International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems

SIMILARITY LOGIC AND TRANSLATIONS

COSTANZA CRISCONIO, DANIEL DONATO and GIANGIACOMO GERLA
Soft-Computer Laboratory
Dipartimento Matematica e Informatica, University of Salerno, via Ponte Don Melillo
84084 Fisciano, Italy
{*gerla,ddonato*}*@unisa.it*

Received (received date)
Revised (revised date)

We propose and examine a simple notion of translation in first order logics to give a basis to similarity-based fuzzy logic.

Keywords: similarity, fuzzy logic, translation, categorical logic, duality principle.

1. Introduction

A similarity-based fuzzy logic was proposed by M.S.Ying in [10] and successively investigated in [2]. The basic idea is that we can make "approximate" reasoning by allowing an inference rule to work also in the case the antecedent clauses match only approximately previously proven formulas. Some tentative to applying such an idea to logic programming was proposed in [1], [6] and [7].

In this note we investigate the possibility of applying the notion of "translation" which is on the basis of several abstract treatment of logic (see, for example, [4], [3], [5], [8]) to the similarity-based logic. This since any similarity relation enables us to define a family of translations. We refer to classical first order logic as defined, as an example, by E. Mendelson in [4]. In particular, we define the class of *well-formed formulae* (*wff*) by assuming that if α and β are *wff* then $(\neg\alpha)$, $(\alpha \wedge \beta)$, $(\alpha \vee \beta)$, and $(\forall x_i \alpha)$ are *wff* too. As usual, sometimes we reduce the number of unnecessary parethesis.

2. Translations

Let $L = (C, F, R, ar)$ be a first order language whose (finite) set of constants, function's names and relation's names is denoted by C, F, and R, respectively, and where $ar : F \cup R \to N$ is the arity-function. We assume $\wedge$, $\vee$, $\neg$ and $\forall$ as primitives and we denote again by L both the related set of well formed formulas and the corresponding first order classical logic. Also, we denote by $Ter(L)$ the set of all terms of L.

Definition 1. Let $L = (C, F, R, ar)$ and $L' = (C', F', R', ar')$ be two first order languages. We call *translation from L to L'* any function $\tau : F \cup R \cup C \to F' \cup R' \cup C'$ such that $\tau(F) \subseteq F'$, $\tau(R) \subseteq R', \tau(C) \subseteq C'$ and $ar'(\tau(x)) = ar(x)$ for every $x \in F \cup R$.

Given a translation τ, we denote by $\tau(L)$ the sublanguage $(\tau(C), \tau(F), \tau(R), ar')$ of L'. Any translation can be extended to the set of terms and the set of formulas as follows. We define the map τ': $Ter(L) \to Ter(L')$ by setting:

- $\tau'(x_i) = x_i$ for every variable x_i,
- $\tau'(c) = \tau(c)$ for every $c \in C$,
- $\tau'(h(t_1, ..., t_n)) = (tau(h)\tau'(t_1), ..., \tau'(t_n)$ for every $h \in F$ s.t. $ar(h) = n$.

Also, we define $\tau'' : L \to L'$ by setting:

- $\tau''(r(t_1, ..., t_n)) = \tau(r)(\tau'(t_1), ..., \tau'(t_n))$ for atomic formula $r(t_1, ..., t_n)$
- $\tau''((\alpha \wedge \beta)) = (\tau''(\alpha) \wedge \tau''(\beta))$ for every α and β in L
- $\tau''((\alpha \vee \beta)) = (\tau''(\alpha) \vee \tau''(\beta))$ for every α and β in L
- $\tau''(\neg(\alpha)) = (\neg\tau''(\alpha))$ for every α in L
- $\tau''((\forall x_i \alpha)) = (\forall x_i \tau''(\alpha))$ for every α in L.

An equivalent way to define τ' and τ'', algebraic in nature, is the following. At first we interpret $Ter(L)$ as the free algebra of terms by associating any n-ary function's name h with the *n-ary* operation $h^* : Ter(L)^n \to Ter(L)$ defined setting

$$h^*(t_1, ..., t_n) = h(t_1, ..., t_n)$$

for any $t_1, ..., t_n$ in $Ter(L)$. The set of variables and constants is a system of generators of such an algebraic structure.

Definition 2. Let τ be a translation from L to L'. Then we denote by $\tau' : Ter(L) \to Ter(L')$ the homomorphism from $Ter(L)$ to $Ter(\tau(L'))$ such that $\tau'(x_i) = x_i$ for every variable x_i and $\tau'(c) = \tau(c)$ for every constant c.

The second step is to define τ'' as a suitable homomorphism from the free algebra of the formulas of L into the free algebra of the formulas of L'. Indeed, define the free algebra of the formulas as the algebraic structure with infinite many operations $A(L) = (L, \wedge', \vee', \neg', \forall' x_1, ..., \forall' x_i, ...)$ in which:

– $\wedge'$ is the operation associating any pair of formulas α, $\beta \in L$ with the formula $(\alpha \wedge \beta)$,

– $\vee'$ is the operation associating any pair of formulas α, $\beta \in L$ with the formula $(\alpha \vee \beta)$,

– $\neg'$ is the operation associating any formula $\alpha \in L$ with the formula $(\neg\alpha)$,

– for any i, $\forall' x_i$ is the operation associating formula $\alpha \in L$ with the formula $(\forall x_i \alpha)$.

Obviously the set of atomic formulas is a system of generators for such a structure.

Definition 3. Let τ be a translation from L to L'. Then we denote by $\tau'' : A(L) \to A(L')$ the homomorphism from $A(L)$ to $A(L')$ that associates any atom $r(t_1, ..., t_n)$ with the atom $\tau(r)(\tau'(t_1), ..., \tau'(t_n))$.

We denote by $\equiv_\tau$, in brief $\equiv$, the congruence defined by τ' and if $\alpha \equiv \alpha'$ we say that α is a *synonymous* of α'. In the following we use again τ to denote both the mappings τ' and τ'' and we call *translation* such a map.
Observe that, in accordance with a basic property of the *homomorphisms*, the *quotient* $A(L)/\equiv$ of $A(L)$ modulo the congruence $\equiv$ is isomorphic with the substructure of $A(L')$ defined by the image $\tau(L)$. Such a quotient is not an algebra of formulas since its elements are classes of formulas. However, the quotient is isomorphic with an algebra of formulas of a suitable sub-language of L. Indeed, assume that in $F \cup R \cup C$ a linear order was fixed. Then we propose the following definition:

Definition 4. We denote by $\underline{\tau} : F \cup R \cup C \to F \cup R \cup C$ the map defined by associating any $x \in F \cup R \cup C$ with the first element $\underline{\tau}(x)$ in the class $[x] = \{x' \in F \cup R \cup C | \tau(x') = \tau(x)\}$. Also, we denote by L_τ the language (F_τ, R_τ, C_τ) obtained by setting $F_\tau = \underline{\tau}(F)$, $R_\tau = \underline{\tau}(R)$ and $C_\tau = \underline{\tau}(C)$.

The language L_τ is a sub-language of L, an impoverished version of L, in a sense. We can imagine that L_τ is obtained from L by deleting the synonymous. The map $\underline{\tau}$ is a translation from the language L into the impoverished version L_τ. The proof of the following proposition is immediate.

Proposition 1. *Let τ be a translation from L to L'. Then τ is an injective translation from L_τ to L'. The stuctures A (L_τ) is a subalgebra of $A(L)$ and τ is an isomorphism between $A(L_\tau)$ and $\tau(A(L))$. Consequently, $A(L_\tau)$ is isomorphic with $A(L)/\equiv$.*

We conclude this section by observing that, in accordance with a basic property of the homomorphisms, a translation preserves also all the connectives we can define in our logic. As an example,

$$\tau(\alpha \Rightarrow \beta) = \tau(\neg\alpha \vee \beta) = \tau(\neg\alpha) \vee \tau(\beta) = \neg\tau(\alpha) \vee \tau(\beta) = \tau(\alpha) \Rightarrow \tau(\alpha),$$

and

$$\tau(\exists x_i \alpha) = \tau(\neg\forall x_i \neg\alpha) = \neg(\forall x_i \neg\tau(\alpha)) = \exists x_i \tau(\alpha).$$

3. Translations of the proofs: Categorical Logic

In accordance with Lambek and Scott [2], it is possible to consider any logic L as a category in which the objects are the formulas of L and, given a formula β, a morphism π from α to β is a proof of β that employs α as the unique hypothesis. In this paper we refer to the usual inferential apparatus obtained by adding to *Modus Ponens* rule and *Generalisation* rule a suitable system of logical axioms, e.g. the following schemas proposed by E. Mendelson in [4]:

1. $\alpha \Rightarrow (\beta \Rightarrow \alpha)$
2. $(\alpha \Rightarrow (\beta \Rightarrow \gamma)) \Rightarrow ((\alpha \Rightarrow \beta) \Rightarrow (\alpha \Rightarrow \gamma))$

3. $(\neg\beta \Rightarrow \neg\alpha) \Rightarrow ((\neg\beta \Rightarrow \alpha) \Rightarrow \beta)$
4. $\forall x_i \alpha(x_i) \Rightarrow \alpha(t)$, if $\alpha(x_i)$ is a formula of L and t is a free term for x_i in $\alpha(x_i)$
5. $\forall x_i(\alpha \Rightarrow \beta(x_i)) \Rightarrow (\alpha \Rightarrow \forall x_i \beta(x_i))$, if α doesn't have free occurrences of x_i.

Given a set X of formulas of L, a proof of α *under hypotheses* X is a sequence of formulas $\alpha_1, ..., \alpha_n$ with $\alpha_n = \alpha$ such that, for every $i = 1, ..., n$, α_i satisfies at least one of the following conditions:

- α_i is a logical axiom
- α_i is an hypothesis, i.e. $\alpha_i \in X$
- α_i is obtained from two previous formulas by *Modus Ponens*
- α_i is obtained from one previous formula by *Generalisation.*

In the category we will define we confine ourselves to proofs whith only an hypothesis and where such an hypothesis is the first formula. We call *normalized* the proofs of such a kind. This is not restrictive. Indeed, let π be any proof, then, since π uses a finite number of hypotheses $\alpha_1, ..., \alpha_m, \pi$ is equivalent to a proof whose first formula γ_1 coincides with $\alpha_1 \wedge ... \wedge \alpha_n$ and such a formula is the only hypothesis. In accordance, we write $X \vdash \alpha$ provided that a normalized proof α_1, ..., α_n of α exists such that α_1 is a conjunction of formulas in X.

Definition 5. Given a first order language L, we denote by $Cat(L)$, the category such that:
- the objects are the formulas in L,
- given α and β in L, a morphism from α to β is a normalized proof of β whose hypothesis is α
- the composition of two morphisms π_1: $\alpha \to \beta$ and π_2: $\beta \to \delta$ is the proof $\pi_1\pi_2$ we obtain adding to the deduction π_1 the deduction π_2 after the deletion of the first formula in π_2.
- given $\alpha \in L$, the identity is the morphism $l_\alpha : \alpha \to \alpha$ equal to by the one-step proof $\alpha_1 = \alpha$, where α is assumed as an hypothesis.

In order to show that any translation define a functor, we need the following lemma:

Lemma 1. *Let $\tau : L \to L'$ be a translation and α a logical axiom in L. Then $\tau(\alpha)$ is a logical axiom in $\tau(L)$. Let β be a logical axiom in the language $\tau(L)$, then a logical axiom α in L exists such that $\tau(\alpha) = \beta$.*

Proof. The first part of the proof is based on the fact that all the logical axioms are defined by a particular syntactical structure and by the identity of some subformulas and that both the structure and the identities are preserved by τ. Indeed, an axiom arising from Schema 1 has the syntactical structure $\alpha \Rightarrow (\beta \Rightarrow \gamma)$ with $\gamma = \alpha$. Its translation $\tau(\alpha \Rightarrow (\beta \Rightarrow \gamma))$ is equal to $\tau(\alpha) \Rightarrow (\tau(\beta) \Rightarrow \tau(\gamma))$, where $\tau(\gamma) = \tau(\alpha)$, and therefore it is a logical axiom of L'. The same holds for Schemas 2 and 3. An axiom arising from Schema 4 has the structure $\forall x_i \alpha(x_i) \Rightarrow \alpha(t)$, where t is a free

term for x_i in $\alpha(x_i)$ and its translation is $\forall x_i \tau(\alpha(x_i)) \Rightarrow \tau(\alpha(t))$. Now, $\tau(\alpha(t))$ can be obtained by substituting $\tau(t)$ in all the occurrences of x_i in $\tau(\alpha(x_i))$ and $\tau(t)$ is a free term for x_i in $\tau(\alpha(x_i))$. So $\tau(\forall x_i \alpha(x_i) \Rightarrow \alpha(t))$ is a logical axiom of L'. An axiom arising from Schema 5 has the structure $\forall x_i(\alpha \Rightarrow \beta(x_i)) \Rightarrow (\alpha \Rightarrow \forall x_i \beta(x_i))$ where α doesn't have free occurrences of x_i. Its translation has the structure $\forall x_i(\tau(\alpha) \Rightarrow \tau(\beta(x_i))) \Rightarrow (\tau(\alpha) \Rightarrow \forall x_i \tau(\beta(x_i)))$ where $\tau(\alpha)$ doesn't have free occurrences of x_i. This means that this translation is a logical axioms.

Conversely, assume that β is an axiom in $\tau(L)$ and assume, for example, that β is obtained from Schema 1. This means that β is a formula as $\tau(\gamma) \Rightarrow (\tau(\delta) \Rightarrow \tau(\gamma'))$ with $\tau(\gamma) = \tau(\gamma')$ and therefore that β is the translation of the logical axiom $\gamma \Rightarrow (\delta \Rightarrow \gamma)$. Assume that β is obtained from Schema 4, i.e. that β is equal to a formula as $\forall x_i \tau(\alpha(x_i)) \Rightarrow \tau(\gamma(t))$ with $\tau(\alpha) = \tau(\gamma)$. Then β is the translation of the logical axiom $\forall x_i \alpha(x_i) \Rightarrow \alpha(t)$. We can work in an analogous way for all the remaining logical axioms. □

The following theorem shows that the translation of a proof is a proof, too.

Theorem 1. *Let L and L' be two languages and $\tau : L \to L'$ be a translation. Denote by F the map associating any formula α with $\tau(\alpha)$ and any proof $\pi = \gamma_1, ..., \gamma_n$ with the sequence, $F(\pi) = \tau(\gamma_1), ..., \tau(\gamma_n)$. Then F is a functor from the category Cat(L) into the category Cat(L').*

Proof. Firstly we have to prove that F is well posed, i.e. that F transforms a proof $\pi = \gamma_1, ..., \gamma_n$ in L into a proof $F(\pi)$ in L'. We proceed by induction on the length n. The case $n = 1$ is trivial. Let $n \neq 1$ and $\pi = \gamma_1, ..., \gamma_n$ be a proof in L. If γ_n is a logical axiom, by Lemma 1 $\tau(\gamma_n)$ is still a logical axiom. Thus, being for hypothesis of induction $\tau(\gamma_1), ..., \tau(\gamma_{n-1})$ a deduction, adding the logical axiom $\tau(\gamma_n)$ a deduction is still obtained.
Let us suppose, that γ_n were obtained for Modus Ponens, and so that exist $i, j \leq n - 1$ such that $\gamma_j = \gamma_i \Rightarrow \gamma_n$. Then, by definition of translation, $\tau(\gamma_j) = \tau(\gamma_i) \Rightarrow \tau(\gamma_n)$. Thus $\tau(\gamma_n)$ is demonstrated for Modus Ponens from $\tau(\gamma_i)$ and $\tau(\gamma_j)$. Therefore even in such a case from the hypothesis of induction it follows that $\tau(\gamma_1)$, ..., $\tau(\gamma_n)$ is a deduction in L'. The procedure is finally analogous if γ_n were obtained by the Generalisation rule, and so if an index $i \leq n - 1$ exists such that $\gamma_n = \forall x \gamma_i$. This proves F transforms morphisms of L into morphisms of L'.

It is immediate that $F(1_\alpha) = 1_{F(\alpha)}$, i.e. that F preserves the identity. Consider two morphisms $\pi_1 : \alpha \to \beta$ and $\pi_2 : \beta \to \delta$. Then, trivially, $F(\pi_2 \pi_1) = F(\pi_2) F(\pi_1)$, i.e. F preserves the composition of two morphisms. □

Theorem 2. *Let T be a theory and β be a formula, then*

$$T \vdash \beta \Rightarrow \tau(T) \vdash \tau(\beta).$$

As a consequence,

$$\tau(T) \text{ consistent} \Rightarrow T \text{ consistent.}$$

The converse implications do not hold, in general.

Proof. Let $T \vdash \beta$. Then a finite set $\{\alpha_1, ..., \alpha_n\}$ of formulas in T and a normalized proof π from $\alpha_1 \wedge ... \wedge \alpha_n$ to β exist. Also, by Theorem 1 e have that $\mathrm{F}(\pi)$ is a proof from $\tau(\alpha_1 \wedge ... \wedge \alpha_n) = \tau(\alpha_1) \wedge ... \wedge \tau(\alpha_n)$ to $\tau(\beta)$, and then $\tau(T) \vdash \tau(\beta)$. This proves also that if $\tau(T)$ is consistent, then T is consistent.

To prove that the converse of the first implication is not true, in general, let T be a theory and α and α' formulas such that $\tau(\alpha) = \tau(\alpha')$ while $\beta = \alpha \vee \neg\alpha'$ is not a theorem of T. Then, since $\tau(\beta) = \tau(\alpha) \vee \neg\tau(\alpha')$ is a tautology, we have that $\tau(T) \vdash \tau(\beta)$ while by hypothesis is not $T \vdash \beta$.

Let α and α' be two formulas which are not logically equivalent and such that $\tau(\alpha) = \tau(\alpha')$. Then, by setting $T = \{\alpha, \neg\alpha'\}$, we have that while T is consistent, $\tau(T)$ is inconsistent. □

Recall that the duality principle for projective geometry says that if α is a theorem, then the dual formula obtained by reversing the roles of points and lines is also a theorem. A similar principle holds for the Boolean algebras theory and lattice theory. The notion of translation enables us to express these principles in a general way. Indeed the proof of the following Corollary is trivial.

Corollary 1. *(duality principle). Let τ be a translation from the language L into L and T a theory such that $\tau(T) \subseteq T$. Then, for any formula α*

$$T \vdash \alpha \Rightarrow T \vdash \tau(\alpha).$$

If τ is involutory, i.e. τ^2 is the identity, then

$$T \vdash \beta \Leftrightarrow T \vdash \tau(\beta)$$

.

We conclude this section by considering the impoverished language L_τ.

Theorem 3. *Let T be a theory in the language L_τ and $\beta \in L_\tau$,then*

$$T \vdash \beta \Leftrightarrow \tau(T) \vdash \tau(\beta).$$

Moreover,

$$T \text{ is consistent} \Leftrightarrow \tau(T) \text{ is consistent.}$$

Proof. Denote by τ' the map from $\tau(F) \cup \tau(R) \cup \tau(C)$ into $F_{\tau'} \cup R_\tau \cup C_\tau$ defined by setting $\tau'(x)$ equal to the first element of the set $\tau^{-1}(x)$ with respect to the fixed order. Then τ' defines a translation from $\tau(L)$ to L_τ and, for any formula x in $L_\tau \tau'(\tau(x)) = x$. Assume that $\tau(T) \vdash \tau(\beta)$, where T is a theory in $L\tau$ and β is a formula in L_τ . Then, by Theorem 1, $\tau'(\tau(T)) \vdash \tau'(\tau(\beta))$ and therefore $T \vdash \beta$. □

4. Relaxed logic

To find an opposite of Theorem 1, we define a new logical apparatus where the Modus Ponens rule is "relaxed", in a sense. Indeed, we admit that from α and $\alpha' \Rightarrow \beta$ we can infer β also in the case $\alpha' \neq \alpha$ provided that $\tau(\alpha) = \tau(\alpha')$. By denoting with $\equiv$ the congruence defined by τ, we can picture this rule as follows:

$$\frac{\alpha, \alpha' \Rightarrow \beta, \alpha \equiv \alpha'}{\beta}$$

(*Relaxed Modus Ponens*, in brief *RMP*)

In a sense such a rule arises from the usual *MP* rule by replacing the identity relation with the congruence induced by τ (see also [2]). We call *τ-logic* the resulting logic, *τ-proof* any proof in *τ-logic* and we set $T \vdash^{\tau} \alpha$ if a *τ-proof* of α exists under hypotheses T. We can also consider a new rule, we call the *equivalence rule* as the rule defined by the schema

$$\frac{\alpha, \alpha \equiv \alpha'}{\alpha'}$$

(*Equivalence Rule*, in brief *ER*)

Proposition 2. *The Equivalence rule is equivalent to the Relaxed Modus Ponens rule.*

Proof. Assume that β is obtained by *RMP* from α and $\alpha' \Rightarrow \beta$, where $\alpha \equiv \alpha'$. Then we can obtain β also by proving, by *ER*, α' from α and, successively, by applying the usual *MP* to α' and $\alpha' \Rightarrow \beta$. Conversely, assume that α' is obtained from α by *ER*. Then we can obtain a *τ-proof* of α' by considering a classical proof of the tautology $\alpha' \Rightarrow \alpha'$ and by applying *RMP* to α and $\alpha' \Rightarrow \alpha'$. □

Theorem 4. *Let T be a theory, then, given any formula α,*

$$T \vdash^{\tau} \alpha \Leftrightarrow \tau(T) \vdash \tau(\alpha) \Leftrightarrow \underline{\tau}(T) \vdash \underline{\tau}(\alpha).$$

Proof. To prove the implication $T \vdash^{\tau} \alpha \Rightarrow \tau(T) \vdash \tau(\alpha)$, we prove that if $\alpha_1, ..., \alpha_n$ is a τ-proof of α under hypotheses T, then $\tau(\alpha_1), ..., \tau(\alpha_n)$ is a proof of $\tau(\alpha)$ under hypotheses $\tau(T)$. Indeed:

- if $\alpha_i \in T$ then $\tau(\alpha_1) \in \tau(T)$,
- if α_i is a logical axiom, then by Lemma 1 $\tau(\alpha_i)$ is a logical axiom
- if α_i is obtained by *RMP*, from the formula α_j and the formula $\alpha_k = \alpha'_j \Rightarrow \alpha_i$, then $\tau(\alpha_j) = \tau(\alpha'_j)$ and therefore, $\tau(\alpha_i)$ is obtained by *MP* from $\tau(\alpha_j)$ and $\tau(\alpha_k) = \tau(\alpha'_j) \Rightarrow \tau(\alpha_i)$

- if α_i is obtained by Generalisation from α_j, then $\alpha_i = \forall x \alpha_j$ and therefore, since $\tau(\alpha_i) = \forall x \tau(\alpha_j), \tau(\alpha_i)$ is obtained by Generalisation from $\tau(\alpha_j)$.

To prove the implication $\tau(T) \vdash \tau(\alpha) \Rightarrow T \vdash^{\tau} \alpha$, firstly we prove that if $\beta_1, ..., \beta_n$ is a proof in the language $\tau(L)$ under hypotheses $\tau(T)$, then a τ-proof $\alpha_1, ..., \alpha_n$ exists under hypotheses T in the language L such that $\beta_i = \tau(\alpha_i), i = 1, ..., n$. We proceed by induction on n.

Consider the case $n = 1$ and assume that β_1 is an hypothesis, i.e. $\beta_1 \in \tau(T)$. Then $\alpha_1 \in T$ exists such that $\beta_1 = \tau(\alpha_1)$. Assume that β_1 is a logical axiom, then by Lemma 1 a logical axiom α_1 exists such that $\beta_1 = \tau(\alpha_1)$. In both the cases α_1 is a τ-proof under hypotheses T such that $\tau(\alpha_1) = \beta_1$.

Assume that $n \neq 1$. Then, since $\beta_1, ..., \beta_{n-1}$ is a τ-proof in the language $\tau(L)$ under hypotheses $\tau(T)$, by induction hypothesis a τ-proof $\alpha_1, ..., \alpha_{n-1}$ under hypotheses T exists such that $\beta_i = \tau(\alpha_i)$, for $i = 1, ..., n-1$. If β_n is a logical axiom, then going on as above we obtain that a logical axiom α_n exists such that $\beta_n = \tau(\alpha_n)$. By adding to $\alpha_1, ..., \alpha_{n-1}$ the formula α_n we obtain the τ-proof we need. Assume that β_n were obtained by *MP* from β_i, and $\beta_j = \beta_i \Rightarrow \beta_n$ where $i, j \leq n-1$. Then, since $\beta_j = \tau(\alpha_j), \alpha_j$ has the structure $\delta \Rightarrow \gamma$ where $\tau(\delta) = \beta_i$ and $\tau(\gamma) = \beta_n$. Consequently, by setting $\alpha_n = \gamma$, we have that α_n can be obtained by *RMP* from α_i and $\alpha_j = \delta \Rightarrow \gamma$. Thence, $\alpha_1, ..., \alpha_n$ is a τ-proof under hypotheses T such that $\beta_i = \tau(\alpha_i), i = 1, ..., n$. Finally, suppose that β_n were obtained by Generalisation, i.e. that an index $i \leq n-1$ exists such that $\beta_n = \forall \mathrm{x} \beta_i$. Then, by setting $\alpha_n = \forall \mathrm{x} \alpha_i$, since $\tau(\alpha_n) = \forall x \tau(\alpha_i) = \forall x \beta_i = \beta_n$, we have that $\alpha_1, ..., \alpha_n$ is a proof under hypotheses T such that $\beta_i = \tau(\alpha_i), i = 1, ..., n$.

Assume that $\tau(T) \vdash \tau(\alpha)$. Then a proof $\beta_1, ..., \beta_n$ of $\tau(\alpha)$ exists under hypothesis $\tau(T)$ and is not restrictive to assume that all the formulas $\beta_1, ..., \beta_n$ are in the language $\tau(L)$. Let $\alpha_1, ..., \alpha_n$ be a τ-proof under hypothesis T such that $\tau(\alpha_1) = \beta_1, ..., \tau(\alpha_n) = \beta_n = \tau(\alpha)$. Then, since $\alpha_n \equiv \tau\alpha$, by the *ER*, $T \vdash \tau\alpha$.

To prove the equivalence $T \vdash^{\tau} \alpha \Leftrightarrow \underline{\tau}(T) \vdash \underline{\tau}(\alpha)$, we observe that from the just proved equivalence we have that $T \vdash^{\tau} \alpha \Leftrightarrow \underline{\tau}(T) \vdash \underline{\tau}(\alpha)$ and that, since τ and $\underline{\tau}$ define the same equivalence relation in L, the relation $\vdash^{\underline{\tau}}$ coincides with the relation $\vdash^{\tau}$. □

5. Similarity-Based Logic

The main task of such a note is to contemplate the possibility of defining a similarity-based logic by the notion of translation. To do this, we recall some basic definitions in fuzzy set theory. Let [0,1] the real numbers interval we consider as a lattice whose operations we denote by $\wedge$ and $\vee$, as usual. Then a *fuzzy subset* of a set S, or more simply a *fuzzy set*, is a map $s : S \to [0, 1]$. We denote by $F(S)$ the class of all fuzzy subsets of S. Given a fuzzy set s, for every $\lambda \in [0, 1]$, the subset $C(s, \lambda) = \{x \in S : s(x) \geq \lambda\}$ is called the (*closed*) λ-*cut* of S. A *fuzzy relation* is any fuzzy subset $R : S \times S \to [0, 1]$ of the Cartesian product $S \times S$. A basic class of fuzzy relations are the similarities.

Definition 6. A *similarity* is a fuzzy relation $Sim : S \times S \to [0,1]$ such that, for any $x, y, z \in S$,
(i) $Sim(x,x) = 1$ (reflexivity)
(ii) $Sim(x,y) = Sim(y,x)$ (symmetry)
(iii)$Sim(x,y) \neq Sim(x,z) \wedge Sim(z,y)$ (transitivity)

Observe that while the similarities can be defined in correspondence with any triangular norm, we confine ourselves to the minimum $\wedge$ since in such a case, in accordance with the following proposition, we can work by the cuts.

Proposition 3. *$Sim : S \times S \to [0,1]$ is a similarity if and only if every cut $C(Sim, \lambda)$ is an equivalence relation.*

Proof. Trivial. □

Let $L = (F \cup R \cup C)$ be a first order language and $Sim : (F \cup R \cup C) \times (C \cup F \cup R) \to [0,1]$ be a similarity such that, for any pair x and y in L such that $Sim(x,y) \neq 0$,
1. either $x, y \in F$ and $ar(x) = ar(y)$ or
2. $x, y \in R$ and $ar(x) = ar(y)$,
3. or $x, y \in C$.

In other words, Sim is the union of a similarity in F, a similarity in R and a similarity in C. According with Proposition 3 we have that, for any fixed $\lambda \in [0,1]$, the λ-cut $C(Sim, \lambda)$ is an equivalence relation we denote by $\equiv_\lambda$. So, we can consider the quotients $F_\lambda = F/\equiv_\lambda, R_\lambda = R/\equiv_\lambda$ and $C_\lambda = C/\equiv_\lambda$ as new sets of function's names, relation's names, and costants, respectively. We denote by $L_\lambda = (C_\lambda \cup F_\lambda \cup R_\lambda)$ the related first order language and we consider the function $\tau_\lambda : (C \cup F \cup R) \to (C_\lambda \cup F_\lambda \cup R_\lambda)$ defined by setting $\tau_\lambda(x) = [x]_\lambda$ for any $x \in C \cup F \cup R$. Such a function defines a translation from L into the language L_λ, we call λ-translation, and therefore we can apply all the notions and results about the translations exposed in the previous sections.

Definition 7. Given a similarity Sim and $\lambda \in [0,1]$, we call *λ-relaxed logic* the τ_λ-logic. Also, we write $T \vdash^\lambda \alpha$ instead of $T \vdash^\tau_\lambda \alpha$.

Then in the λ-relaxed logic we have that the Modus Ponens rule is relaxed by admitting that from α and $\alpha' \to \beta$ we can infer β provided that $\mathrm{Sim}(\alpha, \alpha') \geq \lambda$, i.e. provided that α' is "sufficiently similar" with α. Now we are ready to propose a deduction apparatus based on the notion of similarity.

Definition 8. Given a similarity Sim, the *similarity logic* is the logic whose deduction operator $D : P(L) \to F(L)$ is defined by setting

$$D(T)(\alpha) = Sup\{\lambda \in [0,1] : T \vdash^\lambda \alpha\}, \tag{1}$$

for any theory T and $\alpha \in [0,1]$.

We have to interpret $D(T)$ as the fuzzy subset of formulas we can prove from formulas which either are in T or are "*similar*" to formulas in T. The similarity logic is an extension of the classical logic. Indeed if $T \vdash \alpha$, then $T \vdash^{\lambda} \alpha$ for every $\lambda \in [0,1]$ and therefore $D(T)(\alpha) = 1$. In accordance with Theorem 4, we have the following equivalent way to define $D(T)(\alpha)$.

Proposition 4. *In the similarity logic,*

$$D(T)(\alpha) = Sup\{\lambda \in [0,1] : \tau_{\lambda}(T) \vdash \tau_{\lambda}(\alpha)\} \tag{2}$$

and

$$D(T)(\alpha) = Sup\{\lambda \in [0,1] : \underline{\tau}_{\lambda}(T) \vdash \underline{\tau}_{\lambda}(\alpha)\}. \tag{3}$$

6. Questions and future works

Several questions arise from the ideas exposed in this note. The first one is semantical in nature. Let τ be a translation from a language L into a language L' and let (D, I) be an interpretation of L:

Is there a natural way to define an interpretation (D', I') of L' *in such a way that* τ *is, in some sense, meaning-preserving?*

Another question is about a possible definition of a similarity-based logic programming in accordance with the definition of the previous section. As an example, assume that P is a program and Sim a similarity. Then we can define the *least fuzzy Herbrand model* of T given Sim as the restriction of the fuzzy set $D(P)$ of theorems to the set of facts. In other words, the least fuzzy Herbrand model is the fuzzy set of facts we can prove by (1). To this purpose observe that, for any $\lambda \in [0,1]$, $\underline{\tau}_{\lambda}(P)$ is a program we denote by $P(\lambda)$. Assume that the (finite) co-domain of Sim, is $\{\lambda_0, \lambda_1, ..., \lambda_h\}$ where $0 = \lambda_0 < \lambda_1 < ... < \lambda_h = 1$. Then, given a fact α, we can calculate $D(T)(\alpha)$ as follows:
1. we rewrite T into h programs $P(\lambda_1), ..., P(\lambda_h)$ where $P(\lambda_i)$ is obtained by substituting each symbol $x \in C \cup F \cup R$ occurring in P with the first element $x' \in C \cup F \cup R$ such that $Sim(x, x') \geq \lambda_i$;
2. we activate h parallel processes π_1,...,π_h corresponding with the goals $\tau_{\lambda_1}(\alpha)$, ..., $\tau_{\lambda_h}(\alpha)$ and the programs $P(\lambda_1), ..., P(\lambda_h)$ in accordance with the usual resolution technique;
3. we give as an output the value $Max\{\lambda_i : \pi_i$ converges in a positive way$\}$.
Obviously, such a procedure has the same computational difficulties of the usual resolution process.

References

1. F. Arcelli, F. Formato and G. Gerla, "Extending unification trough similarity relations", *BUSEFAL* **70** (1997) 3-12.
2. L. Biacino, G. Gerla and M. S. Ying, "Approximate reasoning based on similarity", *Math. Log. Quart.* **46** (2000) 77-86.

3. W. A. Carnielli and M. E. Coniglio, "Transfer between Logics and their Applications", *Studia Logica* to appear.
4. M. E. Coniglio and W. A. Carnielli, "A model theoretic approach to translations between logics", *Proceedings of 7th WoLLIC*, Natal, Brazil, August 2000, pp. 55-65
5. M. Cerioli and J. Messeguer, "May I Borrow You Logic? (transporting Logical Structures along Maps)", manuscript.
6. F. Formato, G. Gerla and M. Sessa, "Similarity-based unification", *Fundamenta Informaticae*, **41** (2000) 393-414.
7. G. Gerla and M. I. Sessa, "Similarity in Logic Programming", in *Fuzzy Logic and Soft Computing*, eds. G. Chen, M.Ying and K.-Y. Cay, (Kluwer Ac. Pub., 1999).
8. J. Lambek and P. J. Scott, *Introduction to Higher Order Categorical Logic* (Cambridge University Press, 1986).
9. E. Mendelson, *Introduction to Mathematical Logic* (D. Van Nostrand Company, Princeton 1964).
10. M. S. Ying, "A logic for approximate reasoning", *J. Symbolic Logic* **59** (1994).