can use all the previous work on SWRL and Jess integration [35]. As mentioned earlier one research team has also demonstrated that a mapping between OWL and FuzzyJess is possible [25]. However, they did not use SWRL to create rules. FuzzyJess engine can be embedded in Java applications and provides a flexible two-way run-time communication between FuzzyJess rules and Java.

The FuzzyJess system consists of a rule base, a fact base, and an execution engine. The execution engine matches facts in the fact base with rules in the rule base. These rules can assert new facts and put them in the fact base or execute Java functions.

SWRL rules reason about OWL individuals, primarily in terms of OWL classes and properties. When a SWRL rule is matched, it can create new classifications for existing individuals. In SWRL-F the process is slightly more complicated, but only if we use fuzzy assertions. For binary assertions no changes need to be made. Therefore, the basic OWL classes and properties are represented in the same way as in previous work on SWRL and Jess. However, there were several changes and additions necessary to enable fuzzy rules, which are based on SWRL-F ontology.

Classes and individuals in SWRL-F ontology carry significantly more knowledge than respective elements in other OWL ontologies. Therefore apart from standard representation we had to add appropriate constructs for creating FuzzyVariables, FuzzyTerms and FuzzyValues. No additional constructs were necessary for FuzzySet as those were built basing exactly on FuzzyJ package that we utilize.

All the following examples base on the definitions from Section 4. FuzzyVariables are represented in a following way on example of CurveDelayDuration:

```
(deftemplate CurveDelayDuration (slot name)
  (slot fv) (slot cv))
```

in comparison with standard class representation one can notice two additional slots to represent fuzzy value fv and crisp value cv. We have initially experimented with representing such values in the same way as other object properties. However, due to functioning details of Jess and FuzzyJess it was impossible. The main reason being the impossibility to address them properly in order to handle grouping of multiple assertions to one variable, which is one of key features of SWRL-F.

FuzzyTerms are represented using a totally new construct in comparison with standard OWL to Jess translation. We can also notice the use of the FuzzySet in this example:

```
(?*CurveDelayDurationFVar* addTerm
"LongCurveDelayDuration" (new
TrapezoidFuzzySet 10.0 12.0 30.0 31.0))
```

FuzzyValues definition builds on the definition of FuzzyVariable presented earlier. Only the FuzzyValues that have FuzzySet assigned (input values) are being represented like this. For output values the assignment process happens in rules.

```
(assert (CurveDelayDuration (name
CurveDelayDuration2 ) (fv (new FuzzyValue
?*CurveDelayDurationFVar* (new
SingletonFuzzySet 4.0))))))
```

All the values need to be defuzzified. We have considered two options, either explicit or implicit defuzzification. In the end we have decided on implicit defuzzification. This way SWRL-F is much easier to use for the user. At the same time there is no loss of expressivity due to this solution. Defuzzification results in calling inferOWLPropertyAssertionAxiom. Despite the difference in calling it, it functions in exactly the same way as in standard OWL to Jess implementation.

```
(defrule defuzzificationrule2
(declare (salience -100))
(Risk (name ?vn) (fv ?fv))
(not (test (eq ?fv nil)))
=>
(bind ?cv (call ?fv
weightedAverageDefuzzify))
(inferOWLPropertyAssertionAxiom
hasCrispValue ?vn ?cv)
)
```

the axiom `inferOWLPropertyAssertionAxiom` functions in exactly the same way as in standard OWL to Jess implementation.

Changes also had to be made to rules in order to accommodate fuzzy assertions. This is based on using `fuzzymatch` property from Fuzzy Description Ontology. Fuzzy matching in the rule body looks like this:

```
(CurveDelayDuration (name ?d) (fv
?fv&:(fuzzy-match ?fv
"ShortCurveDelayDuration")))
```

and the assertion in the consequent like this:

```
(assert (Risk (name ?r) (fv (new FuzzyValue
?*RiskFVar* "LowRisk")))).
```

There are other ways to represent the same variables, value and rules in FuzzyJess in sometimes more clear and concise way, as seen in literature [39]. However, they are then incapable to represent knowledge and rules on the generality level required to represent SWRL-F rules. We made an explicit choice here that we prefer more complex implementation in order to provide a simpler and more generic final solution.

7. DISCUSSION

In this section we discuss benefits and drawback of using `FuzzyVariable` class to represent fuzzy knowledge, limitations of our inference support and fuzzy knowledge representation, and usefulness of SWRL-F for expert knowledge representation.

Introducing a new class `FuzzyVariable` was not absolutely necessary. Another option was to allow use of any numerical value and just supplying it with additional information that is necessary to perform fuzzy inference in the rule base. This would allow for easier integration with already existing ontology. On the other hand, it would force user to remember every time about supplying this additional values and tying them by hand with the main value. Additionally, it shortens SWRL-F rules by one atom per each use of `fuzzymatch`. In our opinion, introducing special class to represent a fuzzy variable should ease the usage of SWRL-F. However, it might also limit the applicability of SWRL-F to some of the problems.

SWRL-F does not introduce any inconsistencies into host ontology basing on limiting fuzzy inference to rules only thanks to SWRL-F ontology construction. However, it has the some limitations with regards to OWL representation. It follows the same general limitation of `SWRLJessTab` as explained in [35].

Supplementing DL (or other SW-related logic) with FL is not a trivial issue, especially in the context of generic implementation, as it was demonstrated in Related Work. Therefore, we have limited fuzzy inference to rules only through applying fuzzy control systems approach. This introduces specific limitations to fuzzy knowledge representation that have to be well understood to determine applicability of SWRL-F to particular problem.

SWRL-F will not allow creating ontologies where class definition is dependent on fuzzy concepts or inference. In other words, it will not allow fuzzy reasoning that influences class taxonomy. On the other hand, SWRL-F will allow creating SW applications that require fuzzy rule inference from data and concepts defined in the knowledge base. Therefore, SWRL-F might not be well applicable to problems where fuzzy reasoning is used as a tool for ontology construction. However, it should be useful for problems of decision making or querying where fuzzy inference is necessary from concepts defined in an ontology.

SWRL-F allows easier knowledge management by moving numerical values from rules to ontology. This results in simpler rules and removes hard-coding of those numerical values in rules. SWRL-F also allows, in some cases, to separate large rules into logical part thanks to support of automatic combined assignment. It means that if several rules make a fuzzy assertion about particular fuzzy value, the result will be combined from all assertions instead of just the last one. This is different comparing to standard SWRL implementation and refers only to situation where `fuzzymatch` property is used in the rule consequent. Such separation could be also used to replace lack of disjunction in SWRL in some scenarios. Judging on these elements SWRL-F should be useful tool to represent imprecise expert knowledge. However, this aspect has to further investigated in detail on wider example set, what is not the scope of this paper.

8. CONCLUSIONS

In this paper we presented SWRL-F. It is an extension to SWRL that allows constructing fuzzy rules in SWRL based on language variables described using OWL-based ontology. Its general design is based on fuzzy control systems approach and together with proper construction of SWRL-F ontology it allows to avoid conflicts between FL and the host ontology. SWRL-F allows extending SW applications with FL capabilities basing on Protégé editor and modified `SWRLJessTab`.

9. REFERENCES

- [1] "ISO15926Primer – POSC Caesar – Trac" Available: <http://www.posccaesar.org/wiki/ISO15926Primer>.
- [2] "WITSML Standard" Available: <http://www.energetics.org/witsml-standard>.
- [3] "The Protégé Ontology Editor and Knowledge Acquisition System" Available: <http://protege.stanford.edu/>.
- [4] C. Thomas and A. Sheth, "On the expressiveness of the languages for the Semantic Web — Making a case for "A little more," FUZZY LOGIC AND THE SEMANTIC WEB, CAPTURING INTELLIGENCE, 2006, pp. 3--20.
- [5] U. Straccia and I.V.G. Moruzzi, "Uncertainty and description logic programs over lattices," FUZZY LOGIC AND THE SEMANTIC WEB, CAPTURING INTELLIGENCE, CHAPTER 7, 2004, pp. 115--133.
- [6] U. Straccia, "Reasoning within fuzzy description logics," J. Artif. Int. Res., vol. 14, 2001, pp. 137-166.
- [7] U. Straccia, "Description Logics with Fuzzy Concrete Domains," 2005.
- [8] U. Straccia, "Towards a fuzzy description logic for the semantic web (preliminary report," FUZZY LOGIC AND THE

SEMANTIC WEB, CAPTURING INTELLIGENCE, CHAPTER 4, 2005, pp. 167--181.

[9] S. Hölldobler, T.D. Khang, and H. Störr, "A Fuzzy Description Logic with Hedges as Concept Modifiers," IN PROCEEDINGS INTECH/VJFUZZY'2002, 2002, pp. 25--34.

[10] P. Hájek, "Chapter 5 What does mathematical fuzzy logic offer to description logic?," Fuzzy Logic and the Semantic Web, Elsevier, 2006, pp. 91-100.

[11] G. Stoilos and G. Stamou, "Extending fuzzy description logics for the semantic web," IN: PROCEEDINGS OF THE 3RD INTERNATIONAL WORKSHOP ON OWL: EXPERIENCES AND DIRECTIONS (OWLED, 2007.

[12] V. Loia and S. Senatore, "Chapter 3 Capturing basic semantics exploiting RDF-oriented classification," Fuzzy Logic and the Semantic Web, Elsevier, 2006, pp. 49-69.

[13] R. Li, X. Sun, Z. Lu, K. Wen, and Y. Li, "Towards a type-2 fuzzy description logic for semantic search engine," Proceedings of the joint 9th Asia-Pacific web and 8th international conference on web-age information management conference on Advances in data and web management, Huang Shan, China: Springer-Verlag, 2007, pp. 805-812.

[14] J.Z. Pan, G. Stamou, G. Stoilos, and E. Thomas, "Expressive Querying over Fuzzy DL-Lite."

[15] T. Lukasiewicz and U. Straccia, "Description logic programs under probabilistic uncertainty and fuzzy vagueness," International Journal of Approximate Reasoning, vol. 50, Jun. 2009, pp. 837-853.

[16] T. Lukasiewicz, "Fuzzy Description Logic Programs under the Answer Set Semantics for the Semantic Web," Rules and Rule Markup Languages for the Semantic Web, International Conference on, Los Alamitos, CA, USA: IEEE Computer Society, 2006, pp. 89-96.

[17] X. Wang, Z. Ma, C. Xu, and J. Cheng, "Nonmonotonic Fuzzy Rules in the Semantic Web," Fuzzy Systems and Knowledge Discovery, Fourth International Conference on, Los Alamitos, CA, USA: IEEE Computer Society, 2009, pp. 275-279.

[18] X. Wang, Z.M. Ma, L. Yan, and J. Cheng, "If-Then and If-Then-Unless Rules in the Semantic Web," Proceedings of the 2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology - Volume 01, IEEE Computer Society, 2009, pp. 357-360.

[19] S. Agarwal and P. Hitzler, "Modeling Fuzzy Rules with Description Logics," IN PROCEEDINGS OF WORKSHOP ON OWL EXPERIENCES AND DIRECTIONS, 2005.

[20] C.V. Damásio, J.Z. Pan, G. Stoilos, and U. Straccia, "Representing Uncertainty in RuleML," Fundam. Inf., vol. 82, 2008, pp. 265-288.

[21] N. Wulff and D. Sottara, "Fuzzy Reasoning with a Rete-OO Rule Engine," Rule Interchange and Applications, 2009, pp. 337-344.

[22] G. Stoilos, G. Stamou, V. Tzouvaras, and J. Pan, "Uncertainty and RuleML Rulebases: A Preliminary Report," Rules and Rule Markup Languages for the Semantic Web, 2005, pp. 199-203.

[23] J.Z. Pan, G. Stamou, V. Tzouvaras, and I. Horrocks, "f-SWRL: A Fuzzy Extension of SWRL," Artificial Neural Networks: Formal Models and Their Applications - ICANN 2005, 2005, pp. 829-834.

[24] S. Agarwal and P. Hitzler, "Modeling Fuzzy Rules with Description Logics."

[25] F. Bobillo, M. Delgado, J. Gómez-Romero, and E. López, "A semantic fuzzy expert system for a fuzzy balanced scorecard," Expert Syst. Appl., vol. 36, 2009, pp. 423-433.

[26] G. Stoilos, N. Simou, G. Stamou, and S. Kollias, "Uncertainty and the Semantic Web," IEEE Intelligent Systems, vol. 21, 2006, pp. 84-87.

[27] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. Patel-Schneider, The Description Logic Handbook: Theory, Implementation and Applications, Cambridge University Press, 2003.

[28] "OWL Web Ontology Language Guide" Available: <http://www.w3.org/TR/owl-guide/>.

[29] "SWRL: A Semantic Web Rule Language Combining OWL and RuleML" Available: <http://www.w3.org/Submission/SWRL/>.

[30] L. Zadeh, "Fuzzy sets," pp. 338-353.

[31] T.J. Ross, "Fuzzy Control Systems," Fuzzy Logic with Engineering Applications, Wiley-Blackwell, 2004.

[32] "Built-Ins for SWRL" Available: <http://www.daml.org/2004/04/swrl/builtins.html>.

[33] B. Orchard, "Controlling with fuzzy rules," Jess in Action: Java Rule-Based Systems, Manning Publications, 2003.

[34] "ProtegeWiki: SWRLJess Tab" Available: <http://protege.cim3.net/cgi-bin/wiki.pl?SWRLJessTab>.

[35] M. O'connor, H. Knublauch, S. Tu, B. Grosz, M. Dean, W. Grosso, and M. Musen, "Supporting Rule System Interoperability on the Semantic Web with SWRL," The Semantic Web - ISWC 2005, 2005, pp. 974-986.

[36] "OWL 2 Web Ontology Language Manchester Syntax" Available: <http://www.w3.org/TR/owl2-manchester-syntax/>.

[37] T.W. Włodarczyk, C. Rong, B. Jia, L. Cocanu, C.I. Nyulas, and M.A. Musen, "DataStorm," Services, IEEE Congress on, Los Alamitos, CA, USA: IEEE Computer Society, 2010, pp. 123-127.

[38] "Welcome to Apache Hadoop!" Available: <http://hadoop.apache.org/>.

[39] E. Friedman-Hill, Jess in Action: Java Rule-Based Systems, Manning Publications, 2003.