Available online at www.sciencedirect.com

SCIENCE DIRECT®

Fuzzy Sets and Systems 144 (2004) 43 – 65

FUZZY sets and systems

www.elsevier.com/locate/fss

Adding similarity-based reasoning capabilities to a Horn fragment of possibilistic logic with fuzzy constants

Teresa Alsinet[a,*], Lluís Godo[b]

[a]Department of Computer Science, Universitat de Lleida (UdL), Jaume II 69, Lleida 25001, Spain
[b]AI Research Institute (IIIA), CSIC, Bellaterra 08193, Spain

Abstract

PLFC is a first-order possibilistic logic dealing with fuzzy constants and fuzzily restricted quantifiers. The refutation proof method in PLFC is mainly based on a generalized resolution rule which allows an implicit graded unification among fuzzy constants. However, unification for precise object constants is classical. In order to use PLFC for similarity-based reasoning, in this paper we extend a Horn-rule sublogic of PLFC with similarity-based unification of object constants. The Horn-rule sublogic of PLFC we consider deals only with disjunctive fuzzy constants and it is equipped with a simple and efficient version of PLFC proof method. At the semantic level, it is extended by equipping each sort with a fuzzy similarity relation, and at the syntactic level, by fuzzily "enlarging" each non-fuzzy object constant in the antecedent of a Horn-rule by means of a fuzzy similarity relation.
© 2003 Elsevier B.V. All rights reserved.

Keywords: Possibilistic logic; Fuzzy constants; Horn-rule sublogic; Similarity-based unification

1. Introduction

Possibilistic logic [15] is a logic of uncertainty to reason with classical propositions under incomplete information and partially inconsistent knowledge. Formulas of the necessity-valued fragment of possibilistic logic are of the form (φ, α), where φ is a classical (propositional or first-order) formula and $\alpha \in [0, 1]$ is understood as a lower bound for the necessity degree of φ. To enhance the knowledge representation power, Dubois et al. [16,17] defined a syntactic extension of first-order Possibilistic logic (called PLFC) to deal with fuzzy constants and fuzzily restricted quantifiers inside the language, for which Alsinet et al. [4] defined a formal semantics and a sound resolution-style calculus by refutation. In PLFC, the resolution inference rule includes an *implicit* fuzzy unification

[*] Corresponding author. Tel.: +34-973-702-734; fax: +34-973-702-702.
E-mail addresses: tracy@eup.udl.es (T. Alsinet), godo@iiia.csic.es (L. Godo).

0165-0114/$ - see front matter © 2003 Elsevier B.V. All rights reserved.
doi:10.1016/j.fss.2003.10.013

mechanism between fuzzy constants. Alternatively, Alsinet and Godo defined in [1] a general fuzzy possibilistic logic (called PGL) based on Gödel infinitely-valued logic, and in [2] the Horn-rule sublogic of PGL was extended with fuzzy constants and a modus ponens-style calculus based on an *explicit* fuzzy unification mechanism between fuzzy constants, which was shown to be complete for a restricted class of Horn-rules [3]. Note that the issue of fuzzy unification, within different frameworks, has already been addressed in the literature by a number of relevant authors, for instance, starting in the 1980s by Cayrol et al. [11], Bel et al. [9] and Umano [27], and then following in the 1990s with Baldwin et al. [8], Virtanen [29], Rios–Filho and Sandri [25] and Arcelli et al. [7], and more recently by authors like Gerla and colleagues [18,19] and Vojtáš [30].

In both PLFC and PGL, fuzzy constants can be seen as (flexible) restrictions on an existential quantifier. For instance, in both systems, the fuzzy statement

"it is almost sure that Peter is *about_35* years old"

can be represented by a *certainty-weighted* formula of the form

$(age\_Peter(about\_35), 0.9)$,

where *age_Peter* is a classical predicate and *about_35* is a fuzzy constant defined over the domain $[0, 120]$ (years). In the case in which *about_35* denotes a crisp interval of ages, say $[34, 36]$, the *certainty-weighted* formula $(age\_Peter(about\_35),0.9)$ is interpreted in both systems as

"$\exists x \in [34, 36]$ such that $age\_Peter(x)$" is certain with a necessity of at least 0.9.

In the case in which *about_35* denotes a fuzzy interval with a membership function $\mu_{about\_35} : [0, 120] \rightarrow [0, 1]$, because in each system the necessity measure used for defining the possibilistic semantics is different, in each system the certainty-weighted formula has a different interpretation. In PLFC, it is interpreted as

"$\exists x \in [\mu_{about\_35}]_{0.9}$ such that $age\_Peter(x)$" is certain with a necessity of at least 0.9,

where $[\mu_{about\_35}]_{0.9}$ denotes the crisp interval of ages associated with the α-cut of the fuzzy set μ_{about_35} at the level of 0.9. In PLFC it is interpreted, for each $\alpha \in [0, 1]$, as

"$\exists x \in [\mu_{about\_35}]_{\alpha}$ such that $age\_Peter(x)$" is certain with a necessity of at least $\min(0.9, 1 - \alpha)$,

where $[\mu_{about\_35}]_{\alpha}$ denotes the α-cut of the fuzzy set μ_{about_35}.

In PLFC, the use of variable weights [13,14] is a suitable technique for modeling statements of the form

"the more x is A (or x belongs to A), the more certain is $p(x)$",

where A is a fuzzy set with membership function $\mu_A(x)$. This is formalized in PLFC as,

"for all x, $p(x)$ is certain with a necessity of at least $\mu_A(x)$"

and is represented as $(p(x), A(x))$. When A is imprecise but not fuzzy, the interpretation of such a formula is just "for all $x \in A$, $p(x)$". So variable weights in PLFC act as (flexible if they are fuzzy) restrictions on an universal quantifier, or if you prefer, as a kind of conjunctive constants. Notice that the notion of variable weight has not been introduced in PGL since all fuzzy constants in that system are interpreted disjunctively.

Concerning the unification mechanism, the matching degree between two object (fuzzy) constants is computed in terms of a necessity measure for fuzzy sets in both PLFC and PGL. Namely, given the following possibilistic clauses

$$\{(price\_book(A), 1),\\ (price\_book(B) \rightarrow buy\_book, 1)\},$$

where A and B are fuzzy constants of sort `price_euros` denoting *medium_price* and *not_expensive*, respectively, $price\_book(A)$ would unify with $price\_book(B)$ to the degree $N(B \mid A) = N(not\_expensive \mid medium\_price)$, where $N(\cdot|\cdot)$ is a necessity measure of matching between fuzzy events. Then, we would derive the clause $(buy\_book, N(B \mid A))$. However, the necessity measures N used in PLFC and in PGL are different. In PLFC, $N(B \mid A)$ is a necessity measure defined as $\inf_x \max(1 - A(x), B(x))$, whereas in PGL, $N(B|A)$ is another necessity measure defined as $\inf_x A(x) \Rightarrow B(x)$, where $\Rightarrow$ is the reciprocal of Gödel's many-valued implication.[1] Different degrees can be obtained therefore. For instance, when *medium_price* is modeled by the trapezoidal fuzzy set[2] $[25, 30, 40, 45]$, and *not_expensive* by another trapezoidal fuzzy set $[0, 0, 35, 50]$, then we get $N(not\_expensive \mid medium\_price) = \frac{1}{2}$ in PLFC but $N(not\_expensive \mid medium\_price) = 0$ in PGL.

A detailed comparison of the unification mechanisms in PLFC and in PGL can be found in [5], but in any case, the unification degree between two different and precise constants is null in both systems. Sometimes this is a rather unpleasant behavior, specially if we are trying to model approximate knowledge. To remedy this situation, in this paper we equip each basic sort with a fuzzy proximity relation in order to allow a kind of similarity-based unification [6,7,18] between object constants. For instance, if 34 and 35 are two precise object constants of a given sort `price_euros`, from the set of *certainty-weighted* formulas

$$\{(price\_book(34), 1),\\ (price\_book(35) \rightarrow buy\_book, 1)\},$$

considering that 34 is very close to 35, we would like to infer something of the form

$$(buy\_book, N(around(35) \mid 34)),$$

where $around(\cdot)$ is a fuzzy proximity relation attached to the sort `price_euros` which models the fuzzification of the precise object constant 35 and hence it would allow for the above similarity-based inference pattern

Within this framework, in this paper we tackle two main problems.

(i) We address the problem of similarity-based unification involving fuzzy constants in systems where a separation between general and specific knowledge patterns can be made [25]. Patterns classified in the first class are part of general information, like rules in expert systems, or ungrounded clauses in logic programming languages. The ones classified in the second class come from specific

[1] $x \Rightarrow y = 1$, if $x \leqslant y$, and $x \Rightarrow y = 1 - x$, otherwise.

[2] We use the representation of a trapezoidal fuzzy set as $[t_1; t_2; t_3; t_4]$, where the interval $[t_1, t_4]$ is the support and the interval $[t_2, t_3]$ is the core.

information about a problem, like facts in expert systems, or grounded clauses in logic programming languages. In this frame, the fuzzification mechanism should be only performed on object constants appearing in general patterns, otherwise, we would be adding vagueness to the specific patterns of the knowledge base, and thus, we would be reducing the unification degree between fuzzy events. Notice that in the example above we only fuzzify the constant 35 and not the constant 34.

(ii) Advantages of incorporating such fuzzification mechanism clearly depend on the kind of necessity measures used for computing the partial matching between fuzzy events. For instance, in PGL, in the usual case in which $around(35)(x)<1$ for $x\neq 35$, we would have $N(around(35)\mid 34)=N(35\mid 34)=0$, while in PLFC we would have $N(around(35)\mid 34)=\mu_{around(35)}(34)>0$.

These considerations lead us in this paper to extend a Horn-rule sublogic of PLFC (and not PGL) with a similarity-based unification. To do so we shall introduce two minor changes. At the semantic level, we shall equip each sort with a fuzzy similarity relation. At the syntactic level, we shall replace each non-fuzzy object constant appearing in the antecedent of a Horn-rule by a variable weight fuzzily "enlarged" by means of a fuzzy similarity relation. For instance, the previously considered set of PLFC clauses

$$\{(price\_book(34),1),\\ (price\_book(35)\rightarrow buy\_book,1)\},$$

will be transformed into

$$\{(price\_book(34),1),\\ (price\_book(x)\rightarrow buy\_book, around\_35(x))\},$$

where we take $\mu_{around\_35}(x)=S(35,x)$, with

$$S: D_{\texttt{price\_euros}}\times D_{\texttt{price\_euros}}\rightarrow[0,1]$$

being a fuzzy proximity (i.e. reflexive and symmetric) relation on the domain $D_{\texttt{price\_euros}}$ of the sort price_euros. Then, applying the PLFC resolution inference rule [4], we shall be able to infer

$$(buy\_book, N(around\_35\mid 34)),$$

where, in this case, we would have

$$N(around\_35|34)=\mu_{around\_35}(34)=S(35,34).$$

After this Introduction, the rest of the paper is organized as follows. In Section 2, we describe formal aspects of PLFC. In Section 3, we define a Horn-rule sublogic with only disjunctive fuzzy constants of PLFC and we provide this sublogic with a simple an efficient version of the PLFC refutation proof method and, in Section 5, we show how this proof method works by means of an example. Finally, in Section 5, we extend this sublogic with similarity-based reasoning capabilities.

2. Background on PLFC

As already mentioned, PLFC is an extension of possibilistic logic that provides a powerful framework for reasoning under possibilistic uncertainty and representing disjunctive and conjunctive vague knowledge. Following [4], a general PLFC clause is a pair of the form

$$(\varphi(\bar{x}), f(\bar{y})),$$

where $\bar{x}$ and $\bar{y}$ denote sets of free and implicitly universally quantified variables, each one having its sort, such that $\bar{y} \supseteq \bar{x}$; $\varphi(\bar{x})$, called *base formula*, is a disjunction of (positive and negative) literals with typed classical predicates and possibly with fuzzy constants, each one having its sort; and $f(\bar{y})$ is a valid valuation function, defined for a superset of the variables in the left-hand side, which expresses a lower bound of the certainty of $\varphi(\bar{x})$ in terms of necessity measures. Basically, valuation functions $f(\bar{y})$ are either constant values in the real unit interval $[0,1]$, or membership functions of fuzzy sets (fuzzy constants), or max–min combinations of them, or necessity measures on them. An example of PLFC clause is

$$(p(A,x) \vee q(y), \min(\alpha, B(x), C(y))),$$

where A, B and C are fuzzy constants.

Next, let us briefly recall a semantics for PLFC proposed in [4]. For a given language signature, a *many-valued interpretation* $w = (U, i, m)$ maps:

1. each sort σ into a non-empty domain U_σ of U;
2. a predicate p of type $(\sigma_1, \ldots, \sigma_n)$ into a *crisp* relation $i(p) \subseteq U_{\sigma_1} \times \cdots \times U_{\sigma_n}$; and
3. a (precise or fuzzy) object constant A of sort σ into a normalized fuzzy set $m(A)$ with membership function $\mu_{m(A)} : U_\sigma \to [0,1]$. When A is a precise constant c, then $\mu_{m(A)}$ will represent a singleton, an element of U_σ.

An *evaluation* of variables is a mapping e associating to each variable x of sort σ an element $e(x) \in U_\sigma$. The *truth value* of a base formula under an interpretation $w = (U, i, m)$ and an evaluation of variables e is defined by cases:

1. $w_e(p(x, \ldots, A)) = \sup_{(u,\ldots,v) \in i(p)} \min(\mu_{e(x)}(u), \ldots, \mu_{m(A)}(v))$.
2. $w_e(\neg p(x, \ldots, A)) = \sup_{(u,\ldots,v) \notin i(p)} \min(\mu_{e(x)}(u), \ldots, \mu_{m(A)}(v))$.
3. $w_e(L_1 \vee \cdots \vee L_r) = \max(w_e(L_1), \ldots, w_e(L_r))$, where $L_1, \ldots, L_r$ are (positive or negative) literals.

The present form of truth evaluation for a negative literal deserves some explanation. In PLFC, a negative literal like $\neg\, p(A)$, where A is an imprecise non-fuzzy constant, is to be interpreted as $(\exists x \in A) \neg\, p(x)$ and not as $\neg(\exists x \in A) p(x)$. The above definition is according with this interpretation, and makes $\neg$ to be non-truth functional under this semantics.

Finally, the *truth value* of a base formula φ under an interpretation w is defined as

$$w(\varphi) = \inf\{w_e(\varphi) \mid e \text{ is an evaluation of variables}\}.$$

Notice that $w(\varphi)$ may take any intermediate value between 0 and 1 as soon as φ contains some fuzzy constant and $w(\varphi)$ depends not only on the crisp relations assigned to predicate symbols, but on the fuzzy sets assigned to fuzzy constants.

So far we have defined the *many-valued* semantics for base formulas φ. Now we define the *possibilistic* semantics for certainty-weighted clauses (φ, α). In order to define such semantics, we need to fix a *context* and to consider some extension for fuzzy sets (of interpretations) of the standard notion of necessity measure. Basically a context is the set of interpretations sharing a common domain U and an interpretation of object constants m. So, given U and m, its associated context $\Omega_{U,m}$ is just the set $\{w \text{ interpretation} \mid w=(U,i,m)\}$. Now, for each possibility distribution on the context $\pi : \Omega_{U,m} \to [0,1]$, and each PLFC formula (φ, α), we define

$$\pi \models (\varphi, \alpha) \quad \text{iff } N([\varphi]|\pi) \geqslant \alpha,$$

where $N(\cdot \mid \pi)$ is the necessity measure induced by π on fuzzy sets of interpretations defined by

$$N([\varphi] \mid \pi) = \inf_{w \in \Omega_{U,m}} \max(1-\pi(w), w(\varphi)),$$

where we take $\mu_{[\varphi]}(w) = w(\varphi)$. Here, we have considered a PLFC formula with a constant weight α. If the formula has a variable weight, e.g. $(\varphi(x), A(x))$, then the above definition, always in the same context, extends to

$$\pi \models (\varphi(x), A(x)) \quad \text{iff } \pi \models (\varphi(c), \mu_{m(A)}(m(c)))$$

for all precise object constants c.

An interesting and remarkable consequence of the above notion of possibilistic satisfiability of PLFC clauses is the following one:

$$\pi \models (p(A) \vee q(B), \alpha) \quad \text{iff } \pi \models (p([A]_\alpha) \vee q([B]_\alpha), \alpha),$$

where p and q can be positive or negative literals, and $[A]_\alpha$ and $[B]_\alpha$ denote the imprecise constants corresponding to the α-cuts of the fuzzy constants A and B, respectively. This property has important consequences since it means that in PLFC with (only) fuzzy constants we can in a way forget about fuzzy constants as such and focus only on imprecise but crisp constants.

Notation convention: since we need to fix a context $\Omega_{U,m}$ in order to perform deduction, we can identify a fuzzy constant A with its interpreted fuzzy set $m(A)$ and also with its membership function $\mu_{m(A)}$. Hence, for the sake of a simpler notation, we shall consider fuzzy constants simply as fuzzy sets. Further, if A and B are fuzzy constants, $A \cup B$ will refer to their fuzzy set max-union.

One of the main advantages of the present semantics for PLFC is that it provides a sound *refutation-by-resolution* proof mechanism. Given a context $\Omega_{U,m}$, the *PLFC resolution inference rule*, which implicitly manages the unification mechanism between fuzzy constants, can be expressed as follows:

$$\frac{\begin{array}{l}(\neg p(x) \vee \varphi(x), \min(\beta, A(x)))\\ (p(B) \vee \psi, \alpha)\end{array}}{(\varphi(B) \vee \psi, \min(\beta, \alpha, N(A|[B]_\alpha)))}\ [\text{RE}],$$

where $[B]_\alpha$ denotes the α-cut of B and thus $N(A \mid [B]_\alpha) = \inf_{u \in [B]_\alpha} A(u)$.

When applying the resolution inference rule to PLFC clauses with variable weights, involved variables may disappear in the logical part of the resolvent clause, but still appear in the valuation function. For instance, from

$$(\neg p(x) \vee q, A(x)) \quad \text{and} \quad (p(x), B(x)),$$

with fuzzy constants B and C in the valuation function, we infer

$$(q, \min(A(x), B(x))),$$

which is interpreted as "for all precise object constants c, q is certain with a necessity degree of at least $\min(A(c),B(c))$", and thus, for each precise object constant c we get a clause with the same logical part. To eliminate variables that only appear in the valuation function of a PLFC clause, Dubois et al. [16,17] proposed a *fusion inference rule* which can be expressed in the following way:

$$\frac{(\varphi(\bar{x}), f(\bar{x}, y))}{(\varphi(\bar{x}), \max_c f(\bar{x}, c))} \text{ [FU]},$$

where c varies on the set of precise object constants of the corresponding sort.

The above resolution mechanism produces conclusions which are all the stronger as A is large and B is small. Indeed,

$$N(A \mid B) \geqslant N(A' \mid B) \quad \text{if } A \geqslant A'$$

and

$$N(A \mid B) \geqslant N(A \mid B') \quad \text{if } B \leqslant B'.$$

This points out that, during the refutation proof procedure, it is interesting to have PLFC clauses with the greatest possible weight. In order to get larger variable weights, Dubois et al. [16,17] proposed a *merging inference rule* which can be expressed in the following way:

$$\frac{\begin{array}{c}(p(x) \vee \varphi(\bar{z}), f_1(x, \bar{z}))\\ (p(y) \vee \psi(\bar{v}), f_2(y, \bar{v}))\end{array}}{(p(x) \vee \varphi(\bar{z}) \vee \psi(\bar{v}), \max(f_1(x, \bar{z}), f_2(x, \bar{v})))} \text{ [ME]},$$

p being a positive or a negative literal.

It is worth pointing out that in order to define a refutation proof procedure for PLFC, we cannot borrow the unification concept used in classical first-order logic programming systems. Let us consider one illustrative example. For instance, from

$$(\neg p(A) \vee \psi, 1) \quad \text{and} \quad (p(A), 1),$$

which, if A is not fuzzy, are interpreted, respectively, as

$$\text{“}[\exists x \in A, \neg p(x)] \vee \psi\text{”} \quad \text{and} \quad \text{“}\exists x \in A, \;\; p(x)\text{”},$$

we can infer ψ iff A is a precise constant. Then, resolution for $\neg\, p(A)$ and $p(A)$ must fail unless A is a precise constant, even though, obviously, $p(A)\theta = p(A)\theta$ for each (classical) substitution of variables θ. This points out that before applying the merging inference rule to a knowledge base, it is interesting to transform each precise object constant (appearing in the logical part of PLFC clauses) into a variable weight by means of the following *transformation inference rule*:

$$\frac{(\varphi(\bar{x}, c), f(\bar{y}))}{(\varphi(\bar{x}, z), \min(f(\bar{y}), c(z)))} \text{ [TR]},$$

where $z \notin \bar{y}$ and c is a precise constant.

Finally, given a context $\Omega_{U,m}$, a *refutation-by-resolution* proof method can be defined for PLFC. Let $K = \{(\varphi_i, f_i) \mid i = 1, \ldots, n\}$ be a set of PLFC clauses and let (φ, f) be PLFC query of the form

$(p(A),\alpha)$ or $(p(x),\min(\alpha,A(x)))$. Then the proof method to check whether $K\models(\varphi,f)$ follows the next steps:

1. Negate the query in the following way:
   - $\neg[(p(A),\alpha)]$ is $(\neg p(x),A(x))$.
   - $\neg[(p(x),\min(\alpha,A(x)))]$ is $(\neg p(A_{>0}),1)$, where $A_{>0}$ denotes the support of A, i.e.

$$A_{>0}(u)=\begin{cases}1 & \text{if } A(u)>0,\\ 0 & \text{otherwise.}\end{cases}$$

2. Consider $K'=K\cup\{\neg[(\varphi,f)]\}$.
3. In the context $\Omega_{U,m}$, search for a deduction from K' of $(\perp,\beta)$, with $\beta\geqslant\alpha$, by repeatedly applying the RE, FU, ME and TR inference rules.
4. If this is so, then we know that $K\models(\varphi,f)$.

The soundness of this proof procedure is based on the soundeness of the inference rules (see [4]) and the fact that the following relationship:

$$K\cup\{\neg[(\varphi,f)]\models(\perp,\alpha)\quad\text{if}\quad K\models(\varphi,f),$$

where $\neg[(\varphi,f)]$ is defined as above, always holds true. As for the converse direction, we actually proved in [4] that refutation in PLFC is semantically complete for queries without fuzzy constants in the valuation function and only sound otherwise. That is,

(i) $K\cup\{(\neg p(x),A(x))\}\models(\perp,\alpha)$ iff $K\models(p(A),\alpha)$.
(ii) If A is an imprecise but non-fuzzy constant, then

$$K\cup\{(\neg p(A),1)\}\models(\perp,\alpha)\quad\text{iff}\ K\models(p(x),\min(\alpha,A(x))).$$

(iii) If $K\cup\{(\neg p(A_{>0}),1)\}\models(\perp,\alpha)$, then $K\models(p(x),\min(\alpha,A(x)))$.

Obviously, it is case (iii) that prevents the above refutation procedure to be semantically complete. Nevertheless, refutation for PLFC could indeed be (theoretically) defined in a semantically complete way by modifying (iii). Namely, for queries with fuzzy constants in the valuation function, thanks to (ii) we have that

$$K\cup\{(\neg p(A_{>0}),1)\}\models(\perp,\alpha)\quad\text{iff}\ K\models(p(x),\min(\alpha,A_{>0}(x))).$$

Hence, semantical completeness of refutation for queries of the kind $(p(x),\min(\alpha,A(x)))$ reads as follows:

$$K\models(p(x),\min(\alpha,A(x)))\ \text{iff, for each}\ \beta\in[0,\alpha],\ K\cup\{(\neg p(A_\beta),1)\}\models(\perp,\beta).$$

However, from the automated deduction point of view, this definition has a clear computational limitation since, for each $\beta\in[0,\alpha]$, it would be necessary to search in $K\cup\{(\neg p(A_\beta),1)\}$ for a deduction of $(\perp,\beta)$.

3. A Horn sublogic of PLFC

As already mentioned, our aim in this paper is to extend a Horn-rule sublogic of PLFC with a similarity-based unification mechanism in order to allow similarity reasoning patterns of the kind

(i) Given p' and $p \to q$, then infer q' such that the more similar is p' to p, the more similar is q' to q.
(ii) Given p' and $p \to q$, the more similar is p' to p, the more *truth-like* is q.

These approximate reasoning patterns have been studied for instance in [12] for classical propositions. In PLFC, due to the presence of fuzzy constants and variable weights, we can encode both *disjunctive* and *conjunctive* (fuzzy) information. Consider for instance the following general PLFC clause

$$(\neg s(A) \vee \neg p(x) \vee q(C), \min(\alpha, B(x))),$$

with (fuzzy) constants A and C in the logical part and with a fuzzy constant B in the valuation function. As fuzzy constants in the logical part of PLFC clauses express disjunctive (vague) knowledge, in the case in which A and C are imprecise but not fuzzy, $\neg s(A)$ and $q(C)$ are, respectively, interpreted in PLFC as

$$\text{“}\exists y \in A,\ \neg s(y)\text{”} \quad \text{and} \quad \text{“}\exists y \in C,\ q(y)\text{”}$$

and therefore, if B is also non-fuzzy, the above PLFC clause could be equivalently written in a Horn-rule syntax-style as

$$((\forall x \in B)([\forall y \in A, s(y)] \wedge p(x) \to [\exists y \in C, q(y)]), \alpha),$$

that is,

$$([\forall y \in A, s(y)] \wedge [\exists x \in B, p(x)] \to [\exists y \in C, q(y)], \alpha).$$

Then, using fuzzy constants and variable weights, the above general PLFC clause should be represented in a Horn-rule syntax-style as

$$(s(A) \wedge p(x) \to q(C), \min(\alpha, B(x))).$$

Thus, it becomes clear that when we transform PLFC clauses into a Horn-rule syntax-style, (fuzzy) constants are interpreted as conjunctive information if they appear in the antecedent of a Horn-rule, and as disjunctive information, otherwise. Moreover, variable weights allow us to model disjunctive information in the antecedent of a Horn-rule, and conjunctive information, otherwise. However, the above similarity reasoning patterns actually make sense when the propositions involved express imprecise or disjunctive information. For conjunctive pieces of information is not so evident the usefulness of allowing a similarity-based inference mechanism. Hence, in this paper we shall focus on the Horn fragment of PLFC clauses with only disjunctive fuzzy constants. In the rest of this paper we shall refer to this sublogic as *Disjunctive Horn PLFC*, DH-PLFC for short.

A *DH-PLFC clause* is a PLFC clause $(\varphi(\bar{x}), f(\bar{y}))$ such that in the base formula $\varphi(\bar{x})$ there exists at most one positive literal and where the positive literal does not involve variable weights and negative literals do not involve imprecise and fuzzy constants. From now on, for the sake of a simpler and more standard notation, we write a DH-PLFC clause $(\neg p_1 \vee \cdots \vee \neg p_k \vee q, f)$ as

$(p_1 \wedge \cdots \wedge p_k \rightarrow q, f)$. For instance, the statements “it is more or less sure that Mary is young” and “it is almost sure young people have low salaries”, can be represented in this framework, respectively, as

$$(age(Mary, young), 0.7) \quad \text{and} \quad (age(x, y) \rightarrow salary(x, low), \min(0.9, young(y))),$$

where $age(\cdot,\cdot)$ and $salary(\cdot,\cdot)$ are classical predicates of type $(\texttt{person\_name}, \texttt{years\_old})$ and $(\texttt{person\_name}, \texttt{salary\_euros})$, respectively; $Mary$ is an object constant of sort $\texttt{person\_name}$; $young$ is a fuzzy constant of sort $\texttt{years\_old}$; and low is a fuzzy constant of sort $\texttt{salary\_euros}$.

It should be noticed that in [20] a Horn sublanguage of PLFC was already considered, called PLFC-H, where the only syntactical difference with respect to DH-PLFC is that the positive literal in a clause is not allowed to contain fuzzy constants, only imprecise but non-fuzzy constants.

Although the general refutation-by-resolution proof method of PLFC can be used for Horn clauses, it is possible to develop a simpler and more efficient refutation proof method oriented to queries. In [21] a theorem prover for PLFC-H was proposed, defined on top of the system KOMET [10]. Here below, we shall describe a refutation-by-resolution proof method for DH-PLFC, more general than the one described in [21] (we shall come back to this later).

Given a context $\Omega_{U,m}$, the PLFC resolution and merging inference rules can be particularized for DH-PLFC clauses as follows.

Resolution rule:

$$\frac{\begin{array}{l}(p \wedge s \rightarrow q(B), \alpha)\\ (q(y) \wedge t \rightarrow r, A(y))\end{array}}{(p \wedge s \wedge t \rightarrow r,\ \min(\alpha, N(A \mid [B]_\alpha)))}\ [\mathrm{DH-RE}].$$

Merging rule:

$$\frac{\begin{array}{l}(p(x) \wedge s \rightarrow q(B_1),\ \min(A_1(x), f_1))\\ (p(y) \wedge t \rightarrow q(B_2),\ \min(A_2(y), f_2))\end{array}}{(p(x) \wedge s \wedge t \rightarrow q(B_1 \cup B_2), \max(\min(A_1(x), f_1), \min(A_2(x), f_2)))}\ [\mathrm{DH-ME}].$$

In DH-PLFC, since consequents of Horn-rules cannot involve variable weights, the PLFC fusion inference rule is not necessary any longer and the PLFC transformation inference rule has to be applied only over precise object constants appearing in the antecedent of Horn-rules. Then, for this restricted class of clauses, the PLFC transformation inference rule can be particularized as follows:

$$\frac{(p(c) \wedge s \rightarrow q, f)}{(p(x) \wedge s \rightarrow q, \min(f, c(x)))}\ [\mathrm{DH-TR}],$$

where c is a precise constant.

In classical Horn-based systems, proof methods are oriented to queries, i.e. to existentially quantified atomic formulas. Hence, in our current framework, the refutation-based proof method should be oriented to DH-PLFC clauses of the form $(q(B_1, \ldots, B_n), \alpha)$, where $B_1, \ldots, B_n$ are (fuzzy) object constants and α should be understood as a proof threshold. In doing so, in contrast to PLFC, the proof mechanism for this restricted class of clauses can be divided into three different and sequential phases:

PM1. The first phase consists of *transforming*, by applying the DH-TR inference rule, each precise constant in the antecedent of a Horn-rule to a variable weight.
PM2. The second phase consists of a *completion algorithm* which, as in the PGL system [3], ensures that a knowledge base with disjunctive fuzzy constants is extended with all hidden clauses, and then DH-PLFC clauses may be possibly extended with larger variable weights.
PM3. The third phase properly consists then of a refutation-by-resolution *proof algorithm*.

Given a set P of DH-PLFC clauses and a context $\Omega_{U,m}$, the completion algorithm computes the set P_1 of DH-PLFC clauses that can be derived from P by applying the DH-RE inference rule. Then, the algorithm computes the set P_2 of DH-PLFC clauses that can be derived from $P \cup P_1$ by applying the DH-ME inference rule. Because the DH-ME inference rule stretches variable weights, if P_2 is not empty, the algorithm checks again if a new clause can be derived by applying the DH-RE inference rule. In this case, as the DH-RE inference rule modifies both fuzzy constants and variable weights, the algorithm checks again if a new set of clauses can be derived by applying the DH-ME inference rule. This process is performed until either the DH-RE or the DH-ME inference rules do not derive new clauses. In the worst-case, each combination of clauses of P produces a new clause. However, in general, only few clauses of P can be combined to derive new clauses. Then, the completion algorithm should not systematically check all possible combinations, but only should extend clauses which have been previously computed. Thus, the algorithm should check if three different clauses C_1, C_2 and C_3 of P derive a new clause whenever either C_1 and C_2, or C_1 and C_3, or C_2 and C_3 have already derived a new clause. Let us denote by $\hat{P}$ the set of DH-PLFC clauses obtained after the completion algorithm.

Since the DH-PLFC proof method is oriented to queries of the form $(q(B_1,\dots,B_n),\alpha)$, the refutation-by-resolution proof algorithm can be performed as follows:

1. Negate the query $q(B_1,\dots,B_n)$:

$$\neg[q(B_1,\dots,B_n)] \text{ is } (q(x_1,,\dots,x_n) \to \bot, \min(B_1(x_1),\dots,B_n(x_n))).$$

2. Let $\hat{P}' = \hat{P} \cup \{\neg[q(B_1,\dots,B_n)]\}$.
3. Search in $\hat{P}'$ for a deduction of $(\bot,\beta)$, with $\beta \geqslant \alpha$, by repeatedly applying the following resolution rule for DH-PLFC queries:

$$\frac{(p_1 \wedge \cdots \wedge p_n \to q, f_1);(q_1 \wedge \cdots \wedge q_m \to \bot, f_2)}{[(q_1 \wedge \cdots \wedge q_{i-1} \wedge p_1 \wedge \cdots \wedge p_n \wedge q_{i+1} \wedge \cdots \wedge q_m \to \bot, \min(f_1,f_2))]\theta},$$

where θ is the most general unifier (mgu) that unifies q and q_i.

This rule only allows for the resolution of a non-negative clause (rule or fact) with a negative clause (the query). Therefore, in that framework, a rule $(p_1 \wedge \cdots \wedge p_n \to q, f_1)$ and a fact $(\to p, f_2)$ is not resolved together even if there exists a mgu θ that unifies p and a $p_i, i>0$, contrary to what would happen with the DH-RE inference rule. This does not imply a lack of generality, because if q is ever resolved with the jth query, then p_i will be part of the $(j+1)$th query, which can then be unified with the fact.

In PLFC [4], given a context $\Omega_{U,m}$, the mgu of two atomic formulas is constructed from a mgu of classical first-order logic, with the only remarkable distinction that two object constants A and B of a same sort σ are only unified if they denote a same precise constant, i.e. if $\forall u \in U_\sigma$, $A(u) = B(u)$ and if $A(v) = 1$ for some $v \in U_\sigma$, then $\forall u \neq v$, $A(v) = 0$. In DH-PLFC, after applying the transformation inference rule over the clauses of the knowledge base, the antecedents of Horn-rules only involve variables, then, in this restricted framework, a mgu is a mapping from variables to substitution terms, where a substitution term is either a variable, a precise constant or an imprecise but non-fuzzy constant,[3] and it is written as $\theta = \{x_1/t_1, \ldots, x_n/t_n\}$, where the variables $x_1, \ldots, x_n$ are different and $x_i \neq t_i$, for $i = 1, \ldots, n$. Then, mgu's can be composed in the classical way. Let $\theta = \{x_1/t_1, \ldots, x_n/t_n\}$ and $\eta = \{y_1/s_1, \ldots, y_m/s_m\}$ be two mgu's. The composition of θ and η, written $\theta\eta$, is a mgu defined by removing from the set $\{x_1/t_1\eta, \ldots, x_n/t_n\eta, y_1/s_1, \ldots, y_m/s_m\}$ those pairs $x_i/t_i\eta$ for which $x_i = t_i\eta$ and those pairs y_i/s_i for which $y_i \in \{x_1, \ldots, x_n\}$.

A mgu θ is applied over a resolvent clause of the form $(\varphi(\bar{x}), f(\bar{x}))$ and is performed by simultaneously replacing each occurrence in $\varphi(\bar{x})$ and $f(\bar{x})$ of a variable from the domain of θ by the corresponding substitution term. After applying a mgu θ to a resolvent clause, the valuation function becomes computable in a given context $\Omega_{U,m}$ as soon as all variables $\bar{x}$ are instantiated to some object constant, and then, we can obtain expressions like $f_1(B)$ or $f_2(B_1, \ldots, B_n)$, where f_1 and f_2 are valid valuation functions in the model and $B, B_1, \ldots, B_n$ are imprecise but non-fuzzy constants. Then, given a context $\Omega_{U,m}$, $f_1(B)$ is computed as

$$N(m(f_1) \mid m(B)) = \inf_{u \in m(B)} \mu_{m(f_1)}(u)$$

and $f_2(B_1, \ldots, B_n)$ as

$$N(m(f_2) \mid \min(m(B_1), \ldots, m(B_n))) = \inf_{(u_1, \ldots, u_n) \in m(B_1) \times \cdots \times m(B_n)} \mu_{m(f_2)}(u_1, \ldots, u_n),$$

where $\mu_{m(f_1)}$ and $\mu_{m(f_2)}$ are the membership function of the fuzzy set that results from applying the interpretation function m to the object constants involved in f_1 and f_2, respectively.

The refutation-by-resolution proof algorithm attempts to construct a proof tree for the DH-PLFC clause that results from negating a query of the form $(q(B_1, \ldots, B_n), \alpha)$, beginning at the leaves (the atomic formula of the negated query) and working up towards the root (the contradiction $\perp$). We can think of this process as one of "reducing" a negated DH-PLFC query to $\perp$, with a necessity of at least α. At each reduction step, an atomic formula of the query matching the head of a DH-PLFC program clause (by means of a mgu) is replaced by its body and the mgu is applied over both the logical-part and the valuation function. If the DH-PLFC program clause is chosen correctly at each reduction step, a derivation through the DH-PLFC inference rules is traced out in reverse.

Finally, comparing the proof method described here to the one in [21], one should notice that they differ in the pre-processing of the knowledge base, namely in the completion procedure in step PM2 above: the completion procedure in [21] does not consider applications of the resolution rule, and hence not all the hidden valid clauses are obtained. Moreover, as already mentioned, only imprecise but non-fuzzy constants are allowed in the head of the clauses. Therefore, we can use the theorem prover of [21] as refutation-by-resolution proof algorithm for DH-PLFC queries only when fuzzy constants are not involved in the head of the program clauses and when no chaining of clauses is needed to prove the query.

[3] As already pointed out, because of the possibilistic semantics, in PLFC fuzzy constants appearing in the logical part of PLFC clauses can be replaced by imprecise but non-fuzzy constants.

3.1. Digression: about the completeness of the DH-PLFC proof procedure

As we have already pointed out in Section 2, refutation in PLFC is semantically complete for queries of the form $(q(B_1,\dots,B_n),\alpha)$. However, completeness of the syntactical refutation-based proof method (based on RE, FU, ME and TR inference rules) has not been established so far, not even for this particular class of queries. But in the restricted framework of DH-PLFC, and in the case of contexts with finite domains, we can say something about this.

In DH-PLFC, after transforming precise object constants of the antecedent of rules to variable weights by applying the DH-TR inference rule, clauses are either of the form $(q(C),\alpha)$ or $(p(x)\wedge r(y)\rightarrow q(C),\min(\alpha,A(x),B(y)))$. Then, under the assumption of contexts $\Omega_{U,m}$ with finite domains U, due to the PLFC semantics (see Section 2), DH-PLFC clauses can be transformed into a semantically equivalent set of clauses of classical Possibilistic logic, i.e. clauses with precise object constants and constant weights. The DH-PLFC clause $(q(C),\alpha)$ is semantically equivalent to the clause $(q(C_\alpha),\alpha)$, where C_α denotes an imprecise object constant corresponding to the α-cut of the fuzzy constant C. Then, for instance, if $C_\alpha=\{c_1,\dots,c_k\}$, the DH-PLFC clause

$$(q(C),\alpha)$$

is semantically equivalent to the classical possibilistic logic clause

$$(q(c_1)\vee\cdots\vee q(c_k),\alpha).$$

More generally, if A and B are two fuzzy constants of sorts, respectively, U_{σ_A} and U_{σ_B}, the DH-PLFC clause

$$(p(x)\wedge r(y)\rightarrow q(C),\min(\alpha,A(x),B(y)))$$

is semantically equivalent to the set of clauses

$$\{(p(a)\wedge r(b)\rightarrow q(C_{\alpha_{ab}}),\alpha_{ab})\mid a\in U_{\sigma_A} \text{ and } b\in U_{\sigma_B}\},$$

where $\alpha_{ab}=\min(\alpha,A(a),B(b))$ and $C_{\alpha_{ab}}$ denotes an imprecise object constant corresponding to the cut of the fuzzy constant C at the level of α_{ab}. Hence the above DH-PLFC clause is semantically equivalent to the set of clauses of classical possibilistic logic

$$\{(\neg p(a)\vee\neg r(b)\vee q(c_1)\vee\cdots\vee q(c_{n_{ab}}),\alpha_{ab})|a\in U_{\sigma_A},\ b\in U_{\sigma_B},\ C_{\alpha_{ab}}=\{c_1,\dots,c_{n_{ab}}\}\}.$$

On the other hand, always under the hypothesis of contexts $\Omega_{U,m}$ with finite domains U, one application of resolution inference rule DH-RE is syntactically equivalent to apply a finite-many times the resolution inference rule of classical Possibilistic logic (over the set of transformed clauses), and applying the merging inference rule DH-ME issyntactically equivalent to choose the transformed clause with higher constant weight. For instance, in DH-PLFC, from

$$(p(x)\rightarrow q,\min(\alpha,A(x)))\quad\text{and}\quad(p(B),\beta),$$

by applying the DH-PLFC resolution rule we infer

$$(q,\min(\alpha,\beta,N(A|B_\beta))),$$

where $N(A \mid B_\beta) = \inf_{u \in B_\beta} A(u)$. And, if A and B are two fuzzy constants of sort U_σ and $B_\beta = \{u_1, \ldots, u_k\}$, then the above two DH-PLFC clauses are, respectively, semantically equivalent to the set of clauses of classical possibilistic logic

$$\{(\neg p(u) \vee q, \min(\alpha, A(u))) | u \in U_\sigma\} \quad \text{and} \quad (p(u_1) \vee \cdots \vee p(u_k), \beta).$$

Then, by applying repeatedly the resolution inference rule of classical possibilistic logic to these clauses we infer

$$(q, \min(\alpha, \beta, A(u_1), \ldots, A(u_k))),$$

but $N(A|B_\beta) = \min(A(u_1), \ldots, A(u_k))$, and thus in both systems we deduce q with the same necessity degree. Moreover, in DH-PLFC, from

$$(p(x) \rightarrow q, \min(\alpha, A_1(x))) \quad \text{and} \quad (p(y) \rightarrow q, \min(\beta, A_2(y)))$$

by applying the DH-PLFC merging rule we infer

$$(p(x) \rightarrow q, \max(\min(\alpha, A_1(x)), \min(\beta, A_2(x))))$$

and, resolving this clause with $(p(B), \beta)$, we infer

$$(q, N(\max(\min(\alpha, A_1(x)), \min(\beta, A_2(x)))|B_\beta)),$$

where $N(\max(\min(\alpha, A_1(x)), \min(\beta, A_2(x)))|B_\beta) = \inf_{u \in B_\beta} \max(\min(\alpha, A_1(u)), \min(\beta, A_2(u)))$, and one can easily check that this necessity degree corresponds to the highest necessity degree with which q can be deduced from the transformed set of clauses by applying the resolution inference rule of classical possibilistic logic.

Summarizing, we have that

- a DH-PLFC knowledge base can be transformed into a semantically equivalent set of clauses of classical possibilistic logic, and
- applying the DH-PLFC resolution and merging inference rules is in turn syntactically equivalent to applying a finite-many times the resolution inference rule of classical possibilistic logic.

Hence, since it is well known that refutation by resolution is a complete proof method for classical possibilistic logic [15], we may conjecture that the refutation-based proof method described in this section is also complete for DH-PLFC clauses, and we expect to formally prove it in the near future.

4. An example of derivation in disjunctive Horn PLFC

Let us show how the DH-PLFC proof method, described in the previous section, works by means of the following example. Let P be the following set of DH-PLFC clauses modeling part of a buyer's motivation and decision making system:

r1: $(\mathit{stock\_level}(x) \rightarrow \mathit{order\_units}(\mathit{a\_few}),\ \mathit{low}(x))$,
r2: $(\mathit{product\_price}(y) \rightarrow \mathit{order\_units}(\mathit{many}),\ \min(\mathit{cheap}(y), 0.8))$,

r3: $(stock\_level(x) \wedge product\_price(y) \rightarrow order\_units(some), \min(medium\_l(x), not\_expensive(y), 0.7))$.

These clauses express how a buyer decides to order more or less units of a given product depending on its stock level and the market product price. The minimum number of units to be ordered is 5 and the maximum is 30. Let m be the following interpretation of object constants:

- $m(a\_few) = [5, 5, 5, 8]$ (trapezoidal fuzzy set[4] on a scale from 5 to 30 product-units),
- $m(many) = [8, 10, 20, 25]$ (trapezoidal fuzzy set on a scale from 5 to 30 product-units),
- $m(some) = [5, 5, 10, 13]$ (trapezoidal fuzzy set on a scale from 5 to 30 product-units),
- $m(a\_few \cup some) = m(a\_few) \cup m(some) = m(some)$,
- $m(many \cup some) = m(many) \cup m(some) = [5, 5, 20, 25]$ (trapezoidal fuzzy set on a scale from 5 to 30 product-units),
- $m(a\_few \cup many \cup some) = m(a\_few) \cup m(many) \cup m(some) = m(many \cup some)$,
- $m(low) = [0, 0, 2, 7]$ (trapezoidal fuzzy set on a scale from 0 to 100 level-units),
- $m(medium\_l) = [5, 7, 10, 15]$ (trapezoidal fuzzy set on a scale from 0 to 100 level-units),
- $m(cheap) = [15, 20, 25, 30]$ (trapezoidal fuzzy set on a scale from 0 to 100 euros),
- $m(not\_expensive) = [0, 0, 35, 55]$ (trapezoidal fuzzy set on a scale from 0 to 100 euros), and
- $m(medium\_p) = [25, 30, 40, 45]$ (trapezoidal fuzzy set on a scale from 0 to 100 euros).

Suppose now that what is known is that the stock level for a given product is either 5 or 6 units and, according to the latest market estimates, the product has a *medium* price. This information can be represented in this framework as:

f1: $(stock\_level(\{5, 6\}),\ 1)$,
f2: $(product\_price(medium\_p),\ 0.9)$.

Moreover, suppose that the buyer is interested in determining whether it is advisable to order some product-units in the interval $[5, 20]$ (with a certainty of at least 0.4) which can be represented in this framework by the query $(order\_units([5, 20]), 0.4)$. Then, in order to prove the buyer's query, we apply the DH-PLFC proof method to the program $P' = \{r1, r2, r3, f1, f2\}$ under the above interpretation m of object constants. As the antecedent of rules do not involve precise object constants, no transformation by the DH-TR inference rule is needed. Then, we can next compute, by applying the completion algorithm, the set of DH-PLFC clauses $\widehat{P'}$. The DH-ME rule can be used to merge r1 with r3 and r2 with r3, which yields the new DH-PLFC clauses:

r4: $(stock\_level(x) \wedge product\_price(y) \rightarrow order\_units(a\_few \cup some), f_{r4}(x, y))$,
r5: $(stock\_level(x) \wedge product\_price(y) \rightarrow order\_units(many \cup some),\ f_{r5}(x, y))$,

where

$$f_{r4}(x, y) = \max(low(x), \min(medium\_l(x), not\_expensive(y), 0.7))$$

and

$$f_{r5}(x, y) = \max(\min(cheap(y), 0.8), \min(medium\_l(x), not\_expensive(y), 0.7)).$$

[4] By a four tuple $[a_1, a_2, a_3, a_4]$, where $a_1 \leqslant a_2 \leqslant a_3 \leqslant a_4$, we denote the trapezoidal fuzzy set whose core is the closed interval $[a_2, a_3]$ and whose support is the open interval (a_1, a_4) when $a_1 < a_2$ and $a_3 < a_4$ (when $a_1 = a_2$ or $a_3 = a_4$ then a_1 or a_4 are, respectively, added to the support).

In turn, the DH-ME rule can be used again to merge r4 with r2, r5 with r1, and r4 with r5, which yields to a unique new DH-PLFC clause:

r6: $(stock\_level(x) \wedge product\_price(y) \rightarrow order\_units(a\_few \cup many \cup some),\ f_{r6}(x,y))$,

where $f_{r6}(x,y) = \max(low(x), \min(cheap(y), 0.8), \min(medium\_l(x), medium\_p(y), 0.7))$. Hence, r4, r5 and r6 can be seen as valid (hidden) clauses of the extended knowledge base obtained through the DH-PLFC merging inference rule, and thus, $\widehat{P'} = \{r1, r2, r3, r4, r5, r6, f1, f2\}$.

Now, we apply the refutation-by-resolution proof algorithm to $\widehat{P'}$ extended with the negated query

q: $(order\_units(z) \rightarrow \perp,\ [5,20](z))$.

Obviously, the proof tree for q in $\widehat{P'}$ under the interpretation m of object constants is not unique.

Namely, resolving r1 with q one gets

q1: $[(stock\_level(x) \rightarrow \perp,\ f_{q1}(x,z))]\theta_1$,

with $f_{q1}(x,z) = \min(low(x), [5,20](z))$ and $\theta_1 = \{z/[a\_few]_{low(x)}\}$ being the mgu of $order\_units(z)$ and $order\_units([a\_few]_{low(x)})$.[5] Then, applying the mgu θ_1 we get

q1: $(stock\_level(x) \rightarrow \perp,\ f_{q1}(x, [a\_few]_{low(x)}))$.

Next, resolving q1 with f1 yields

q2: $[(\perp,\ f_{q2}(x, [a\_few]_{low(x)}))]\theta_2$,

with $f_{q2} = \min(1, f_{q1}) = f_{q1}$, and $\theta_2 = \{x/\{5,6\}\}$ being the mgu of $stock\_level(x)$ and $stock\_level(\{5,6\})$. Then, applying the mgu θ_2 we get

q2: $(\perp,\ f_{q2}(\{5,6\}, [a\_few]_{low(\{5,6\})}))$.

As $low(\{5,6\})$ is computed as $N(low | \{5,6\}) = \inf_{u \in \{5,6\}} low(u) = \mu_{[0,0,2,7]}(6) = 0.2$, we have $[a\_few]_{low(\{5,6\})} = [a\_few]_{0.2} = [5,5,5,8]_{0.2} = \{5,6,7\}$. Then, $f_{q2}(\{5,6\}, \{5,6,7\})$ is computed as

$$N(f_{q2} | \min(\{5,6\}, \{5,6,7\})) = \min_{(u,v) \in \{5,6\} \times \{5,6,7\}} \min(low(u), [5,20](v)) = low(6) = 0.2.$$

Hence, using r1, the query can be only proved with a necessity degree of $0.2 < 0.4$.

One can easily check that using r2, r3 and r5 the query can be only proved with a lower bound for the necessity degree of 0. However, it can be proved with a necessity of at least 0.4 when using r4 and r6. Namely, resolving r4 with q one gets[6]

q3: $[(stock\_level(x) \wedge product\_price(y) \rightarrow \perp,\ f_{q3}(x,y,z))]\theta_3$,

[5] Because of the possibilistic semantics of PLFC, r1 is semantically equivalent to $(stock\_level(x) \rightarrow order\_units([a\_few]_{low(x)}),\ low(x))$, where $[a\_few]_{low(x)}$ denotes the α-cut of the fuzzy set $m(a\_few)$ at the level of $low(x)$ and can be computed as soon as variable x is instantiated.

[6] As $m(a\_few \cup some) = m(some)$, for the sake of a simpler notation, we shall use the object constant $some$ instead of the object constant $a\_few \cup some$.

with $f_{q3}(x,y,z) = \min(f_{r4}(x,y),[5,20](z))$ and where $\theta_3 = \{z/[some]_{f_4(x,y)}\}$ is the mgu of $order\_units(z)$ and $order\_units([some]_{f_4(x,y)})$.[7] Then, applying the mgu θ_3 we get

q3: $(stock\_level(x) \wedge product\_price(y) \rightarrow \bot,\ f_{q3}(x,y,[some]_{f_{r4}(x,y)}))$.

Next, resolving q3 with f1 yields

q4: $[(product\_price(y) \rightarrow \bot,\ f_{q4}(x,y,[some]_{f_{r4}(x,y)}))]\theta_4$,

with $f_{q4} = \min(f_{q3},1) = f_{q3}$ and $\theta_4 = \{x/\{5,6\}\}$ being the mgu of $stock\_level(x)$ and $stock\_level(\{5,6\})$. Applying the mgu θ_4 we get

q4: $(product\_price(y) \rightarrow \bot,\ f_{q4}(\{5,6\},y,[some]_{f_{r4}(\{5,6\},y)}))$.

Next, resolving q4 with f2 yields

q5: $[(\bot,\ f_{q5}(\{5,6\},y,[some]_{f_{r4}(\{5,6\},y)}))]\theta_5$,

with $f_{q5} = \min(f_{q4},0.9)$, hence $f_{q5}(x,y,z) = \min(f_{r4}(x,y),[5,20](z),0.9)$, and $\theta_5 = \{y/[medium\_p]_{0.9}\}$ being the mgu of $product\_price(y)$ and $product\_price([medium\_p]_{0.9})$.[8] Then, applying the mgu θ_5 we get

q5: $(\bot,\ f_{q5}(\{5,6\},[medium\_p]_{0.9},[some]_{f_{r4}(\{5,6\},[medium\_p]_{0.9})}))$.

As $[medium\_p]_{0.9} = [29,41]$, we have

$$\begin{aligned}
&f_{r4}(\{5,6\},[medium\_p]_{0.9})\\
&\quad= \min_{(u_1,u_2)\in\{5,6\}\times[29,41]} \{\max(low(u_1),\min(medium\_l(u_1),not\_expensive(u_2),0.7))\}\\
&\quad= \min_{(u_1,u_2)\in\{5,6\}\times[29,41]} \{\max(\mu_{[0,0,2,7]}(u_1),\min(\mu_{[5,7,10,15]}(u_1),\mu_{[0,0,35,55]}(u_2),0.7))\}\\
&\quad= \min_{(u_1,u_2)\in\{5,6\}\times\{41\}} \{\max(\mu_{[0,0,2,7]}(u_1),\min(\mu_{[5,7,10,15]}(u_1),\mu_{[0,0,35,55]}(u_2),0.7))\}\\
&\quad= \min(\max(\mu_{[0,0,2,7]}(5),\min(\mu_{[5,7,10,15]}(5),\mu_{[0,0,35,55]}(41),0.7)),\\
&\qquad\max(\mu_{[0,0,2,7]}(6),\min(\mu_{[5,7,10,15]}(6),\mu_{[0,0,35,55]}(41),0.7)))\\
&\quad= \min(\max(0.4,\min(0,0.7,0.7)),\max(0.2,\min(0.5,0.7,0.7)))\\
&\quad= \min(0.4,0.5) = 0.4.
\end{aligned}$$

Remark that $f_{r4}(\{5,6\},[medium\_p]_{0.9})$ has been computed as $N(f_{r4} \mid \min(\{5,6\},[29,41]))$, which is different to compute

$$\begin{aligned}
&\max(low(\{5,6\}),\min(medium\_l(\{5,6\}),not\_expensive([29,41]),0.7))\\
&\quad= \max(N(low|\{5,6\}),\min(N(medium\_l|\{5,6\}),N(not\_expensive|[29,41]),0.7))\\
&\quad= \max(0.2,\min(0,0.7,0.7)) = 0.2.
\end{aligned}$$

[7] r4 is semantically equivalent to $(stock\_level(x) \wedge product\_price(y) \rightarrow order\_units([some]_{f_4(x,y)}),\ f_4(x,y))$, where $[some]_{f_{r4}(x,y)}$ denotes the α-cut of the fuzzy set $m(some)$ at the level of $f_{r4}(x,y)$ and can be computed as soon as variables x and y are instantiated.

[8] f2 is semantically equivalent to $(product\_price([medium\_p]_{0.9}),\ 0.9)$, where $[medium\_p]_{0.9}$ denotes the α-cut of the fuzzy set $m(medium\_p)$ at the level of 0.9.

Then, $[some]_{f_{r4}(\{5,6\},[medium\_p]_{0.9})} = [some]_{0.4} = [5, 11]$ and

$$\begin{aligned} & f_{q5}(\{5,6\},[medium\_p]_{0.9},[some]_{0.4}) \\ &= \min_{(u_1,u_2,u_3)\in\{5,6\}\times[29,41]\times[5,11]} \{\min(f_{r4}(u_1,u_2),[5,20](u_3),0.9)\} \\ &= \min_{(u_1,u_2,u_3)\in\{5,6\}\times[29,41]\times[5,11]} \{\min(\max(low(u_1), \\ &\quad \min(medium\_l(u_1), not\_expensive(u_2), 0.7)), [5,20](u_3), 0.9)\} \\ &= 0.4. \end{aligned}$$

Therefore, $(\perp, 0.4)$ is the output of the DH-PLFC proof method, hence, by soundness, we know that $P' \models (order\_units([5,20]), 0.4)$. Accordingly, the buyer should order some product-units in the interval $[5, 20]$, this decision being supported with a necessity degree of at least 0.4. Moreover, composing the computed mgus θ_3, θ_4 and θ_5, we get $(\theta_3\theta_4)\theta_5 = \{z/[some]_{f_{r4}(\{5,6\},[medium\_p]_{0.9})}, x/\{5,6\}, y/[med-ium\_p]_{0.9}\}$, i.e. $\{z/[5,11], x/\{5,6\}, y/[29,41]\}$, which gives to the buyer the following additional and more precise information: the buyer should order between 5 and 11 units of the product (supported with a necessity degree of at least 0.4), provided the product stock level is either 5 or 6 units and the market product price is between 29 and 41 euros.

5. Extending DH-PLFC with a similarity-based unification

In this section, we formally extend DH-PLFC with a similarity-based unification mechanism of object constants.

Indeed, our intention in extending DH-PLFC with fuzzy proximity relations is to interpret a clause of the form $(p(x) \to q(B), \min(\alpha, A(x)))$, where A is a precise or an imprecise but non-fuzzy constant, as $(p(x) \to q(B), \min(\alpha, around\_A(x)))$, where $around\_A$ is the result of fuzzifying A by means of some fuzzy similarity relation. Hence, at the syntactic level, we are lead to an *extra-logical* transformation of DH-PLFC clauses with precise and imprecise constants in the antecedents of Horn-rules and modeled by means of variable weights, to DH-PLFC clauses with fuzzily enlarged variable weights. Thus, in general, for each precise and imprecise object constant A we shall assume there exists a fuzzy constant $\hat{A}$ corresponding to the fuzzification of A by means of some fuzzy proximity relation. At the semantic level, in each context $\Omega_{U,m}$, we need to introduce a collection $\mathscr{S}$ of fuzzy similarity relations $S_\sigma : U_\sigma \times U_\sigma \to [0, 1]$, one into each domain U_σ, in order to provide the meaning of the new fuzzy constants $\hat{A}$'s.

Summarizing, given an initial set of DH-PLFC clauses K, a context $\Omega_{U,m}$ and a collection of similarity relations $\mathscr{S}$, to perform possibilistic reasoning extended with similarity-based unification (of precise and imprecise constants) we propose the following steps:

1. Compute the set of DH-PLFC clauses K_1 that results of applying the DH-TR inference rule over the clauses of K, i.e. K_1 is the result of transforming each precise constant in the antecedent of a Horn-rule of K to a variable weight.
2. Define a syntactic transformation of DH-PLFC clauses

$$\Phi\colon\ K_1 \mapsto K_2,$$

which substitutes each precise and imprecise constant A appearing in the valuation function of a DH-PLFC clause by the corresponding fuzzily enlarged variable weight $\hat{A}(x)$. For instance $\Phi((p(x) \rightarrow q(B), \min(\alpha, A(x))) = (p(x) \rightarrow q(B), \min(\alpha, \hat{A}(x)))$.

3. Define a new extended context $\Omega_{U,m_{\mathscr{S}}}$, where $m_{\mathscr{S}}$ is like m but mapping each new fuzzy constant $\hat{A}$ of sort σ to a fuzzy set $m_{\mathscr{S}}(\hat{A})$: $U_\sigma \rightarrow [0,1]$ defined as the image of $m(A)$ by the corresponding fuzzy similarity relation S_σ on U_σ, written $m_{\mathscr{S}}(\hat{A})(u) = S_\sigma \circ m(A)$, where $\circ$ denotes max-min composition.
4. Use sequentially the completion and the refutation-by-resolution proof algorithms of DH-PLFC over the DH-PLFC clauses of K_2 under the new context Ω_{U,m_S}.

Let us briefly discuss the interest of this extension by means of one example. Let K be the following set of DH-PLFC clauses modeling part of an student evaluation assessment:

r1: $(extra\_work(x) \rightarrow interest(very\_high),\ exercise(x))$,
r2: $(exam\_grade(x) \wedge interest(y) \rightarrow final\_grade(5),\ \min([4,5)(x), high(y)))$,
r3: $(exam\_grade(x) \wedge extra\_work(y) \rightarrow final\_grade(6),\ \min([4,5)(x), experiment(y)))$,

where $[4,5)$ denotes a (semi-open) crisp interval of grades on a decimal scale from 1 to 10. These clauses express the fact that a student with an exam-grade between 4 and 5 can get a final higher grade if he/she shows high interest by developing an exercise or if he develops an experiment as an extra-work. Let m be the following interpretation of object constants:

- $m(experiment) = \{e_1, e_2\}$,
- $m(exercise) = \{x_1, x_2, x_3\}$,
- $m(experiment \cup exercise) = m(experiment) \cup m(exercise)$,
- $m(high) = [7,10]$ (on a decimal scale from 1 to 10),
- $m(very\_high) = [9,10]$ (on a decimal scale from 1 to 10), and
- $m(pass) = [5,10]$ (on a decimal scale from 1 to 10).

Suppose now that a student did not pass the exam, with a grade 3.9, and he is interested in checking whether he can still get an improved final grade by either developing one experiment or one exercise. This data can be represented in this framework as:

f1: $(exam\_grade(3.9),\ 1)$,
f2: $(extra\_work(experiment \cup exercise),\ 1)$.

Let us denote the program $P = \{\mathrm{r1}, \mathrm{r2}, \mathrm{r3}, \mathrm{f1}, \mathrm{f2}\}$. The student is interested in the query *final_grade* (*pass*) but obviously, as $3.9 \notin [4,5)$, *final_grade*(*pass*) can be proved in P, after completing the knowledge base and applying the refutation-by-resolution proof mechanism, only to the degree 0. However, if the student assumes that professors make use of the above rules in an approximate rather than crisp way (when referring to grades), he can extend the program P with some fuzzy similarity relation for the sort `grade` and see what could happen. To do so, one should first apply the transformation rule to replace precise constants in the antecedent of rules, if any, by variable weights (this is not the case), and then, apply the syntactic transformation Φ to fuzzify non-fuzzy constants of valuation functions and get

r2′: $(exam\_grade(x) \wedge interest(y) \rightarrow final\_grade(5),\ \min(\widehat{[4,5)}(x), high(y)))$, r3′: $(exam\_grade(x) \wedge extra\_work(y) \to final\_grade(6),\ \min(\widehat{[4,5)}(x), experiment(y)))$.

Now, one has to extend the above context to interpret the new fuzzified constant $\widehat{[4,5)}$. Assume $m(\widehat{[4,5)}) = S_{\texttt{grade}} \circ [4,5)$, where the similarity relation $S_{\texttt{grade}}$ is defined as

$$S_{\texttt{grade}}(u,v) = \max(0, 1 - 3 \cdot |u - v|).$$

With this definition, $m(\widehat{[4,5)})$ becomes the trapezoidal fuzzy set $[3.7, 4, 5, 5.3]$.

Next step is the completion of the new program $P' = \{r1, r2', r3', f1, f2\}$. The resolution rule can be applied to r1 and r2′ which yields the new DH-PLFC clause

r4: $[(exam\_grade(x) \wedge extra\_work(z) \to final\_grade(5),\ f_{r4}(x,y,z))]\theta$,

with $f_{r4}(x,y,z) = \min(\widehat{[4,5)}(x), high(y), exercise(z))$ and $\theta = \{y/very\_high\}$ being the mgu of $int-erest(very\_high)$[9] and $interest(y)$. Then, applying the mgu θ we get

r4: $(exam\_grade(x) \wedge extra\_work(z) \to final\_grade(5),\ f_{r4}(x, very\_high, z))$,

where $f_{r4}(x, very\_high, z) = \min(\widehat{[4,5)}(x), exercise(z))$, since for all $u \in m(very\_high)$, $\mu_{m(high)}(u) = 1$. In turn, the merging rule can be now applied to r3′ and r4, which yields the new DH-PLFC clause

r5: $(exam\_grade(x) \wedge extra\_work(y) \to final\_grade(\{5,6\}),\ f_{r5}(x,y))$,

with $f_{r5}(x,y) = \max(\min(\widehat{[4,5)}(x), experiment(y)), \min(\widehat{[4,5)}(x), exercise(y)))$, that is, $f_{r5}(x,y) = \min(\widehat{[4,5)}(x), \max(experiment(y), exercise(y)))$. Hence, r4 and r5 are valid (hidden) clauses of the completed program $\widehat{P'} = \{r1, r2', r3', r4, r5, f1, f2\}$.

Finally, we can apply the refutation-by-resolution proof algorithm of DH-PLFC to $\widehat{P'}$ extended with the negated query

q: $(final\_grade(z) \to \perp,\ pass(z))$.

Namely, resolving q with r5 one gets

q1: $[(exam\_grade(x) \wedge extra\_work(y) \to \perp, f_{q1}(x,y,z))]\theta_1$,

with $f_{q1}(x,y,z) = \min(f_{r5}(x,y), pass(z))$ and $\theta_1 = \{z/\{5,6\}\}$ being the mgu of $final\_grade(z)$ and $final\_grade(\{5,6\})$. Then, applying the mgu θ_1 we get

q1: $(exam\_grade(x) \wedge extra\_work(y) \to \perp,\ f_{q1}(x,y,\{5,6\}))$,

where $f_{q1}(x,y,\{5,6\}) = \min(f_{r5}(x,y), pass(\{5,6\}) = f_{r5}(x,y)$, since $pass(5) = pass(6) = 1$ and hence $pass(\{5,6\}) = N(pass|\{5,6\}) = 1$. Next, resolving q1 with f1 yields

q2: $[(extra\_work(y) \to \perp,\ f_{q2}(x,y))]\theta_2$,

with $f_{q2}(x,y) = \min(f_{r5}(x,y), 1)$ and $\theta_2 = \{x/3.9\}$ being the mgu of $exam\_grade(x)$ and $exam\_gr-ade(3.9)$. Then, applying the mgu θ_2 we get

q2: $(extra\_work(y) \to \perp,\ f_{r5}(3.9, y))$,

[9] Remark that in this particular context $very\_high$ is an imprecise but non-fuzzy constant.

where, as $\widehat{[4,5)}(3.9) = [3.7, 4, 5, 5.3](3.9) = \frac{2}{3}$,

$$\begin{aligned} f_{r5}(3.9, y) &= \min(\widehat{[4,5)}(3.9), \max(\mathit{exercise}(y), \mathit{experiment}(y))) \\ &= \min(\tfrac{2}{3}, \max(\mathit{exercise}(y), \mathit{experiment}(y))). \end{aligned}$$

Finally, resolving q2 with f2 one gets

q3: $[(\perp,\ f_{q3}(y))]\theta_3$,

with $f_{q3}(y) = \min(\frac{2}{3}, \max(\mathit{exercise}(y), \mathit{experiment}(y)), 1)$ and $\theta_3 = \{y/\mathit{experiment} \cup \mathit{exercise}\}$ being the mgu of $\mathit{extra\_work}(y)$ and $\mathit{extra\_work}(\mathit{experiment} \cup \mathit{exercise})$. Then, applying the mgu θ_3 we get

q3: $(\perp,\ f_{q3}(\mathit{experiment} \cup \mathit{exercise}))$,

where

$$\begin{aligned} & f_{q3}(\mathit{experiment} \cup \mathit{exercise}) \\ &\quad = \min\left(\tfrac{2}{3}, \min_{u \in m(\mathit{experiment} \cup \mathit{exercise})} \max(m(\mathit{exercise})(u), m(\mathit{experiment})(u))\right) \\ &\quad = \tfrac{2}{3}. \end{aligned}$$

Remark again here that $f_{q3}(\mathit{experiment} \cup \mathit{exercise})$ has been computed as $N(m(f_{q3}) \mid m(\mathit{experiment} \cup \mathit{exercise}))$ and not as compute

$$\begin{aligned} & \min\left(\tfrac{2}{3}, \max(\mathit{exercise}(\mathit{experiment} \cup \mathit{exercise}), \mathit{experiment}(\mathit{experiment} \cup \mathit{exercise}))\right) \\ &\quad = \min(\tfrac{2}{3}, \max(N(m(\mathit{exercise}) | m(\mathit{experiment} \cup \mathit{exercise})), \\ &\qquad N(m(\mathit{experiment}) | m(\mathit{experiment} \cup \mathit{exercise})))) \\ &\quad = 0. \end{aligned}$$

Therefore, $(\perp, \frac{2}{3})$ is the output of the DH-PLFC proof method, hence, by soundness, we know that $P' \models (\mathit{final\_grade}(\mathit{pass}), \frac{2}{3})$. Accordingly, the student can still have some hope to finally pass the subject.

Actually, notice that for this particular example, the computed answer $(\mathit{final\_grade}(\mathit{pass}), \frac{2}{3})$ could be equivalently derived from the original program P using the DH-PLFC proof method after replacing the imprecise constant $[4,5)$ appearing in the variable weights of rules r2 and r3 by the $(\frac{2}{3})$-cut of the fuzzified constant $\widehat{[4,5)}$, namely the bigger interval $[3.9, 5.1)$, and adding the constant value $\frac{2}{3}$ as a new term in the min-expressions of the variable weights. This comes from the fact that $\frac{2}{3} = N(\widehat{[4,5)} \mid 3.9) = \sup\{\alpha \in [0,1] \mid 3.9 \in [\widehat{[4,5)}]_\alpha\}$ and $[\widehat{[4,5)}]_{2/3} = [3.9, 5.1)$.

6. Conclusions

Within the framework of Possibilistic logic programming, in this paper we have addressed the issue of extending the (graded) unification of fuzzy constants to cope with a similarity-based unification of precise and imprecise object constants. For simplicity and practical reasons, we have focused on the sublogic of Horn-like PLFC clauses expressing disjunctive information, hence fuzzy constants are only allowed in the head of a clause. Then, each precise and imprecise object constant attached with the body of a Horn-rule is fuzzified by means a similarity relation, and the fuzzified constant is placed as a variable weight. With this we enlarge the applicability of the clause to constants *close* to the original ones.

The similarity-based fuzzification of precise and imprecise constants can be easily extended to fuzzy constants themselves. On the other hand, the proposed methodology can be used solely as a pure similarity-based reasoning with classical (precise) constants (that is, with no imprecise and fuzzy constants at all). The comparison of the resulting system with the ones proposed by Arcelli et al. [7,18,26] on the one hand, and the ones proposed by Vinař and Vojtáš [28,30] and Medina et al. [22–24], in different frameworks, will be a matter of high interest.

Acknowledgements

The authors acknowledge the anonymous referees for interesting comments. This research has been partially supported by the Spanish CYCIT project LOGFAC (TIC2001-1577-C03-01/03).

References

[1] T. Alsinet, L. Godo, A complete calculus for possibilistic logic programming with fuzzy propositional variables, in: Proc. UAI’2000 Conf., Stanford, CA, 2000, pp. 1–10.

[2] T. Alsinet, L. Godo, A complete proof method for possibilistic logic programming with semantical unification of fuzzy constants, in: Proc. ESTYLF’2000 Conf., Sevilla, Spain, 2000, pp. 279–284. http://fermat.eup.udl.es/∼tracy/report002.ps.

[3] T. Alsinet, L. Godo, A proof procedure for possibilistic logic programming with fuzzy constants, in: Proc. ECSQARU’2001 Conf., Toulouse, France, Lecture Notes in Artificial Intelligence, Vol. 2143, Springer, Berlin, 2001, pp. 760–771.

[4] T. Alsinet, L. Godo, S. Sandri, On the semantics and automated deduction for PLFC, a logic of possibilistic uncertainty and fuzziness, in: Proc. UAI’99 Conf., Stockholm, Sweden, 1999, pp. 3–12. Extended version as IIIA Tech. Report 99-09. Available at http://www.iiia.csic.es/Publications/Reports/1999.

[5] T. Alsinet, L. Godo, S. Sandri, Two formalisms of extended possibilistic logic programming with context-dependent fuzzy unification: a comparative description, Electron. Notes Comput. Sci. 66(5) (2002) 21, URL: http://www.elsevier.nl/locate/entcs/volume66.html.

[6] F. Arcelli, F. Formato, G. Gerla, Extending unification through similarity relations, BUSEFAL, Vol. 70, IRIT, Toulouse, France, 1997, pp. 3–12.

[7] F. Arcelli, F. Formato, G. Gerla, Fuzzy unification as a foundation of fuzzy logic programming, in: F. Arcelli, T.P. Martin (Eds.), Logic Programming and Soft Computing, Research Studies Press, Baldock, Hertfordshire, UK, 1998, pp. 51–68 (Chapter 3).

[8] J.F. Baldwin, T.P. Martin, B.W. Pilsworth, Fril—Fuzzy and Evidential Reasoning in Artificial Intelligence, Research Studies Press, Taunton, Somerset, UK, 1995.

[9] G. Bel, D. Farreny, H. Prade, Towards the use of fuzzy rule-based systems in the monitoring of manufacturing systems, in: J.F. Mc Waters, J.P. Crestin (Eds.), Software for Discrete Manufacturing, North-Holland, Amsterdam, 1986, pp. 525–535.

[10] J. Calmet, S. Jekutsch, P. Kullmann, S. Schü, KOMET—A system for the integration of heterogeneous information sources, in: Z.W. Raś, A. Skowron (Eds.), Foundations of Intelligent Systems, Lecture Notes in Artificial Intelligence, Vol. 1325, Springer, Berlin, 1997, pp. 318–327.

[11] M. Cayrol, H. Farreny, H. Prade, Fuzzy pattern matching, Kybernetes 11 (1982) 103–116.

[12] D. Dubois, F. Esteva, P. Garcia, L. Godo, H. Prade, A logical approach to interpolation based on similarity relations, Internat. J. Approx. Reason. 17 (1) (1997) 1–36.
[13] D. Dubois, J. Lang, H. Prade, Automated reasoning using possibilistic logic: semantics, belief revision and variable certainty weights, IEEE Trans. Data Knowledge Eng. 1 (6) (1994) 64–71.
[14] D. Dubois, J. Lang, H. Prade, Handling uncertainty, contex vague predicates and partial inconsistency in possibilistic logic, in: P.W. Eklund, D. Driankov, A.L. Ralescu (Eds.), Fuzzy Logic and Fuzzy Control, Lecture Notes in Artificial Intelligence, Vol. 833, Springer, Berlin, 1994, pp. 45–55.
[15] D. Dubois, J. Lang, H. Prade, Possibilistic logic, in: D.M. Gabbay, C.J. Hogger, J.A. Robinson (Eds.), Handbook of Logic in Artificial Intelligence and Logic Programming, Oxford University Press, Oxford, 1994, pp. 439–513.
[16] D. Dubois, H. Prade, S. Sandri, Possibilistic logic augmented with fuzzy unification, in: Proc. IPMU'96 Conf., Granada, Spain, 1996, pp. 1009–1014.
[17] D. Dubois, H. Prade, S. Sandri, Possibilistic logic with fuzzy constants and fuzzily restricted quantifiers, in: F. Arcelli, T.P. Martin (Eds.), Logic Programming and Soft Computing, Research Studies Press, Baldock, Hertfordshire, UK, 1998, pp. 69–90 (Chapter 4).
[18] F. Formato, G. Gerla, M. Sessa, Similarity-based unification, Fund. Inform. 40 (2000) 1–22.
[19] G. Gerla, M.I. Sessa, Similarity in logic programming, in: G. Chen, M. Ying, K. Cai (Eds.), Fuzzy Logic and Soft Computing, Kluwer, Dordrecht, 1999, pp. 19–31 (Chapter 2).
[20] P. Kullmann, S. Sandri, Possibilistic logic as an annotated logic, in: Proc. Fuzz-IEEE'99 Conf., Seoul, South Korea, 1999, pp. 210–215.
[21] P. Kullmann, S. Sandri, Implementation of an Extended Possibilistic Logic in an Annotated Logic Theorem Prover, in: Proc. IFSA-NAFIPS'2001 Conf., Vancouver, Canada, 2001, pp. 1529–1534.
[22] J. Medina, M. Ojeda-Aciego, P. Vojtáš, Multi-adjoint logic programming with continuous semantics, in: Proc. LPNMR'01 Conf., Lecture Notes in Artificial Intelligence, Vol. 2173, Springer, Berlin, 2001, pp. 351–364.
[23] J. Medina, M. Ojeda-Aciego, P. Vojtáš, A procedural semantics for multi-adjoint logic programming, in: Proc. EPIA'01 Conf., Lecture Notes in Artificial Intelligence, Vol. 2258, Springer, Berlin, 2001, pp. 290–297.
[24] J. Medina, M. Ojeda-Aciego, P. Vojtáš, Similarity-based unification: a multi-adjoint approach, in: Proc. EUSFLAT Conf. in Fuzzy Logic and Technology, De Montfort University, Leicester, UK, 2001, pp. 273–276.
[25] L.G. Rios-Filho, S. Sandri, Contextual fuzzy unification, in: Proc. IFSA'95 Conf., São Paulo, Brazil, 1995, pp. 81–84.
[26] M.I. Sessa, Approximate reasoning by similarity-based SLD resolution, Theoret. Comput. Sci. 275 (1–2) (2002) 389–426.
[27] M. Umano, Fuzzy set prolog, in: Proc. 2nd IFSA Congress, Tokyo, Japan, 1987, pp. 750–753.
[28] J. Vinař, P. Vojtáš, A formal model for fuzzy knowledge based systems with similarities, Neural Network World 10 (5) (2000) 891–905.
[29] H.E. Virtanen, Linguistic logic programming, in: F. Arcelli, T.P. Martin (Eds.), Logic Programming and Soft Computing, Research Studies Press, Baldock, Hertfordshire, UK, 1998, pp. 91–128 (Chapter 5).
[30] P. Vojtáš, Fuzzy logic programming, Fuzzy Sets and Systems 124 (3) (2001) 361–370.