Proceedings of the 13th International Conference
on Computational and Mathematical Methods
in Science and Engineering, CMMSE 2013
24–27 June, 2013.

Relaxing the Role of Adjoint Pairs in Multi-adjoint Logic Programming

G. Moreno, J. Penabad and C. Vázquez[1]

[1] *University of Castilla-La Mancha, Faculty of Computer Science Engineering*
02071, Albacete (Spain),

emails: {Gines.Moreno,Jaime.Penabad,Carlos.Vazquez}@uclm.es

Abstract

Multi-adjoint logic programming, MALP in brief, represents a modern and powerful approach for fuzzifying pure logic programming, by dealing with truth degrees and connectives collected from a lattice more complex than the trivial one based on two simple values *true* and $false$. Each MALP program is associated with its own multi-adjoint lattice, equipped with a complete or partial ordering among the (possibly infinite set of) elements and a wide set of fuzzy connectives where it is crucial to connect each implication symbol with a proper conjunction thus conforming constructs of the form $(\leftarrow_i, \&_i)$. Initially used for modeling a fuzzy variant of the classical *modus ponens* inference rule, the use of the so-called *adjoint pairs* strongly affects the definitions of both declarative and operational semantics (as well as their corresponding soundness and completeness properties) of the MALP framework. In this work we relax this impact in two stages. Firstly, we show a sub-class of MALP programs for which it is possible to define simpler versions of models and computational steps non depending on adjoint pairs. Then, and what is the best, we show a simple technique able to map every MALP program with other one in this sub-class, thus moving the need for using adjoint pairs from the semantic core of MALP to a purely syntactic pre-process.

Key words: Multi-adjoint Logic Programming, Adjoint Pairs
MSC 2000: Multi-adjoint Lattice

1 Introduction

⋆

TO DO...

 ISBN: 978-84-616-2723-3

$\&_{\mathtt{P}}(x,y)$	$\triangleq$	$x * y$	$\leftarrow_{\mathtt{P}}(x,y)$	$\triangleq$	$\min(1, x/y)$	*Product*
$\&_{\mathtt{G}}(x,y)$	$\triangleq$	$\min(x,y)$	$\leftarrow_{\mathtt{G}}(x,y)$	$\triangleq$	$\begin{cases} 1 & \text{if } y \leq x \\ x & \text{otherwise} \end{cases}$	*Gödel*
$\&_{\mathtt{L}}(x,y)$	$\triangleq$	$\max(0, x+y-1)$	$\leftarrow_{\mathtt{L}}(x,y)$	$\triangleq$	$\min\{x-y+1, 1\}$	*Łukasiewicz*

Figure 1: Adjoint pairs in $[0,1]$ for *Łukasiewicz*, *Gödel* and *product* fuzzy logics

2 MALP Syntax and Operational/Declarative Semantics

In essence, the notion of multi-adjoint lattice considers a *carrier* set L (whose elements verify a concrete ordering $\leq$) equipped with a set of connectives like implications, conjunctions, disjunctions and other *hybrid operators* (not always belonging to an standard taxonomy) with the particularity that for each implication symbol there exists its adjoint conjunction used for modeling the *modus ponens* inference rule in a fuzzy logic setting. For instance, some adjoint pairs -i.e. conjunctors and implications- in the lattice $([0,1], \leq)$ are presented in Figure 1, where labels `L`, `G` and `P` mean respectively *Łukasiewicz logic*, *Gödel logic* and *product logic* (with different capabilities for modeling *pessimist*, *optimist* and *realistic scenarios*, respectively).

Definition 2.1. *Let $(L, \leq)$ be a lattice. A* multi-adjoint lattice *is a tuple $(L, \leq, \leftarrow_1, \&_1, \ldots, \leftarrow_n, \&_n)$ such that:*

i) $(L, \leq)$ *is a complete lattice, namely,* $\forall S \subset L, S \neq \emptyset, \exists\ inf(S), sup(S)$[1].

ii) $(\&_i, \leftarrow_i)$ *is an* adjoint pair *in* $(L, \leq)$*, i.e.:*

1) $\&_i$ *is non-decreasing in both arguments, for all* i, $i = 1, \ldots, n$.

2) $\leftarrow_i$ *is non-decreasing in the first argument and non-increasing in the second one.*

3) $x \leq (y \leftarrow_i z)$ *if and only if* $(x \&_i z) \leq y$*, for any* $x, y, z \in L$ *(adjoint property).*

iii) $\top \&_i v = v \&_i \top = v$ *for all* $v \in L, i = 1, \ldots, n$*, where* $\top = sup(L)$.

This last condition, called *adjoint property*, is the most important feature of the framework.

In what follows, we present a short summary of the main features of our language (we refer the reader to [6, 4, 5] for a complete formulation). We work with a first order language, $\mathcal{L}$, containing variables, function symbols, predicate symbols, constants, quantifiers ($\forall$ and $\exists$), and several (arbitrary) connectives to increase language expressiveness. In our fuzzy setting, we use implication connectives $(\leftarrow_1, \leftarrow_2, \ldots, \leftarrow_m)$ and also other connectives

[1] Then, it is a bounded lattice, i.e. it has bottom and top elements, denoted by $\bot$ and $\top$, respectively.

 ISBN: 978-84-616-2723-3

which are grouped under the name of *aggregators* or *aggregation operators*. They are used to combine/propagate truth values through the rules. The general definition of aggregation operators subsumes conjunctive operators (denoted by $\&_1, \&_2, \ldots, \&_k$), disjunctive operators ($\vee_1, \vee_2, \ldots, \vee_l$), and average and hybrid operators (usually denoted by $@_1, @_2, \ldots, @_n$). Although the connectives $\&_i$, $\vee_i$ and $@_i$ are binary operators, we usually generalize them as functions with an arbitrary number of arguments. By definition, the truth function for an n-ary aggregation operator $[\![@]\!] : L^n \to L$ is required to be monotone and fulfills $[\![@]\!](\top, \ldots, \top) = \top$, $[\![@]\!](\bot, \ldots, \bot) = \bot$. Additionally, our language $\mathcal{L}$ contains the elements of a multi-adjoint lattice, $(L, \preceq, \leftarrow_1, \&_1, \ldots, \leftarrow_n, \&_n)$, equipped with a collection of adjoint pairs $(\leftarrow_i, \&_i)$, where each $\&_i$ is a conjunctor intended to the evaluation of *modus ponens*. In general, the set of truth values L may be the carrier of any complete lattice.

A *rule* is a formula $H \leftarrow_i \mathcal{B}$, where H is an atomic formula (usually called the *head*) and $\mathcal{B}$ (which is called the *body*) is a formula built from atomic formulas $B_1, \ldots, B_n$ ($n \geq 0$), truth values of L and conjunctions, disjunctions and aggregations. Rules with an empty body are called *facts*. A *goal* is a body submitted as a query to the system. Variables in a rule are assumed to be governed by universal quantifiers. Roughly speaking, a MALP program is a set of pairs $\langle \mathcal{R}; v \rangle$, where $\mathcal{R}$ is a rule and v is a *truth degree* (a value of L) expressing the confidence which the user of the system has in the truth of the rule $\mathcal{R}$.

2.1 MALP Operational Semantics

In order to describe the procedural semantics of the MALP language, in the following we denote by $\mathcal{C}[A]$ a formula where A is a sub-expression (usually an atom) which occurs in the –possibly empty– context $\mathcal{C}[]$ whereas $\mathcal{C}[A/A']$ means the replacement of A by A' in context $\mathcal{C}[]$. Moreover, $\mathcal{V}ar(s)$ denotes the set of distinct variables occurring in the syntactic object s, $\theta[\mathcal{V}ar(s)]$ refers to the substitution obtained from θ by restricting its domain to $\mathcal{V}ar(s)$ and $mgu(E)$ denotes the *most general unifier* of a set of expressions E. In the next definition, we always consider that A is the selected atom in goal $\mathcal{Q}$ and L is the multi-adjoint lattice associated to $\mathcal{P}$.

Definition 2.2 (Admissible Step)**.** *Let $\mathcal{Q}$ be a goal and let σ be a substitution. The pair $\langle \mathcal{Q}; \sigma \rangle$ is a* state*. Given a program $\mathcal{P}$, an* admissible computation *is formalized as a state transition system, whose transition relation $\overset{AS}{\rightsquigarrow}$ is the smallest relation satisfying the following* admissible rules*:*

1) $\langle \mathcal{Q}[A]; \sigma \rangle \overset{AS}{\rightsquigarrow} \langle (\mathcal{Q}[A/v\&_i\mathcal{B}])\theta; \sigma\theta \rangle$ *if $\theta = mgu(\{H = A\})$, $\langle H \leftarrow_i \mathcal{B}; v \rangle$ in $\mathcal{P}$ and $\mathcal{B}$ is not empty.*

2) $\langle \mathcal{Q}[A]; \sigma \rangle \overset{AS}{\rightsquigarrow} \langle (\mathcal{Q}[A/v])\theta; \sigma\theta \rangle$ *if $\theta = mgu(\{H = A\})$, and $\langle H \leftarrow_i; v \rangle$ in $\mathcal{P}$.*

3) $\langle \mathcal{Q}[A]; \sigma \rangle \overset{AS}{\rightsquigarrow} \langle (\mathcal{Q}[A/\bot]); \sigma \rangle$ *if there is no rule in $\mathcal{P}$ whose head unifies with A (this case copes with possible unsuccessful branches).*

 ISBN: 978-84-616-2723-3

An *admissible derivation* is a sequence $\langle \mathcal{Q}; id\rangle \overset{\text{AS}}{\rightsquigarrow} {}^{*}\langle \mathcal{Q}'; \theta\rangle$. As usual, rules are taken renamed apart. We shall use the symbols $\overset{\text{AS1}}{\rightsquigarrow}$, $\overset{\text{AS2}}{\rightsquigarrow}$ and $\overset{\text{AS3}}{\rightsquigarrow}$ to distinguish between computation steps performed by applying one of the specific admissible rules. The application of a rule on a step will be annotated as a superscript of the $\overset{\text{AS}}{\rightsquigarrow}$ symbol.

If we exploit all atoms of a given goal, by applying admissible steps as much as needed during the operational phase, then it becomes a formula with no atoms (a L-expression) which can be then interpreted w.r.t. lattice L as follows.

Definition 2.3 (Interpretive Step and Fuzzy Computed Answer)**.** *Let $\mathcal{P}$ be a program, $\mathcal{Q}$ a goal and σ a substitution. Assume that $[\![@]\!]$ is the truth function of connective @ in the lattice $(L, \leq)$ associated to $\mathcal{P}$, such that, for values $r_1, \ldots, r_n, r_{n+1} \in L$, we have that $[\![@]\!](r_1, \ldots, r_n) = r_{n+1}$. Then, we formalize the notion of* interpretive computation *as a state transition system, whose transition relation $\overset{IS}{\rightsquigarrow}$ is defined as the least one satisfying:*

$$\langle \mathcal{Q}[@(r_1, \ldots, r_n)]; \sigma\rangle \quad \overset{IS}{\rightsquigarrow} \quad \langle \mathcal{Q}[@(r_1, \ldots, r_n)/r_{n+1}]; \sigma\rangle$$

An interpretive derivation *is a sequence $\langle \mathcal{Q}; \sigma\rangle \overset{IS}{\rightsquigarrow} \cdots \overset{IS}{\rightsquigarrow} \langle \mathcal{Q}'; \sigma\rangle$. When $\mathcal{Q}' = r \in L$, the state $\langle r; \sigma\rangle$ is called a* fuzzy computed answer *(f.c.a.) for that derivation.*

Moreover, in the MALP framework [6, 4, 5], each program has its own associated multi-adjoint lattice and each program rule is "weighted" with an element of L, whereas the components in its body (i.e., atoms and elements of L) are *linked* with connectives of the lattice.

2.2 MALP Declarative Semantics

We formally introduce now the semantic notions of Herbrand interpretation and Herbrand model, or directly, interpretation and model for short, for a MALP program $\mathcal{P}$, in a similar way to [6] and [2].

Definition 2.4 (Herbrand Interpretation)**.** *A Herbrand interpretation is a map $\mathcal{I} : B_{\mathcal{P}} \to L$, where $B_{\mathcal{P}}$ is the Herbrand base of the* MALP *program $\mathcal{P}$ and $(L, \leq)$ is the multi-adjoint lattice associated to program $\mathcal{P}$.*

$\mathcal{I}$ is extended in a natural way to the set of ground formulae of the language. In order to interpret a non ground formula A (closed, and universally quantified in the case of the MALP language), it suffices to take $\mathcal{I}(A) = inf\{\mathcal{I}(A\xi) : A\xi \text{ is a ground instance of A}\}$. Let $\mathcal{H}$ be the set of interpretations whose order is induced from the order of L:

$$\mathcal{I}_j \leq \mathcal{I}_k \Longleftrightarrow \mathcal{I}_j(F) \leq \mathcal{I}_k(F), \forall F \in B_{\mathcal{P}}$$

It is trivial to check that $(\mathcal{H}, \leq)$ inherits the structure of complete lattice from the multi-adjoint lattice $(L, \leq)$.

 ISBN: 978-84-616-2723-3

Definition 2.5 (Herbrand Model)**.** *An interpretation $\mathcal{I}$ satisfies (or is* model *of) a rule $\langle H \leftarrow_i \mathcal{B}; \alpha_i \rangle$ if, and only if, $v \leq \mathcal{I}(H \leftarrow_i \mathcal{B})$. An interpretation $\mathcal{I}$ is a Herbrand model of $\mathcal{P}$ if, and only if, all rules in $\mathcal{P}$ are satisfied by $\mathcal{I}$.*

In [2] we have defined, for the first time in the literature using model theory, a declarative semantics for multi-adjoint logic programming in terms of the least fuzzy Herbrand model. This construction reproduces, in our fuzzy context, the classic construction of least Herbrand model of pure logic programming [3, 1], which has been traditionally accepted as the declarative semantics of logic programs.

In the last years, other adaptations of this concept have been provided, using model theory too ([9, 7, 8]), in alternative fuzzy logic programming frameworks different from MALP.

Furthermore, in [2] we have related our notion of least fuzzy model with the already existing procedural semantics and fix-point semantics, and we have given revealing examples in which our declarative semantics has still sense beyond the multi-adjoint case, while the previously mentioned ones remain undefined.

Definition 2.6 (Least Fuzzy Herbrand Model)**.** *Let $\mathcal{P}$ be a* MALP *program with associated lattice $(L, \leq)$. The interpretation $\mathcal{I}_{\mathcal{P}} = inf\{\mathcal{I}_j : \mathcal{I}_j \text{ is model of } \mathcal{P}\}$ is called least fuzzy Herbrand*[2] *model of $\mathcal{P}$.*

The following result justifies that the previous interpretation $\mathcal{I}_{\mathcal{P}}$ can be thought really as the least fuzzy Herbrand model.

Theorem 2.7 ([2])**.** *Let $\mathcal{P}$ be a* MALP *program with associated lattice L. The map $\mathcal{I}_{\mathcal{P}} = inf\{\mathcal{I}_j : \mathcal{I}_j \text{ is a model of } \mathcal{P}\}$ is the least model of $\mathcal{P}$.*

In the proof of this result provided in [2], it is essential that the lattice associated to the program be a multi-adjoint lattice. In this reference it is possible to contrast the necessity of this hypothesis for Theorem 2.7.

In Figure 2 we illustrate most definitions presented in this section, where we wish to remark that:

- In the first rule of $\mathcal{P}$, we mix connectives belonging to three different fuzzy logics, whose truth functions appear in Figure 1, having too that $|_{\mathtt{L}}(x, y) \triangleq min(x + y, 1)$.
- Note that since there are no rules defining predicate s, the last step in the admissible derivation reduces $s(Y_2)$ to 0 by applying an $\overset{\text{AS3}}{\rightsquigarrow}$ step, which contrasts with crisp logic languages such as Prolog which would abort the whole derivation. Hence, in our fuzzy setting we can reach computed answers (at the end of the interpretive phase) even in the presence of non defined predicates.

[2] Sometimes we will say only least fuzzy model or least model.

 ISBN: 978-84-616-2723-3

Multi-adjoint logic program:

$$\mathcal{P} = \left\{ \begin{array}{lllll} \mathcal{R}_1: & \langle p(X) & \leftarrow_{\mathtt{P}} & q(X,Y) \;\&_{\mathtt{G}}\; (r(Y) \;|_{\mathtt{L}}\; s(Y)) & ; \;\; 0.8\rangle \\ \mathcal{R}_2: & \langle q(a,Y) & \leftarrow & & ; \;\; 0.9\rangle \\ \mathcal{R}_3: & \langle r(b) & \leftarrow & & ; \;\; 1\rangle \end{array} \right.$$

Admissible derivation:

$$\begin{array}{ll} \langle \underline{p(X)}; id\rangle & \overset{\text{AS1}}{\rightsquigarrow}\ \mathcal{R}_1 \\ \langle 0.8 \;\&_{\mathtt{P}}\; (\underline{q(X_1,Y_1)} \;\&_{\mathtt{G}}\; (r(Y1) \;|_{\mathtt{L}}\; s(Y1)));\{X/X_1\}\rangle & \overset{\text{AS2}}{\rightsquigarrow}\ \mathcal{R}_2 \\ \langle 0.8 \;\&_{\mathtt{P}}\; (0.9 \;\&_{\mathtt{G}}\; (\underline{r(Y2)} \;|_{\mathtt{L}}\; s(Y2)));\{X/a, X_1/a, Y_1/Y_2\}\rangle & \overset{\text{AS2}}{\rightsquigarrow}\ \mathcal{R}_3 \\ \langle 0.8 \;\&_{\mathtt{P}}\; (0.9 \;\&_{\mathtt{G}}\; (1 \;|_{\mathtt{L}}\; \underline{s(Y2)}));\{X/a, X_1/a, Y_1/b, Y_2/b\}\rangle & \overset{\text{AS3}}{\rightsquigarrow} \\ \langle 0.8 \;\&_{\mathtt{P}}\; (0.9 \;\&_{\mathtt{G}}\; (1 \;|_{\mathtt{L}}\; \overline{0}));\{X/a, X_1/a, Y_1/b, Y_2/b\}\rangle & \end{array}$$

Interpretive derivation:

$$\begin{array}{ll} \langle 0.8 \;\&_{\mathtt{P}}\; (0.9 \;\&_{\mathtt{G}}\; \underline{(1 \;|_{\mathtt{L}}\; 0)});\{X/a\}\rangle & \overset{\text{IS}}{\rightsquigarrow} \\ \langle 0.8 \;\&_{\mathtt{P}}\; \underline{(0.9 \;\&_{\mathtt{G}}\; 1)};\{X/a\}\rangle & \overset{\text{IS}}{\rightsquigarrow} \\ \langle \underline{0.8 \;\&_{\mathtt{P}}\; 0.9};\{X/a\}\rangle & \overset{\text{IS}}{\rightsquigarrow} \\ \langle 0.72;\{X/a\}\rangle & \end{array}$$

— f.c.a. means "*p(X)* is proved with truth degree 0.72 when $X = a$".

Least fuzzy Herbrand model:

$$\mathcal{I}_{\mathcal{P}}(p(a)) = 0.72,\ \mathcal{I}_{\mathcal{P}}(q(a,a)) = \mathcal{I}_{\mathcal{P}}(q(a,b)) = 0.9,\ \mathcal{I}_{\mathcal{P}}(r(b)) = 1.$$

Figure 2: Illustrative examples of MALP syntax and semantics

- When describing the least fuzzy Herbrand model of $\mathcal{P}$, we omit those elements of the Herbrand Base interpreted as 0.

3 A MALP Sub-class non Depending on Adjoint Pairs

From now on, we call $\mathcal{M}_\top$ to the set of MALP programs whose rules are always labeled with the top element $\top$ of their associated lattices. We now speak about $\top$-programs, $\top$-rules and so on. For executing these programs, we can conceive the following operational semantics which is simpler than the one seen in the previous section (see Definition 2.2).

 ISBN: 978-84-616-2723-3

Definition 3.1 ($\top$-Admissible Step)**.** *Let $\mathcal{Q}$ be a goal and let σ be a substitution. The pair $\langle \mathcal{Q}; \sigma \rangle$ is a* state*. Given a $\top$-program $\mathcal{P} \in \mathcal{M}_\top$, a* $\top$-admissible computation *is formalized as a state transition system, whose transition relation $\overset{AS^\top}{\rightsquigarrow}$ is the smallest relation satisfying the following* $\top$-admissible rules*:*

1) $\langle \mathcal{Q}[A]; \sigma \rangle \overset{AS^\top}{\rightsquigarrow} \langle (\mathcal{Q}[A/\mathcal{B}])\theta; \sigma\theta \rangle$ *if $\theta = mgu(\{H = A\})$, $\langle H \leftarrow_i \mathcal{B}; \top \rangle$ in $\mathcal{P}$ and $\mathcal{B}$ is not empty.*

2) $\langle \mathcal{Q}[A]; \sigma \rangle \overset{AS^\top}{\rightsquigarrow} \langle (\mathcal{Q}[A/\top])\theta; \sigma\theta \rangle$ *if $\theta = mgu(\{H = A\})$, and $\langle H \leftarrow_i; \top \rangle$ in $\mathcal{P}$.*

3) $\langle \mathcal{Q}[A]; \sigma \rangle \overset{AS^\top}{\rightsquigarrow} \langle (\mathcal{Q}[A/\bot]); \sigma \rangle$ *if there is no $\top$-rule in $\mathcal{P}$ whose head unifies with A (this case copes with possible unsuccessful branches).*

A $\top$-*admissible derivation* is a sequence $\langle \mathcal{Q}; id \rangle \overset{\mathrm{AS}^\top}{\rightsquigarrow} {}^*\langle \mathcal{Q}'; \theta \rangle$. We shall use the symbols $\overset{\mathrm{AS1}^\top}{\rightsquigarrow}$, $\overset{\mathrm{AS2}^\top}{\rightsquigarrow}$ and $\overset{\mathrm{AS3}^\top}{\rightsquigarrow}$ to distinguish between computation steps performed by applying one of the specific admissible rules. Note that this definition (which is very close to the classical one of SLD-resolution used in Prolog) differs for Definition 2.2 just in the first case, since it does not make use on states of the $\&_i$ conjunction adjoint to the $\leftarrow_i$ implication symbol of $\top$-rules.

The following result establishes that, when dealing with $\top$-programs, the derivations built with admissible (together with subsequent interpretive) steps as well as those ones based on $\top$-admissible (and interpretive) steps, lead to the same set of fuzzy computed answers.

Theorem 3.2. *Let $\mathcal{P} \in \mathcal{M}_\top$ be a* MALP *$\top$-program with associated lattice L, $\mathcal{Q}$ a goal, σ a substitution and $v \in L$. Then,*

$$\langle \mathcal{Q}; id \rangle \overset{AS}{\rightsquigarrow} {}^* \cdots \overset{IS}{\rightsquigarrow} {}^*\langle v; \sigma \rangle \text{ w.r.t. } \mathcal{P} \quad \text{if and only if} \quad \langle \mathcal{Q}; id \rangle \overset{AS^\top}{\rightsquigarrow} {}^* \cdots \overset{IS}{\rightsquigarrow} {}^*\langle v; \sigma \rangle \text{ w.r.t. } \mathcal{P}.$$

Proof. In our our proof we simply need to show that the effects produced by $\overset{\mathrm{AS}}{\rightsquigarrow}$ steps on a generic state of the form $\langle \mathcal{Q}[A]; \sigma \rangle$, are replicated by $\overset{\mathrm{AS}^\top}{\rightsquigarrow}$ steps and viceversa. We consider three different cases:

1) If $\langle H \leftarrow_i \mathcal{B}; \top \rangle \in \mathcal{P}$, where $\mathcal{B}$ is not empty and $\theta = mgu(\{H = A\})$, then $\langle \mathcal{Q}[A]; \sigma \rangle \overset{\mathrm{AS1}}{\rightsquigarrow} \langle (\mathcal{Q}[A/\top \&_i \mathcal{B}])\theta; \sigma\theta \rangle$ if and only if $\langle \mathcal{Q}[A]; \sigma \rangle \overset{\mathrm{AS1}^\top}{\rightsquigarrow} \langle (\mathcal{Q}[A/\mathcal{B}])\theta; \sigma\theta \rangle$, since $\top \&_i \mathcal{B} \equiv \mathcal{B}$ and hence $(\mathcal{Q}[A/\top \&_i \mathcal{B}])\theta \equiv (\mathcal{Q}[A/\mathcal{B}])\theta$.

2) If $\langle H \leftarrow_i; \top \rangle \in \mathcal{P}$, where $\theta = mgu(\{H = A\})$, then $\langle \mathcal{Q}[A]; \sigma \rangle \overset{\mathrm{AS2}}{\rightsquigarrow} \langle (\mathcal{Q}[A/\top])\theta; \sigma\theta \rangle$ if and only if $\langle \mathcal{Q}[A]; \sigma \rangle \overset{\mathrm{AS2}^\top}{\rightsquigarrow} \langle (\mathcal{Q}[A/\top])\theta; \sigma\theta \rangle$.

 ISBN: 978-84-616-2723-3

3) if there is no $\top$-rule in $\mathcal{P}$ whose head unifies with A then, $\langle \mathcal{Q}[A]; \sigma\rangle \overset{\text{AS3}}{\rightsquigarrow} \langle (\mathcal{Q}[A/\bot]); \sigma\rangle$ if and only if $\langle \mathcal{Q}[A]; \sigma\rangle \overset{\text{AS3}^{\top}}{\rightsquigarrow} \langle (\mathcal{Q}[A/\bot]); \sigma\rangle$.

□

Let us continue now with declarative semantics aspects related with $\top$-programs.

Definition 3.3 ($\top$-model)**.** *An interpretation $\mathcal{I}$ $\top$-satisfies (or is* $\top$-model *of) a $\top$-rule $\langle H \leftarrow_i \mathcal{B}; \top\rangle$ if, and only if, $\mathcal{I}(\mathcal{B}) \leq \mathcal{I}(H)$. An interpretation $\mathcal{I}$ is $\top$-model of a $\top$-program $\mathcal{P}$ if, and only if, all $\top$-rules in $\mathcal{P}$ are $\top$-satisfied by $\mathcal{I}$.*

Definition 3.4 (Least Fuzzy Herbrand $\top$-Model)**.** *Let $\mathcal{P}$ be a* MALP $\top$*-program with associated lattice $(L, \leq)$. The interpretation $\mathcal{I}_{\mathcal{P}}^{\top} = inf\{\mathcal{I}_j : \mathcal{I}_j$ is $\top$-model of $\mathcal{P}\}$ is the least fuzzy Herbrand $\top$-model of $\mathcal{P}$.*

In the following result we prove that the notion of least fuzzy Herbrand model of a given MALP $\top$-program is just the same construct than its least fuzzy Herbrand $\top$-model.

Theorem 3.5. *The least fuzzy Herbrand model of a* MALP $\top$*-program $\mathcal{P}$ coincides with its least fuzzy Herbrand $\top$-model, that is, $\mathcal{I}_{\mathcal{P}} = \mathcal{I}_{\mathcal{P}}^{\top}$.*

Proof. TO DO... □

It is noteworthy that, in the proof of this result, we don't require that lattice associated to $\top$-programs be a multi-adjoint lattice (in fact, we never use adjoint pairs), as occurred too when defining the new operational semantics for $\top$-programs. For this reason, from now on we can simplify the syntax of $\top$-rules, by removing the label of their implication symbols as well as their weights (or associated truth degrees), i.e., instead of $\langle H \leftarrow_i \mathcal{B}; \top\rangle$ we will simply write $H \leftarrow \mathcal{B}$.

4 A Mapping from MALP Programs to $\mathcal{M}_{\top}$

The following definition represents a very simple, purely syntactic pre-process which, by making use of adjoint pairs, is able to link MALP programs with $\top$-programs.

Definition 4.1. *We define a mapping that associates to each* MALP *program $\mathcal{P}$ a $\top$-program in $\mathcal{M}_{\top}$ with the following shape:*

$$\mathcal{P}^{\mathcal{M}_{\top}} = \{\mathcal{R}^{\mathcal{M}_{\top}} \ : \ \mathcal{R} \in \mathcal{P}\}$$

where for each $\top$-rule $\mathcal{R} : \langle H \leftarrow_i \mathcal{B}; v\rangle \in \mathcal{P}$, the mapping is defined too as:

$$\mathcal{R}^{\mathcal{M}_{\top}} \quad = \quad \begin{cases} H \leftarrow v \&_i \mathcal{B} & \text{if} \quad v \neq \top \\ H \leftarrow \mathcal{B} & \text{otherwise} \end{cases}$$

 ISBN: 978-84-616-2723-3

Multi-adjoint logic $\top$-program:

$$\mathcal{P}' = \mathcal{P}^{\mathcal{M}_\top} = \left\{ \begin{array}{lll} \mathcal{R}'_1: & p(X) & \leftarrow \quad 0.8 \;\&_{\mathtt{P}}\; (q(X,Y) \;\&_{\mathtt{G}}\; (r(Y) \;|_{\mathtt{L}}\; s(Y)) \\ \mathcal{R}'_2: & q(a,Y) & \leftarrow \quad 0.9 \\ \mathcal{R}'_3: & r(b) & \leftarrow \end{array} \right.$$

$\top$-Admissible derivation:

$$\begin{array}{ll} \langle \underline{p(X)}; id \rangle & \overset{\text{AS1}^\top}{\rightsquigarrow} \mathcal{R}'_1 \\ \langle 0.8 \;\&_{\mathtt{P}}\; (\underline{q(X_1,Y_1)} \;\&_{\mathtt{G}}\; (r(Y1) \;|_{\mathtt{L}}\; s(Y1))); \{X/X_1\} \rangle & \overset{\text{AS1}^\top}{\rightsquigarrow} \mathcal{R}'_2 \\ \langle 0.8 \;\&_{\mathtt{P}}\; (0.9 \;\&_{\mathtt{G}}\; (\underline{r(Y2)} \;|_{\mathtt{L}}\; s(Y2))); \{X/a, X_1/a, Y_1/Y_2\} \rangle & \overset{\text{AS2}^\top}{\rightsquigarrow} \mathcal{R}'_3 \\ \langle 0.8 \;\&_{\mathtt{P}}\; (0.9 \;\&_{\mathtt{G}}\; (1 \;|_{\mathtt{L}}\; \underline{s(Y2)})); \{X/a, X_1/a, Y_1/b, Y_2/b\} \rangle & \overset{\text{AS3}^\top}{\rightsquigarrow} \\ \langle 0.8 \;\&_{\mathtt{P}}\; (0.9 \;\&_{\mathtt{G}}\; (1 \;|_{\mathtt{L}}\; 0)); \{X/a, X_1/a, Y_1/b, Y_2/b\} \rangle & \end{array}$$

Figure 3: Illustrative examples of concepts defined in Sections 3 and 4

In Figure 3 we illustrate this definition as well as other concepts introduced in the previous section. Note that:

- The $\top$-program $\mathcal{P}'$ coincides with the transformation of the MALP program $\mathcal{P}$ seen in Figure 2, that is $\mathcal{P}' = \mathcal{P}^{\mathcal{M}_\top}$, since $\mathcal{R}'_1 = \mathcal{R}_1^{\mathcal{M}_\top}$, $\mathcal{R}'_2 = \mathcal{R}_2^{\mathcal{M}_\top}$ and $\mathcal{R}'_3 = \mathcal{R}_3^{\mathcal{M}_\top}$. In this last case, we have simply removed the weight of the rule (since it is just 1, i.e., the top element of lattice $[0,1]$) and both $\mathcal{R}_1$ and $\mathcal{R}'_1$ are facts in $\mathcal{P}$ and $\mathcal{P}'$, respectively.
- On the other hand, note that even when $\mathcal{R}_2 \in \mathcal{P}$ is a fact, $\mathcal{R}'_2 \in \mathcal{P}'$ is not a fact, since its body is not empty (it is composed by just a truth degree). For this reason, while the second admissible step of the admissible derivation in Figure 2 is of kind $\overset{\text{AS2}}{\rightsquigarrow}$, the corresponding second $\top$-admissible step of the $\top$-admissible derivation in Figure 3 is not a $\overset{\text{AS2}^\top}{\rightsquigarrow}$ but a $\overset{\text{AS1}^\top}{\rightsquigarrow}$ step.
- The sequence of states in the admissible derivation of Figure 2 coincides with the sequence of states in the $\top$-admissible of Figure 3, and after applying exactly the same sequence of interpretive steps drawn in Figure 2 (for this reason we have omitted it in Figure 3), the same fuzzy computed answer is reached.
- Note that even when the notion of $\top$-model[3] is less involved than the one of model, the least fuzzy Herbrand $\top$-model of $\mathcal{P}'$ coincides with $\mathcal{I}_\mathcal{P}$ in Figure 2, as wanted.

[3]This concept does not make use of adjoint pairs and weights of program rules.

 ISBN: 978-84-616-2723-3

The following result establishes that derivations built with admissible (together with subsequent interpretive) steps lead to the same set of fuzzy computed answers than those ones based on $\top$-admissible (and interpretive) steps when dealing with $\top$-programs obtained from previous MALP programs after being transformed according Definition 4.1.

Theorem 4.2. *Let $\mathcal{P}$ be a* MALP *program with associated lattice L, $\mathcal{Q}$ a goal, σ a substitution and $v \in L$. Then,*

$\langle \mathcal{Q}; id\rangle \overset{AS}{\rightsquigarrow} * \cdots \overset{IS}{\rightsquigarrow} * \langle v; \sigma\rangle$ *w.r.t.* $\mathcal{P}$ *if and only if* $\langle \mathcal{Q}; id\rangle \overset{AS^{\top}}{\rightsquigarrow} * \cdots \overset{IS}{\rightsquigarrow} * \langle v; \sigma\rangle$ *w.r.t.* $\mathcal{P}^{\mathcal{M}_{\top}}$.

Proof. We distinguish four cases for showing that the effects produced by $\overset{\text{AS}}{\rightsquigarrow}$ steps on a generic state of the form $\langle \mathcal{Q}[A]; \sigma\rangle$, are replicated by $\overset{\text{AS}^{\top}}{\rightsquigarrow}$ steps and viceversa.

1) Note that $\langle H \leftarrow_i \mathcal{B}; v\rangle \in \mathcal{P}$, where $\mathcal{B}$ is not empty and $\theta = mgu(\{H = A\})$, if and only if $\langle H \leftarrow v \&_i \mathcal{B}\rangle \in \mathcal{P}^{\mathcal{M}_{\top}}$, and hence $\langle \mathcal{Q}[A]; \sigma\rangle \overset{\text{AS1}}{\rightsquigarrow} \langle (\mathcal{Q}[A/v\&_i\mathcal{B}])\theta; \sigma\theta\rangle$ if and only if $\langle \mathcal{Q}[A]; \sigma\rangle \overset{\text{AS1}^{\top}}{\rightsquigarrow} \langle (\mathcal{Q}[A/v\&_i\mathcal{B}])\theta; \sigma\theta\rangle$.

2) Now, $\langle H \leftarrow; v\rangle \in \mathcal{P}$, where $v \neq \top$ and $\theta = mgu(\{H = A\})$, if and only if $\langle H \leftarrow v\rangle \in \mathcal{P}^{\mathcal{M}_{\top}}$, and hence $\langle \mathcal{Q}[A]; \sigma\rangle \overset{\text{AS2}}{\rightsquigarrow} \langle (\mathcal{Q}[A/v])\theta; \sigma\theta\rangle$ if and only if $\langle \mathcal{Q}[A]; \sigma\rangle \overset{\text{AS1}^{\top}}{\rightsquigarrow} \langle (\mathcal{Q}[A/v])\theta; \sigma\theta\rangle$ (note this particular correspondence between $\overset{\text{AS2}}{\rightsquigarrow}$ and $\overset{\text{AS1}^{\top}}{\rightsquigarrow}$ steps).

3) Observe that $\langle H \leftarrow; \top\rangle \in \mathcal{P}$, where $\theta = mgu(\{H = A\})$, if and only if $\langle H \leftarrow\rangle \in \mathcal{P}^{\mathcal{M}_{\top}}$, and hence $\langle \mathcal{Q}[A]; \sigma\rangle \overset{\text{AS2}}{\rightsquigarrow} \langle (\mathcal{Q}[A/\top])\theta; \sigma\theta\rangle$ if and only if $\langle \mathcal{Q}[A]; \sigma\rangle \overset{\text{AS2}^{\top}}{\rightsquigarrow} \langle (\mathcal{Q}[A/\top])\theta; \sigma\theta\rangle$ (now we have shown the equivalence between $\overset{\text{AS2}}{\rightsquigarrow}$ and $\overset{\text{AS2}^{\top}}{\rightsquigarrow}$ steps).

4) Finally, there is no rule in $\mathcal{P}$ whose head unifies with A if and only if there is no rule in $\mathcal{P}^{\mathcal{M}_{\top}}$ whose head unifies with A and so, $\langle \mathcal{Q}[A]; \sigma\rangle \overset{\text{AS3}}{\rightsquigarrow} \langle (\mathcal{Q}[A/\bot]); \sigma\rangle$ if and only if $\langle \mathcal{Q}[A]; \sigma\rangle \overset{\text{AS3}^{\top}}{\rightsquigarrow} \langle (\mathcal{Q}[A/\bot]); \sigma\rangle$.

□

In the following result we prove that the notion of least fuzzy Herbrand model of MALP programs is just the same construct than the least fuzzy Herbrand $\top$-model of those $\top$-programs obtained by applying the transformation process described in Definition 4.1.

Theorem 4.3. *The least fuzzy Herbrand model of a* MALP *program $\mathcal{P}$ coincides with the least fuzzy Herbrand $\top$-model of its associated $\top$-program $\mathcal{P}^{\mathcal{M}_{\top}}$, that is, $\mathcal{I}_{\mathcal{P}} = \mathcal{I}^{\top}_{\mathcal{P}^{\mathcal{M}_{\top}}}$.*

Proof. TO DO... □

 ISBN: 978-84-616-2723-3

5 Conclusions and Future Work

TO DO...

References

[1] K. R. Apt. *From Logic Programming to Prolog.* International Series in Computer Science, Prentice Hall, 1997.

[2] P. Julián, G. Moreno, and J. Penabad. On the declarative semantics of multi-adjoint logic programs. In *Proc. of the 10th International Work-Conference on Artificial Neural Networks, IWANN'09*, pages 253–260. Springer, Lecture Notes in Computer Science 5517, 2009.

[3] J. W. Lloyd. *Foundations of Logic Programming.* Springer-Verlag, Berlin, 1987.

[4] J. Medina, M. Ojeda-Aciego, and P. Vojtáš. Multi-adjoint logic programming with continuous semantics. *Proc. of Logic Programming and Non-Monotonic Reasoning, LPNMR'01, Lecture Notes in Artificial Intelligence*, 2173:351–364, 2001.

[5] J. Medina, M. Ojeda-Aciego, and P. Vojtáš. A procedural semantics for multi-adjoint logic programing. *Progress in Artificial Intelligence, EPIA'01, Springer-Verlag, Lecture Notes in Artificial Intelligence*, 2258(1):290–297, 2001.

[6] J. Medina, M. Ojeda-Aciego, and P. Vojtáš. Similarity-based Unification: a multi-adjoint approach. *Fuzzy Sets and Systems*, 146:43–62, 2004.

[7] M. I. Sessa. Approximate reasoning by similarity-based SLD resolution. *Fuzzy Sets and Systems*, 275:389–426, 2002.

[8] U. Straccia, M. Ojeda-Aciego, and C. V. Damásio. On fixed-points of multivalued functions on complete lattices and their application to generalized logic programs. *SIAM Journal on Computing*, 38(5):1881–1911, 2009.

[9] P. Vojtáš and L. Paulík. Soundness and completeness of non-classical extended SLD-resolution. In R. Dyckhoff et al, editor, *Proc. of ELP'96 Leipzig*, pages 289–301. Lecture Notes in Computer Science 1050, Springer Verlag, 1996.

 ISBN: 978-84-616-2723-3