Fundamenta Informaticae 40 (2000) 393–414
IOS Press

Similarity-based Unification

Ferrante Formato, Giangiacomo Gerla and Maria I. Sessa*
DMI University of Salerno,
84081 Baronissi (SA), Italy
formato@dia.unisa.it gerla@matna2.dma.unina.it mis@dia.unisa.it

Abstract. Unification plays a central rule in Logic Programming. We "soften" the unification process by admitting that two first order expressions can be "similar" up to a certain degree and not necessarly identical. An extension of the classical unification theory is proposed accordingly. Indeed, in our approach, inspirated by the unification algorithm of Martelli-Montanari, the systems of equations go through a series of "sound" transformations until a solvable form is found yielding a substitution that is proved to be a most general extended unifier for the given system of equations.

1. Introduction

The recent outbursts in the field of data bases and information retrieval caused an ever compelling demand for systems capable to deal with flexible queries and answers. Deductive data bases, in particular, are requested to cope with approximate reasoning and more flexible queries. Researches in such a direction was pioneered by [8] with a threshold-based resolution process within the semantic framework of a many-valued logic; later, Mukaidono [10] replaced the threshold-based inference with a graded resolution based on possibility distribution over clauses. This approach has been followed in [4]. A quite different extension of resolution has been considered in [3], where an evidential distribution has been used in the unification process and classical first order terms have been replaced by fuzzy sets; in a more general framework, Virtannen has proposed a similar approach in [15]. Despite its fallouts and motivations are to be found mainly in the extensions of logic programming, the extension of unification theory is a subject of its own, as proved in [14].

*Address for correspondence: DMI University of Salerno, 84081 Baronissi (SA), Italy

In this work classical unification is turned into a "relaxed" unification which is particularly significant when, in the classical process, a failure occurs. In particular, let us recall that in the declarative paradigm of Logic Programming the unification plays a central rule [7]. Moreover, such a technique could be usefully exploited in the context of deductive databases, so, we will considered languages where no function symbol occurs.

Now, classical unification fails because of a "mismatch" among constants. Therefore, it is necessary to fade the concept of "mismatch" and this requires a weakening of the classical identity. Similarities (see [6],[12]) seem to be a promising candidate to do this. More precisely, in this paper a basic concept is the one of "extended constant", replacing the one of usual constant. Extended constants, we call *clouds*, are obtained by fading the notion of singleton into that of set of pairwise "similar" elements. It is worthy to point out that the concept of cloud, lying on the notion of similarity, does not coincide neither with the notion of fuzzy set nor with the one of classical set. Clouds have a strong connection with the "conceptual spaces", an attempt to formalize structural analogies among objects [5]. As suggested by [16] and [2], we introduce a similarity in a first-order language; in our case, constants are clouds. The notion of identity between constants is replaced with the notion of "equivalence" between clouds. This allows us to define an extended version of most general unifier (e-mgu) and supply an algorithm to compute the e-mgu of a set of equations. Such an algorithm of fuzzy unification provides both an e-mgu of the set of atomic formulae and an associated numerical value, called unification degree, intuitively representing the "cost" payed to strike a success in an unification that, classically, would result into a failure. The proposed algorithm follows the lines of Martelli-Montanari [9], and it is an extension of the classical case.

The paper, that focuses and extends some ideas sketched in [1], is arranged as follows: in section 2 we introduce similarity and clouds; in Sections 3 and 4, we describe some operators on clouds and sets of clouds; in Section 5 we introduce a similarity on a first order language and we extend the notions of unifier of a system of extended equations; in Sections 6 and 7 we give a computational way to get an extended unifier; in Section 8 we introduce the notion of e-mgu. Finally, in Section 9, we give a computational way to complete the information conveyed within an e-mgu.

2. Similarity Relations and Clouds

The notion of similarity on which is based our approach is a many valued extension of the classical notion of equality. In the sequel we define two binary operations $\vee$ and $\wedge$ in [0,1] by setting $x \wedge y = \min\{x, y\}$, $x \vee y = \max\{x, y\}$ for any $x, y \in [0, 1]$. Also, we denote by $\bigvee$ and $\bigwedge$ the lowest upper bound and the greatest lower bound operators, respectively.

Definition 2.1. A *similarity* on a domain $\mathcal{U}$ is a fuzzy subset $\mathcal{R} : \mathcal{U} \times \mathcal{U} \to [0, 1]$ of $\mathcal{U} \times \mathcal{U}$ such that the following properties hold:

i) $\mathcal{R}(x, x) = 1$ for any $x \in \mathcal{U}$ (reflexivity)

ii) $\mathcal{R}(x, y) = \mathcal{R}(y, x)$ for any $x, y \in \mathcal{U}$ (symmetry)

iii) $\mathcal{R}(x,z) \geq \mathcal{R}(x,y) \wedge \mathcal{R}(y,z)$ for any $x, y, z \in \mathcal{U}$ (transitivity).

For any $\lambda \in [0,1]$, *the closed* $\lambda-$*cut* of $\mathcal{R}$ is the classical relation

$$\mathcal{R}_\lambda = \{(x,y) \in \mathcal{U} \times \mathcal{U} \mid \mathcal{R}(x,y) \geq \lambda\}.$$

If $(x,y) \in \mathcal{R}_\lambda$, we say that x is λ-*similar* to y. Such a notion enables us to identify the similarities with suitable families of equivalence relations.

Proposition 2.1. *Let $\mathcal{R}$ be a similarity on a domain $\mathcal{U}$ and, for any $\lambda \in [0,1]$, denote by $\mathcal{R}_\lambda$ the closed $\lambda-$cut $\{(x,y) \in \mathcal{U} \times \mathcal{U} \mid \mathcal{R}(x,y) \geq \lambda\}$. Then $(\mathcal{R}_\lambda)_{\lambda \in [0,1]}$ is a family of equivalence relations such that*

i) $\lambda \leq \mu \Rightarrow \mathcal{R}_\lambda \supseteq \mathcal{R}_\mu$ for any μ and λ in $[0,1]$

ii) $\bigcap\limits_{\lambda \leq \mu} \mathcal{R}_\lambda = \mathcal{R}_\mu$ for any μ in $[0,1]$.

Conversely, let $(\mathcal{R}_\lambda)_{\lambda \in [0,1]}$ be a family of equivalence relations satisfying i) and ii). Then the fuzzy relation $\mathcal{R}$ defined by setting

$$\mathcal{R}(x,y) = \bigvee\{\lambda \in [0,1] \mid x\mathcal{R}_\lambda y\}$$

is a similarity whose $\lambda-$cuts are the relations $\mathcal{R}_\lambda$.

In order to illustrate such a proposition, we give a simple example.

Example 2.1. Let $\mathcal{U} = \{circle, square, polygon, rectangle, ellipse\}$ and consider the following family of partitions $(\Pi_\lambda)_{\lambda \in [0,1]}$ of $\mathcal{U}$:

$\Pi_\lambda = \{\{square,\ circle,\ polygon, rectangle,\ ellipse\}\}$ for $\lambda \in [0, 0.1]$

$\Pi_\lambda = \{\{square, polygon,\ rectangle\}, \{circle,\ ellipse\}\}$ for $\lambda \in (0.1, 0.5]$

$\Pi_\lambda = \{\{square, rectangle\}, \{polygon\}, \{circle,\ ellipse\}\}$ for $\lambda \in (0.5, 0.8]$

$\Pi_\lambda = \{\{square\}, \{circle\}, \{polygon\}, \{rectangle\}, \{ellipse\}\}$ for $\lambda \in (0.8, 1]$.

We obtain a similarity $\mathcal{R}$ on the domain $\mathcal{U}$ by considering the family of equivalence relations $(\mathcal{R}_\lambda)_{\lambda \in [0,1]}$ associated with $(\Pi_\lambda)_{\lambda \in [0,1]}$. As an example, we have that

$$\mathcal{R}(square, rectangle) = 0.8,\ \ \mathcal{R}(square, polygon) = 0.5,\ \ \mathcal{R}(square, circle) = 0.1.$$

We can associate to every subset X of $\mathcal{U}$ a number $\mu(X)$ intuitively expressing the "worst" value of similarity between pairs of elements in X,

$$\mu(X) = \bigwedge_{x,x' \in X} \mathcal{R}(x,x').$$

Note that, since $\bigwedge\limits_{x,x' \in \emptyset} \mathcal{R}(x,x') = \bigwedge \emptyset = 1$, it is $\mu(\emptyset) = 1$. For instance, in Example 1 we have $\mu(\{square,\ polygon, rectangle\}) = 0.5$ and $\mu(\mathcal{U}) = 0.1$. We call $\mu(X)$ *co-diameter of X* since the function $\mathcal{R}'(x,y) = 1 - \mathcal{R}(x,y)$ behaves like a distance whose related diameter is the function $\mu'(X) = \bigvee\limits_{x,y \in X} \mathcal{R}'(x,y) = 1\text{-}\mu(X)$. In the sequel we call *clouds* the subsets of a domain

$\mathcal{U}$ where a similarity $\mathcal{R}$ is defined, and therefore we say that $\mu(X)$ is the *co-diameter of the cloud* X. We stress that a cloud is not a fuzzy subset since no membership function is defined on the elements. Roughly speaking, we can regard a non-empty cloud X as a point and the number $\mu(X)$ as a many valued evaluation of the claim that X is a point. If $\lambda \in (0,1]$, and X is a nonempty cloud such that $\mu(X) \geq \lambda$, we say also that X is a *λ-point*. Then a λ-point is a part of a complete class of equivalence modulo the λ-similarity. The following proposition shows that $\mu(X)$ is the largest level for which the elements of X can be considered pairwise λ-similar, i.e. for which X is a λ-point.

Proposition 2.2. *Let X be a subset of $\mathcal{U}$. Then*

$$\mu(X) = \bigvee\{\lambda \in [0,1] \mid X \text{ is a } \lambda\text{-point}\}.$$

Proof:
Indeed

$$\begin{aligned} \mu(X) &= \bigvee\{\lambda \in [0,1] \mid \lambda \leq \mu(X)\} \\ &= \bigvee\{\lambda \in [0,1] \mid \lambda \leq \mathcal{R}(x,y)\ \forall x,y \in X\}. \end{aligned}$$

□

The following is a simple characterization of the co-diameter of a nonempty cloud.

Proposition 2.3. *Assume that $c \in X$, then*

$$\mu(X) = \bigwedge_{x \in X} \mathcal{R}(x,c).$$

Consequently, given a finite sequence $X_1, ..., X_n$ of clouds such that $\bigcap_{i=1}^{n} X_i \neq \emptyset$,

$$\mu(\bigcup_{i=1}^{n} X_i) = \bigwedge_{i=1}^{n} \mu(X_i).$$

Proof:
Clearly, $\bigwedge_{x \in X} \mathcal{R}(x,c) \geq \mu(X)$. Moreover, for any $x, y \in X$

$$\mathcal{R}(x,y) \geq \mathcal{R}(x,c) \wedge \mathcal{R}(c,y) \geq \bigwedge_{x \in X} \mathcal{R}(x,c).$$

Therefore $\mu(X) \geq \bigwedge_{x \in X} \mathcal{R}(x,c)$. □

A *system of clouds* will be a finite set of clouds. Given a system of clouds $Z = \{X_1, ..., X_n\}$ we call *crispness degree* of Z the number

$$\xi(Z) = \bigwedge_{i=1}^{n} \mu(X_i).$$

As usual we have that $\xi(\emptyset) = 1$. In a sense $\xi(Z)$ is a many-valued evaluation of the claim that each element in Z is a point. It is immediate that if Z' is obtained from Z by deleting the empty cloud and all the singletons, then $\xi(Z) = \xi(Z')$.

3. The Compactification of systems of clouds

We say that two clouds X and Y *overlap* if $X \cap Y \neq \emptyset$. A system of clouds Z is *compact* provided that no pair of clouds in Z overlaps. To define a compactification procedure for a system of clouds, it is useful the following definition.

Definition 3.1. Given a system of clouds Z we set, for $M \in Z$

$$T_Z(M) = \bigcup\{M' \in Z \mid M \cap M' \neq \emptyset, \}.$$

Moreover, we define an operator T by setting

$$T(Z) = \{T_Z(M) \mid M \in Z\}.$$

The following example clarify the previous notion

Example 3.1. Let us consider the domain $U = \{a, b, c, d, e, f, p, q\}$. Given the system of clouds $Z = \{\{a, c\}, \{d\}, \{c, d\}\}$ we have that
$T_Z(\{a, c\}) = \{a, c, d\}$, $T_Z(\{d\}) = \{c, d\}$, $T_Z(\{c, d\}) = \{a, c, d\}$.
Then, $T(Z) = \{\{a, c, d\}, \{c, d\}\}$.

Obviously, Z is a compact system of clouds if and only if Z is a fixed point for T. Moreover, T does not change the crispness degree of a system of clouds, as the following proposition shows.

Proposition 3.1. *Let Z be a system of clouds. Then*

$$\xi(Z) = \xi(T(Z)).$$

Proof:
By Proposition 2.3

$$\xi(T(Z)) = \bigwedge_{M \in Z} \left(\bigwedge_{Y \cap M \neq \emptyset, Y \in Z} \mu(Y) \right).$$

Since, for any $M \in Z$, $\bigwedge_{Y \cap M \neq \emptyset, Y \in Z} \mu(Y) \geq \xi(Z)$, it follows that

$$\xi(T(Z)) \geq \xi(Z).$$

Conversely, since $M \cap M \neq \emptyset$, for any $M \in Z$, $\bigwedge_{Y \cap M \neq \emptyset, Y \in Z} \mu(Y) \leq \mu(M)$.

Therefore

$$\xi(T(Z)) \leq \xi(Z).$$

□

The following definition enables us to obtain the least compact system of clouds "containing" a given system. As usual, we set $T^0(Z) = Z$ and $T^{n+1}(Z) = T(T^n(Z))$.

Definition 3.2. Let Z be a system of clouds. We call $T-$*degree* of Z the number

$$\Phi(Z) = \min\{n \mid T^n(Z) = T^{n+1}(Z)\}.$$

Moreover, the operator *Compact* is defined by setting

$$Compact(Z) = T^{\Phi(Z)}(Z).$$

Since Z is a finite system of finite clouds, the $T-$degree of Z is a finite non-negative integer. By the given definitions, Z is a compact system of clouds if and only if $\Phi(Z) = 0$. Since $T^{\Phi(Z)}(Z)$ is a fixed point for T, we have that $Compact(Z)$ is compact. So *Compact* transforms a system of clouds into a compact system. The following properties summarize the main features of *Compact*.

Proposition 3.2. *Let Z be a system of clouds, then*
i) $\xi(Z) = \xi(Compact(Z))$
ii) $Compact(Compact(Z)) = Compact(Z)$
iii) Z is compact if and only if Z is a fixed point for Compact
iv) $Compact(Z)$ is compact.

Proof:
The proof of *ii)*, *iii)* and *iv)* are immediate. Property *i)* is a consequence of the fact that

$$\xi(Z) = \xi(T^n(Z)).$$

That, in turn, is a consequence of Proposition 3.1. □

4. The enlargement of a System of Clouds

It is possible to extend a cloud without changing its co-diameter. We will introduce a closure operator providing the largest extension of a cloud with respect to this property. With this aim, we define a class of operators associated with the closed cuts $\mathcal{R}_\lambda$.

Definition 4.1. Let λ be any element in $[0, 1]$. Then the operator $H_\lambda : \mathcal{P}(\mathcal{U}) \to \mathcal{P}(\mathcal{U})$ is defined by setting, for every X in $\mathcal{P}(\mathcal{U})$

$$H_\lambda(X) = \{x \in \mathcal{U} \mid x\mathcal{R}_\lambda x' \text{ for some } x' \in X\}.$$

In other words, $H_\lambda(X)$ is obtained by adding to X all the elements in $\mathcal{U}$ that are $\lambda-$similar to some element in X. Notice that, by Proposition 2.2, if $\lambda \leq \mu(X)$ then we have that

$$H_\lambda(X) = \{x \in \mathcal{U} \mid x\mathcal{R}_\lambda x' \text{ for any } \; x' \in X\}.$$

The following example clarify the previous notion

Example 4.1. Let us consider the domain $U = \{a, b, c, d, e, f, p, q\}$ and a similarity $\mathcal{R}$ such that $\mathcal{R}(a, b) = \mathcal{R}(d, e) = \mathcal{R}(e, f) = \mathcal{R}(f, d) = .7$, $\mathcal{R}(p, q) = .5$, $\mathcal{R}(a, c) = \mathcal{R}(c, b) = .4$, and zero otherwise.
Then $H_{.6}(\{c, d\}) = \{c, d, e, f\}$, and $H_{.3}(\{c, d\}) = \{a, b, c, d, e, f\}$.

The following result is well known in the literature on rough sets [11].

Proposition 4.1. *Let $\lambda \in [0, 1]$, then H_λ is a topological closure operator, i.e., for any X and Y in $\mathcal{P}(\mathcal{U})$,*

$$\begin{aligned}
&H_\lambda(X) \supseteq X\\
&H_\lambda(Y) \supseteq H_\lambda(X) \textit{ if } Y \supseteq X\\
&H_\lambda(H_\lambda(X)) = H_\lambda(X)\\
&H_\lambda(\emptyset) = \emptyset\\
&H_\lambda(X \cup Y) = H_\lambda(X) \cup H_\lambda(Y).
\end{aligned}$$

Moreover

$$\lambda_1 \leq \lambda_2 \Rightarrow H_{\lambda_1}(X) \supseteq H_{\lambda_2}(X).$$

The following proposition gives some useful information about the co-diameter of $H_\lambda(X)$.

Proposition 4.2. *Let X be a cloud and $\lambda \in [0, 1]$. Then*

$$\mu(X) \geq \mu(H_\lambda(X)) \geq \lambda \wedge \mu(X).$$

In particular,
i) $\lambda \geq \mu(X) \Longrightarrow \mu(H_\lambda(X)) = \mu(X)$
ii) $\mu(X) \geq \lambda \Longrightarrow \mu(H_\lambda(X)) \geq \lambda$.

Proof:
For any $x, y \in H_\lambda(X)$ there exist $x', y' \in X$ such that $\mathcal{R}(x, x') \geq \lambda$ and $\mathcal{R}(y, y') \geq \lambda$.
Then

$$\mathcal{R}(x, y) \geq \mathcal{R}(x, y') \wedge \mathcal{R}(y', y) \geq \mathcal{R}(x, x') \wedge \mathcal{R}(x', y') \wedge \mathcal{R}(y', y) \geq \lambda \wedge \mu(X)$$

and this proves that $\mu(H_\lambda(X)) \geq \lambda \wedge \mu(X)$. □

From ii) it follows that if X is a λ-point then $H_\lambda(X)$ is a λ-point, too. Indeed, $H_\lambda(X)$ is the largest λ-point containing X or, equivalently, the complete class of equivalence modulo the λ-similarity containing X.

Proposition 4.3. *Let $\lambda \in [0, 1]$ and let A and B be two λ-points. Then the following are equivalent*
i) $\mu(A \cup B) \geq \lambda$, *i.e. $A \cup B$ is a λ-point*
ii) $H_\lambda(A) = H_\lambda(B) = H_\lambda(A \cup B)$
iii) $H_\lambda(A) \cap H_\lambda(B) \neq \emptyset$.

Proof:
i) $\Rightarrow$ *ii)* Since $\mu(A \cup B) \geq \lambda$, by definition it follows that $H_\lambda(A) \supseteq A \cup B$ and $H_\lambda(B) \supseteq A \cup B$. By Proposition 4.1

$$H_\lambda(A) = H_\lambda(H_\lambda(A)) \supseteq H_\lambda(A \cup B) \text{ and } H_\lambda(B) = H_\lambda(H_\lambda(B)) \supseteq H_\lambda(A \cup B).$$

Moreover,

$$H_\lambda(A) \subseteq H_\lambda(A \cup B) \text{ and } H_\lambda(B) \subseteq H_\lambda(A \cup B).$$

This proves that

$$H_\lambda(A) = H_\lambda(B) = H_\lambda(A \cup B).$$

ii) $\Rightarrow$ *iii)* Immediate.

iii) $\Rightarrow$ *i)* Suppose that an element d in $H_\lambda(A) \cap H_\lambda(B)$ exists. Then there exist two elements d_1 and d_2 in A and B, respectively, such that $\mathcal{R}(d, d_1) \geq \lambda$, $\mathcal{R}(d, d_2) \geq \lambda$. Also, by transitivity of $\mathcal{R}$, $\mathcal{R}(d_1, d_2) \geq \lambda$. Therefore, since $\mu(A) \geq \lambda$ and $\mu(B) \geq \lambda$

$$\begin{aligned} \mu(A \cup B) &= \bigwedge_{x,x' \in A \cup B} \mathcal{R}(x,y) \\ &= \bigwedge_{x,y \in A} \mathcal{R}(x,y) \wedge \bigwedge_{x,y \in B} \mathcal{R}(x,y) \wedge \bigwedge_{x \in A, y \in B} \mathcal{R}(x,y) \\ &\geq \lambda \wedge \bigwedge_{x \in A, y \in B} \mathcal{R}(x,y). \end{aligned}$$

Since $\mu(A) \geq \lambda$ and $\mu(B) \geq \lambda$ and $\mathcal{R}(d_1, d_2) \geq \lambda$, it follows that, for any $x \in A$ and for any $y \in B$

$$\mathcal{R}(x,y) \geq \mathcal{R}(x, d_1) \wedge \mathcal{R}(d_1, y) \geq \mathcal{R}(x, d_1) \wedge \mathcal{R}(d_1, d_2) \wedge \mathcal{R}(d_2, y) \geq \lambda.$$

Therefore $\bigwedge_{x \in A, y \in B} \mathcal{R}(x,y) \geq \lambda$. □

The following corollary is an immediate consequence of the equivalence between *i)* and *ii)* in the previous proposition.

Corollary 4.1. *Let $\lambda \in [0,1]$ and define the relation $\cong_\lambda$by setting, for any pair of clouds X and Y*

$$X \cong_\lambda Y \iff \mu(X \cup Y) \geq \lambda.$$

Then $\cong_\lambda$ is a relation of equivalence on the set of λ-points.

Now we define an operator that allows extending a cloud without changing its co-diameter.

Definition 4.2. The operator $H : \mathcal{P}(\mathcal{U}) \to \mathcal{P}(\mathcal{U})$ is defined by setting

$$H(X) = H_{\mu(X)}(X).$$

The following proposition shows that $H(M)$ is the maximum among the clouds X such that $\mu(X) = \mu(M)$ and $X \supseteq M$.

Proposition 4.4. *Let M be a cloud. Then*
i) $\mu(M) = \mu(H(M))$
ii) if $\mu(X) = \mu(M)$ and $X \supseteq M$, then $H(M) \supseteq X$.

Proof:
Let x, x' be two elements of $H(M)$; by definition of $H(M)$, there exist $m, m' \in M$ such that $\mathcal{R}(x, m) \geq \mu(M)$ and $\mathcal{R}(x', m') \geq \mu(M)$. Then, since $\mathcal{R}(m, m') \geq \mu(M)$,

$$\mathcal{R}(x, x') \geq \mathcal{R}(x, m) \wedge \mathcal{R}(m, x') \geq \mathcal{R}(x, m) \wedge \mathcal{R}(m, m') \wedge \mathcal{R}(m', x') \geq \mu(M).$$

Consequently

$$\mu(H(M)) = \bigwedge_{x,x' \in H(M)} \mathcal{R}(x, x') \geq \mu(M).$$

Since $H(M) \supseteq M$, it is immediate that $\mu(H(M)) \leq \mu(M)$; then we can conclude that $\mu(H(M)) = \mu(M)$.

Let X be a cloud such that $\mu(X) = \mu(M)$ and $X \supseteq M$. If $x \in X$ and m is any element in M, then since $\mathcal{R}(x, m) \geq \mu(X) = \mu(M)$, x is an element in $H(M)$. As a consequence, $H(M) \supseteq X$.
□

As a consequence, we have the following proposition.

Proposition 4.5. *H is a closure operator on $\mathcal{P}(\mathcal{U})$, i.e., for every M and M' in $\mathcal{P}(\mathcal{U})$,*
i) $H(M) \supseteq M$
ii) if $M' \supseteq M$ then $H(M') \supseteq H(M)$
iii) $H(H(M)) = H(M)$.

Proof:

i) Immediate.
ii) We note that $M' \supseteq M$ implies $\mu(M') \leq \mu(M)$. The thesis follows by Proposition 4.1. Indeed,
$H(M') = H_{\mu(M')}(M') \supseteq H_{\mu(M')}(M) \supseteq H_{\mu(M)}(M) = H(M)$.
iii) The case $M = \emptyset$ is obvious; Assume that $M \neq \emptyset$. In this case

$$\begin{aligned} H(H(M)) &= H_{\mu(H(M))}(H(M)) = H_{\mu(H(M))}(H_{\mu(M)}(M)) \\ &= H_{\mu(M)}(H_{\mu(M)}(M)) = H_{\mu(M)}(M) = H(M). \end{aligned}$$

□

The above defined operators can be extended to the systems of clouds.

Definition 4.3. Let $Z = \{M_1, ..., M_n\}$ be a system of clouds and λ any element in $[0,1]$. Then we set

$$H_\lambda(Z) = \{H_\lambda(M_1), ..., H_\lambda(M_n)\}$$

and

$$H(Z) = H_{\xi(Z)}(Z).$$

The following proposition shows that $H(Z)$ gives the largest extension of Z preserving the crispness degree.

Proposition 4.6. *Let $Z = \{M_1, ..., M_n\}$ be a system of clouds in $\mathcal{U}$. Then*
i) $\xi(H(Z)) = \xi(Z)$
ii) $H(H(Z)) = H(Z)$
iii) if $Z' = \{M_1^{'}, ..., M_n^{'}\}$ with $\xi(Z) = \xi(Z')$ and $M_i^{'} \supseteq M_i$ then, for any $i = 1, ..., n$, $H_{\xi(Z)}(M_i) \supseteq M_i^{'}$.

Proof:

i) Since $M_i \subseteq H_{\xi(Z)}(M_i)$ for any $i = 1, ..., n$ it is sufficient to prove that $\xi(H(Z)) \geq \xi(Z)$. With this aim, we prove that, for any $i = 1, ..., n$, $\mu(H_{\xi(Z)}(M_i)) \geq \xi(Z)$. Let $\bar{x} \in M_i$. By definition, for any $x \in H_{\xi(Z)}(M_i)$, there exists an y in M_i such that $\mathcal{R}(x, y) \geq \xi(Z)$. So

$$\mathcal{R}(\bar{x}, x) \geq \mathcal{R}(\bar{x}, y) \wedge \mathcal{R}(y, x) \geq \mu(M_i) \wedge \xi(Z) = \xi(Z).$$

By Proposition 2.3 the thesis follows.

ii) Immediate from the definition of H, by *i)*.

iii) Since, $\mu(M_i') \geq \xi(Z)$ and $M_i^{'} \supseteq M_i$, we have that, for every $m' \in M_i'$ and $m \in M_i$, $\mathcal{R}(m, m') \geq \xi(Z)$. This proves that $m' \in H_{\xi(Z)}(M_i)$. □

Proposition 4.7. *Let Z be a system of clouds, then $H_\lambda(Z)$ is compact for any $\lambda \leq \xi(Z)$. In particular, $H(Z)$ is compact.*

Proof:
Indeed, by Proposition 4.3, two clouds in $H_\lambda(Z)$ are either coincident or disjoint. □

5. Extending Unification Through a Similarity Relation

Let $\mathcal{L}$ be a function-free first-order language and denote with
- $\mathcal{V}$ its *set of variables*
- $\mathcal{C}$ its *set of constants*
- $\mathcal{P}$ its set of *predicate symbols.*

We assume that $\mathcal{V}$ is ordered in a sequence $x_1, x_2, \ . \ . \ .$ Also, denote by $\mathcal{L}'$ the language with the same variables and predicate symbols as $\mathcal{L}$ but whose set of constants is the set $\mathcal{C}'$ of nonempty clouds of constants in $\mathcal{L}$. $\mathcal{L}'$ is an extension of $\mathcal{L}$, indeed we can identify any constant c in $\mathcal{C}$ with the singleton $\{c\}$ in $\mathcal{C}'$. In the sequel, we call *extended-type* (briefly *e-type*), with *type*$\in$*{constant, term, formula}* a type of $\mathcal{L}'$. Also, we denote with $\mathcal{T}_{\mathcal{V},\mathcal{C}'}$ the *set of e-terms* , i.e., the set $\mathcal{V} \cup \mathcal{C}'$. An *extended substitution* (denoted by *e-substitution*) is a map $\theta : \mathcal{V} \to \mathcal{T}_{\mathcal{V},\mathcal{C}'}$. We denote with Θ the set of e-substitutions.

We assume as primitive a similarity relation *eq* on $\mathcal{P} \cup \mathcal{C} \cup \mathcal{V}$ such that $eq(t,t') = 0$ whenever
-t and t' are not both in $\mathcal{V}$ or $\mathcal{C}$ or P
-t and t' are in $\mathcal{V}$ and $t \neq t'$
-t and t' are predicates with different arities.

In other words, *eq* is union of a similarity in $\mathcal{C}$, the identity in $\mathcal{V}$ and a similarity among symbols of predicates with the same arities. Given two e-terms t, t', we set

$$t\dot{\sqcup}t' = \begin{cases} t \cup t' \text{ if } t \text{ and } t' \in \mathcal{C}' \\ \{t\} \cup t' \text{ if } t \in \mathcal{V}, t' \in \mathcal{C}' \\ \{t'\} \cup t \text{ if } t \in \mathcal{C}', t' \in \mathcal{V} \\ \{t, t'\} \text{ if } t, t' \in \mathcal{V}. \end{cases}$$

We define a relation $\overline{eq}$ on the set of atomic e-formulae by setting

$$\overline{eq}(p(t_1, \ldots, t_n), q(t'_1, \ldots, t'_m)) = \begin{cases} eq(p,q) \wedge \left(\bigwedge_{i=1}^{n} \mu(t_i\dot{\sqcup}t'_i) \right) \text{ if } n = m \\ 0 \text{ otherwise.} \end{cases}$$

Such a relation is *not* a similarity since it may be *not* reflexive. Indeed, if t is not a singleton, $\mu(t\dot{\sqcup}t) = \mu(t) \neq 1$, in general.

Definition 5.1. An extended equation (briefly *e-equation*) is an equation between two atomic e-formulas with the same arity.

Given a system of e-equations S, and an e-substitution θ, we define the *unification degree* $\nu(S,\theta)$ *of* S *with respect to* θ as follows

$$\nu(S,\theta) = \bigwedge_{e=e' \in S} \overline{eq}(\theta(e), \theta(e')).$$

Finally, the *unification degree* $U(S)$ of a system of e-equations S is defined as

$$U(S) = \bigvee_{\theta \in \Theta} \nu(S,\theta).$$

The following example clarifies the previous notion

Example 5.1. Let us consider the domain $U = \{a, b, c, d, e, f, p, q\}$ and a similarity $\mathcal{R}$ such that $\mathcal{R}(a,b) = \mathcal{R}(d,e) = \mathcal{R}(e,f) = \mathcal{R}(f,d) = .7$, $\mathcal{R}(p,q) = .5$, $\mathcal{R}(a,c) = \mathcal{R}(c,b) = .4$.
If we consider the substitution $\theta = \{x/\{a\},\ y/\{b\}\}$ and the set of e-equations
$S = \{p(x, \{e,d\}) = q(\{a\}, \{f\}),\ p(\{a\}, \{y\}) = p(\{a,b\}, \{b\})\}$,
since it is

$\overline{eq}(\theta(p(x,\{e,d\})) = \theta(q(\{a\},\{f\}))) = eq(p,q) \wedge \mu(\{a\}) \wedge \mu(\{e,d,f\}) =$
$.5 \wedge 1 \wedge .7 = .5$
$\overline{eq}(\theta(p(\{a\},\{y\})) = \theta(p(\{a,b\},\{b\}))) = eq(p,p) \wedge \mu(\{a,b\}) \wedge \mu(\{b\}) =$
$1 \wedge .7 \wedge 1 = .7$
it follows that
$\nu(S,\theta) =$
$\overline{eq}(\theta(p(x,\{e,d\})) = \theta(q(\{a\},\{f\}))) \wedge \overline{eq}(\theta(p(\{a\},\{y\})) = \theta(p(\{a,b\},\{b\}))) =$
$.5 \wedge .7. = .5.$
It is easy to verify that .5 is the maximum possible unification degree, then $U(S) = \nu(S,\theta) = .5$.

Notice that, differently from the classical case, the fact that the left and the right sides of the equations in S coincide, does not entails that $U(S) = 1$. As an example, let r be an unary predicate and a and b two constants such that $eq(a,b) \neq 1$. Then, by setting $S = \{r(\{a,b\}) = r(\{a,b\})\}$, we obtain that

$$U(S) = eq(r,r) \wedge \mu(\{a,b\}) = eq(a,b) \neq 1.$$

Definition 5.2. Given a system of e-equations S such that $U(S) \neq 0$ we say that an e-substitution θ is an *extended unifier* (briefly *e-unifier*) for S provided that $\nu(S,\theta) = U(S)$.

It is worth noticing that the notion of e-unifier coincides with the usual one of unifier when S is a system of equations and eq is the identity relation.

6. e-Unifiers of Systems of clouds

In the sequel we consider clouds that are either subsets of $\mathcal{C} \cup \mathcal{V}$ or subsets of $\mathcal{P}$. If M is a cloud in $\mathcal{C} \cup \mathcal{V}$, we write $M = X \sqcup D$ to denote that X is the set of variables and D the set of constants occurring in M. In our approach, the systems of clouds are exploited to compute e-unifiers for systems of e-equations. With this aim, we define a function associating to any compact system of clouds $Z = \{X_1 \sqcup D_1, ..., X_n \sqcup D_n\}$ an e-substitution $\theta_Z = Assoc\_sub(Z)$ where, for any $x \in \mathcal{V}$
$\theta_Z(x) = x$ if $x \notin X_1 \cup ... \cup X_n$
$\theta_Z(x) = D_i$ if $x \in X_i$ and $D_i \neq \emptyset$
$\theta_Z(x) = x_j$ *if* $x \in X_i, D_i = \emptyset$ and x_j is the first variable occurring in X_i.
Also, we consider two functions $Trans\_t$ and $Trans\_p$ transforming a
system of e-equations into two systems of clouds in $\mathcal{C} \cup \mathcal{V}$ and $\mathcal{P}$, respectively. First, we obtain two systems of clouds by setting

$$\begin{aligned} Split\_p(p(t_1,...,t_m) &= q(t'_1,...,t'_m)) = \{\{p,q\}\} \\ Split\_t(p(t_1,...,t_m) &= q(t'_1,...,t'_m)) = \{t_1 \dot{\sqcup} t'_1, ..., t_m \dot{\sqcup} t'_m\}. \end{aligned}$$

Moreover, if $S = \bigcup_{i=1}^{n} \{e_i = e'_i\}$, we set

$$\begin{aligned} Trans\_p(S) &= \bigcup_{i=1\ldots n} Split\_p(e_i = e'_i) \\ Trans\_t(S) &= \bigcup_{i=1\ldots n} Split\_t(e_i = e'_i). \end{aligned}$$

Let us stress that both $Trans\_p(S)$ and $Trans\_t(S)$ are systems of clouds. As an example, let $S = \{p(x, \{a, c\}, y) = q(a, b, b),\, r(x, x, y) = s(a, \{b, c\}, c)\}$.

Then

$$\begin{aligned} Trans\_p(S) &= \{\{p, q\}, \{r, s\}\} \\ Trans\_t(S) &= \{\{x, a\}, \{a, b, c\}, \{y, b\}, \{x, b, c\}, \{y, c\}\} \end{aligned}$$

As we will prove, an e-unifier of a system of e-equations S is computed through the associated system of clouds $Compact(Trans\_t(S))$. More precisely, we will prove that the function $Assoc\_sub(Compact(Trans\_t(S)))$ provides an e-unifier of S.

Given a cloud M in $\mathcal{V} \cup \mathcal{C}$ and an e-substitution θ, we denote by $\theta(M)$ the cloud obtained by substituting each variable x in M with

-$\theta(x)$ if $\theta(x)$ is a variable

-the elements in $\theta(x)$ if $\theta(x)$ is a cloud of constants.

Obviously, since θ operates only on the variables, if $M = X \sqcup D$, then $\theta(X \sqcup D) = \theta(X) \cup D$. Given a system $Z = \{M_1, ..., M_n\}$ where $M_1, ..., M_n$ are clouds in $\mathcal{C} \cup \mathcal{V}$, we denote with $\theta(Z)$ the system $\{\theta(M_1), ..., \theta(M_n)\}$.

Now, we will extend some definitions given for systems of e-equations to systems of clouds.

Definition 6.1. Given a system of clouds Z, and an e-substitution θ, the *unification degree of Z with respect to θ* is defined by

$$\nu(Z, \theta) = \xi(\theta(Z)).$$

The *unification degree of Z* is the number

$$U(Z) = \bigvee_{\theta \in \Theta} \nu(Z, \theta).$$

If $U(Z) \neq 0$, we call *e-unifier* of Z any e-substitution θ such that $\nu(Z, \theta) = U(Z)$.

Definition 6.2. Let $M = X \sqcup D$ be a cloud. Then we set $Ground(M) = D$. If $Z = \{M_1, \ldots, M_t\}$ is a system of clouds, we set

$$Ground(Z) = \{Ground(M_1), \ldots, Ground(M_t)\}.$$

The proof of the following proposition is immediate.

Proposition 6.1. *Let Z be a compact system of clouds such that $U(Z) \neq 0$. Then $Assoc\_sub(Z)$ is an e-unifier of Z. Moreover,*

$$U(Z) = \xi(Ground(Z)).$$

Proof:
For any e-substitution θ,

$$\begin{aligned}\xi(\theta(Z)) &= \bigwedge_{X \sqcup D \in Z} \mu(\theta(X \sqcup D)) \\ &= \bigwedge_{X \sqcup D \in Z} \mu(\theta(X) \sqcup D) \leq \bigwedge_{X \sqcup D \in Z} \mu(D) = \xi(Ground(Z)).\end{aligned}$$

Since $\xi(Assoc\_sub(Z)) = \xi(Ground(Z))$, the thesis follows. □

The following proposition shows that Z and $Compact(Z)$ have the same unifiers.

Proposition 6.2. *Let Z be a system of clouds. Then, for any e-substitution θ*

$$\nu(Compact(Z), \theta) = \nu(Z, \theta).$$

Proof:
We first prove that

$$\xi(\theta(T(Z))) = \xi(\theta(Z)). \tag{1}$$

We begin by proving that, for $i = 1, ..., n$,

$$\mu(\theta(T_Z(M_i))) \geq \xi(\theta(Z)). \tag{2}$$

Indeed, since from $M_i \cap M_j \neq \emptyset$ we have that $\theta(M_i) \cap \theta(M_j) \neq \emptyset$,
by Proposition 2.3

$$\begin{aligned}\mu(\theta\,(T_Z(M_i))) &= \mu\left(\theta\left(\bigcup_{M_j \cap M_i \neq \emptyset} M_j\right)\right) \\ &= \mu\left(\bigcup_{M_j \cap M_i \neq \emptyset} \theta(M_j)\right) = \bigwedge_{M_j \cap M_i \neq \emptyset} \mu\,(\theta(M_j)) \\ &\geq \bigwedge_{i=1}^{n} \mu\,(\theta(M_i)) = \xi(\theta(Z)).\end{aligned}$$

That proves inequality (2).
From (2), it follows that

$$\xi(\theta(T(Z))) = \bigwedge_{i=1}^{n} \mu(\theta\,(T_Z(M_i))) \geq \xi(\theta(Z)).$$

On the other hand, since for any $i = 1, ..., n$ $M_i \subseteq T_Z(M_i)$,

$$\xi(\theta(Z)) = \bigwedge_{i=1}^{n} \mu\left(\theta(M_i)\right) \geq \bigwedge_{i=1}^{n} \mu(\theta\left(T_Z(M_i)\right) = \xi(\theta(T(Z))).$$

And this completes the proofs of (1).

We proceed by induction on the $T-$degree $\Phi(Z)$ of Z.

-If $\Phi(Z) = 0$ then Z is compact and therefore the thesis follows.

-If $\Phi(Z) = n$, then $\Phi(T(Z)) = n - 1$. Therefore, by induction hypothesis

$$\xi(\theta(Compact(T(Z)))) = \xi(\theta(T(Z))).$$

Since Z has $T-$degree n, it is $Compact(Z) = T^n(Z) = T^{n-1}(T(Z)) = Compact(T(Z))$. Then, by (1), $\xi(\theta(Compact(Z))) = \xi(\theta(Z))$. □

7. Computing e-unifiers of systems of e-equations

Now we are ready to prove that the unification degree of a system of clouds or e-equations is an invariant with respect to the transformations *Compact* and *Trans*, as stated by the following theorem.

Theorem 7.1. *Let S be a system of e-equations. Then, for any extended substitution θ*
i) $\nu(S,\theta) = \nu(Trans\_t(S),\theta) \wedge \xi(Trans\_p(S))$
and therefore
ii) $\nu(S,\theta) = \nu(Compact(Trans\_t(S)),\theta) \wedge \xi(Compact(Trans\_p(S)))$.

Proof:

i) Observe that, if $e = p(t_1, ..., t_n)$ and $e' = q(t'_1, ..., t'_n)$ then, since it is $Split\_t(\theta(e_i) = \theta(e'_i)) = \theta(Split\_t(e_i = e'_i))$ for any $i = 1, .., n$

$$\begin{aligned} \overline{eq}(\theta(e), \theta(e')) &= \left(\bigwedge_{i=1}^{n} \mu\left(\theta(t_i)\dot{\sqcup}\theta(t'_i)\right)\right) \wedge eq\,(p,q) \\ &= \left(\bigwedge_{i=1}^{n} \mu\left(\theta(t_i\dot{\sqcup}t'_i)\right)\right) \wedge eq\,(p,q) \\ &= \xi(\theta(Split\_t(e = e'))) \wedge \xi(Split\_p(e = e')). \end{aligned}$$

Therefore

$$\begin{aligned} \nu(S,\theta) &= \bigwedge_{e=e' \in S} \overline{eq}(\theta(e), \theta(e')) \\ &= \bigwedge_{e=e' \in S} \xi(\theta(Split\_t(e = e'))) \wedge \bigwedge_{e=e' \in S} \xi(Split\_p(e = e')) \end{aligned}$$

$$= \xi\left(\theta\left(\bigcup_{e=e'} Split\_t\,(e=e')\right)\right) \wedge \xi\left(\bigcup_{e=e'} Split\_p\,(e=e')\right)$$
$$= \xi(\theta(Trans\_t(S))) \wedge \xi(Trans\_p(S)).$$

ii) Follows immediately from *i*) and Proposition 6.2. □

Given a system S of e-equation, we set

$$Cond(S) = Ground(Compact(Trans\_t(S))) \cup Compact(Trans\_p(S)).$$

and we say that $Cond(S)$ is *the system of conditions for the unification of* S. If every cloud in $Cond(S)$ is a singleton then S admits a classical unifier. Otherwise, as we will prove in the next theorem, the degree of crispness of $Cond(S)$ gives the degree of unification of S. In other words, $\xi(Cond(S))$ is the "cost" we have to pay to allow the unification process.

Theorem 7.2. *Let S be a system of e-equations, then*

$$U(S) = \xi(Cond(S)).$$

Moreover, if $U(S) \neq 0$ then

$$\theta_S = Assoc\_sub(Compact(Trans\_t(S)))$$

is an e-unifier of S.

Proof:
It is an immediate consequence of Theorem 7.1 and Proposition 6.1. □

This just proven theorem gives an effective procedure to compute the unification degree of a system of e-equations S, and, at the same time, an e-unifier of S.

Corollary 7.1. *Let S be a system of e-equations, then θ is an e-unifier of S if and only if either*

- *θ is an unifier of $Compact(Trans\_t(S))$*

or

- *$\xi(\theta(Compact(Trans\_t(S)))) \geq \xi(Compact(Trans\_p(S)))$.*

Another characterization of the class of e-unifiers of a system of e-equations is given by the following proposition.

Proposition 7.1. *Let S be a system of e-equations such that $U(S) \neq 0$. An e-substitution θ is an e-unifier of S if and only if*

$$\xi(\theta(Compact(Trans\_t(S)))) \geq U(S).$$

Proof:

$\Longrightarrow$ Since θ is an e-unifier, $\nu(S,\theta) = U(S)$. Then, by theorem 7.1,

$$U(S) = \nu(Compact(Trans\_t(S)),\theta) \wedge \xi(Compact(Trans\_p(S))).$$

As a consequence, $\nu(Compact(Trans\_t(S)),\theta) \geq U(S)$.

$\Longleftarrow$ Since $\xi(\theta(Compact(Trans\_p(S)))) \geq U(S)$, by Theorem 7.1, $\nu(S,\theta) \geq U(S)$. The thesis follows immediately. □

Example 7.1. Consider the system of e-equations

$$S = \{p(c,w,c) = q(x,x,y),\ p(x,z,x) = r(d,\{a,e\},b)\}.$$

Then,

$$Trans\_p(S) = \{\{p,q\},\{p,r\}\},$$

$$Trans\_t(S) = \{\{x,c\},\{w,x\},\{c,y\},\{x,d\},\{z,a,e\},\{x,b\}\}$$

and therefore,

$$Compact(Trans\_p(S)) = \{\{p,q,r\}\},$$

$$Compact(Trans\_t(S)) = \{\{x,y,w,c,b,d\},\{z,a,e\}\},$$

This entails that

$$Cond(S) = \{\{p,q,r\},\{c,b,d\},\{a,e\}\}$$

and therefore

$$U(S) = \mu(\{p,q,r\}) \wedge \mu(\{c,b,d\}) \wedge \mu(\{a,e\}).$$

Finally, if $U(S) \neq 0$, the e-substitution

$$\theta_S = \{x/\{c,b,d\}, y/\{c,b,d\}, w/\{c,b,d\}, z/\{a,e\}\}$$

is an e-unifier of S.

8. The Most General Unifier

A further step is to prove that the unifier θ_S furnished in the previous section is the best one and that, in a sense, it represents the whole class of unifiers for S. To this purpose, we have to extend the classical concept of most general unifier (*mgu*). Recall that, given two substitutions θ and θ', we say that θ' is *more general than* θ, and we write $\theta' \preceq \theta$, if there exists a substitution τ such that, for any variable x, $\theta(x) = \tau(\theta'(x))$. In order to extend such a notion, we replace the identity with the relation $\cong_\lambda$. As usual, if τ and θ are two e-substitutions, we define $\theta\tau$ as the e-substitution which sets $\theta\tau(x) = \theta(x)$, if $\theta(x)$ is a ground e-term, and $\theta\tau(x) = \tau(\theta(x))$ otherwise.

Definition 8.1. Let $\lambda \in (0,1]$,we say that θ is a λ-substitution if and only if $\theta(x)$ is a λ-point for any variable x.

Obviously, if both τ and θ are λ-substitution then $\theta\tau$ is a λ-substitution, too. Also, we extend $\cong_\lambda$ to λ-substitutions in a pointwise manner by setting $\theta \cong_\lambda \theta'$ whenever $\theta(x) \cong_\lambda \theta'(x)$ for any $x \in \mathcal{V}$.

Definition 8.2. Let $\lambda \in (0,1]$ and let θ and θ' be two λ-substitutions. Then we set $\theta \preceq_\lambda \theta'$ if a λ-substitution τ exists such that, $\theta' \cong_\lambda \theta\tau$. In this case we say that θ *is more general than* θ' (*w.r.t. the level* λ).

Proposition 8.1. *For any* $\lambda \in (0,1]$, *the relation* $\preceq_\lambda$ *is a pre-order over the set of* λ*-substitutions.*

Proof:
Since $\equiv_\lambda$ is reflexive, it is immediate that $\preceq_\lambda$is reflexive. Let θ, θ' and θ'' be λ-substitutions such that $\theta \preceq_\lambda \theta'$ and $\theta' \preceq_\lambda \theta''$. By definition, there exist two λ-substitutions τ and τ' such that, for any variable x in $\mathcal{V}$, $\theta'(x) \equiv_\lambda \tau(\theta(x))$ and $\theta''(x) \equiv_\lambda \tau'(\theta'(x))$. We claim that $\theta''(x) \equiv_\lambda \tau'(\tau(\theta(x)))$ and therefore that $\theta \preceq_\lambda \theta''$. Indeed, if $\theta'(x)$ is ground then

$$\theta''(x) \equiv_\lambda \tau'(\theta'(x)) = \theta'(x) \equiv_\lambda \tau(\theta(x)).$$

If $\theta'(x) \in \mathcal{V}$ then, since $\lambda \neq 0, \tau(\theta(x))$ is a variable, too, and $\theta'(x) = \tau(\theta(x))$. By substituting in $\theta''(x) \equiv_\lambda \tau'(\theta'(x))$ we obtain $\theta''(x) \equiv_\lambda \tau'(\tau(\theta(x)))$. □

The following is a characterization of $\preceq_\lambda$ that does not use e-substitutions.

Proposition 8.2. *Let* θ *and* θ' *be two* λ*-substitutions. Then* $\theta \preceq_\lambda \theta'$ *if and only if the following conditions hold*
i) $\theta(x)$ *ground* $\Longrightarrow \theta'(x) \cong_\lambda \theta(x)$
ii) $\theta(x)$ *variable* $\Longrightarrow [\theta(x) = \theta(y) \Longrightarrow \theta'(x) \cong_\lambda \theta'(y)]$.

Proof:
Suppose that $\theta \preceq_\lambda \theta'$ and therefore that an e-substitution τ exists such that $\theta'(x) \cong_\lambda \tau(\theta(x))$. Then, if $\theta(x)$ is ground, $\theta'(x) \cong_\lambda \tau(\theta(x)) = \theta(x)$. Assume that $\theta(x) \in \mathcal{V}$ and that $\theta(x) = \theta(y)$. Then

$$\theta'(x) \cong_\lambda \tau(\theta(x)) = \tau(\theta(y)) \cong_\lambda \theta'(y).$$

Conversely, suppose that conditions i) and ii) hold. Then, the substitution τ, defined as follows

$$\tau(z) = \begin{cases} \theta'(y) \text{ if } y \text{ is the first variable such that } z = \theta(y) \\ z \text{ if no variable } y \text{ exists such that } z = \theta(y) \end{cases}$$

is well defined. Moreover, for any x in $\mathcal{V}$, $\theta'(x) \cong_\lambda \tau(\theta(x))$. In fact, if $\theta(x)$ is a variable, then there exists the first variable y such that $\theta(x) = \theta(y)$. By construction, $\tau(\theta(x)) = \theta'(y)$ and by property *ii)* $\tau(\theta(x)) \cong_\lambda \theta'(x)$. If $\theta(x)$ is ground, it is immediate that $\tau(\theta(x)) = \theta(x)$ and, by property *i)*, $\tau(\theta(x)) \cong_\lambda \theta'(x)$. □

Given a system of e-equations S, we focus our attention upon the relation $\preceq_{U(S)}$.

Definition 8.3. Let S be a system of e-equations with $U(S) \neq 0$ and θ an e-substitution. Then θ is an *extended most general unifier* (denoted with *e-mgu*) for S if

- θ is an e-unifier of S
- for any e-unifier γ for S, $\theta \preceq_{U(S)} \gamma$.

It is easy to see that the e-unifier in the Example 5.1 is an e-mgu.

We can conclude this section by showing that the unifier furnished in the previous section is the best one.

Theorem 8.1. *Let S be a system of e-equations such that $U(S) \neq 0$ and*

$$\theta_S = Assoc\_sub(Compact(Trans\_t(S))).$$

Then θ_S is a most general e-unifier of S.

Proof:
Set

$$Compact(Trans\_t(S)) = \{X_1 \sqcup D_1, ..., X_n \sqcup D_n\},$$

and let θ be an e-unifier of S. Let x be a variable such that $\theta_S(x)$ is ground. Then X_i exists such that $x \in X_i$ and $D_i \neq \emptyset$. Since θ is an e-unifier of S, by Proposition 7.1, $\mu(\theta(x) \cup D_i) \geq \mu(\theta(X_i) \cup D_i) \geq U(S) \neq 0$, then $\theta(x)$ is ground, too and therefore $\theta(x) \cong_{U(S)} \theta_S(x)$.

Assume that $\theta_S(x)$ is a variable, and that $\theta_S(x) = \theta_S(y)$, then X_i exists such that both x and y belong to X_i and $D_i = \emptyset$. In this case, since $\mu(\theta(x)\dot{\sqcup}\theta(y)) \geq \mu(\theta(X_i)) \geq U(S)$, we can conclude that $\theta(x) \cong_{U(S)} \theta(y)$. □

9. The structure of the set of e-unifiers

Unlike the classical case, an instantiation of an e-unifier is *not* an e-unifier, in general. As an example, $\theta = \{x/y, z/b\}$ is an e-unifier of the system $S = \{r(x,z) = r(y,b)\}$, and $U(S) = 1$. By composing θ with the e-substitution $\tau = \{y/\{a,b\}\}$, we obtain the e-substitution $\theta' = \{x/\{a,b\}, z/b, y/\{a,b\}\}$. If we assume that $eq(a,b) \neq 1$, then it is immediate that θ' is not an e-unifier of S. The following proposition shows that if the instantiation was made by a $U(S)-$substitution then we obtain an unifier, too.

Proposition 9.1. *Let S be a system of e-equations and θ an e-unifier of S. Then, for every $U(S)$-substitution τ, $\theta\tau$ is an e-unifier of S.*

Proof:
On account of Proposition 7.1, we have to prove that $\mu(\tau(\theta(X)) \cup D) \geq U(S)$ for any $X \sqcup D \in Compact(Trans\_t(S))$. Since θ is an e-unifier only two cases are possible. In the firt one $\theta(x) \in C'$ for any $x \in X$. Then $\tau(\theta(x)) = \theta(x)$ and, since θ is an e-unifier

$$\mu(\tau\theta(X) \cup D) = \mu(\theta(X) \cup D) \geq U(S).$$

In the second case, $\theta(X)$ reduces to a variable z and $D = \emptyset$, then, since τ is a $U(S)$-substitution,

$$\mu(\tau(\theta(X))) = \mu(\tau(z)) \geq U(S).$$

$\square$

In the following proposition we show that every e-substitution $U(S)$-similar to an e-unifier is an e-unifier.

Proposition 9.2. *Let θ be an e-unifier of a system of e-equations S and assume that $\theta \cong_{U(S)} \theta'$, then θ' is an e-unifier of S.*

Proof:
Let $M = X \sqcup D$ be a cloud in $Compact(Trans\_t(S))$ and observe that, since $H_{U(S)}(\theta(x)) = H_{U(S)}(\theta'(x))$ for every $x \in \mathcal{V}$, it is

$$\begin{aligned} H_{U(S)}(\theta'(X) \cup D)) &= H_{U(S)}(D)) \cup (\bigcup\{H_{U(S)}(\theta'(x)) \mid x \in \mathcal{V}\}) \\ &= H_{U(S)}(D)) \cup (\bigcup\{H_{U(S)}(\theta(x)) \mid x \in \mathcal{V}\}) \\ &= H_{U(S)}(\theta(X) \cup D). \end{aligned}$$

Now, since θ is an unifier, on account of Proposition 7.1, $\mu(\theta(X) \cup D) \geq U(S)$ and therefore, by ii) in Proposition 8, $\mu(H_{U(S)}(\theta(X) \cup D)) \geq U(S)$. Thus,

$$\mu(\theta'(X) \cup D) \geq \mu(H_{U(S)}(\theta'(X) \cup D)) = \mu(H_{U(S)}(\theta(X) \cup D)) \geq U(S),$$

and therefore, by Proposition 7.1, θ' is an unifier. $\square$

As an immediate consequence of the definition of e-mgu and of the just proved propositions, we have the following basic theorem.

Theorem 9.1. *Let S be a system of e-equations and θ an e-mgu of S. Then θ' is an e-unifier of S if and only if $\theta \preceq_{U(S)} \theta'$.*

We conclude by observing that the unifier θ_S furnished by the proposed algorithm is still unsatisfactory. In fact, consider the Example 7.1 and assume that

$$eq(a,e) = eq(c,b) = eq(d,c) = 0.6,\ eq(f,d) = 0.9,\ eq(p,q) = eq(q,r) = 1.$$

Then

$$U(S) = \nu(S, \theta_S) = \xi(\{\{c,b,d\},\{a,e\}\}) = 0.6.$$

Since $eq(f,d) = 0.9 \geq 0.6$, we have also that $\mu(\{f,b,c,d\}) = \mu(\{c,b,d\})$. This means that the e-unifier

$$\theta' = \{x/\{f,c,b,d\}, y/\{f,c,b,d\}, z/\{a,e\}, w/\{f,c,b,d\}\}$$

solves the unification problem at the same "cost" of θ. Moreover, θ' is more significative than θ from the point of view of the information conveyed. Indeed, if we admit the threshold of validity 0.6, then no reason exists in distinguishing f from the elements in $\{c,b,d\}$.

In account of these observations, we present a manner of expanding the clouds in an unifier without altering the cost payed for the unification.

Definition 9.1. Let θ be an e-substitution and $\lambda \in [0,1]$. We call $\lambda-$*completion of* θ the e-substitution θ_λ defined by setting $\theta_\lambda(x) = H_\lambda(\theta(x))$.

Obviously, if θ is a λ-completion then θ_λ is an λ-substitution λ-similar to θ. The proof of the following proposition is straightforward.

Proposition 9.3. *Let S be a system of e-equations such that $U(S) \neq 0$, and let $\overline{\theta}_S$ be the λ-completion of $\theta_S = Assoc\_sub(Compact(Trans\_t(S)))$. Then $\overline{\theta}_S$ is an e-mgu for S.*

On account of Proposition 9.3, given a system of e-equations S, the completion of an e-mgu can be actually computed and it is still an extended mgu of the system S.

10. Conclusions and Future Developments

In this paper we introduced a similarity in a function-free first order language and we defined an extended unification theory, accordingly. Then, the first perspective is to use the proposed relaxed unification for a relaxed resolution rule in Logic Programming and automated deduction. At this time we are working actively in this direction. Whenever, we envisage its use in equational unification and rewriting systems, as well. Also, we will extend unification theory in such a way that similarity can be introduced in a first order language with a full-fledged set of function symbols.

In this paper the min-transitive similarity relation has been considered as starting point to study a similarity-based extension of the crisp notion of unification. However, in many applications it is useful to consider similarities related with T-norms different from the minimum. The usual product between real number or the Lukasievicz product look to be the natural candidates. Then the question arises to analyze the behavior of the proposed extended unification also in these cases. Finally, the obtained results could be framed in the general approach to fuzzy set theory based on similarity (see, e.g., [13]).

Acknowledgments

The author would like to thank the referees for useful comments and suggestions.

References

[1] F.Arcelli, F.Formato and G.Gerla, Similitude-Based Unification as a Foundation of Fuzzy Logic Programming. In *Proceedings of Internat. Workshop on Logic Programming and Soft Computing*, Bonn, sept.,1996.

[2] Biacino L., Gerla G., Logics with approximate premises, International J. of Intell. Syst. 13 (1998) 1-10.

[3] J.F.Baldwin, T.P.Martin and B.W. Pilsworth, *FRIL - Fuzzy and Evidential Reasoning in AI.* Research Studies Press, 1995.

[4] D.Dubois, J.Lang and H.Prade, Towards Possibilistic Logic programming. In *Proceedings of ICLP91*, Paris, 1991.

[5] Gärdenfors, P. 1992, "A geometric model of concept formation", pp. 1-16 in Information Modelling and Knowledge Bases III, ed. by S. Ohsuga et al., IOS Press, Amsterdam.

[6] L.Godo and P. Hájek, Fuzzy Inference as Deduction. *Journal of Applied non-classical Logic.*To appear in 1998.

[7] Kowalski, R.A., Predicate logic as a programming language, in: *Proc. IFIP'74* (1974) 569-574.

[8] R.C.T.Lee, "Fuzzy Logic and the Resolution Principle", *Journal of the ACM*, 19, 1972, pp.109-119.

[9] A. Martelli and U.Montanari, An Efficient Unification Algorithm. *ACM Transactions on Programming Languages and Systems*, Vol 4, No 2, April 1982.

[10] M.Mukaidono, Z.Shen and L.Ding, Fundamentals of Fuzzy Prolog. *Int. Journal of Approximated Reasoning*, 3,2, 1989.

[11] Z. Pawlak, Rough Sets, Rough Relations and Rough Functions. *Fundamenta Informaticae* 27(2/3): 103-108 (1996)

[12] E. Plaza, F. Esteva, P. García, L. Godo, R. López de Mántaras, A logical Approach to Case-Based Reasoning Using Fuzzy Similarity Relations. Information Sciences.

[13] Ruspini, E.H., On the semantics of fuzzy logic, *Int. Journal of Approximate Reasoning* 5 (1991) 45-88.

[14] P. Arenas-Sanchez and A. Dovier, A Minimality Study for Set Unification. *The Journal of Functional and Logic Programming*, Volume 1997, No. 7.

[15] H.Virtanen. A Study in Fuzzy Logic Programming, In *Proceedings of the 12th European meeting on Cybernetics and Systems'94*, pp. :249–256, Austria, April 1994.

[16] M.S.Ying, A Logic for Approximated Reasoning, *The Journal of Symbolic Logic*, 59, 1994.