Appl. Comput. Math., V.23, N.4, 2024, pp.520-540
DOI: 10.30546/1683-6154.23.4.2024.520

COMPUTATIONAL TECHNIQUE FOR OBTAINING CONSEQUENCES IN MULTI-ADJOINT LOGIC PROGRAMMING WITH GENERALIZED QUANTIFIERS

J. MEDINA[1,*], J.A. TORNÉ-ZAMBRANO[1]

Abstract. This paper will introduce an operational semantics for the flexible framework of propositional multi-adjoint logic programming with generalized quantifiers. This approach extends diverse general logic programming frameworks, weakening the existential feature of the supremum operator and considering a malleable algebraic structure.

Keywords: Immediate Consequences Operator, Procedural Semantics, Fixed-Point Semantics, Fuzzy Sets, Generalized Quantifiers.

AMS Subject Classification: 03B52, 47J26, 68Q55, 68N17, 94D05.

1. Introduction

Artificial Intelligence is a fundamental discipline nowadays. Many tools in the fields of mathematics, computer science, and transversal ones, have been developed and applied to a huge range of real problems [4, 5, 32, 35]. However, a lot of challenges remain to be solved. A relevant research area which helps to achieve these challenges is mathematical logic and its variants considering fuzzy sets [13, 20, 37].

One of these variants is fuzzy logic programming [16, 18, 21, 23, 36], which is focused on modelling (real) problems through the use of logic rules. Fuzzy logic programming has been used in different areas, such as in supply-chain management [1], ontology [17], and environmental automatic control [1]. Specifically, it allows an executable language and so, it can be used to design algorithms and procedures for complementing the design of decision support systems [2,8,22,25].

In this research line, the definition of an operational semantics associated with a logic allows the implementation of an automatic reasoning procedure for obtaining information and deductions to be considered in the designed decision support system.

This paper will introduce an operational semantics for the adaptable framework of propositional multi-adjoint logic programming with generalized quantifiers [30, 31]. This approach extends diverse general logic programming frameworks [10, 21, 29, 36] weakening the existential feature of the supremum operator and considering a flexible algebraic structure [7]. These properties are relevant because the semantics of a logic program is characterized through the post-fixed points of the immediate consequences operator $T_{\mathbb{P}}$, which is defined from the supremum of a set of values. The strict nature of the supremum can cause that the presence of noise in the data alters the final result of the computation. For this reason, a more general definition of the immediate consequences operator $T_{\mathbb{P}}^{\mathrm{M}}$, defined on the multi-adjoint framework and based on generalized quantifiers, was proposed in [30]. The multi-adjoint approach presents several advantages that allow the representation of more complex systems than with the classical logic programming (monotone and residuated) [10] and the fuzzy approach given in [36], among others. The use of several adjoint pairs in the construction of logic programs is allowed, as

[1]Department of Mathematics, University of Cádiz, Cádiz, Spain, e-mail: joseantonio.torne@uca.es
[*]Corresponding author e-mail: jesus.medina@uca.es
Manuscript received 14 December 2023.

well as less restrictive properties in the considered connectives, such as the requirement of the commutativity and associativity properties. Therefore, this generalization provides a more flexible and versatile operator with interesting properties, which have been considered in different frameworks, such as in formal concept analysis [6].

Specifically, this paper will adapt the notions of computed and correct answers, and the procedure for obtaining computed answers to the flexible framework given by the multi-adjoint algebras and generalized quantifiers. In addition, a novel definition of reductant will be introduced in order to prove the soundness of the introduced procedure. Reductants were originally introduced in generalized annotated logic programming [19], and latter was considered in the multi-adjoint logic programming framework in [14, 26]. The notion of reductant will be fundamental in this paper to analyze the completeness, with the goal of obtaining an equivalence in terms of the corresponding maximum computed and correct answers.

The structure of the paper is as follows: in Section 2 some preliminaries results are recalled, such as the generalization of the immediate consequences operator based on generalized quantifiers. Section 3 introduces the operational semantics of the new flexible framework of this paper, and the necessity of the reductants. This last notion is adapted in Section 4, while the completeness results are given in Section 5. Finally, some conclusions and future work are considered in Section 6.

2. Preliminaries

First of all, some definitions and outcomes will be reviewed in this section to help understand the following sections. For more details see [24, 29, 31].

2.1. Generalized quantifiers.

The immediate consequences operator, which will be taken into consideration in this paper, was extended in [30, 31] using generalized quantifiers, in order to weaken the existential feature of the supremum. Therefore, some important issues from generalized quantifiers are recalled next. For more information see [3, 20, 34].

Definition 1. *[34] Let $U = \{u_1, \ldots, u_n\}$ be a finite universe, $\mathcal{P}(U)$ denote the powerset of U and $\mu\colon \mathcal{P}(U) \longrightarrow [0,1]$ be a fuzzy measure, i.e., a monotone mapping with $\mu(\varnothing) = 0$. We say that the fuzzy measure μ is invariant with respect to cardinality, if the following condition holds*

$$\text{If } |A| = |B|, \text{ then } \mu(A) = \mu(B), \text{ for all } A, B \in \mathcal{P}(U),$$

where $|\cdot|$ denotes the cardinality of a set.

Now, the definition of generalized quantifier is recalled.

Definition 2. *[34] Let U be a non-empty finite universe, $\mu\colon \mathcal{P}(U) \to [0,1]$ a fuzzy measure invariant with respect to the cardinality and $\otimes$ an operator which is increasing on the first argument and satisfies the following boundary conditions:*[1] *$x \otimes 1 = x$, $x \otimes 0 = 0 \otimes x = 0$, for all $x \in [0,1]$. A mapping $Q_\mu\colon \mathcal{F}(U) \to [0,1]$ defined, for all $C \in \mathcal{F}(U) = \{C \mid C\colon U \to [0,1]\}$, as*

$$Q_\mu(C) = \bigvee_{D \in \mathcal{P}(U)\setminus\{\varnothing\}} \left(\left(\bigwedge_{u \in D} C(u) \right) \otimes \mu(D) \right)$$

is called generalized quantifier determined by the fuzzy measure μ, where $\bigvee$ and $\bigwedge$ represent the supremum and infimum operators in the unit interval.

Next, two of the most significant examples of generalized quantifiers (universal and existential) are shown in the example below, with the corresponding fuzzy measures necessary in order to obtain them.

[1] These conditions are required to obtain the universal and existential quantifiers as particular cases of Q_μ.

Example 1. *The universal and existential quantifiers can be obtained from Definition 2 considering the following fuzzy measures $\mu^{\forall}$ and $\mu^{\exists}$, respectively, which are defined as follows:*

$$\mu^{\forall}(D) = \begin{cases} 1 & \text{if} \quad D = U, \\ 0 & \text{if} \quad D \neq U, \end{cases} \qquad \mu^{\exists}(D) = \begin{cases} 0 & \text{if} \quad D = \varnothing, \\ 1 & \text{if} \quad D \neq \varnothing. \end{cases}$$

Furthermore, it can be proved that the following equations are satisfied [6]:

$$Q^{\forall}(C) = \bigwedge_{u \in U} C(u) \qquad Q^{\exists}(C) = \bigvee_{u \in U} C(u).$$

It is important to note that $\mu^{\exists}$ is the maximum fuzzy measure, whereas $\mu^{\forall}$ is the minimum measure of the normalized ones[2]. Eventually, as the generalized quantifier provided by Definition 2 is difficult to compute, the following characterization of generalized quantifiers is recalled next.

Theorem 1. *[34] Let Q_μ be a generalized quantifier in the universe $U = \{u_1, \ldots, u_n\}$ determined by a fuzzy measure μ invariant with respect to the cardinality. Then, for every $C \in \mathcal{F}(U)$,*

$$Q_\mu(C) = \bigvee_{i=1}^{n} C\left(u_{\pi(i)}\right) \otimes \mu\left(\{u_1, \ldots, u_i\}\right),$$

where π is a permutation on $\{1, 2, \ldots, n-1, n\}$ such that $C\left(u_{\pi(n)}\right) \leq C\left(u_{\pi(n-1)}\right) \leq \ldots \leq C\left(u_{\pi(2)}\right) \leq C\left(u_{\pi(1)}\right)$.

2.2. **Multi-adjoint structure and syntax of multi-adjoint logic programs.** The adjoint pair is the basic operator taken into account in multi-adjoint logic programming [28, 29].

Definition 3. *Let $\langle P, \preceq \rangle$ be a poset. An adjoint pair in $\langle P, \preceq \rangle$ is a pair of binary operators $(\leftarrow, \&)$ defined on P satisfying that*

1. *The operator $\&$ is increasing in both arguments.*
2. *The operator $\leftarrow$ is increasing in the first argument (consequent) and decreasing in the second one (antecedent).*
3. *The adjoint property holds, that is, for every $x, y, z \in P$ the following equivalence holds:*
$$x \preceq (y \leftarrow z) \text{ if and only if } (x \& z) \preceq y.$$

The following definition sets the algebraic structure containing different adjoint pairs.

Definition 4. *Let $\langle L, \preceq \rangle$ be a lattice. A multi-adjoint lattice $\mathcal{L}$ is a tuple $(L, \preceq, \leftarrow_1, \&_1, \ldots, \leftarrow_n, \&_n)$ satisfying the following conditions:*

1. *$\langle L, \preceq \rangle$ is bounded, i.e., it has a bottom $\bot$ and a top $\top$ element.*
2. *$(\leftarrow_i, \&_i)$ is an adjoint pair in $\langle L, \preceq \rangle$, for all $i \in \{1, \ldots, n\}$.*
3. *$\top \&_i \vartheta = \vartheta \&_i \top = \vartheta$, for all $\vartheta \in L$ and $i \in \{1, \ldots, n\}$.*

Now, we recall the algebraic structure that will be used in this paper, allowing us to take into account additional operators together with those of the multi-adjoint lattice. For more information about the definitions of graded set, Ω-algebra and Ω-subalgebra see [24, 29].

Definition 5. *Let Ω be a graded set containing operators $\leftarrow_i$ and $\&_i$ for $i \in \{1, \ldots, n\}$ and possibly some extra operators, and let $\mathfrak{L} = \langle L, I \rangle$ be an Ω-algebra whose carrier set L is a complete lattice under $\preceq$. We say that $\mathfrak{L}$ is a multi-adjoint Ω-algebra with respect to the pairs $(\leftarrow_i, \&_i)$ for $i \in \{1, \ldots, n\}$ if $\mathcal{L} = (L, \preceq, I(\leftarrow_1), I(\&_1), \ldots, I(\leftarrow_n), I(\&_n))$ is a multi-adjoint lattice.*

[2]We say that a fuzzy measure μ is normalized if $\mu(U) = 1$.

Next, we introduce an example of a multi-adjoint Ω-algebra. From now on, the notation $\dot{\nu} = I(\nu)$ will be considered, where $\nu \in \Omega$.

Example 2. *$\langle [0,1], I\rangle$ is a multi-adjoint Ω-algebra with respect to the graded set $\Omega = \{\leftarrow_P, \&_P, \leftarrow_G, \&_G, \wedge_L, @_{2,1}\}$, where $\dot{\&}_P$ and $\dot{\&}_G$ are the product and Gdel t-norms respectively, together with their residuated implications, $\dot{\wedge}_L$ is the Łukasiewicz t-norm and $\dot{@}_{(1,2)}$ is an aggregation operator. These operators are defined for all x, y, $z \in [0,1]$ as follows*

$$\dot{\&}_G(x,y) = \min\{x,y\} \qquad z\dot{\leftarrow}_G y = \begin{cases} 1 & \text{if} \quad y \leq z \\ z & \text{if} \quad \text{otherwise} \end{cases}$$

$$\dot{\&}_P(x,y) = x \cdot y \qquad z\dot{\leftarrow}_P y = \begin{cases} 1 & \text{if} \quad y \leq z \\ \frac{z}{y} & \text{otherwise} \end{cases}$$

$$\dot{\wedge}_L(x,y) = \max\{0, x+y-1\} \qquad \dot{@}_{(1,2)}(x,y) = \frac{x+2y}{3}.$$

2.3. **Syntax and semantics of multi-adjoint logic programs.** The fundamental notion that allows us to formulate the syntax of multi-adjoint logic programming is that of a multi-adjoint logic program, being a set of rules and facts that enables different intelligent systems to be modeled. In order to define such program, a multi-adjoint Ω-algebra $\mathfrak{L}$ will be considered. The definitions of alphabet $A_{\Omega,\Pi}$, Ω-algebra of expressions, well-formed formulas $\mathfrak{F}$ and the homomorphic extension theorem can be found in [29].

Definition 6. *Let Π be a set of propositional symbols. A multi-adjoint logic program is a set $\mathbb{P}$ of weighted rules of the form $\langle (A \leftarrow_i \mathcal{B}), \vartheta\rangle$ such that:*

1. *The rule $(A \leftarrow_i \mathcal{B})$ is a formula of $\mathfrak{F}$.*
2. *The confidence factor ϑ is an element (a truth-value) of L.*
3. *The head of the rule A is a propositional symbol of Π.*
4. *The body formula $\mathcal{B}$ is a formula of $\mathfrak{F}$ built from propositional symbols $B_1, \ldots, B_n$ ($n \geq 0$) by the use of conjunctors $\&_1, \ldots, \&_n$ and $\wedge_1, \ldots, \wedge_k$, disjunctors $\vee_1, \ldots, \vee_l$ and aggregators $@_1, \ldots, @_m$.*

A fact is a weighted rule where the body $\mathcal{B}$ is $\top$. It must be taken into account that in the Ω-graded set, the element $\top$ is regarded as a constant. Furthermore, the rule $\langle A \leftarrow_i \mathcal{B}, \vartheta\rangle$ can also be represented as $\langle A \leftarrow @[B_1, \ldots, B_n], \vartheta\rangle$, where $@$ is the aggregator defined as a composition of all operators symbols in $\mathcal{B}$, with $B_1, \ldots, B_n$ being the propositional symbols appearing in the body $\mathcal{B}$, verifying that $\mathcal{B} = @[B_1, \ldots, B_n]$. Now, an example of a multi-adjoint logic program is shown.

Example 3. *In Example 2, identifying the operator symbols in Ω with their interpretations, we can consider the multi-adjoint Ω-algebra*

$$\left\langle [0,1], \leq, \leftarrow_G, \&_G, \leftarrow_P, \&_P, \wedge_L, @_{(1,2)} \right\rangle,$$

where $\&_G$ and $\&_P$ are the Gdel and the product conjunctions respectively, with $\leftarrow_G$ and $\leftarrow_P$ being their respective adjoint implications. Moreover, we use the extra operator $\wedge_L$, the Łukasiewicz conjunction and the aggregator $@_{(1,2)}$ given in Example 2. We define the set of propositional symbols Π as:

$$\Pi = \{\texttt{covid}, \texttt{diarrhoea}, \texttt{headache}, \texttt{moderate_fever}, \texttt{cough}, \texttt{waist_pain}\}.$$

We can consider, in an specific situation, the multi-adjoint logic program $\mathbb{P}$ defined through the rules

$$\begin{aligned} \texttt{covid} &\overset{0.8}{\leftarrow_G} @_{(1,2)}\left(\texttt{diarrhoea} \wedge_L \texttt{headache}, \texttt{waist_pain}\right) \\ \texttt{covid} &\overset{0.9}{\leftarrow_P} \texttt{moderate_fever} \\ \texttt{covid} &\overset{0.7}{\leftarrow_P} \texttt{cough}. \end{aligned}$$

These rules are completed with observations about each particular person, in the form of facts. For example, a particular individual may have the symptoms represented by the following facts.

$$\begin{aligned} \texttt{moderate_fever} &\overset{0.6}{\leftarrow} 1 \\ \texttt{cough} &\overset{0.8}{\leftarrow} 1 \\ \texttt{diarrhoea} &\overset{0.6}{\leftarrow} 1 \end{aligned} \qquad \begin{aligned} \texttt{headache} &\overset{0.7}{\leftarrow} 1 \\ \texttt{waist_pain} &\overset{0.3}{\leftarrow} 1. \end{aligned}$$

The concept of interpretation is fundamental in order to provide a meaning to syntactic formulae.

Definition 7. *An interpretation is a mapping $I\colon \Pi \to L$ which asigns values of the complete lattice L to the set of propositional symbols. The set of all interpretations of the formulas defined by the Ω-algebra $\mathfrak{F}$ in the Ω-algebra $\mathfrak{L}$ is denoted $\mathcal{I}_{\mathfrak{L}}$.*

Notice that each interpretation $I\colon \Pi \longrightarrow L$ can be extended to the set of formulas F_Ω, i.e. $\hat{I}\colon F_\Omega \longrightarrow L$, using the unique homomorphic extension theorem that was given in [29]. Furthermore, it is possible to extend the ordering $\preceq$ of $\langle L, \preceq\rangle$ to a point-wise ordering in $\mathcal{I}_{\mathfrak{L}}$. This ordering, together with $\mathcal{I}_{\mathfrak{L}}$, forms a complete lattice, denoted by $\langle \mathcal{I}_{\mathfrak{L}}, \sqsubseteq\rangle$ as in [29]. The least interpretation $\triangle$ maps every propositional symbol to the least element $\bot$ of L.

Next, the definition of model and satisfaction is recalled.

Definition 8. *Given an interpretation $I \in \mathcal{I}_{\mathfrak{L}}$, a weighted rule $\langle A \leftarrow_i \mathcal{B}, \vartheta\rangle$ is satisfied by I if and only if $\vartheta \preceq \hat{I}(A \leftarrow_i \mathcal{B})$. An interpretation $I \in \mathcal{I}_{\mathfrak{L}}$ is a model of a multi-adjoint logic program $\mathbb{P}$ if and only if all weighted rules in $\mathbb{P}$ are satisfied by I.*

The following definition recalls the generalization of the immediate consequences operator presented in [12] to the multi-adjoint logic programming framework [29].

Definition 9. *Let $\mathbb{P}$ be a multi-adjoint logic program. The immediate consequences operator $T^{\mathfrak{L}}_{\mathbb{P}}\colon \mathcal{I}_{\mathfrak{L}} \to \mathcal{I}_{\mathfrak{L}}$, mapping interpretations to interpretations, is defined by considering*

$$T^{\mathfrak{L}}_{\mathbb{P}}(I)(A) = \sup\left\{\vartheta \dot{\&}_i \hat{I}(\mathcal{B}) \mid A \overset{\vartheta}{\leftarrow_i} \mathcal{B} \in \mathbb{P}\right\}.$$

From now on, in order to simplify, we will refer to the immediate consequences operator in the multi-adjoint framework just as $T_{\mathbb{P}}$. The next theorem illustrates how models of a logic program are characterized as post-fixed points of the $T_{\mathbb{P}}$.

Theorem 2. *[29] An interpretation $I \in \mathcal{I}_{\mathfrak{L}}$ is a model of a multi-adjoint logic program $\mathbb{P}$ if and only if I is a post-fixed point of $T^M_{\mathbb{P}}$, that is, $T_{\mathbb{P}}(I) \sqsubseteq I$.*

Moreover, this operator is increasing, as the following result shows.

Theorem 3. *[29] The operator $T_{\mathbb{P}}$ is monotonic.*

Since $T_{\mathbb{P}}$ is monotonic, it has a least fixed point [24]. The following result shows that the least fixed point of $T_{\mathbb{P}}$ is equal to the least model of the program, which is denoted as $M_{\mathbb{P}}$. Furthermore, it is possible to compute it by iterating $T_{\mathbb{P}}$ an ordinal number of times, starting from the least interpretation $\triangle$.

Theorem 4. *[29] Let L be a complete lattice and $\mathbb{P}$ a multi-adjoint logic program. We have that $M_{\mathbb{P}} = \inf\{I \in \mathcal{I}_{\mathfrak{L}} \mid I \text{ is a model of } \mathbb{P}\}$ is a model and, for some ordinal α, it is satisfied that $M_{\mathbb{P}} = lfp\,(T_{\mathbb{P}}) = T^{\alpha}_{\mathbb{P}}(\triangle)$.*

2.4. **Definition of $T_{\mathbb{P}}$ through an arbitrary generalized quantifier.** First of all, we need the following technical definition, in order to present the adaptation of the immediate consequences operator using generalized quantifiers. For more details about this section, see [30, 31]. From now on, a non-empty multi-adjoint logic program $\mathbb{P}$, a set of propositional symbols Π, a multi-adjoint Ω-algebra $\mathfrak{L}$ and a family of fuzzy measures invariant with respect to the cardinality $\mathrm{M} = \{\mu_A \colon \mathcal{P}(\mathbb{P}_A) \to [0,1] \,|\, A \in \Pi, \mathbb{P}_A \neq \varnothing\}$ will be fixed.

Definition 10. *Let I be an interpretation of $\mathbb{P}$ and A a propositional symbol. We define the mapping $C_I^A \colon \mathbb{P} \longrightarrow [0,1]$, for each $\langle D \leftarrow_i \mathcal{B}, \vartheta\rangle \in \mathbb{P}$, as follows*

$$C_I^A\left(\langle D \leftarrow_i \mathcal{B}, \vartheta\rangle\right) = \begin{cases} \vartheta \dot{\&}_i \hat{I}(\mathcal{B}) & \text{if } D = A \\ 0 & \text{otherwise .} \end{cases}$$

In addition, this new operator is monotonic with respect to the interpretation I [31].

Now, the general definition of quantified immediate consequences operator is recalled, in which we consider a family of fuzzy measures, where the most beneficial measure can be considered for each propositional symbol.

Definition 11. *[31] Let $\mathbb{P}_A$ be the set of rules of $\mathbb{P}$ with head A. We define the M-quantified immediate consequences operator obtained from the family M as the mapping $T_{\mathbb{P}}^M \colon \mathcal{I}_{\mathfrak{L}} \to \mathcal{I}_{\mathfrak{L}}$ given by*

$$T_{\mathbb{P}}^M(I)(A) = \begin{cases} Q_{\mu_A}\left(C_I^A\right) & \text{if } \ \mathbb{P}_A \neq \varnothing \\ 0 & \text{if } \ \mathbb{P}_A = \varnothing \end{cases} \tag{1}$$

for all $A \in \Pi$ and $I \in \mathcal{I}_{\mathfrak{L}}$.

Notice that, if $|\mathbb{P}_A| = 1$, then $\mu = \mu^{\exists}$ and $T_{\mathbb{P}}^M(I)(A) = Q^{\mu_A}(C_I^A) = Q^{\exists}(C_I^A)$. Several illustrative examples can be found in [31]. This new definition is monotonic as the following result recalls.

Proposition 1. *[31] The immediate consequences operator $T_{\mathbb{P}}^M$ is increasing in $\mathcal{I}_{\mathfrak{L}}$.*

In [31], we present some examples on generalized quantifiers, for which a declarative definition of model was given. These definitions were based on solving an specific problem, which can be represented with a quantifier. For example, the declarative definition of μ^1-model was given as "an interpretation $I \in \mathcal{I}_{\mathfrak{L}}$ such that, for each $A \in \Pi$, all the rules with head A are satisfied by I, except possibly one of them", which clearly can be understood as the quantifier Q_{μ^1}, and where the fuzzy measure $\mu^1 \colon \mathcal{P}(\mathbb{P}) \to [0,1]$ is defined as:

$$\mu^1(D) = \begin{cases} 0, & \text{if } |D| \leq 1 \\ 1, & \text{otherwise .} \end{cases} \tag{2}$$

However, it is not possible to give a general declarative definition, but only particular for each family of quantifiers. Therefore, aesthetically, the notion of model had to be adapted to the new flexible framework based on the considered generalized quantifier.

Definition 12. *An interpretation $I \in \mathcal{I}_{\mathfrak{L}}$ is an M-model of $\mathbb{P}$, if $Q_{\mu_A}\left(C_I^A\right) \leq I(A)$ for all $A \in \Pi$. In this case, we say that I satisfies the set of rules in $\mathbb{P}$ concerning the family of generalized quantifiers M.*

Clearly, an interpretation M-model of $\mathbb{P}$ it is a post-fixed point of $T_{\mathbb{P}}^{\mathrm{M}}$. The following result is an adaptation of Theorem 4 to this new framework.

Theorem 5. *[31] We have that $M_{\mathbb{P}} = \inf\{I \in \mathcal{I}_{\mathfrak{L}} \,|\, I \text{ is an } M\text{-model of } \mathbb{P}\}$ is an M-model and there is an ordinal α such that $M_{\mathbb{P}} = lfp\left(T_{\mathbb{P}}^M\right) = \left(T_{\mathbb{P}}^M\right)^{\alpha}(\triangle)$.*

3. A procedural semantics for multi-adjoint logic programming

The main goal of this section is to provide an algorithm for query answering extending the one given in [28]. Recall that a query (or goal) is a propositional symbol intended as a question prompting the system, denoted as $?A$. As a consequence, it can also be used for computing the least M-model of a given program. For this purpose, we will consider the usual definitions, such as the extended language $\mathfrak{F}^e$, and the admissible rules, which will provide a procedural semantics for giving computed answers to queries of programs. Later, we will adapt the notion of correct answer and analyze how this procedure can be considered in the framework of this paper. Recall that we have fixed a non-empty multi-adjoint logic program $\mathbb{P}$, a family of fuzzy measures M, a set of propositional symbols Π and a multi-adjoint Ω-algebra $\mathfrak{L}$.

Definition 13. *[28] Let $V \subseteq [0,1]$ be the set of truth values of the rules in $\mathbb{P}$. The extended language $\mathfrak{F}^e$ is the corresponding Ω-algebra of formulas freely generated from the disjoint union of Π and V. We will refer to the formulas in the language $\mathfrak{F}^e$ simply as extended formulas or e-formulas. An operator symbol ν interpreted under $\mathfrak{F}^e$ will be denoted as $\bar{\nu}$.*

The procedural semantics is based on the admissible rules [15, 28].

Definition 14. *[28] The admissible rules are defined as follows:*

- R1. *Substitute an atom A in an extended formula by $\left(\vartheta \bar{\&}_i \mathcal{B}\right)$ whenever there exists a rule $\langle A \leftarrow_i \mathcal{B}, \vartheta\rangle$ in $\mathbb{P}$.*
- R2. *Substitute an atom A in an extended formula by 0.*
- R3. *Substitute an atom A in an extended formula by ϑ whenever there exists a fact $\langle A \leftarrow_i \top, \vartheta\rangle$ in $\mathbb{P}$.*

The formal definition of computed answer is given next.

Definition 15. *[28] Let $?A$ be a goal. An element $\dot{@}\,[r_1, \ldots, r_m]$, with $r_i \in L$, for all $i \in \{1, \ldots, m\}$ is said to be a computed answer for $?A$ (with respect to $\mathbb{P}$) if there is a sequence $G_0, \ldots, G_{n+1}$ such that*

1. *$G_0 = A$ and $G_{n+1} = \bar{@}\,[r_1, \ldots, r_m]$ where $r_i \in L$ for all $i \in \{1, \ldots, m\}$.*
2. *Every G_i, for $i \in \{1, \ldots, n\}$, is a formula in $\mathfrak{F}'$.*
3. *Every G_{i+1} is inferred from G_i by one of the admissible rules.*

Therefore, the procedural semantics consists of applying admissible rules successively until obtaining an extended formula without propositional symbols $\bar{@}\,[r_1, \ldots, r_m]$. Hence, the computed answer is the interpretation of this formula as an element of $[0,1]$, i.e., $\dot{@}\,(r_1, \ldots, r_m)$.

Example 4. *Considering the program $\mathbb{P}$ presented in Example 3, we will calculate a computed answer for $?$`covid`. For that, we start with $G_0 =$ `covid`. Applying the admissible rule R1 to $\langle$`covid` $\leftarrow_P$ `moderate_fever`$, 0.9\rangle$, we obtain the e-formula $G_1 = 0.9\bar{\&}_P$`moderate_fever` and with the rule R3 applied to the fact $\langle$`moderate_fever` $\leftarrow_P 1, 0.6\rangle$, $G_2 = 0.9\bar{\&}_P 0.6$ is obtained, which does not contain any propositional symbol. Therefore, we can conclude that $\lambda = 0.9\bar{\&}_P 0.6 = 0.54$ is a computed answer for $?$`covid` w.r.t. $\mathbb{P}$.*

Notice that, we could have also started from $\langle$`covid` $\leftarrow_P$ `cough`$, 0.7\rangle$, obtaining the computed answer $\lambda' = 0.7\bar{\&}_P 0.8 = 0.56$. Therefore, different computed answers can be obtained depending on the rules considered for its calculation. □

Next, we must pay attention to the soundness of the procedure. First of all, we need to adapt the classical definition of correct answer.

Definition 16. *An element $\lambda \in L$ is an M-correct answer for the program $\mathbb{P}$ and a query $?A$ if for each M-model $I\colon \Pi \longrightarrow L$ of $\mathbb{P}$, the inequality $\lambda \preceq I(A)$ holds.*

Moreover, we can see from Definition 16 that the least value 0 of the lattice is always an M-correct answer. However, it does not provide any information. In order to solve these drawbacks, an outstanding particular case of M-correct answer will be considered.

Definition 17. *We define the maximum M-correct answer λ^M_{MC}, for a program $\mathbb{P}$ and a query $?A$ as*

$$\lambda^M_{MC} = \sup\{\lambda \in [0,1] \mid \lambda \text{ is an M-correct answer for } \mathbb{P} \text{ and } ?A\}.$$

From this definition, by the properties of the supremum, we can deduce that the maximum M-correct answer is unique for each family M. On the other hand, in the following lemma we prove that, indeed, the supremum of M-correct answers is an M-correct answer. Consequently, this lemma provides the basis for the terminology (maximum) used in the previous definition.

Lemma 1. *Let A be a propositional symbol, then λ^M_{MC} is an M-correct answer for $\mathbb{P}$ and the query $?A$.*

Proof. Let $\lambda \in [0,1]$ be an M-correct answer for the program $\mathbb{P}$ and the query $?A$. Then, it is satisfied that $\lambda \leq I(A)$, for any M-model I of $\mathbb{P}$. Therefore, by the supremum property, the following inequality is satisfied.

$$\begin{aligned} \lambda^{\mathrm{M}}_{\mathrm{MC}} &= \sup\{\lambda \in L \mid \lambda \text{ is an M-correct answer for } \mathbb{P} \text{ and } ?A\} \\ &\leq I(A). \end{aligned}$$

Consequently, from Definition 16, it is clear that $\lambda^{\mathrm{M}}_{\mathrm{MC}}$ is an M-correct answer for $\mathbb{P}$ and the query $?A$.

□

In the following theorem, we will show that the immediate consequences operator given in Definition 11 is the maximum M-correct answer.

Theorem 6. *Let A be a propositional symbol and α an ordinal such that $\left(T^M_{\mathbb{P}}\right)^{\alpha}(\triangle) = lfp(T^M_{\mathbb{P}})$. Then, $\left(T^M_{\mathbb{P}}\right)^{\alpha}(\triangle)(A)$ is the maximum M-correct answer for $\mathbb{P}$ and $?A$, where $T^M_{\mathbb{P}}$ is given by Definition 11.*

Proof. From Theorem 5, there exists an ordinal α such that $\left(T^{\mathrm{M}}_{\mathbb{P}}\right)^{\alpha}(\triangle)$ is the least M-model of $\mathbb{P}$. Then, for any M-model I of $\mathbb{P}$ it is satisfied that $\left(T^{\mathrm{M}}_{\mathbb{P}}\right)^{\alpha}(\triangle)(A) \leq I(A)$. From Definition 16, if $\lambda \in [0,1]$ is an M-correct answer for $\mathbb{P}$ and the query $?A$, then for every M-model I the inequality $\lambda \leq I(A)$ holds. In particular, for $\left(T^{\mathrm{M}}_{\mathbb{P}}\right)^{\alpha}(\triangle)$ it is verified that $\lambda \leq \left(T^{\mathrm{M}}_{\mathbb{P}}\right)^{\alpha}(\triangle)(A)$. In addition, from the definition of supremum, we reach that $\lambda^{\mathrm{M}}_{\mathrm{MC}} \leq \left(T^{\mathrm{M}}_{\mathbb{P}}\right)^{\alpha}(\triangle)(A)$.

On the other hand, as $\left(T^{\mathrm{M}}_{\mathbb{P}}\right)^{\alpha}(\triangle)(A) \leq I(A)$, for all M-model I of $\mathbb{P}$, then $\left(T^{\mathrm{M}}_{\mathbb{P}}\right)^{\alpha}(\triangle)(A)$ is an M-correct answer for $\mathbb{P}$ and $?A$. And as $\lambda^{\mathrm{M}}_{\mathrm{MC}}$ is the maximum M-correct answer, then $\left(T^{\mathrm{M}}_{\mathbb{P}}\right)^{\alpha}(\triangle)(A) \leq \lambda^{\mathrm{M}}_{\mathrm{MC}}$. Consequently, $\left(T^{\mathrm{M}}_{\mathbb{P}}\right)^{\alpha}(\triangle)(A) = \lambda^{\mathrm{M}}_{\mathrm{MC}}$.

□

As it is said in [28], when we are working in a practical system, continuity is one of the most important properties that the operator $T^{\mathrm{M}}_{\mathbb{P}}$ needs to satisfy, because it implies the existence of such computational model and that it can be reached in numerable iterations ω.

Furthermore, in the example presented by Morishita [33], the program $\mathbb{P}$, composed of the rules $\langle A \leftarrow B, a\rangle$, $\langle A \leftarrow B, b\rangle$ and the fact $\langle B \leftarrow \top, \top\rangle$, is considered, where $\bot < a < \top$ and $\bot < b < \top$. The computed answers are $\bot$, a or b, but as a and b are incomparable, it can be proved that the maximum correct answer for the query $?A$ is $\top$. Therefore, the most interesting answer cannot be computationally computed. This fact motivated the definition of reductant, which will be the central notion of the following section.

4. A new definition of reductants adapted to this framework

This section focuses on the resolution of the problem highlighted above, with the use of reductants. Recall that we have considered a non-empty multi-adjoint logic program $\mathbb{P}$, a family of fuzzy measures M, a set of propositional symbols Π and a multi-adjoint Ω-algebra $\mathfrak{L}$.

Definition 18. *Assume that all the rules in $\mathbb{P}$ with head A are $\langle A \leftarrow_{j_i} \mathcal{B}_i, \vartheta_i\rangle$, for $i \in \{1,\ldots,l\}$, in which some rules can be facts, and so some formulas $\mathcal{B}_i$ can be the $\top$ symbol. An M-reductant for A is a rule $\langle A \leftarrow @(\mathcal{B}_1,\ldots,\mathcal{B}_l), \vartheta\rangle$ such that for any $b_1,\ldots,b_l \in [0,1]$ it is satisfied that*

$$Q_{\mu_A}(C_b^A) \;=\; \vartheta \dot{\&} \dot{@}\,(b_1,\ldots,b_l)\,,$$

where $(\leftarrow, \&)$ is an adjoint pair and the mapping $C_b^A\colon \mathbb{P} \to [0,1]$, with $b = (b_1,\ldots,b_l)$, is defined as

$$C_b^A\left(\left\langle D \leftarrow_i \mathcal{B}, \vartheta'\right\rangle\right) = \begin{cases} \vartheta' \dot{\&}_i b_i & \text{if } D = A \\ 0 & \text{otherwise}\,. \end{cases}$$

The following result shows an interesting properties of C_b^A mappings.

Lemma 2. *The mapping $C^A\colon [0,1]^n \to [0,1]^{\mathbb{P}}$, defined as $C^A(b) = C_b^A$, for all $b \in [0,1]^n$, is increasing.*

Proof. The proof is analogous to that given in Proposition 4 in [31].

□

Now, with the following example, we can see that an M-reductant for A can be a fact despite the fact that not all the rules with head A must be facts.

Example 5. *Consider the multi-adjoint logic program $\mathbb{P}_1$ given by the rules $\langle A \leftarrow_{j_1} \top, 1\rangle$ and $\langle A \leftarrow_{j_2} B, 0.1\rangle$, and the measure $\mu^{\exists}$ associated with the existential quantifier. It can be shown that $\langle A \leftarrow_{j_1} \top, 1\rangle$ is an M-reductant for A. Moreover, the rule $\langle A \leftarrow @(\top, B), 1\rangle$, where $\dot{@}(b_1,b_2) = \sup\{1\dot{\&}_{j_1} b_1, 0.1\dot{\&}_{j_2} b_2\}$, for all $b_1, b_2 \in [0,1]$, is clearly another M-reductant for A.*

The following example shows that different M-reductants also arise properly changing the aggregator operator.

Example 6. *Let $\dot{\&}_P^*\colon [0,1]_{10} \times [0,1]_4 \to [0,1]_4$ be a discrete conjunctor [27], where $[0,1]_{10}$ and $[0,1]_4$ are the granular sets of the unit interval into 10 and 4 pieces, respectively. Moreover, we consider the program $\mathbb{P}_2$ consisting of the rules $\langle A \leftarrow_P \top, 1\rangle$ and $\langle A \leftarrow_P B, 0.9\rangle$ and the family of fuzzy measures $M = \{\mu_D = \mu^{\exists} \mid D \in \Pi\}$. Then, it can be proved that the following rules are M-reductants for A.*

$$\begin{aligned} u_1 &= \langle A \leftarrow @(\top, B), 1\rangle\,, & u_5 &= \left\langle A \leftarrow_P^* @'(\top, B), 1\right\rangle\,,\\ u_2 &= \langle A \leftarrow_P^* @(\top, B), 0.9\rangle\,, & u_6 &= \left\langle A \leftarrow_P^* @''(\top, B), 1\right\rangle\,,\\ u_3 &= \langle A \leftarrow_P^* @(\top, B), 1\rangle\,, & u_7 &= \left\langle A \leftarrow_P^* @'''(\top, B), 1\right\rangle\,,\\ u_4 &= \langle A \leftarrow_P^* @(\top, B), 0.8\rangle\,, & u_8 &= \left\langle A \leftarrow_P^* @''''(\top, B), 1\right\rangle\,, \end{aligned}$$

where

$$\begin{aligned} \dot{@}(b_1,b_2) &= \sup\{1\dot{\&}_P b_1, 0.9\dot{\&}_P b_2\},\\ \dot{@}'(b_1,b_2) &= \sup\{1\dot{\&}_P b_1, 0.9\dot{\&}_P^* b_2\},\\ \dot{@}''(b_1,b_2) &= \sup\{1\dot{\&}_P b_1, 1\dot{\&}_P^* b_2\},\\ \dot{@}'''(b_1,b_2) &= \sup\{1\dot{\&}_P^* b_1, 1\dot{\&}_P b_2\},\\ \dot{@}''''(b_1,b_2) &= \sup\{0.9\dot{\&}_P^* b_1, 1\dot{\&}_P b_2\}, \end{aligned}$$

for all $b_1, b_2 \in [0,1]_4$. Therefore, we can see the large number of M-reductants that exist for each A.

The following result shows that the M-models of a program also satisfy the M-reductants.

Proposition 2. *Any M-reductant $\langle A \leftarrow \mathcal{B}, \vartheta\rangle$ of $\mathbb{P}$ is satisfied by any M-model of $\mathbb{P}$.*

Proof. Let I be an M-model of $\mathbb{P}$ and $\langle A \leftarrow \mathcal{B}, \vartheta\rangle$ be an M-reductant of $\mathbb{P}$, with $\mathcal{B} = @(\mathcal{B}_1, \ldots, \mathcal{B}_l)$, where $\langle A \leftarrow_{j_i} \mathcal{B}_i, \vartheta_i\rangle$ are the rules in $\mathbb{P}$ with head A, for all $i \in \{1, \ldots, l\}$. From Definition 12, it holds that $T_{\mathbb{P}}^{\mathrm{M}}(I) \sqsubseteq I$. Consequently,

$$\begin{aligned}
\vartheta \dot{\&} \hat{I}(\mathcal{B}) &= \vartheta \dot{\&} \hat{I}\left(@(\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_l)\right) \\
&= \vartheta \dot{\&} \dot{@}\left(\hat{I}(\mathcal{B}_1), \hat{I}(\mathcal{B}_2), \ldots, \hat{I}(\mathcal{B}_l)\right) \\
&\overset{(1)}{=} Q_{\mu_A}\left(C_I^A\right) \\
&= T_{\mathbb{P}}^{M}(I)(A) \\
&\leq I(A),
\end{aligned}$$

where (1) holds from Definition 18 (M-reductant), considering the particular values given by the interpretation I on $\mathcal{B}_i$, with $i \in \{1, \ldots, l\}$. Notice that, if we consider mappings C_I^A, with $I \in \mathcal{I}_{\mathfrak{L}}$ and $A \in \Pi$, it is satisfied that $T_{\mathbb{P}}^{\mathrm{M}}(I)(A) = \vartheta \dot{\&} \dot{@}\left(b_1, \ldots, b_l\right)$, where $b_i = I(\mathcal{B}_i)$, for all $i \in \{1, \ldots, l\}$. □

Following a similar procedure for construction of reductants given in [28], it is also possible to construct M-reductants for any head $A \in \Pi$ of any rule in a given program $\mathbb{P}$.

Definition 19. *The M-reductant built according to this procedure is called canonical M-reductant for A.*

(1) *If all the rules with head A are $\langle A \leftarrow_{j_i} \mathcal{B}_i, \vartheta_i\rangle$ for all $i \in \{1, \ldots, l\}$, where at least one rule is not a fact, an M-reductant for A is any rule $\langle A \leftarrow @(\mathcal{B}_1, \ldots, \mathcal{B}_l), \top\rangle$ where $\leftarrow$ is any implication with an adjoint conjunctor and $@$ is defined, for all $b = (b_1, \ldots, b_l) \in [0,1]^l$, as*

$$\dot{@}(b_1, \ldots, b_l) = Q_{\mu_A}(C_b^A).$$

(2) *If all the rules with head A are facts $\langle A \leftarrow_{j_i} \top, \vartheta_i\rangle$, for $i \in \{1, \ldots, l\}$, an M-reductant for A is any rule $\langle A \leftarrow \top, \vartheta\rangle$ with $\leftarrow$ any implication with an adjoint conjunctor and $\vartheta = Q_{\mu_A}(C_b^A)$.*

Notice that the canonical M-reductant for a propositional symbol is unique except for the considered implication symbols, although it does not affect the calculations because of its construction.

With the following two examples, we will see how M-reductants are obtained from following Definition 19.

Example 7. *Consider the facts $\langle A \leftarrow_1 \top, 0.2\rangle$ and $\langle A \leftarrow_2 \top, 0.9\rangle$ and the family of fuzzy measures $M = \{\mu_A = \mu^1\}$, which is a singleton. Now, by Definition 19, in order to obtain a canonical M-reductant for A $\langle A \leftarrow \top, \vartheta\rangle$, we need to compute the value ϑ as follows.*

$$\begin{aligned}
\vartheta &\overset{(1)}{=} Q_{\mu_A}(C_b^A) \\
&= \bigvee_{i=1}^{2} C_b^A\left(u_{\pi(i)}\right) \otimes \mu_A\left(\{u_1, \ldots, u_i\}\right) \\
&= \sup\left\{\left(0.9 \&_2 \hat{I}(\top)\right) \otimes \mu_A(\{u_1\}), \left(0.2 \&_1 \hat{I}(\top)\right) \otimes \mu_A(\{u_1, u_2\})\right\} \\
&\overset{(2)}{=} \sup\{0.9 \otimes 0, 0.2 \otimes 1\} \\
&= 0.2,
\end{aligned}$$

where in (1) $b = (\top, \top)$ and (2) holds taking into account that $\mu_A = \mu^1$. Thus, the canonical M-reductant for A is $\langle A \leftarrow \top, 0.2\rangle$.

The following example determines the M-reductants in Example 4.

Example 8. *Taking into account Example 4, and the family of measures $M = \{\mu_{\texttt{covid}} = \mu^1, \mu_{\texttt{moderate_fever}} = \mu^\exists, \mu_{\texttt{cough}} = \mu^\exists\}$, the M-reductant for $\texttt{covid}$ is $\langle \texttt{covid} \leftarrow_G @(\texttt{moderate_fever}, \texttt{cough}), \top\rangle$ where we have that*

$$\dot{@}(b_1, b_2) \;=\; Q_{\mu_{\texttt{covid}}}\left(C_b^{\texttt{covid}}\right),$$

for every $b = (b_1, b_2) \in [0,1]^2$. Moreover, due to the fact that the measures for $\texttt{moderate_fever}$ and $\texttt{cough}$ are the existential ones, the M-reductants are the same facts given in the program for them.

The following result asserts that canonical M-reductants are really M-reductants.

Proposition 3. *Given $A \in \Pi$, the canonical M-reductant for A is an M-reductant for A in $\mathbb{P}$.*

Proof. It straightforwardly follows from the construction of $\langle A \leftarrow \top, \vartheta\rangle$ and $\langle A \leftarrow @(\mathcal{B}_1, \ldots, \mathcal{B}_l), \top\rangle$ in Definitions 18 and 19.

□

Next, we present a new definition of multi-adjoint logic program which only includes the M-reductants for each propositional symbol $A \in \Pi$.

Definition 20. *We define a reduced (canonical) program by M-reductants of $\mathbb{P}$ or (canonical) r-program in short, denoted by $\mathbb{P}_r$ (resp. $\mathbb{P}_r^c$, when canonical reductants are considered), as a multi-adjoint logic program composed of an M-reductant (resp. the canonical M-reductant) for each propositional symbol $A \in \Pi$.*

Notice that, given a program $\mathbb{P}$, the canonical r-program $\mathbb{P}_r^c$ is a particular case of r-program. Therefore, every result true in $\mathbb{P}_r$ will also be true in $\mathbb{P}_r^c$. The consideration of $\mathbb{P}_r$ offers a different point of view of the program $\mathbb{P}$. For example, it provides an alternative for computing the M-models of the program, as the following proposition shows.

Proposition 4. *Let $\mathbb{P}_r$ be an r-program of $\mathbb{P}$. Then, I is an M-model of $\mathbb{P}$ if and only if I is a model of $\mathbb{P}_r$.*

Proof. On the one hand, let $I \in \mathcal{I}_{\mathfrak{L}}$ be an M-model of $\mathbb{P}$. Then, from Proposition 2, it holds that I is a model of $\mathbb{P}_r$.

On the other hand, consider a model $I \in \mathcal{I}_{\mathfrak{L}}$ of $\mathbb{P}_r$. Given a propositional symbol A, if no rule with head A exists in $\mathbb{P}$, then no rule exists in $\mathbb{P}_r$ and the inequality $T_{\mathbb{P}}^{\mathrm{M}}(I)(A) \leq I(A)$ trivially holds. Otherwise, we consider the M-reductant $\langle A \leftarrow @(\mathcal{B}_1, \ldots, \mathcal{B}_l), \vartheta\rangle$ in $\mathbb{P}_r$, and we obtain that

$$\begin{aligned} T_{\mathbb{P}}^{\mathrm{M}}(I)(A) &= Q_{\mu_A}\left(C_I^A\right) \\ &\overset{(1)}{=} \vartheta\dot{\&}\dot{@}(b_1, \ldots, b_l) \\ &= \vartheta\dot{\&}\dot{@}(I(\mathcal{B}_1), \ldots, I(\mathcal{B}_l)) \\ &= T_{\mathbb{P}_r}(I)(A) \\ &\overset{(2)}{\leq} I(A), \end{aligned}$$

where (1) follows from the definition of M-reductant and (2) is verified due to the fact that I is a model of $\mathbb{P}_r$. Thus, $I \in \mathcal{I}_{\mathfrak{L}}$ is an M-model of $\mathbb{P}$.

□

The following result shows the equivalence between the concept of M-correct answer for $\mathbb{P}$ and correct answer for $\mathbb{P}_r$.

Proposition 5. *Consider an r-program $\mathbb{P}_r$, a query $?A$ and $\lambda \in [0,1]$. Then, λ is an M-correct answer for $\mathbb{P}$ and $?A$ if and only if λ is a correct answer for $\mathbb{P}_r$ and $?A$.*

Proof. Given $\lambda \in [0,1]$, the proof follows from the following chain of equivalencies. λ is an M-correct answer for $\mathbb{P}$ and $?A$ if and only if, for all M-model I of $\mathbb{P}$, $\lambda \leq I(A)$, which is equivalent, by Proposition 4, to $\lambda \leq I(A)$ for all model I of $\mathbb{P}_r$, that is, λ is a correct answer of $\mathbb{P}_r$. □

Hence, in order to obtain a correct answer for a query $?A$, it is not possible to consider the original program $\mathbb{P}$, but a reduced one $\mathbb{P}_r$. This also shows the relevance of the reductants. Moreover, the following example shows that $\mathbb{P}$ cannot be considered as the program from which obtain the M-computed answers for $?A$.

Example 9. *Consider a multi-adjoint logic program $\mathbb{P}$ composed of the rules $\langle \mathtt{A} \leftarrow_P \mathtt{B}, 0.7\rangle$ and $\langle \mathtt{A} \leftarrow_P \mathtt{C}, 0.8\rangle$; and the facts $\langle \mathtt{B} \leftarrow_P \top, 0.9\rangle$, $\langle \mathtt{B} \leftarrow_P \top, 0.7\rangle$, $\langle \mathtt{C} \leftarrow_P \top, 0.9\rangle$ and $\langle \mathtt{C} \leftarrow_P \top, 0.5\rangle$. In addition, we consider the family of fuzzy measures $M = \{\mu_D = \mu^1 \mid D \in \Pi\}$, where μ^1 is defined in (2).*

On the one hand, the possible computed answers are $\lambda_1 = 0,72$, $\lambda_2 = 0.63$, $\lambda_3 = 0.49$ and $\lambda_4 = 0.4$. On the other hand, $\left(T_{\mathbb{P}}^M\right)^2(\triangle) = lfp\left(T_{\mathbb{P}}^M\right)$ is an M-model of $\mathbb{P}$ and it can be proved that $\left(T_{\mathbb{P}}^M\right)^2(\triangle)(\mathtt{A}) = 0.4$. Therefore, as $0.72 \not\leq 0.4$, we have that 0.72 is a computed answer but not an M-correct answer. Then, not all computed answer for a query ?A is an M-correct answer for the same query. Indeed, the "quantification" of the set of computed answers with the fuzzy measure μ^1 does not provide an M-correct answer, since we obtain 0.63, which is not lesser than 0.4.

Consequently, we need to adapt the notion of computed answer to capture the new philosophy provided by the use of generalized quantifiers, in which the reductants are the key.

Definition 21. *Let $\mathbb{P}_r$ be an r-program of $\mathbb{P}$ and a query $?A$. We say that $\lambda \in [0,1]$ is a $\mathbb{P}_r$-computed answer for $?A$, if it is a computed answer for $?A$ with respect to $\mathbb{P}_r$, that is, using the admissible rules on the M-reductants in $\mathbb{P}_r$.*

The soundness of the new procedure arises from the following result.

Theorem 7. *Let $\mathbb{P}_r$ be an r-program of $\mathbb{P}$ and a query $?A$. Then, every $\mathbb{P}_r$-computed answer for the query $?A$ is an M-correct answer for $\mathbb{P}$ and $?A$.*

Proof. Let λ be a computed answer for $\mathbb{P}_r$ and the query $?A$. We are going to prove that λ is an M-correct answer by complete induction over the length n of the computation sequence of λ.

For $n = 1$, we have applied the admissible rule R2 or R3.

(1) If we apply R2, then the propositional symbol A is not the head of any rule or fact in $\mathbb{P}_r$. The corresponding sequence of e-formulas is $G_0 = A$, $G_1 = \bot$. Hence, $\lambda = \bot$, which is always an M-correct answer.

(2) If we apply R3, then there is a fact $\langle A \leftarrow \top, \vartheta\rangle \in \mathbb{P}_r$, where $\vartheta = Q_{\mu_A}\left(C_b^A\right)$, and the corresponding sequence of e-formulas is $G_0 = A$, $G_1 = \vartheta$. Therefore, $\lambda = \vartheta$. Now, from the definition of M-model (Definition 12), if $I \in \mathcal{I}_{\mathfrak{L}}$ is an M-model, then $T_{\mathbb{P}}^{\mathrm{M}}(I) \sqsubseteq I$ and, consequently, $\lambda = \vartheta = Q_{\mu_A}\left(C_b^A\right) = T_{\mathbb{P}}^{Q_{\mu_A}}(I)(A) \leq I(A)$, where $b = (b_1, \ldots, b_{|\mathbb{P}_A|}) = (I(\mathcal{B}_1), \ldots, I(\mathcal{B}_{|\mathbb{P}_A|}))$. Therefore, λ is an M-correct answer in $\mathbb{P}_r$.

Suppose that the result is true for a length of the computation sequence of λ less than n, where $1 < n$. In this context, there must necessarily be some rule with head A, since otherwise the length of the sequence would be 1. Therefore, we can construct the M-reductant $\langle A \leftarrow @\left(\mathcal{B}_1, \ldots, \mathcal{B}_l\right), \vartheta\rangle$, where $@\left(\mathcal{B}_1, \ldots, \mathcal{B}_l\right) = @'\left[B_1, \ldots, B_k\right]$, and we have the sequence $G_0 = A$, $G_1 = \vartheta \bar{\&} @'\left[B_1, \ldots, B_k\right]$, …, $G_r = \vartheta \bar{\&} @'\left[\lambda_1, \ldots, \lambda_k\right] = \lambda$, where the calculation sequence for each λ_i is less than n and, from the induction hypothesis, $\lambda_i \leq I(B_i)$ for all M-model I and

$i \in \{1, \ldots, k\}$. Consequently, we reach the following chain of inequalities.

$$\begin{aligned}
\lambda &= \vartheta \bar{\&} \dot{@}' [\lambda_1, \ldots, \lambda_k] \\
&\overset{(1)}{\leq} \vartheta \bar{\&} \dot{@}' [I(B_1), \ldots, I(B_k)] \\
&= \vartheta \bar{\&} \hat{I} \left(@' [B_1, \ldots, B_k] \right) \\
&= \vartheta \bar{\&} \hat{I} \left(@ (\mathcal{B}_1, \ldots, \mathcal{B}_l) \right) \\
&= \vartheta \bar{\&} \dot{@} \left(\hat{I}(\mathcal{B}_1), \ldots, \hat{I}(\mathcal{B}_l) \right) \\
&\overset{(2)}{=} Q_{\mu_A} \left(C_I^A \right) \\
&= T^{\mathrm{M}}_{\mathbb{P}_r}(I)(A) \\
&\overset{(3)}{\leq} I(A),
\end{aligned}$$

where (1) follows from the monotonicity of $\dot{@}$ and $\bar{\&}$ and the induction hypothesis $\lambda_i \leq I(B_i)$, for all $i \in \{1, \ldots, k\}$, (2) holds from the definition of M-reductant and (3) is verified due to the fact that I is an M-model. Thus, λ is an M-correct answer for the query $?A$ with respect to $\mathbb{P}_r$. □

5. Completeness results

The proof of the completeness theorems follows from some technical results. Recall that a non-empty multi-adjoint logic program $\mathbb{P}$, a family of fuzzy measures M, a set of propositional symbols Π and a multi-adjoint Ω-algebra $\mathfrak{L}$ have been fixed.

Lemma 3. *Let α be an ordinal such that $\left(T^M_{\mathbb{P}}\right)^{\alpha}(\triangle) = lfp(T^M_{\mathbb{P}})$ and $\lambda \in [0,1]$. We have that $\lambda \leq \left(T^M_{\mathbb{P}}\right)^{\alpha}(\triangle)(A)$ if and only if λ is an M-correct answer for $\mathbb{P}$ and $?A$.*

Proof. Consider $\lambda \in [0,1]$ verifying that $\lambda \leq \left(T^{\mathrm{M}}_{\mathbb{P}}\right)^{\alpha}(\triangle)(A)$. From Theorem 5, $\left(T^{\mathrm{M}}_{\mathbb{P}}\right)^{\alpha}(\triangle)$ is the least M-model of $\mathbb{P}$. Consequently, it is clear that $\left(T^M_{\mathbb{P}}\right)^{\alpha}(\triangle)(A) \leq I(A)$ for all M-model I of $\mathbb{P}$ and $A \in \Pi$. Therefore, it holds

$$\lambda \leq \left(T^{\mathrm{M}}_{\mathbb{P}}\right)^{\alpha}(\triangle)(A) \leq I(A),$$

for all M-model $I \in \mathcal{I}_{\mathfrak{L}}$, concluding that λ is an M-correct answer for $\mathbb{P}$ and $?A$.

Suppose now that λ is an M-correct answer for $\mathbb{P}$ and $?A$. As $\left(T^{\mathrm{M}}_{\mathbb{P}}\right)^{\alpha}(\triangle)$ is in particular an M-model of $\mathbb{P}$, we obtain that $\lambda \leq \left(T^{\mathrm{M}}_{\mathbb{P}}\right)^{\alpha}(\triangle)(A)$. □

The following result shows that each iteration of the immediate consequences operator is a $\mathbb{P}_r$-computed answer for $?A$ (Definition 21).

Proposition 6. *Let $\mathbb{P}_r$ be an r-program of $\mathbb{P}$. Then, for every $n \in \mathbb{N}$ and propositional symbol $A \in \Pi$, $\left(T^M_{\mathbb{P}}\right)^n(\triangle)(A)$ is a $\mathbb{P}_r$-computed answer for $?A$.*

Proof. We are going to prove this result by induction over the number of iterations n.

For $n = 0$ and $A \in \Pi$, applying R2, then a $\mathbb{P}_r$-computed answer for the query $?A$ is $G_1 = \bot = \left(T^{\mathrm{M}}_{\mathbb{P}}\right)^0(\triangle)(A)$.

Suppose that the result is true for $1 \leq n$ and $Q \in \Pi$, that is, $\left(T^{\mathrm{M}}_{\mathbb{P}}\right)^n(\triangle)(Q)$ is a $\mathbb{P}_r$-computed answer for all query $?Q$. Consider $A \in \Pi$, we are going to prove that $\left(T^{\mathrm{M}}_{\mathbb{P}}\right)^{n+1}(\triangle)(A)$ is a $\mathbb{P}_r$-computed answer for $?A$. Clearly, if no rule with head A exists in $\mathbb{P}$, then no rule exists in $\mathbb{P}_r$ and we trivially obtain that result. Otherwise, we consider the M-reductant. $\langle A \leftarrow @ (\mathcal{B}_1, \ldots, \mathcal{B}_l), \vartheta \rangle \in \mathbb{P}_r$, with $@ (\mathcal{B}_1, \ldots, \mathcal{B}_l) = @' [B_1, \ldots, B_k]$, and where $B_1, \ldots, B_k$ are the propositional symbols of the bodies of the rules with head A. Notice that, if the particular case $\mathbb{P}^c_r$ is considered instead of $\mathbb{P}_r$, this M-reductant could represent the fact or the rule

of the canonical M-reductant for A (Definition 19), depending on whether A has some fact or not. Therefore, admissible rule R1 or R3 will be applied, depending on the case, and a similar procedure can be considered for proving the result.

From the induction hypothesis, $\left(T_{\mathbb{P}}^{\mathrm{M}}\right)^n(\triangle)(B_i)$, which we will denote as β_i, is a $\mathbb{P}_r$-computed answer for the query $?B_i$, for all $i \in \{1,\ldots,k\}$. Consequently, there is a sequence $H_0 = B_i, \ldots, H_{s_i} = \left(T_{\mathbb{P}}^{\mathrm{M}}\right)^n(\triangle)(B_i) = \beta_i$, for all $i \in \{1,\ldots,k\}$.

In order to prove that $\left(T_{\mathbb{P}}^{\mathrm{M}}\right)^{n+1}(\triangle)(A)$ is a $\mathbb{P}_r$-computed answer for $?A$, we have to construct a sequence of extended formulas beginning with A and ending with $\left(T_{\mathbb{P}}^{\mathrm{M}}\right)^{n+1}(\triangle)(A)$, applying admissible rules successively. Starting with $G_0 = A$ and applying the admissible rule R1 to the M-reductant $\langle A \leftarrow @\left(\mathcal{B}_1,\ldots,\mathcal{B}_l\right), \vartheta\rangle$, we obtain $G_1 = \vartheta \bar{\&} \bar{@}'\left[B_1,\ldots,B_k\right]$. Now, each B_i can be substituted by β_i computed previously, for all $i \in \{1,\ldots,k\}$. Therefore, we obtain $G_r = \vartheta \bar{\&} \bar{@}'\left[\beta_1,\ldots,\beta_k\right]$, where $r = 1 + s_1 + \ldots + s_k$.

Finally, from the homomorphic extension theorem, the following chain of equalities holds.

$$\begin{aligned}
\vartheta \dot{\&} \dot{@}'[\beta_1,\ldots,\beta_k] &= \vartheta \dot{\&} \dot{@}'[\left(T_{\mathbb{P}}^{\mathrm{M}}\right)^n(\triangle)(B_1),\ldots,\left(T_{\mathbb{P}}^{\mathrm{M}}\right)^n(\triangle)(B_k)] \\
&= \vartheta \dot{\&} \widehat{\left(T_{\mathbb{P}}^{\mathrm{M}}\right)^n}(\triangle)\left(@'\left[B_1,\ldots,B_k\right]\right) \\
&= \vartheta \dot{\&} \widehat{\left(T_{\mathbb{P}}^{\mathrm{M}}\right)^n}(\triangle)\left(@\left(\mathcal{B}_1,\ldots,\mathcal{B}_l\right)\right) \\
&= \vartheta \dot{\&} \dot{@}\left(\widehat{\left(T_{\mathbb{P}}^{\mathrm{M}}\right)^n}(\mathcal{B}_1),\ldots,\widehat{\left(T_{\mathbb{P}}^{\mathrm{M}}\right)^n}(\triangle)(\mathcal{B}_l)\right) \\
&\overset{(1)}{=} \bigvee_{i=1}^{l}\left(\vartheta_i \dot{\&}_i \widehat{\left(T_{\mathbb{P}}^{\mathrm{M}}\right)^n}(\triangle)\left(\mathcal{B}_{\pi(i)}\right)\right) \otimes \mu_A\left(\{u_1,\ldots,u_i\}\right) \\
&= \left(T_{\mathbb{P}}^{\mathrm{M}}\right)^{n+1}(\triangle)(A),
\end{aligned}$$

where (1) follows from the definition of M-reductant (Definition 18). Consequently, we can conclude that $\left(T_{\mathbb{P}}^{\mathrm{M}}\right)^{n+1}(\triangle)(A) = \vartheta \dot{\&} \dot{@}'\left(\beta_1,\ldots,\beta_k\right)$ is a $\mathbb{P}_r$-computed answer for $A?$. □

For the following results, we need to recall the concept of continuity. We say that an operator T is Scott-continuous, or just continuous, if it preserves suprema of directed sets, i.e., given a directed set $X \subseteq L$, it is satisfied that $T\left(\sup X\right) = \sup\left\{T(x) \,|\, x \in X\right\}$ [29]. We have now the necessary tools to prove the completeness results.

Theorem 8. *If $T_{\mathbb{P}}^{M}$ is continuous, then for each M-correct answer $\lambda \in [0,1]$ for a program $\mathbb{P}$ and a query $?A$ and for all $\varepsilon < \lambda$, there is an $\mathbb{P}_r$-computed answer δ such that $\varepsilon < \delta$.*

Proof. On the one hand, we are going to prove by induction over n that

$$\left(T_{\mathbb{P}}^{\mathrm{M}}\right)^n(\triangle) \;\sqsubseteq\; \left(T_{\mathbb{P}}^{\mathrm{M}}\right)^{n+1}(\triangle). \tag{3}$$

For $n = 0$, this result is clear since $\triangle \sqsubseteq T_{\mathbb{P}}^{\mathrm{M}}(\triangle)$ always holds. Suppose that $\left(T_{\mathbb{P}}^{\mathrm{M}}\right)^{n-1}(\triangle) \sqsubseteq \left(T_{\mathbb{P}}^{\mathrm{M}}\right)^n(\triangle)$ is true for $1 \leq n$. From Proposition 1, $T_{\mathbb{P}}^{\mathrm{M}}$ is monotonic, then

$$T_{\mathbb{P}}^{\mathrm{M}}\left(\left(T_{\mathbb{P}}^{\mathrm{M}}\right)^{n-1}(\triangle)\right) \sqsubseteq T_{\mathbb{P}}^{\mathrm{M}}\left(\left(T_{\mathbb{P}}^{\mathrm{M}}\right)^{n}(\triangle)\right).$$

On the other hand, as λ is an M-correct answer, then for all M-model I, it is verified that $\lambda \leq I(A)$. Since $T_{\mathbb{P}}^{\mathrm{M}}$ is continuous, then $\mathrm{lfp}(T_{\mathbb{P}}^{\mathrm{M}}) = \left(T_{\mathbb{P}}^{\mathrm{M}}\right)^{\omega}(\triangle)$ and, by Lemma 3, we have that $\lambda \leq \left(T_{\mathbb{P}}^{\mathrm{M}}\right)^{\omega}(\triangle)(A)$. Taking into account the continuity of $T_{\mathbb{P}}^{\mathrm{M}}$ and Expression (3), we obtain that

$$\varepsilon < \lambda \leq \left(T_{\mathbb{P}}^{\mathrm{M}}\right)^{\omega}(\triangle)(A) = \sup\left\{\left(T_{\mathbb{P}}^{\mathrm{M}}\right)^{n}(\triangle)(A) \,|\, n \in \mathbb{N}\right\}.$$

Therefore, since $\left\{\left(T_{\mathbb{P}}^{\mathrm{M}}\right)^{n}(\triangle)(A) \,|\, n \in \mathbb{N}\right\}$ is an increasing sequence of elements in $[0,1]$ with limit greater or equal than λ, then there exists some $\delta \in X$ (or equivalently $n_0 \in \mathbb{N}$), that is

$\delta = \left(T_{\mathbb{P}}^{\mathrm{M}}\right)^{n_0}(\triangle)(A)$, such that

$$\varepsilon < \delta \leq \left(T_{\mathbb{P}}^{\mathrm{M}}\right)^{\omega}(\triangle)(A),$$

which finishes the proof because of, by Proposition 6, $\left(T_{\mathbb{P}}^{\mathrm{M}}\right)^{n_0}(\triangle)(A)$ is an $\mathbb{P}_r$-computed answer. □

Moreover, the following quasi-completeness theorem holds.

Theorem 9. *If $T_{\mathbb{P}}^{M}$ is continuous, then for each M-correct answer $\lambda \in [0,1]$ for a program $\mathbb{P}$ and a query $?A$, there is a sequence of elements $\lambda_n \in [0,1]$, with $\lambda \leq \sup \lambda_n$, such that, for any $n_0 \in \mathbb{N}$, there exists a $\mathbb{P}_r$-computed answer δ verifying $\lambda_{n_0} \leq \delta$.*

Proof. Consider $\lambda_n = \left(T_{\mathbb{P}}^{\mathrm{M}}\right)^{n}(\triangle)(A)$ and the M-correct answer $\lambda \in [0,1]$. As λ is an M-correct answer and $\left(T_{\mathbb{P}}^{\mathrm{M}}\right)^{\omega}(\triangle)$ is an M-model, then

$$\lambda \leq \left(T_{\mathbb{P}}^{\mathrm{M}}\right)^{\omega}(\triangle)(A) = \sup\left\{\left(T_{\mathbb{P}}^{\mathrm{M}}\right)^{n}(\triangle)(A) \,|\, n \in \mathbb{N}\right\} = \sup \lambda_n.$$

Now, given $n_0 \in \mathbb{N}$, we consider the value $\delta = \left(T_{\mathbb{P}}^{\mathrm{M}}\right)^{n_0}(\triangle)(A) = \lambda_{n_0}$, which is a $\mathbb{P}_r$-computed answer for $?A$ by Proposition 6.

□

5.1. **Maximum computed answer.** This section introduces the notion of maximum M-computed answer and studies its computation and properties as well.

Definition 22. *Consider an r-program of $\mathbb{P}$, $\mathbb{P}_r$, and a query $?A$. We say that λ is the maximum $\mathbb{P}_r$-computed answer, if it is a $\mathbb{P}_r$-computed answer verifying that $\lambda' \leq \lambda$ for all $\mathbb{P}_r$-computed answer λ' for $?A$.*

Notice that, the maximum $\mathbb{P}_r$-computed answer could not exist, because it is possible to obtain an infinite increasing chain of computed answers without a maximum, as we will show in the following example. However, this is not a relevant problem since the completeness results (Theorems 8 and 9) proved that we can be as close as we require to the maximum $\mathbb{P}_r$-computed answer for a query $?A$. Indeed, Proposition 6 showed the relationship between the iterations of $T_{\mathbb{P}}^{\mathrm{M}}$ and the $\mathbb{P}_r$-computed answers for $?A$, which will be complemented with new results at the end of this section. Thus, the termination results of the immediate consequences operator provided in [9, 11] can also give sufficient conditions to ensure the existence of the maximum $\mathbb{P}_r$-computed answer. This remark complements the developments considered, for instance, in [14, 28].

Example 10. *Considering the multi-adjoint lattice introduced in Example 2, and the multi-adjoint logic program $\mathbb{P}$ consisting of the weighted rules $\langle A \leftarrow @_{(1,2)}(B_1, B_2), 1\rangle$, $\langle C \leftarrow_P A, 0.7\rangle$, $\langle B_1 \leftarrow_P C, 0.9\rangle$, $\langle B_1 \leftarrow \top, 0.2\rangle$ and $\langle B_2 \leftarrow \top, 0.8\rangle$. An r-program $\mathbb{P}_r$ for $\mathbb{P}$ has the rules $\langle A \leftarrow @(B_1, B_2), 1\rangle$, $\langle C \leftarrow @'(A), 1\rangle$, $\langle B_1 \leftarrow @''(C, \top), 1\rangle$, and $\langle B_2 \leftarrow \top, 0.8\rangle$, where $\dot{@}(b_1, b_2) = 1\dot{\&}_P\dot{@}_{(1,2)}(b_1, b_2)$, $\dot{@}'(a) = 0.7\dot{\&}_P a$ and $\dot{@}''(b_1, b_2) = \sup\{0.9\dot{\&}_P b_1, 0.2\dot{\&}_P b_2\}$, for all $a, b_1, b_2 \in [0,1]$.*

The procedure for computing the $\mathbb{P}_r$-computed answer for $?A$ is presented next; where the first column shows the extended formula, while the second one shows the evaluation of the $\mathbb{P}_r$-computed answer, if the admissible rule R2 is used on the remaining propositional, in order to stop the computation. For example, in the second row, B_1 and B_2 have been substituted in the formula by 0. Hence, the obtained value is 0. In the third row, C is substituted by 0 and the evaluation produces $\lambda_2 = 0.53$.

Extended formulae	*R2*
$G_0 = A$	$\lambda_0 = 0$
$G_1 = 1\bar{\&}\bar{@}(B_1, B_2)$	$\lambda_1 = 0$
$G_2 = 1\bar{\&}\bar{@}(B_1, 0.8)$	$\lambda_2 = 0.53$
$G_3 = 1\bar{\&}\bar{@}\left(1\bar{\&}\bar{@}''(C, \top), 0.8\right)$	$\lambda_3 = 0.6$
$G_4 = 1\bar{\&}\bar{@}\left(1\bar{\&}\bar{@}''(1\bar{\&}\bar{@}'(A), \top), 0.8\right)$	$\lambda_4 = 0.6$
$G_5 = 1\bar{\&}\bar{@}\left(1\bar{\&}\bar{@}''(1\bar{\&}\bar{@}'(1\bar{\&}\bar{@}(B_1, B_2)), \top), 0.8\right)$	$\lambda_5 = 0.6$
$G_6 = 1\bar{\&}\bar{@}\left(1\bar{\&}\bar{@}''(1\bar{\&}\bar{@}'(1\bar{\&}\bar{@}(B_1, 0.8)), \top), 0.8\right)$	$\lambda_6 = 0.64$
$G_7 = 1\bar{\&}\bar{@}\left(1\bar{\&}\bar{@}''(1\bar{\&}\bar{@}'(1\bar{\&}\bar{@}((1\bar{\&}\bar{@}(C, \top)), 0.8)), \top), 0.8\right)$	$\lambda_7 = 0.66$

Note that, the computation of the $\mathbb{P}_r$-computed answer has not finished after these seven steps because we have still propositional symbols on it. Consequently, the maximum $\mathbb{P}_r$-computed answer is not reached yet. Moreover, we could have been tempted to define the maximum $\mathbb{P}_r$-computed answer as the value λ_7 obtained in G_7 using the rule R2. Nevertheless, λ_7 would not be the maximum $\mathbb{P}_r$-computed answer for $?P$ since, if we continue the computing process, we will obtain another answer λ_j, with $\lambda_7 < \lambda_j$.

The following theorem provides a mechanism for computing the maximum $\mathbb{P}_r$-computed answer.

Theorem 10. *Let $\mathbb{P}_r$ be an r-program of $\mathbb{P}$, and $?A$ a query. If the maximum $\mathbb{P}_r$-computed answer exists, then it is obtained by applying the rule R2 on atoms B for which no rules or facts with head B exists; otherwise the canonical M-reductant will be used.*

Proof. First of all, notice that at most one canonical M-reductant exists for every propositional symbol. Given an r-program $\mathbb{P}_r$ of $\mathbb{P}$, $\lambda \in [0, 1]$, the value obtained following the procedure detailed in the statement of the theorem, and $\lambda' \in [0, 1]$ a $\mathbb{P}_r$-computed answer for the query $?A$. We are going to prove $\lambda' \leq \lambda$, by induction over the length n in the calculation of the $\mathbb{P}_r$-computed answer λ'.

If $n = 1$, then it has been applied rule R2 or rule R3.

- R2. In this case, the $\mathbb{P}_r$-computed answer for the query $?A$ is $\lambda' = \bot \leq \lambda$.
- R3. In $\mathbb{P}_r$ there exists the M-reductant $\langle A \leftarrow \top, \vartheta' \rangle$, where $Q_{\mu_A}(C_b^A) = \vartheta' \dot{\&} \top = \vartheta'$ for all $b \in [0, 1]^l$, with l being the number of rules in $\mathbb{P}$ with head A. Hence, through the sequence of e-formulas $G'_0 = A$, $G'_1 = \vartheta'$, the $\mathbb{P}_r$-computed answer for $?A$ is $\lambda' = \vartheta'$. Now, for the canonical M-reductant (following the procedure given in the statement of the theorem) there exist two options.
  (1) All the rules with head A in $\mathbb{P}$ are facts $\langle A \leftarrow_{j_i} \top, \vartheta_i \rangle$, with $i \in \{1, \ldots, l\}$. The canonical M-reductant for A is $\langle A \leftarrow \top, \vartheta \rangle$, where $\vartheta = Q_{\mu_A}(C_b^A)$, with $b = (b_1, \ldots, b_l) \in [0, 1]^l$. Consequently, in this case, $\lambda' = \vartheta' = Q_{\mu_A}(C_b^A) = \vartheta = \lambda$.
  (2) The rules in $\mathbb{P}$ with head A are $\langle A \leftarrow_{j_i} \mathcal{B}_i, \vartheta_i \rangle$, with $i \in \{1, \ldots, l\}$, where some rule is not a fact. Notice that, an M-reductant can be a fact although not all rules with head A must be facts, as Example 5 shown. The canonical M-reductant for $?A$ is $\langle A \leftarrow @(\mathcal{B}_1, \ldots, \mathcal{B}_l), \top \rangle$, where $@(b_1, \ldots, b_l) = Q_{\mu_A}(C_b^A)$, for all $b = (b_1, \ldots, b_l) \in [0, 1]^l$. Moreover, we have that λ is obtained from the sequence of e-formulas $G_0 = A$, $G_1 = \top \bar{\&} \bar{@}(\mathcal{B}_1, \ldots, \mathcal{B}_l) = \bar{@}'[B_1, \ldots, B_k]$, ..., $G_r = \bar{@}'[\lambda_1, \ldots, \lambda_k] = \lambda$, where λ_i is obtained following the procedure given in the statement of the theorem, for all $i \in \{1, \ldots, k\}$. By the definition of $@'$, there exist $b'_1, \ldots, b'_l \in [0, 1]$, such that $\dot{@}'[\lambda_1, \ldots, \lambda_k] = \dot{@}(b'_1, \ldots, b'_l)$. Finally, the following chain of equalities is reached.

$$\begin{aligned}\lambda' &= \vartheta' \\ &= Q_{\mu_A}(C^A_{b'}) \\ &\overset{(*)}{=} \dot{@}(b'_1,\ldots,b'_l) \\ &= \dot{@}'[\lambda_1,\ldots,\lambda_k] \\ &= \lambda,\end{aligned}$$

where (*) follows from the definition of M-reductant with $b' = (b'_1,\ldots,b'_l)$.

Now, suppose that the result is true for all length of computations less than n, with $1 < n$. In this case, the first admissible rule applied must necessarily be R1. Let $\langle A \leftarrow @(\mathcal{B}_1,\ldots,\mathcal{B}_l), \vartheta\rangle$ be the M-reductant from $\mathbb{P}_r$ for A, where $@(\mathcal{B}_1,\ldots,\mathcal{B}_l) = @'[B_1,\ldots,B_k]$, with $B_1,\ldots,B_k$ the propositional symbols of the bodies of the rules with head A. On the one hand, the $\mathbb{P}_r$-computed answer λ' is obtained from the sequence of e-formulas $G'_0 = A$, $G'_1 = \vartheta\bar{\&}\bar{@}(\mathcal{B}_1,\ldots,\mathcal{B}_l) = \vartheta\bar{\&}\bar{@}'[B_1,\ldots,B_k]$, ..., $G'_n = \vartheta\bar{\&}\bar{@}'[\lambda'_1,\ldots,\lambda'_k] = \lambda'$, where λ'_i is the $\mathbb{P}_r$-computed answer for the query $?B_i$, for all $i \in \{1,\ldots,k\}$.

On the other hand, the canonical M-reductant for $?A$ is $\langle A \leftarrow @''(\mathcal{B}_1,\ldots,\mathcal{B}_l), \top\rangle$ and $G_0 = A$, $G_1 = \top\bar{\&}\bar{@}''(\mathcal{B}_1,\ldots,\mathcal{B}_l) = \top\bar{\&}\bar{@}'''[B_1,\ldots,B_k]$, ..., $G_r = \bar{@}'''[\lambda_1,\ldots,\lambda_k] = \lambda$, where λ_i is the value obtained following the procedure given for the query $?B_i$ in the statement of the theorem, for all $i \in \{1,\ldots,k\}$. By the definition of $@'$ and $@'''$, there exist $b_1,\ldots,b_l,b'_1,\ldots,b'_l \in [0,1]$, such that $\bar{@}'[\lambda'_1,\ldots,\lambda'_l] = \dot{@}'(b'_1,\ldots,b'_l)$ and $\bar{@}'''[\lambda_1,\ldots,\lambda_l] = \dot{@}'''(b_1,\ldots,b_l)$ respectively. In addition, by the induction hypothesis, we have that $\lambda'_i \leq \lambda_i$ for all $i \in \{1,\ldots,k\}$, and the following chain of inequalities is satisfied.

$$\begin{aligned}\lambda' &= \vartheta\bar{\&}\bar{@}'\left[\lambda'_1,\ldots,\lambda'_l\right] \\ &= \vartheta\bar{\&}\dot{@}'\left(b'_1,\ldots,b'_l\right) \\ &\overset{(1)}{=} Q_{\mu_A}\left(C^A_{b'}\right) \\ &\overset{(2)}{\leq} Q_{\mu_A}\left(C^A_{b}\right) \\ &= \dot{@}'''(b_1,\ldots,b_l) \\ &\overset{(3)}{=} \bar{@}'''[\lambda_1,\ldots,\lambda_l] \\ &= \lambda,\end{aligned}$$

where (1) and (3) follow from the definition of M-reductant and canonical M-reductant respectively, with $b = (\lambda_1,\ldots,\lambda_l)$, $b' = (\lambda'_1,\ldots,\lambda'_l)$; while (2) is satisfied due to the induction hypothesis, Lemma 2 and the monotonicity of the quantifier.

□

Theorem 10 is really important because it says that the maximum $\mathbb{P}_r$-computed answer for a query $?A$ can be obtained from the reduced program $\mathbb{P}^c_r$ applying admissible rules $R1$ and $R3$ whenever possible. Since $|\mathbb{P}^c_r(A)| \leq 1$, for all $A \in \Pi$, then the sequence of the computation of the maximum $\mathbb{P}^c_r$-computed answer is unique.

Theorem 11. *Let $\mathbb{P}_r$ be an r-program of $\mathbb{P}$, $\left(T^M_{\mathbb{P}}\right)^{\alpha}(\triangle)$ the least fixed point of $T^M_{\mathbb{P}}$ and a query $?A$. If the maximum $\mathbb{P}_r$-computed answer for the query $?A$ with respect to $\mathbb{P}$ exists, then it is $\left(T^M_{\mathbb{P}}\right)^{\alpha}(\triangle)(A)$.*

Proof. Given an r-program $\mathbb{P}_r$ of $\mathbb{P}$, we are going to prove this result by induction over the length n of the sequence of the computation of the maximum $\mathbb{P}_r$-computed answer λ for the query $?A$ following the mechanism in Theorem 10.

If $n = 1$, then we have applied admissible rule R2 or R3.

- If the rule applied is R2, following Theorem 10, then there are neither rules nor facts with head A. Consequently, $\left(T_{\mathbb{P}}^{\mathrm{M}}\right)^{\alpha}(\triangle)(A) = \bot = \lambda$, which is the maximum $\mathbb{P}_r$-computed answer.
- If R3 is applied, since the length in the computation of λ is $n = 1$, then the canonical M-reductant in $\mathbb{P}_r^c$ is a fact, that is, $\langle A \leftarrow \top, \vartheta \rangle$. From Definition 19, only facts with head A exist in $\mathbb{P}$, that is

$$\langle A \leftarrow_{j_1} \top, \vartheta_1 \rangle, \ldots, \langle A \leftarrow_{j_l} \top, \vartheta_l \rangle$$

are all the rules with head A in $\mathbb{P}$. Therefore, the $\mathbb{P}_r^c$-computed answer is obtained from the sequence $G_0 = A$, $G_1 = \vartheta = Q_{\mu_A}\left(C_b^A\right)$, for all $b = (b_1, \ldots, b_l) \in [0,1]^l$. Finally, it only remains to prove that $\lambda = \left(T_{\mathbb{P}}^{\mathrm{M}}\right)^{\alpha}(\triangle)(A)$, which arises from the following chain of equalities.

$$\begin{aligned}
\left(T_{\mathbb{P}}^{\mathrm{M}}\right)^{\alpha}(\triangle)(A) &\overset{(1)}{=} T_{\mathbb{P}}^{M}\left(\left(T_{\mathbb{P}}^{\mathrm{M}}\right)^{\alpha}(\triangle)\right)(A) \\
&\overset{(2)}{=} Q_{\mu_A}\left(C^A_{\left(T_{\mathbb{P}}^{\mathrm{M}}\right)^{\alpha}(\triangle)}\right) \\
&= \bigvee_{i=1}^{l} C^A_{\left(T_{\mathbb{P}}^{\mathrm{M}}\right)^{\alpha}(\triangle)}\left(u_{\pi(i)}\right) \dot{\otimes} \mu_A\left(\{u_1, \ldots, u_i\}\right) \\
&= \bigvee_{i=1}^{l} \left(\vartheta_{\pi(i)} \dot{\&}_{\pi(i)} \widehat{\left(T_{\mathbb{P}}^{\mathrm{M}}\right)^{\alpha}}(\triangle)(\top)\right) \dot{\otimes} \mu_A\left(\{u_1, \ldots, u_i\}\right) \\
&= \bigvee_{i=1}^{l} \left(\vartheta_{\pi(i)} \dot{\&}_{\pi(i)} \top\right) \dot{\otimes} \mu_A\left(\{u_1, \ldots, u_i\}\right) \\
&\overset{(3)}{=} Q_{\mu_A}\left(C_b^A\right) \\
&\overset{(4)}{=} \vartheta \\
&= \lambda,
\end{aligned}$$

where (1) follows from the fact that α is an ordinal such that $\left(T_{\mathbb{P}}^{\mathrm{M}}\right)^{\alpha}(\triangle) = \mathrm{lfp}(T_{\mathbb{P}}^{\mathrm{M}})$, (2) from the definition of the $T_{\mathbb{P}}^{\mathrm{M}}$, (3) holds because of the definition of the generalized quantifier Q_{μ_A}, with $b = \left(\top, \overset{(l)}{\ldots}, \top\right)$, and (4) follows from the definition of M-reductant.

Suppose that the result is true for all length of computation less than n, with $1 < n$. Let $\langle A \leftarrow @\left(\mathcal{B}_1, \ldots, \mathcal{B}_l\right), \top \rangle$ be the canonical M-reductant for A, where $@\left(\mathcal{B}_1, \ldots, \mathcal{B}_l\right) = @'[B_1, \ldots, B_k]$, where B_i, with $i \in \{1, \ldots, k\}$, are the propositional symbols that appear in $@\left(\mathcal{B}_1, \ldots, \mathcal{B}_l\right)$. From the induction hypothesis, it is verified that $\beta_i = \left(T_{\mathbb{P}}^{\mathrm{M}}\right)^{\alpha}(\triangle)(B_i)$ is the maximum $\mathbb{P}_r$-computed answer for the query $?B_i$, for all $i \in \{1, \ldots, k\}$. Consequently, we obtain the chain of extended formulas $G_0 = A, G_1 = \top \bar{\&} \bar{@}'\left[B_1, \ldots, B_k\right], \ldots, G_r = \top \bar{\&} \bar{@}'\left[\beta_1, \ldots, \beta_k\right] = \lambda$. Therefore, by Theorem 10, we obtain that λ is the maximum $\mathbb{P}_r$-computed answer for $?A$. Moreover,

$$\begin{aligned}
\left(T_{\mathbb{P}}^{\mathrm{M}}\right)^{\alpha}(\triangle)(A) &= T_{\mathbb{P}}^{M}\left(\left(T_{\mathbb{P}}^{\mathrm{M}}\right)^{\alpha}(\triangle)\right)(A)\\
&= Q_{\mu_A}\left(C^{A}_{\left(T_{\mathbb{P}}^{\mathrm{M}}\right)^{\alpha}(\triangle)}\right)\\
&\overset{(*)}{=} \top\dot{\&}\dot{@}\left(\widehat{\left(T_{\mathbb{P}}^{\mathrm{M}}\right)^{\alpha}}(\triangle)(\mathcal{B}_1),\ldots,\widehat{\left(T_{\mathbb{P}}^{\mathrm{M}}\right)^{\alpha}}(\triangle)(\mathcal{B}_l)\right)\\
&= \top\dot{\&}\widehat{\left(T_{\mathbb{P}}^{\mathrm{M}}\right)^{\alpha}}(\triangle)\left(@\left(\mathcal{B}_1,\ldots,\mathcal{B}_l\right)\right)\\
&= \top\dot{\&}\widehat{\left(T_{\mathbb{P}}^{\mathrm{M}}\right)^{\alpha}}(\triangle)\left(@'\left[B_1,\ldots,B_k\right]\right)\\
&= \top\dot{\&}\dot{@}'\left(\left(T_{\mathbb{P}}^{\mathrm{M}}\right)^{\alpha}(\triangle)(B_1),\ldots,\left(T_{\mathbb{P}}^{\mathrm{M}}\right)^{\alpha}(\triangle)(B_k)\right)\\
&= \top\dot{\&}\dot{@}'\left[\beta_1,\ldots,\beta_k\right]\\
&= \lambda,
\end{aligned}$$

where (*) is verified due to the definition of canonical M-reductant, where the vector b is given by the interpretation of the mapping $\widehat{\left(T_{\mathbb{P}}^{\mathrm{M}}\right)^{\alpha}}(\triangle)$ through its image, that is, considering the vector $b = \left(\widehat{\left(T_{\mathbb{P}}^{\mathrm{M}}\right)^{\alpha}}(\triangle)(\mathcal{B}_1),\ldots,\widehat{\left(T_{\mathbb{P}}^{\mathrm{M}}\right)^{\alpha}}(\triangle)(\mathcal{B}_l)\right)$. We can conclude that the maximum M-computed answer is $\left(T_{\mathbb{P}}^{\mathrm{M}}\right)^{\alpha}(\triangle)(A)$.

□

Finally, the maximum M-correct and $\mathbb{P}_r$-computed answers for a query $?A$ are related.

Corollary 1. *Let $\mathbb{P}_r$ be an r-program of $\mathbb{P}$ and a query $?A$. Then, the maximum M-correct answer for $\mathbb{P}$ and $?A$ is equal to the maximum $\mathbb{P}_r$-computed answer for $\mathbb{P}$ and $?A$, if it exists.*

Proof. On the one hand, from Theorem 11, the maximum $\mathbb{P}_r$-computed answer for $?A$ is $\lambda = \left(T_{\mathbb{P}}^{\mathrm{M}}\right)^{\alpha}(\triangle)(A)$, where α is an ordinal such that $\left(T_{\mathbb{P}}^{\mathrm{M}}\right)^{\alpha}(\triangle) = \mathrm{lfp}(T_{\mathbb{P}}^{\mathrm{M}})$. On the other hand, from Theorem 6, the maximum M-correct answer for $?A$ is $\lambda_{\mathrm{MC}}^{\mathrm{M}} = \left(T_{\mathbb{P}}^{\mathrm{M}}\right)^{\alpha}(\triangle)(A)$. □

6. Conclusions and future work

In this paper, we have continued the work started in [30, 31], regarding the new definition of the immediate consequences operator considering generalized quantifiers. We have focused on the introduction of an operational semantics in this new framework. In particular, the definitions of correct answers and computed answers for a query $?A$ and a program $\mathbb{P}$ have been adapted and the relationships between them have been studied. Furthermore, a new definition of reductant has been presented to obtain a semantically equivalent program from the original one and improving the soundness of the introduced operational semantics. The program reduced by the new notion of reductant has paid a fundamental role in the proposed procedural semantics.

In addition, some completeness results have been presented in the flexible framework considered in this paper, in which the requirement of the continuity of $T_{\mathbb{P}}^{\mathrm{M}}$ has been essential in several results. Moreover, we have proved that the maximum M-correct answer and the maximum M-computed answer coincide, and can be calculated using admissible rules.

As future work, the introduced operational semantics will be applied to real datasets and more properties will be studied in the near future. Furthermore, other flexible approaches will be analyzed for complementing the one considered in this paper and others in the literature. For example, we will study the definition of the introduced quantified immediate consequences operator on general lattices and on the first order logic programming framework of the multi-adjoint setting [14].

7. Acknowledgments

Partially supported by the project PID2019-108991GB-I00 funded by MICIU/AEI/10.13039/501100011033, the project PID2022-137620NB-I00 funded by MICIU/AEI/10.13039/501100011033 and FEDER, UE, by the grant TED2021-129748B-I00 funded by MCIN/AEI/10.13039/501100011033 and European Union NextGeneration EU/PRTR

References

[1] Amine Belhadi, S.F.W., Kamble, S., Queiroz, M.M. Building supply-chain resilience: An artificial intelligence-based technique and decision-making framework, *Int. J. Prod. Res.*, V.60, N.14, 2022, pp.4487-4507.

[2] Aragón, R.G., Cornejo, M.E., Medina, J., Moreno-García, J., Ramírez-Poussa, E. Decision support system for photovoltaic fault detection avoiding meteorological conditions, *Int. J. Inf. Technol. Decis. Mak.*, V.21, N.03, 2022, pp.911-932.

[3] Cao, N., Holčapek, M., Štěpnička, M. Extensions of fuzzy relational compositions based on generalized quantifiers, *Fuzzy Sets Syst.*, V.339, 2018, pp.73-98.

[4] Castillo, O., Hammouch, Z., Atangana, A., Vargas-De-Leon, C., Allali, K. Preface of the special issue: Fuzzy logic and its application to modeling epidemies: Coronavirus and beyond (special issue), *Appl. Comput. Math.*, V.20, N.1, 2021, pp.3-4.

[5] Chovanec, A., Barták, R. History-driven template generation in inductive logic programming, *Appl. Comput. Math.*, V.11, N.2, 2012, pp.200-213.

[6] Cornejo, M.E., Díaz-Moreno, J.C., Medina, J. Generalized quantifiers in formal concept analysis, *J. Comput. Appl. Math.*, V.404, 2022, pp.113772.

[7] Cornejo, M.E., Medina, J. Impact Zadeh's theory to algebraic structures. multi-adjoint algebras, *TWMS J. Pure Appl. Math.*, V.12, N.1, 2021, pp.126-141.

[8] Costa, V., Dellunde, P., Falomir, Z. The logical style painting classifier based on Horn clauses and explanations (l-SHE), *Log. J. IGPL*, V.29, N.1, 2019, pp.96-119.

[9] Damásio, C., Medina, J., Ojeda-Aciego, M. Termination of logic programs with imperfect information: applications and query procedure, *J. Appl. Log.*, V.5, 2007, pp.435-458.

[10] Damásio, C.V., Pereira, L.M. Monotonic and residuated logic programs, In: *Symbolic and Quantitative Approaches to Reasoning with Uncertainty, ECSQARU'01*, pp.748-759, Lecture Notes in Artificial Intelligence, Springer, Berlin, V.2143, 2001, 818 p.

[11] Daz-Moreno, J.C., Medina, J., Portillo, J.R. Fuzzy logic programs as hypergraphs. Termination results, *Fuzzy Sets Syst.*, V.445, 2022, pp.22-42.

[12] Emden, M.V., Kowalski, R. The semantics of predicate logic as a programming language, *J. ACM*, V.23, N.4, 1976, pp.733-742.

[13] Hájek, P. *Metamathematics of Fuzzy Logic, Trends in Logic*, Kluwer Academic Publishers, Dordrecht, Boston and London, 1998, 307 p.

[14] Julián-Iranzo, P., Medina, J., Ojeda-Aciego, M. On reductants in the framework of multi-adjoint logic programming, *Fuzzy Sets Syst.*, V.317, 2017, pp.27-43.

[15] Julin-Iranzo, P., Moreno, G., Penabad, J. On fuzzy unfolding: A multi-adjoint approach, *Fuzzy Sets Syst.*, V.154, N.1, 2005, pp.16-33.

[16] Julin-Iranzo, P., Moreno, G., Riaza, J.A. Some properties of substitutions in the framework of similarity relations, *Fuzzy Sets Syst.*, V.465, 2023, pp.108510.

[17] Julin-Iranzo, P., Senz-Prez, F. Implementing wordnet measures of lexical semantic similarity in a fuzzy logic programming system, *Theory Pract. Log. Program.*, V.21, N.2, 2021, pp.264-282.

[18] Julin-Iranzo, P., Senz-Prez, F. Proximity-based unification: An efficient implementation method, *IEEE Trans. Fuzzy Syst.*, V.29, N.5, 2021, pp.1238-1251.

[19] Kifer, M., Subrahmanian, V.S. Theory of generalized annotated logic programming and its applications, *J. Log. Program.*, V.12, 1992, pp.335-367.

[20] Klir, G., Yuan, B. *Fuzzy sets and fuzzy logic: Theory and Applications*, Prentice Hall, New Jersey, 1995, 593 p.

[21] Krajči, S., Lencses, R., Vojtáš, P. A comparison of fuzzy and annotated logic programming, *Fuzzy Sets Syst.*, V.144, N.1, 2004, pp.173-192.

[22] Lanzinger, M., Sferrazza, S., Gottlob, G. MV-Datalog -: Effective rule-based reasoning with uncertain observations, *Theory Pract. Log. Program.*, V.22, N.5, 2022, pp.678-692.

[23] Le, V.H. Extending fuzzy linguistic logic programming with negation, *Math.*, V.10, N.17, 2022, pp.3105.

[24] Lloyd, J. *Foundations of Logic Programming*, Springer Verlag, Berlin, 1987, 214 p.

[25] López-Garza, E., Domínguez-Cruz, R. F., Martell-Chávez, F., Salgado-Tránsito, I. Fuzzy logic and linear programming-based power grid-enhanced economical dispatch for sustainable and stable grid operation in eastern Mexico, *Energies*, V.15, N.11, 2022, pp.4069.
[26] Medina, J., Ojeda-Aciego, M., Ruiz-Calviño, J. Formal concept analysis via multi-adjoint concept lattices, *Fuzzy Sets Syst.*, V.160, N.2, 2009, pp.130-144.
[27] Medina, J., Ojeda-Aciego, M., Valverde, A., Vojtáš, P. Towards biresiduated multi-adjoint logic programming, *Lect. Notes Artif. Intell.*, V.3040, 2004, pp.608-617.
[28] Medina, J., Ojeda-Aciego, M., Vojtáš, P. A procedural semantics for multi-adjoint logic programming, In: *Progress in Artificial Intelligence, EPIA'01*, pp. 290-297, Lecture Notes in Artificial Intelligence, Springer, Berlin, V.2258, 2001, 418 p.
[29] Medina, J., Ojeda-Aciego, M., Vojtáš, P. Multi-adjoint logic programming with continuous semantics, *Lect. Notes Artif. Intell.*, V.2173, 2001, pp.351-364.
[30] Medina, J., Torné-Zambrano, J.A. Fuzzy logic programming with generalized quantifiers, *Stud. Comput. Intell.*, V.1040, 2023, pp.17-23.
[31] Medina, J., Torné-Zambrano, J.A. Immediate consequences operator on generalized quantifiers, *Fuzzy Sets Syst.*, V.456, 2022, pp.72-91.
[32] Mohapatra, D., Chakraverty, S., Castillo, O. Numerical investigation of fluid dynamic model in uncertain environment, *Appl. Comput. Math.*, V.22, N.3, SI, 2023, pp.297-316.
[33] Morishita, S. *A Unified Approach to Semantics of Multi-Valued Logic Programs*, Technical Report RT 5006, IBM Tokyo, 1990.
[34] Štěpnička, M., Holčapek, M. Fuzzy relational compositions based on generalized quantifiers, In: *Information Processing and Management of Uncertainty in Knowledge-Based Systems*, pp.224-233, Communications in Computer and Information Science, Springer International Publishing, Berlin, V.442, 2014, 603 p.
[35] Valdez, F., Castillo, O., Cortes-Antonio, P., Melin, P. Applications of intelligent optimization algorithms and fuzzy logic systems in aerospace: A review. *Appl. Comput. Math.*, V.21, N.3, 2022, pp.233-245.
[36] Vojtáš, P. Fuzzy logic programming, *Fuzzy Sets Syst.*, V.124, N.3, 2001, pp.361-370.
[37] Zadeh, L. A. Fuzzy sets, *Inf. Control*, V.8, 1965, pp.338-353.



Jesus Medina - is a Full Professor at the Department of Mathematics, University of Cádiz, Spain. He received the M.Sc. degree in Mathematics in 1998 from the University of Granada, Spain, and the Ph.D. degree in Mathematics in 2002 from the University of Málaga, Spain. His is interested in the research areas of fuzzy sets, fuzzy rough sets, fuzzy logic, residuated and multiadjoint logic programming, relational data analysis, and algebraic structures for soft computing.



José Antonio Torné-Zambrano - is Ph.D. student at the Department of Mathematics, University of Cádiz, Spain, in addition to being a secondary school mathematics teacher. He received the M.Sc. degree in Mathematics in 2021, from the University of Cádiz. He is interested in the research areas of fuzzy logic, residuated and multi-adjoint logic programming, generalized quantifiers, relational data analysis, and algebraic structures for soft computing.