On the incorporation of interval-valued fuzzy sets into the Bousi-Prolog system: declarative semantics, implementation and applications

Clemente Rubio-Manzano\textsuperscript{a}, Martín Pereira-Fariña\textsuperscript{b}

\textsuperscript{a}Dep. of Information Systems, University of the Bío-Bío, Chile.
\textsuperscript{b}Centre for Argument Technology, University of Dundee, UK. Departamento de Filosofía e Antropología, Universidade de Santiago de Compostela, Spain.

Abstract

In this paper we analyse the benefits of incorporating interval-valued fuzzy sets into the Bousi-Prolog system. A syntax, declarative semantics and implementation for this extension is presented and formalised. We show, by using potential applications, that fuzzy logic programming frameworks enhanced with them can correctly work together with lexical resources and ontologies in order to improve their capabilities for knowledge representation and reasoning.

Keywords: Interval-valued fuzzy sets, Approximate Reasoning, Lexical Knowledge Resources, Fuzzy Logic Programming, Fuzzy Prolog.

1. Introduction and Motivation

Nowadays, lexical knowledge resources as well as ontologies of concepts are widely employed for modelling domain independent knowledge \[1, 2\] or
by automated reasoners [3]. In the case of approximate reasoning, this makes possible to incorporate general knowledge into any system, which is independent of the programmer’s background [4].

Inside the former and current frameworks of fuzzy logic programming [5, 6, 7, 8, 9, 10], we argue that lexical reasoning might be an appropriate way for tackling this challenge, because of this type of knowledge is usually expressed linguistically. However, from a computational point of view, this source of information involves vagueness and uncertainty and, consequently, it must be specifically addressed. Fuzzy set theory (FS) is a good candidate, but it shows some particular limitations to this aim: i) sometimes, words mean different things to different people and this generates and additional layer of uncertainty that cannot be adequately handled by FS; ii) the definition of membership functions for word meaning is also a debatable question and, therefore, achieving an agreement by means of a standard fuzzy set is difficult; and, iii) with respect to semantic similarity measures used in this proposal, there is not a dominant one and, therefore, for two given words, different degrees of resemblance can be obtained with the resulting additional level of uncertainty.

In the specific field of fuzzy logic programming and fuzzy Prolog systems, little attention has been paid to the impact of this type of high degree of uncertainty and vagueness inherent to lexical knowledge, which is used in the definition of knowledge bases and inference processes. Next, a very simple example is introduced in order to illustrate i) and ii) in the building of a
Prolog knowledge base.

**Example 1.** Suppose that we extract from Internet two people’s opinions about a particular football player. The first one says “a is a normal player” and the second one says “a is a bad player”. If we consider the label for qualifying the highest quality (e.g., “good”) as a basic component, this lexical knowledge could be modelled by using two annotated facts as: “football_player(a,good):-0.8.” and “football_player(a,good):-0.6.”, respectively. In this case, we use “football_player(a,good):-0.6.” given the infimum is usually employed. However, as it can be observed, the information of the first person is lost.

Case iii) deserves a special attention, given it involves the use of independent linguistic resources (such as WordNet Similarity [11]). As we said, this tool provide us different measures according to alternative criteria for assessing the degree of similarity between two words. In Example 2 we illustrate this situation by means of a simple case.

**Example 2.** Suppose we have the fact “loves(a,b)” and we extract the closeness between “loves” and “desires” by using two different semantics measures obtaining 0.8 and 0.6. Therefore, in order to represent this semantic knowledge we could employ two facts either “desires(a,b):-0.8” or “desires(a,b):-0.6”.

In order to address both Examples 1 and 2 inside the same frame, we propose to enhance the Bousi-Prolog system with interval-valued fuzzy sets (IVFSs), since they allow us to capture the uncertainty associated to lexical knowledge better than FS. Several advantages have pointed out for dealing with environments with high uncertainty or imprecision using IVFSs, such as [12]; other authors have also shown that IVFSs can generate better results than standard FSs [13]. Additionally, the use of intervals for describing
uncertain information has been successfully applied in the realms of decision making, risk analysis, engineering design, or scheduling [14].

Both Example 1 and Example 2 can be easily modelled by means of IVFSs, using an interval for combining information of the different sources into a single fact such as “football_player(a,good):-[0.6,0.8]” or “desires(a,b):-[0.6,0.8]”, respectively.

The main contribution of this paper is to design and implement an interval-valued fuzzy logic language, and to incorporate it into the Bousi-Prolog system [15]. This task involves different challenges both from theoretical and implementation points of view. The former entails adding a IVFSs arithmetic into the Warren Abstract Machine based on Similarity (SWAM) [16]; the latter, means to establish a (model-theoretic) declarative semantics for the language in the classical way, formalising the notion of least interval valued fuzzy Herbrand model for interval-valued fuzzy definite programs.

This paper is divided into the following sections: section 2 introduces the concepts that support our approach; section 3 describes the details of the syntax, semantics and implementation of the proposed language; section 4 analyses different realms where this programming language can be applied; in section 5 the main differences between this proposal and others that are described in the literature are discussed; and, finally, section 6 summarizes our main conclusions and some ideas for future work.
2. Preliminary Concepts

2.1. Interval-Valued Fuzzy Sets

IVFSs are a fuzzy formalism based on two membership mappings instead of a single one, like in standard FSs. Each one of these membership functions are called, lower membership function and upper membership function. Both are established on a universe of discourse \( \mathcal{X} \), and they map each element from \( \mathcal{X} \) to a real number in the \([0, 1]\) interval, where the elements of \( \mathcal{X} \) belongs to \( \mathcal{A} \) according to an interval.

**Definition 2.1.** An interval-valued fuzzy set \( \mathcal{A} \) in \( \mathcal{X} \) is a (crisp) set of ordered triples: 
\[
\mathcal{A} = \{ (x, \mu_A(x), \overline{\mu}_A(x)) : x \in \mathcal{X}; \mu_A(x), \overline{\mu}_A(x) : \mathcal{X} \rightarrow [0, 1] \}
\]
where: \( \mu, \overline{\mu} \) are the lower and the upper membership functions, respectively, satisfying the following condition: 
\[
0 \leq \mu_A(x) \leq \overline{\mu}_A(x) \leq 1 \forall x \in \mathcal{X}
\]

As can be observed in Definition 2.1 these intervals are included in \([0, 1]\) and closed at both ends. On the other hand, some arithmetic operations on interval-numbers have been recalled since they are useful in operating on cardinalities of IVFSs. Let \( a = [a, \overline{a}], b = [b, \overline{b}] \) be intervals in \( R \), and \( r \in R^+ \). The arithmetic operations '+', '-', '·' and power are defined as follows:

\[
[a, \overline{a}] + [b, \overline{b}] = [a + b, \overline{a} + \overline{b}]; \tag{1}
\]
\[
[a, \overline{a}] - [b, \overline{b}] = [a - b, \overline{a} - \overline{b}]; \tag{2}
\]
\[
[a, \overline{a}] \cdot [b, \overline{b}] = [\min(a \cdot b, \overline{a} \cdot \overline{b}, a \cdot \overline{b}, \overline{a} \cdot b), \max(a \cdot b, a \cdot \overline{b}, \overline{a} \cdot b, \overline{a} \cdot \overline{b})]; \tag{3}
\]
\[
([a, \overline{a}])^r = [a^r, \overline{a}^r] \text{ for non-negative } a, \overline{a} \tag{4}
\]
The operations of union and intersection for IVFSs are defined by triangular norms. Let $A, B$ be IVFSs in $\mathcal{X}$, $t$ a t-norm and $s$ a t-conorm. The union of $A$ and $B$ is the interval-valued fuzzy set $A \cup B$ with the membership function:

$$\mu_{A \cup B}(x) = [s(\mu_A(x), \mu_B(x)), s(\overline{\mu}_A(x), \overline{\mu}_B(x))]$$

The intersection of $A$ and $B$ is the IVFS $A \cap B$ in which $\mu_{A \cap B}(x) = [t(\mu_A(x), \mu_B(x)), t(\overline{\mu}_A(x), (\overline{\mu}_B(x))]$.

Thus, de Morgan’s laws for IVFSs $A, B$ in $\mathcal{X}$ are: $(A \cup B)^c = A^c \cap B^c$ and $(A \cap B)^c = A^c \cup B^c$.

Let $L$ be a lattice of intervals in $[0, 1]$ that satisfies:

$$L = [x_1, x_2] \in [0, 1]^2 \text{ with } x_1 \leq x_2; \quad (5)$$

$$[x_1, x_2] \leq_L [y_1, y_2] \text{ iff } x_1 \leq y_1 \text{ and } x_2 \leq y_2. \quad (6)$$

Also by definition:

$$[x_1, x_2] <_L [y_1, y_2] \iff x_1 < y_1, x_2 \leq y_2 \text{ or } x_1 \leq y_1, x_2 < y_2; \quad (7)$$

$$[x_1, x_2] =_L [y_1, y_2] \iff x_1 = y_1, x_2 = y_2. \quad (8)$$

Hence, $0_L = [0, 0]$ and $1_L = [1, 1]$ are the smallest and the greatest elements in $L$.

2.2. Approximate Deductive Reasoning

When we consider a collection of imprecise premises and a possible imprecise conclusion inferred from them in a Prolog program, we are applying a process of approximate deductive reasoning. These set of statements can
be interpreted under two different frames in a Prolog program: conditional and set-based interpretations. If the former is assumed, an imprecise premise is an assertion qualified by a degree of truth; e.g., “John is tall with [0.2,0.5]” means that the degree of truthfulness of this sentence using and IVFS is [0.2,0.5]. On the other hand, if the latter is adopted, the interval that qualifies the sentence means the degree of membership of an element to a specific set; e.g., “John is tall with [0.2,0.5]” means that the membership of John to the set of tall people is [0.2,0.5]. The conclusion inferred from an imprecise premise must be also qualified by the same type of degree; e.g. “John is a good player with [0.2,0.5]”.

In order to preserve the coherence with classical Prolog, we adopt the propositional interpretation (the interval indicates the degree of truth of the assertion) and, consequently, approximate deductive reasoning is based on multi-valued modus ponens:

\[
Q, [\alpha, \overline{\alpha}] \\
A \leftarrow Q, [\beta, \overline{\beta}] \\
A, T([\alpha, \overline{\alpha}], [\beta, \overline{\beta}])
\]

If we have (9) and (10), we can deduce (11) with \(T\) a t-norm defined on the lattice \(L([0,1])\).
3. Simple Interval-valued fuzzy prolog: syntax, semantics and implementation

The design of a programming language involves three main steps. Firstly, the definition of the syntax; secondly, the elaboration of a formal study of its semantics; and thirdly, an implementation of the system. In order to address the tasks related with syntax and semantics, we will follow the guidelines established in [19] and [20]; for the implementation task, we will follow the guidelines detailed in [16].

3.1. Syntax

An Interval-valued fuzzy program conveys a classical Prolog knowledge base and a set of IVFSs, which are used for annotating the facts by means of an interval-valued fuzzy degree: \( p(t_1, \ldots, t_n)[\alpha, \overline{\alpha}] \).

**Definition 3.1.** An interval-valued fuzzy definite clause is a Horn clause of the form \( A[\alpha, \overline{\alpha}] \) or \( A \leftarrow B_1, \ldots, B_n[\beta, \overline{\beta}] \), where \( A \) is called the head, and \( B_1, \ldots, B_n \) denote a conjunction which is called the body (variables in a clause are assumed to be universally quantified).

**Definition 3.2.** An interval-valued fuzzy definite program is a finite set of interval-valued fuzzy clauses.

**Example 3.** Let \( \Pi = \{ p(X) \leftarrow q(X), q(a)[0.8, 0.9], q(b)[0.7, 0.8] \} \) be an interval-valued fuzzy definite program, \( \Pi \) generates a first order language, \( \mathcal{L} \), whose alphabet is comprised of the set of variable symbols, \( \mathbf{X} \), constant symbols, \( \mathbf{C} \), function symbols, \( \mathbf{F} \) and predicate symbols, \( \mathbf{P} \), which appear in the clauses of \( \Pi \). We assume that the first order language \( \mathcal{L} \) has, at least one constant symbol; i.e., an assertion. If there are not constants available in the alphabet, an artificial constant “a” must be added to it. The first order language \( \mathcal{L} \) generated by \( \Pi \) is: \( \mathbf{X} = \{ x \}, \mathbf{C} = \{ a, b, c \}, \mathbf{F} = \emptyset \) and \( \mathbf{P} = \{ p, q, r \} \).

\(^1\)We assume familiarity with the theory and practice of logic programming.
3.2. Declarative Semantics

In logic programming, the declarative semantics for a program is traditionally formulated on the basis of the least Herbrand model (conceived as the infimum of a set of interpretations). In this section, we formally introduce the semantic notions of Herbrand interpretation, Herbrand model and least Herbrand model for an interval-valued fuzzy program Π, in order to characterise it.

In our framework, truth-values of the facts are modelled in terms of interval-valued degrees $[\alpha, \overline{\alpha}]$ with $0 \leq \alpha \leq \overline{\alpha} \leq 1$. An interval-valued fuzzy interpretation $\mathcal{I}$ is a pair $\langle \mathcal{D}, \mathcal{J} \rangle$ where $\mathcal{D}$ is the domain of the interpretation and $\mathcal{J}$ is a mapping which assigns meaning to the symbols of $\mathcal{L}$: specifically n-ary relation symbols are interpreted as mappings $\mathcal{D}^n \rightarrow L([0, 1])$. In order to evaluate open formulas, we have to introduce the notion of variable assignment. A variable assignment, $\vartheta$, w.r.t. an interpretation $\mathcal{I} = \langle \mathcal{D}, \mathcal{J} \rangle$, is a mapping $\vartheta : V \rightarrow \mathcal{D}$, from the set of variables $V$ of $\mathcal{L}$ to the elements of the interpretation domain $\mathcal{D}$. This notion can be extended to the set of terms of $\mathcal{L}$ by structural induction as usual. The following definition formalises the notion of valuation of a formula in our framework.

**Definition 3.3.** Given an interval-valued fuzzy interpretation $\mathcal{I}$ and a variable assignment $\vartheta$ in $\mathcal{I}$, the valuation of a formula w.r.t. $\mathcal{I}$ and $\vartheta$ is:

1. (a) $\mathcal{I}(p(t_1, \ldots, t_n))[\vartheta] = \overline{p}(t_1\vartheta, \ldots, t_n\vartheta)$, where $\mathcal{J}(p) = \overline{p}$;
   
   (b) $\mathcal{I}(A_1, \ldots, A_n))[\vartheta] = \inf\{\mathcal{I}(A_1)[\vartheta], \ldots, \mathcal{I}(A_n)[\vartheta]\}$;

2. $\mathcal{I}(A \leftarrow Q)[\vartheta] = 1$ if $I(A) \geq I(Q)$; $\mathcal{I}(A \leftarrow Q)[\vartheta] = \mathcal{I}(A)[\vartheta]$ if $I(A) < I(Q)$;
3. $I((\forall x)C)[\vartheta] = \inf \{ I(C)[\vartheta'] \mid \vartheta' \ x\text{-equivalent to } \vartheta \}$ where $p$ is a predicate symbol, $A$ and $A_i$ atomic formulas and $Q$ any body, $C$ any clause, $T$ is any left-continuous t-norm defined on $L([0,1])$. An assignment $\vartheta'$ is $x$-equivalent to $\vartheta$ when $z\vartheta' = z\vartheta$ for all variables $z \neq x$ in $V$.

**Definition 3.4.** Let $L$ be a first order language. The Herbrand universe $U_L$ for $L$, is the set of all ground terms, which can be formed out of the constants and function symbols appearing in $L$.

**Definition 3.5.** Let $L$ be a first order language. The Herbrand base $B_L$ for $L$ is the set of all ground atoms which can be formed by using predicate symbols from $L$ with ground terms from the Herbrand universe as arguments.

**Example 4.** Let us consider again the language $L$ generated by the program $\Pi$ of Example 3, the Herbrand universe $U_L = \{a, b, c\}$ and the Herbrand base: $B_L = \{p(a), p(b), p(c), q(a), q(b), q(c), r(a), r(b), r(c)\}$.

It is well-known that, in the classical case, it is possible to identify a Herbrand interpretation with a subset of the Herbrand base. Therefore, a convenient generalization of the notion of Herbrand interpretation to the interval-valued fuzzy case consists in establishing an interval-valued fuzzy Herbrand interpretation as an interval-valued fuzzy subset of the Herbrand base.

**Definition 3.6 (Interval-valued fuzzy interpretation).** Given, a first order language $L$, an interval-valued fuzzy Herbrand interpretation for $L$ is a mapping $I : B_L \rightarrow L([0,1])$.

Hence, the truth value of a ground atom $A \in B_L$ is $I(A)$. Sometimes we will represent an interval-valued fuzzy Herbrand interpretation $I$ extensively: as a set of pairs $\{(A, [\underline{a}, \overline{a}]) \mid A \in B_L \text{ and } [\underline{a}, \overline{a}] = I(A)\}$.

Now, we introduce the notion of Interval-valued Fuzzy Herbrand Model, which is formalised in Definitions 3.7 and 3.8. We employ a declarative
semantics based on a threshold $[\lambda, \overline{\lambda}]$. Intuitively, a threshold $[\lambda, \overline{\lambda}]$ is delimiting truth degrees equal or greater that $[\lambda, \overline{\lambda}]$ as true. Therefore, we are going to speak of Interval-valued Fuzzy Herbrand Model at level $[\lambda, \overline{\lambda}]$ or simply $[\lambda, \overline{\lambda}]$-model.

**Definition 3.7.** An Interval-valued fuzzy Herbrand Interpretation is a $[\lambda, \overline{\lambda}]$-model of an interval-valued fuzzy clause $C[\alpha, \overline{\alpha}]$ if and only if $I(C) \geq [\alpha, \overline{\alpha}] \geq [\lambda, \overline{\lambda}]$.

**Definition 3.8.** An Interval-valued fuzzy Herbrand Interpretation is a $[\lambda, \overline{\lambda}]$-model of an interval-valued fuzzy program $\Pi$ if and only if $I$ is a $[\lambda, \overline{\lambda}]$-model for each clause $C[\alpha, \overline{\alpha}] \in \Pi$.

**Theorem 3.9.** Let $\Pi$ be an Interval-valued fuzzy program and suppose $\Pi$ has a $[\lambda, \overline{\lambda}]$-model. Then $\Pi$ has a Herbrand $[\lambda, \overline{\lambda}]$-model.

**Proof 1.** Suppose that $M$ is a $[\lambda, \overline{\lambda}]$-model of $\Pi$. Let $M'$ be an Interval-valued fuzzy Herbrand interpretation: $M' = \{A \in B_{\Pi} \mid M(A) \geq [\lambda, \overline{\lambda}]\}$. We are going to prove that this interpretation is a $[\lambda, \overline{\lambda}]$-model for all clauses of $\Pi$. Let $C$ any clause, by initial supposition and by definition of $[\lambda, \overline{\lambda}]$-model for an interval-valued fuzzy program, we have that:

$$C \equiv \forall x_1, \ldots, x_n(p(x_1, \ldots, x_n) \leftrightarrow q_1(x_1, \ldots, x_n) \land \ldots \land q_m(x_1, \ldots, x_n)) [\beta, \overline{\beta}]$$

$M$ is a $[\lambda, \overline{\lambda}]$-model of $C$ iff $\forall a_1, \ldots, a_n \in U_L, M(C) \geq [\beta, \overline{\beta}] \geq [\lambda, \overline{\lambda}]$. Let $a_1, \ldots, a_n \in U_L$ then we have that $M(p(a_1, \ldots, a_n)) = [\beta, \overline{\beta}] \geq [\lambda, \overline{\lambda}]$ what implies that $M'(p(a_1, \ldots, a_n)) \geq [\lambda, \overline{\lambda}]$

**Definition 3.10.** Let $\Pi$ be an interval-valued fuzzy program. Let $A$ be an interval-valued fuzzy clause of $\Pi$. Then $A$ is a logical consequence of $\Pi$ at level $[\lambda, \overline{\lambda}]$ if and only if for each interval-valued fuzzy interpretation $I$, if $I$ is a $[\lambda, \overline{\lambda}]$-model for $\Pi$ then $I$ is a $[\lambda, \overline{\lambda}]$-model for $A$.

**Proposition 3.11.** $A$ is a logical consequence of an interval-valued fuzzy program $\Pi$ at level $[\lambda, \overline{\lambda}]$ if and only if for every interval-valued fuzzy Herbrand interpretation $I$ for $\Pi$, if $I$ is a $[\lambda, \overline{\lambda}]$-model for $\Pi$, it is an interval-valued fuzzy Herbrand $[\lambda, \overline{\lambda}]$-model for $A$. 

11
Proof 2. First, let us suppose that \( A \) is a logical consequence for \( \Pi \) at level \( [\lambda, \overline{\lambda}] \), then, by definition, for any interval-valued fuzzy interpretation \( \mathcal{I} \) if \( \mathcal{I} \) is \( [\lambda, \overline{\lambda}] \)-model for \( \Pi \), it is a \( [\lambda, \overline{\lambda}] \)-model for \( A \). Moreover, by the Theorem 3.9, there must exist \( \mathcal{I}' \) which being an interval-valued fuzzy Herbrand model for \( \Pi \) at level \( [\lambda, \overline{\lambda}] \), it is a \( [\lambda, \overline{\lambda}] \)-model for \( A \). This establishes the first side of the argument. Now, we have that for every interpretation \( \mathcal{I} \), if \( \mathcal{I} \) is a Herbrand model for \( \Pi \) at level \( [\lambda, \overline{\lambda}] \), it is a Herbrand \( [\lambda, \overline{\lambda}] \)-model for \( A \). Let \( \mathcal{M} \) be an interpretation, not necessarily Herbrand, which is a \( [\lambda, \overline{\lambda}] \)-model for \( \Pi \). We have that: 

\[
\mathcal{M}' = \left\{ \alpha \mid \mathcal{M}(\alpha) \geq [\lambda, \overline{\lambda}] \right\}
\]

and by the Theorem 3.9 \( \mathcal{M}' \) is a \( [\lambda, \overline{\lambda}] \)-model for \( \Pi \). And so it is for \( A \). So, \( \mathcal{M} \) is a \( [\lambda, \overline{\lambda}] \)-model for all ground instances \( A' \) of \( A \). As result \( \mathcal{M} \) is a \( [\lambda, \overline{\lambda}] \)-model for \( A' \), hence for \( A \) and \( A' \). This establishes the other side of the argument.

The ordering \( \leq \) in the lattice \( L([0,1]) \) can be extended to the set of interval-valued fuzzy interpretation as follows: \( I_1 \leq I_2 \) iff \( I_1(A) \leq I_2(A) \) for all interval-valued fuzzy atom \( A \in B_L \). It is important note that the pair \( \langle H^{IVF}, \subseteq \rangle \) is a complete lattice. Then it comes equipped with t-norms and t-conorms, that is, \( T(I_1, I_2) \) is an interval-valued fuzzy interpretation for all \( A \in B_L \), and \( t(I_1, I_2) \) an interval-valued fuzzy interpretation for all \( A \in B_L \). Therefore, the top element of this lattice is \( \langle A, [1,1] \rangle \) with \( A \in B_L \) and the bottom element is \( \langle A, [0,0] \rangle \) with \( A \in B_L \).

Interval-valued fuzzy interpretations have an important property which allow us to characterize the semantics of an interval-valued fuzzy program \( \Pi \).

Definition 3.12. If \( M_1 \) is a model of \( \Pi \) at level \( [\lambda_1, \overline{\lambda}_1] \) and \( M_2 \) is a model of \( \Pi \) at level \( [\lambda_2, \overline{\lambda}_2] \), then \( M_1 \cap M_2 \) contains the interval-valued fuzzy atom in both \( M_1 \) and \( M_2 \) but to degree \( \min([\lambda_1, \overline{\lambda}_1], [\lambda_2, \overline{\lambda}_2]) \).

Proposition 3.13 (Intersection Property of Models: Min-Model). Let \( \Pi \) be an interval-valued fuzzy program. Let \( \mathcal{M}_1, \ldots, \mathcal{M}_n \) be a non-empty set of
model for $\Pi$ at levels $[\lambda_1, \lambda_1] \ldots [\lambda_n, \lambda_n]$, respectively. Then $\bigcap(M_1, \ldots, M_n) \geq \min([\lambda_1, \lambda_1] \ldots [\lambda_n, \lambda_n])$ is a min-model for $\Pi$.

Proof 3. We prove this proposition by induction on the number of interpretations $i$:

1. **Base Case** ($i=2$). Let $M_1$ and $M_2$ be models for $\Pi$ at levels $[\lambda_1, \lambda_1]$ and $[\lambda_2, \lambda_2]$. Then for all interval-valued fuzzy clause $C$, $M_1(C) \geq [\lambda_1, \lambda_1]$ and $M_2(C) \geq [\lambda_2, \lambda_2]$, so $M_1 \cap M_2$ is a $\min([\lambda_1, \lambda_1], [\lambda_2, \lambda_2])$-model for $\Pi$;

2. **Inductive Case** ($i=n$). Let $M_1, M_2, \ldots M_n$ be models for $\Pi$ at levels $[\lambda_1, \lambda_1] \ldots [\lambda_n, \lambda_n]$. Then for all interval-valued fuzzy clause $C$, $M_i(C) \geq \min([\lambda_i, \lambda_i])$, so by the properties of the minimum.

**Definition 3.14.** Let $\Pi$ be an interval-valued fuzzy program. The least model for $\Pi$ is defined as follows: $M = \bigcap\{ I(A) \geq [\lambda, \lambda] | A \in B_L \}$. We call it a min-interval-valued fuzzy degree $[\lambda, \lambda]_{\min}$.

**Theorem 3.15.** Let $\Pi$ an interval-valued fuzzy program. Let $M$ be the least model of $\Pi$. Let $A \in B_L$ a ground atom of the interval-valued fuzzy Herbrand base. $M(A) \geq [\lambda, \lambda]_{\min}$ if and only if $A$ is logical consequence of $\Pi$ at level $[\lambda, \lambda]_{\min}$.

**Proof 4.** First, by definition $M = \bigcap\{ I(A) \geq [\lambda, \lambda] | A \in B_L \}$. Hence, for all model $I$ of $\Pi$, $I(A) \geq M(A) \geq [\lambda, \lambda]_{\min}$. That is, $A$ is a logical consequence for $\Pi$ at level $[\lambda, \lambda]_{\min}$. This establishes the first side of the argument. Now, If $A$ is a logical consequence of $\Pi$ by definition all model $I$ for $\Pi$, $I$ is a $[\lambda, \lambda]_{\min}$-model for $A$. That is, $I(A) \geq [\lambda, \lambda]_{\min}$. Therefore, by definition of least model, $\bigcap(I(A)) \geq [\lambda, \lambda]$ what implies that $M \geq [\lambda, \lambda]_{\min}$. This establishes the another side of the argument.

3.3. Fixpoint Semantics

In this section, we give a deeper characterisation of the least Herbrand model for an interval-valued fuzzy program $\Pi$ using fixpoint concepts.

This is possible because of each interval-valued fuzzy program has associated a complete lattice of interval-valued fuzzy Herbrand interpretations and
we can define a continuous operator on that lattice. This allows us to provide
a constructive vision of the meaning of a program by defining an immediate
consequences operator and to construct the least Herbrand model by means
of successive applications.

Definition 3.16 (Fixpoint Characterization of the least Herbrand model).
Let \( \Pi \) be an interval-valued fuzzy program, the mapping \( O : 2^{B_L} \rightarrow 2^{B_L} \) is
defined as follows. Let \( \mathcal{I} \) be an interval-valued fuzzy Herbrand interpretation,
then:

\[
O = \{ A \in B_L : A \leftarrow B_1, \ldots, B_n[\alpha, \overline{\alpha}] \text{ is a ground instance of a clause in } \Pi \\
\text{ and } \mathcal{I}(B_i) \geq [\alpha, \overline{\alpha}] \geq [\lambda, \overline{\lambda}] \text{ where } \mathcal{I}(A) \geq \inf(\mathcal{I}(B_1, \ldots, B_n))\}
\]

As in the case of classical logic programming, interval-valued fuzzy Her-
brand interpretations which are models can be characterised in terms of the
operator \( O \).

Theorem 3.17. Let \( \Pi \) be an interval-valued fuzzy program. Let \( \mathcal{I} \) be an
interval-valued fuzzy Herbrand interpretation of \( \Pi \). \( \mathcal{I} \) is \([\lambda, \overline{\lambda}]\)-model for \( \Pi \) if
and only if \( O(\mathcal{I}) \subseteq \mathcal{I} \).

Proof 5. \( \mathcal{I} \) is a \([\lambda, \overline{\lambda}]\)-model for \( \Pi \) if and only if for all clause \( \mathcal{C} \) in \( \Pi \) then
\( \mathcal{I}(\mathcal{C}) \geq [\lambda, \overline{\lambda}] \). Therefore, it is fulfilled if and only if for every variable assign-
ment \( \vartheta \), \( \mathcal{I}(\mathcal{C}\vartheta) \geq [\lambda, \overline{\lambda}] \). Therefore, supposing without loss of generality that
\( \mathcal{C} \equiv A \leftarrow B_1, \ldots, B_n[\alpha, \overline{\alpha}] \) then \( \mathcal{I}(A \leftarrow B_1, \ldots, B_n\vartheta) \geq [\lambda, \overline{\lambda}] \), by the
properties of the t-norm minimun \( \mathcal{I}(B_1, \ldots, B_n) \geq [\alpha, \overline{\alpha}] \geq [\lambda, \overline{\lambda}] \) that implies
that \( \mathcal{I}(B_1, \ldots, B_n) \subseteq O \) and hence \( \mathcal{I}(A\vartheta) \subseteq O(\mathcal{I}) \), again by the properties
of the t-norm minimun \( \mathcal{I}(A\vartheta) \geq \inf(\mathcal{I}(B_1\vartheta), \ldots, \mathcal{I}(B_n\vartheta)) \) what implies that
\( O(\mathcal{I}) \subseteq \mathcal{I} \)

Now we are ready to demonstrate the main theorem of this subsection,
but first we recall the following results from fixpoint theory.
Theorem 3.18 (FixPoint Theorem). Let \( \langle L, \leq \rangle \) be a complete lattice and \( O : L \rightarrow L \) be a monotonic mapping. Then \( O \) has a least fixpoint \( \text{lfp}(O) = \inf \{ x \mid O(x) = x \} = \inf \{ x \mid T(x) \leq x \} \).

Proposition 3.19. Let \( \langle L, \leq \rangle \) be a complete lattice and \( O : L \rightarrow L \) be a continuous mapping. Then \( \text{lfp}(O) = O^\uparrow \omega \).

Proof 6. See [19]

Theorem 3.20. Let \( \Pi \) be an interval-valued fuzzy definite program. Then \( M = \text{lfp}(\mathcal{O}^\Pi) = O^\uparrow \omega \).

Proof 7. \( M \) is the least model which is the intersection of any \([\Delta, \Xi]\)-model for \( \Pi \). As the lattice of interval-valued fuzzy Herbrand models is a complete one, then we can use the Theorem 3.18 the Proposition 3.19 and the Theorem 3.17. Applying them and the continuity of \( \mathcal{O} \) establishes the theorem.

Example 5. Given the program \( \Pi \) of Example 3, the least Herbrand model for \( \Pi \):

\[
\begin{align*}
O \uparrow 0 &= I_\perp; \\
O \uparrow 1 &= O(O \uparrow 0) = \{ \langle p(a), [0, 0] \rangle, \langle p(b), [0, 0] \rangle, \langle q(a), [0.8, 0.9] \rangle, \langle q(a), [0.7, 0.8] \rangle \} \\
O \uparrow 2 &= O(O \uparrow 1) = \{ \langle p(a), [0.8, 0.9] \rangle, \langle p(b), [0.7, 0.8] \rangle, \langle q(a), [0.8, 0.9] \rangle, \langle q(a), [0.7, 0.8] \rangle \} \\
O \uparrow 3 &= O \uparrow 2
\end{align*}
\]

Therefore, as the fixpoint is reached at the next item: \( M = O \uparrow 2 \)

3.4. Operational Semantics

We begin by providing definitions of an interval-valued SLD-derivation and an interval-valued fuzzy SLD-refutation that will be used later for showing the soundness and the completeness of the system.
Definition 3.21. Let \( G \) be \( A_1, \ldots, A_m, \ldots, A_k \) and \( C \) be either \( A[\alpha, \overline{\alpha}] \) or \( A \leftarrow B_1, \ldots, B_q[\beta, \overline{\beta}] \). Then \( G' \) is derived from \( G \) and \( C \) using \( \text{mgu} \ \theta \) if the following conditions hold (\( G' \) is the interval-fuzzy resolvent of \( G \) and \( C \)): i) \( A_m \) is an atom called the selected atom in \( G \); ii) \( \theta \) is a mgu of \( A_m \) and \( A \); iii) \( G' \) is the interval-valued fuzzy goal \( \leftarrow (A_1, \ldots, B_1, \ldots, B_q, \ldots, A_k) \theta \) with \( [\alpha_{G'}, \overline{\alpha_{G'}}] = \min([\alpha_C, \overline{\alpha_C}], [\alpha_G, \overline{\alpha_G}]) \)

Definition 3.22. An interval-valued fuzzy SLD-derivation of \( \Pi \cup G \) is a successful interval-valued SLD-derivation of \( \Pi \cup G \) which has the empty clause as the last goal in the derivation. If \( G_n \) is the empty clause, we say that the derivation has length \( n \). The empty clause is derived from \( \leftarrow (A_1, \ldots, A_m, \ldots, A_k)[\alpha_G, \overline{\alpha_G}] \) and \( A(t_1, \ldots, t_q)[\alpha_A, \overline{\alpha_A}] \leftarrow [\alpha_{G_n}, \overline{\alpha_{G_n}}] = \min([\alpha_A, \overline{\alpha_A}], [\alpha_G, \overline{\alpha_G}]) \)

Definition 3.23. Let \( \Pi \) be an interval-valued fuzzy program and \( G \) be an interval-valued fuzzy goal. An interval-valued fuzzy computed answer \( \langle \theta, [\beta, \overline{\beta}] \rangle \) for \( \Pi \cup G \) is the substitution obtained by restricting the composition \( \theta_1, \ldots, \theta_n \) to the variables of \( G \), where \( \theta_1, \ldots, \theta_n \) is the sequence of mgus employed in the finite interval-valued fuzzy SLD-derivation of \( \Pi \cup G \) with an interval-valued approximation degree \( [\beta, \overline{\beta}] \)

Definition 3.24. Let \( \Pi \) be an interval-valued fuzzy program, \( G \) be an interval-valued fuzzy goal \( \leftarrow (A_1, \ldots, A_k) \) and \( \langle \theta, [\beta, \overline{\beta}] \rangle \) be an answer for \( \Pi \cup G \). We say that \( \langle \theta, [\beta, \overline{\beta}] \rangle \) is an interval-valued fuzzy correct answer if \( \forall (A_1, \ldots, A_k) \theta \) is a logical consequence of \( \Pi \) at level \( [\lambda, \overline{\lambda}]_{\min} \), that is, \( [\beta, \overline{\beta}] \geq [\lambda, \overline{\lambda}]_{\min} \).

3.5. Implementation

In this section, we briefly explain how interval-valued fuzzy sets are incorporated into the Bousi-Prolog system\(^2\). Here, we describe the structure and main features of its abstract machine. It was created as extension of the SWAM for the execution of Bousi-Prolog programs. We have appropriately modified the compiler, some machine instructions and SWAM structures in

\(^2\)A beta version can be founded at the URL: http://www.face.ubiobio.cl/~clrubio/bousiTools/
order to trigger the interval-valued fuzzy resolution. It is worth noting that, to the best of our knowledge, this is the first SWAM implementation that supports interval-valued fuzzy resolution.

A mandatory step to achieve this result is to include a new data structure into the architecture for computing with interval-valued fuzzy sets. This data structure has been implemented by using a class called `IntervalFS` which is formed by two private attributes of double type: `upper_limit`, `lower_limit`. We define the public method constructor `IntervalFS(double l1, double l2)` and the four methods (sets and gets): `double getUpperLimit(); double getLowerLimit(); void setUpperLimit(double v); void setLowerLimit(double v)`. Additionally, we overwrite both the `toString` and the equals methods in the usual way. Finally, methods for adding, subtracting and computing minimum of interval valued fuzzy set are implemented: `IntervalFS add(IntervalFS a, IntervalFS b); IntervalFS subtract(IntervalFS a, IntervalFS b); IntervalFS min(IntervalFS a, IntervalFS b).

The following example illustrates the new features of the SWAM enhanced with IVFSs.

**Example 6.** Let us suppose that we want to represent the following knowledge: a football player is good when he is fast, tall and coordinated. We know a particular player that is fast, quite tall but he is not very coordinated. Thus, is he a good player? Answering this question and in this scenario, the linguistic expression “is not very coordinate” could be represented by the fact “coordinate(a) [0.2, 0.4]”, the linguistic term “fast” could be represented by the fact “fast(a) [0.9, 1.0]” and “quite tall” could be represented by the fact “tall(a) [0.8, 0.9]”. A possible solution by employing a Bousi-Prolog program is described as follows:
%% FACTS
coordinate(a) [0.2,0.4]
fast(a) [0.9,1.0]
tall(a )[0.8,0.9]

%% RULES
good_player(X):-tall(X), fast(X), coordinate(X)

The SWAM enhanced with IVFSs allows us to obtain the answer: “X=a with [0.2,0.4]”. The SWAM code generated for this program is as follows:

00:good_player:trust_me [1.0,1.0]
01: allocate
02: get_variable Y0 A0
03: put_value Y0 A0
04: call coordinate (11)
05: put_value Y0 A0
06: call fast (14)
07: put_value Y0 A0
08: call tall (17)
09: deallocate
10: proceed
11:coordinate:trust_me [0.2,0.4]
12: get_constant a A0
13: proceed
14: fast: trust_me [0.9,1.0]
15: get_constant a A0
16: proceed
17:tall: trust_me [0.8,0.9]
18: get_constant a A0
19: proceed
20:query: trust_me
21: create_variable Q0 X
22: put_value Q0 A0
23: call good_player (00)
24: halt

The first instruction to be executed is the one labelled with the key “query”, hence the execution starts at the position 20 with a degree $D = [1.0, 1.0]$ (which is fixed in the instruction trust_me). After that, from line 20 to line 23 the query is launched and the variable $X$ is created (create_variable instruction). After that from line 00 to line 04 the first subgoal (coordinate(X)) is launched, then the execution goes to line 11 and the unification with the term "coordinate(a) is produced (from line 11 to 13) (put_value
and *get\_constant* instructions), a new approximation degree is established
\[ D = \min([1.0, 1.0], [0.2, 0.4]) \] (trust\_me instruction), as these terms unify the
following subgoal (*fast(X)*, line 05 and from line 14 to line 16) is launched
with an approximation degree \[ D = \min([0.2, 0.4], [0.9, 1.0]) \]; as the terms
unify, then the following subgoal (*tall(X)*, line 08 and from line 17 to 19)
is launched with an approximation degree \[ D = \min([0.2, 0.4], [0.9, 1.0]) \]. Fi-
nally, the assignation \[ X = a \] with [0.2, 0.4] is produced.

We have implemented a limit to the expansion of the search space in a
computation by what we called a “\(\lambda\)-cut for IVFSs”. When the LambdaC-
putIVFS flag is set to a value different than [0.0, 0.0], the weak unification
process fails if the computed approximation degree goes below the stored
LambdaCutIVFS value. Therefore, the computation also fails and all possi-
ble branches starting from that choice point are discarded. By default the
LambdaCutIVFS value is [0.0, 0.0]. However, the lambda cut flag can be set
to a different value by means of a \(\lambda\)-cut directive: “:-lambdaCutIVFS(N).”,
where N is an interval between [0.0, 0.0] and [1.0, 1.0]. For example, a \(\lambda\)-
cut of [0.5, 0.5] could be established by using the following directive: “:-
lambdaCutIVFS([0.5,0.5])”.

4. Applications

The main realms for the application of the IVFSs programming language
described in this paper are those which involve natural language semantics
processing. In this section, we will discuss two of them: linguistic knowledge
modelling and proximity-based logic programming using linguistic resources.

4.1. Linguistic Knowledge Modelling

Linguistic knowledge modelling handles the computational representation of knowledge that is embedded in natural language. This framework can be enhanced by combining multiadjoint paradigm with interval-valued fuzzy sets [14]. For example, we can define interval-valued annotated atoms. Let us assume the same definition of suitable journal given in [22], that is, a journal with a high impact factor, a medium immediacy index, a relatively big half-life and with a not bad position in the listing of the category. Now, we introduce in the program the following inference rule:

\[
\text{suitable\_journal}(X) :\neg \text{impact\_factor}(X)[0.8,0.9], \\
\text{immediacy\_index}(X)[0.4,0.6], \\
\text{cited\_half\_life}(X)[0.6,0.7], \\
\text{best\_position}(X)[0.4,0.6].
\]

Now, let us suppose the IEEE Transactions of Fuzzy System journal has the following properties: “high” impact factor, “small” immediacy index, “relatively small” cited half life and the “best position”. Regarding the linguistic variables: “high”, “medium”, “relatively big” and “not bad”, which can be related to the following truth-values: [0.8, 0.9], [0.4, 0.6], [0.6, 0.7] and [0.4, 0.6], respectively, considering the variables “medium” and “not a bad” with a similar meaning. This knowledge could be model in an interval-valued fuzzy logic language as follows:
When the query “suitable\_journal(X)” is launched, then the system answers: “X = ieee\_fs” with [0.3, 0.6].

4.2. Proximity-based Logic Programming based on WordNet

Proximity-based Logic Programming is a framework that provides us with the capability of enriching semantically classical logic programming languages by using Proximity Equations (PEs). A limitation of this approach is that PEs are mostly defined for a specific domain [6, 23], being the designer who manually fixes the values of these equations. This fact makes harder to use PLP systems in real applications.

A possible solution consists in obtaining the proximity equations from WordNet which requires to employ interval-valued fuzzy sets in order to deal with the high uncertainty generated by the possibility of using several
different semantic similarity metrics. Let us assume a fragment of a deductive
database that stores information about people and their preferences. The
proximity equations can be generated from WordNet, we only put here some
of them (see [4] for more detail).

%% m loves mountaineering
loves(mary,mountaineering).

%% j likes football
likes(john,football).

%% peter plays basketball
plays(peter,basketball).

%% if a person practises sports
%% the he/she is a healthy person
healthy(X):- practices(X,sport).

%% automatically generated from wordnet
love~passion=[0.25,0.8].
basketball~hoops=[1,1].
play~act=[0.25,0.7].
practice~rehearse=[1,1].
sport~variation=[0.1,0.5].
sport~fun=[0.3,0.8].
5. Related Work

In the literature, other proposals that address our same goal can be found [24, 25]. One of the most relevant ones is Ciao-Prolog [25] and, for that reason, we will analyse in detail the differences between it and Bousi-Prolog in order to clarify and reinforce the novelty of our proposal:

- **From the point of view of its implementation.** In Ciao-Prolog, IVFSs are included by means of constrains and hence a translator must be implemented. As a result, the programmer must code the variables in order to manage the truth values and get the answers from the system based on those constraints. In Bousi-Prolog, on the other hand, IVFSs are included in a different way, where the compiler and the warren abstract machine are enhanced by using a IVFSs data structure which has been created and adapted for this architecture. As a result, intervals work as a standard data structure in the code of the program instead of a particular set of variables defined ad hoc by the programmer. This feature allows us to include IVFSs in both fuzzy unification (see [4]) and fuzzy resolution. In addition, this framework also allows other possible extensions, such as the incorporation of a reasoning module using WordNet (see [4]).

- **From the point of view of its syntax.** Ciao-Prolog and Bousi-Prolog, although both are Prolog languages, they have a well differentiated syntax. The former only allows the annotation of facts, rules
cannot be annotated because these only allow the use of an aggregator operator for the computing of the annotated IVFSs. The latter, on the other hand, allows the user both the annotation the fact and rules by means of IVFSs. In addition, if we focus on the inference engine, while Ciao-Prolog only extends the resolution mechanism, Bousi-Prolog uses interval-valued proximity equations (e.g., young teenager=[0.6,0.8]), which extends both the resolution and unification process.

- **From the point of view of its semantics.** Ciao-Prolog and Bousi-Prolog have relevant differences at the semantic levels as well. Firstly, Bousi-Prolog implements the concept of cut-level, which allows to the user imposes a threshold in the system, and according to it you can be as precise as you want in your answer. This is a substantial change due to the introduction of a threshold operational semantics. Therefore, our operational mechanism behaves very much as the one of a Prolog system (obtaining correct answers one by one), while this option is not available in Ciao semantics. As we mentioned in section 3.5, a \( \lambda \)-cut for IVFS approximation degrees has been implemented. The concepts of interpretation, least model semantics, model, so on, are presented and defined in a different way, in Bousi-Prolog the operational semantics is based an extension of SLD Resolution. In [25] the type of resolution is based on the classical SLD Resolution of Prolog Systems.
6. Conclusions and future work

We have formally defined and efficiently implemented a simple interval-valued fuzzy programming language using interval-valued fuzzy sets for modelling the uncertainty and imprecision of the knowledge associated to lexical resources. As future work, we propose to extend our language and to provide results of soundness and completeness. Additionally, we want to develop a fully integrated framework in which interval-valued fuzzy sets and interval-valued fuzzy relations can be combined in a same framework.

Acknowledgements

The authors gratefully acknowledge the comments made by reviewers. This work has been partially supported by FEDER and the State Research Agency (AEI) of the Spanish Ministry of Economy and Competition under grants TIN2016-76843-C4-2-R (AEI/FEDER, UE) and TIN2014-56633-C3-1-R, the Consellería de Cultura, Educación e Ordenación Universitaria (the Postdoctoral Training Grants 2016 and Centro singular de investigación de Galicia accreditation 2016-2019, ED431G/08) and European Regional Development Fund (ERDF). This work has been done in collaboration with the research group SOMOS (SOftware-MOdelling-Science) funded by the Research Agency and the Graduate School of Management of the Bio-Bío University.

References

[1] Miller, G.A.: Wordnet: A lexical database for english. Communications of the ACM 38(11) (1995) 39–41
[2] Liu, H., Singh, P.: Commonsense reasoning in and over natural language. In Negoita, M.G., Howlett, R.J., Jain, L.C., eds.: Proceedings of the 8th International Conference on Knowledge-Based Intelligent Information and Engineering Systems (KES 2004), Wellington, New Zealand, September 20-25, Berlin, Heidelberg, Springer (2004) 293–306

[3] León-Aráuz, P., Gómez-Romero, J., Bobillo, F.: A fuzzy ontology extension of wordnet and eurowordnet for specialized knowledge. In: Proceedings of the TKE 2012. (2012)

[4] Rubio-Manzano, C., Pereira-Fariña, M.: Towards fuzzy lexical reasoning. Journal of Intelligent & Fuzzy Systems 32(3) (Feb 2017) 2425–2436

[5] Medina, J., Ojeda-Aciego, M.: Multi-adjoint logic programming. In: Proceedings of the 10th International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems (IPMU 2004). (2004) 823–830

[6] Julián-Iranzo, P., Moreno, G., Penabad, J., Vázquez, C. In: A Declarative Semantics for a Fuzzy Logic Language Managing Similarities and Truth Degrees. Springer International Publishing, Cham (2016) 68–82

[7] Julián-Iranzo, P., Moreno, G., Penabad, J., Vázquez, C.: A fuzzy logic programming environment for managing similarity and truth degrees. (2015)
[8] Muñoz-Hernandez, S., Pablos-Ceruelo, V., Strass, H.: Rfuzzy: Syntax, semantics and implementation details of a simple and expressive fuzzy tool over prolog. Information Sciences 181(10) (2011) 1951 – 1970 Special Issue on Information Engineering Applications Based on Lattices.

[9] Straccia, U. In: Managing Uncertainty and Vagueness in Description Logics, Logic Programs and Description Logic Programs. Springer Berlin Heidelberg, Berlin, Heidelberg (2008) 54–103

[10] Le, V.H., Liu, F., Tran, D.K.: Fuzzy linguistic logic programming and its applications. Theory and Practice of Logic Programming 9(03) (may 2009) 309–341

[11] Pedersen, T., Patwardhan, S., Michelizzi, J.: Wordnet::similarity: Measuring the relatedness of concepts. In: Demonstration Papers at HLT-NAACL 2004. HLT-NAACL–Demonstrations ’04, Stroudsburg, PA, USA, Association for Computational Linguistics (2004) 38–41

[12] Turksen, I.: Four methods of approximate reasoning with interval-valued fuzzy sets. International Journal of Approximate Reasoning 3(2) (mar 1989) 121–142

[13] Bustince, H.: Interval-valued fuzzy sets in soft computing. International Journal of Computational Intelligence Systems 3(2) (2010) 215–222

[14] Medina, J.: Adjoint pairs on interval-valued fuzzy sets. In: Proceedings of the 13th International Conference on Information Processing and
[15] Rubio-Manzano, C., Julián-Iranzo, P.: A fuzzy linguistic prolog and its applications. Journal of Intelligent & Fuzzy Systems 26(3) (2014) 1503–1516

[16] Julián-Iranzo, P., Rubio-Manzano, C. In: A Similarity-Based WAM for Bousi-Prolog. Springer Berlin Heidelberg, Berlin, Heidelberg (2009) 245–252

[17] Pereira-Fariña, M.: Some reflections on the set-based and the conditional-based interpretations of statements in syllogistic reasoning. Archives for the Philosophy and History of Soft Computing 1/2014(1) (2014) 1–16

[18] Hajek, P.: Metamathematics of fuzzy logic. Springer Science and Business Media (1998)

[19] Lloyd, J.: Foundations of Logic Programming. Springer-Verlag, Berlin (1987)

[20] Julián-Iranzo, P., Rubio-Manzano, C.: A declarative semantics for bousi prolog. In: Proceedings of the 11th ACM SIGPLAN Conference on Principles and Practice of Declarative Programming. PPDP ’09, New York, NY, USA, ACM (2009) 149–160
[21] Ebrahim, R.: Fuzzy logic programming. Fuzzy Sets and Systems 117(2) (2001) 215 – 230

[22] Medina, J., Ojeda-Aciego, M., Ruíz-Calviño, J.: Formal concept analysis via multi-adjoint concept lattices. Fuzzy Sets and Systems 160(2) (2009) 130 – 144

[23] Rodríguez-Artalejo, M., Romero-Díaz, C.A.: A declarative semantics for clp with qualification and proximity. Theory and Practice of Logic Programming 10(4-6) (2010) 627642

[24] Atanassov, K., Georgiev, C.: Intuitionistic fuzzy prolog. Fuzzy Sets and Systems 53(2) (1993) 121 – 128

[25] Guadarrama, S., Muoz, S., Vaucheret, C.: Fuzzy prolog: a new approach using soft constraints propagation. Fuzzy Sets and Systems 144(1) (2004) 127 – 150