




Article

Relating Multi-Adjoint Normal Logic Programs to Core Fuzzy Answer Set Programs from a Semantical Approach

M. Eugenia Cornejo * , David Lobo  and Jesús Medina 

Department of Mathematics, University of Cádiz, 11510 Puerto Real, Cádiz, Spain; david.lope@uca.es (D.L.); jesus.medina@uca.es (J.M.)

* Correspondence: mariaeugenia.cornejo@uca.es (M.E.C.)

Received: 20 April 2020; Accepted: 26 May 2020; Published: 1 June 2020



Abstract: This paper relates two interesting paradigms in fuzzy logic programming from a semantical approach: core fuzzy answer set programming and multi-adjoint normal logic programming. Specifically, it is shown how core fuzzy answer set programs can be translated into multi-adjoint normal logic programs and vice versa, preserving the semantics of the starting program. This translation allows us to combine the expressiveness of multi-adjoint normal logic programming with the compactness and simplicity of the core fuzzy answer set programming language. As a consequence, theoretical properties and results which relate the answer sets to the stable models of the respective logic programming frameworks are obtained. Among others, this study enables the application of the existence theorem of stable models developed for multi-adjoint normal logic programs to ensure the existence of answer sets in core fuzzy answer set programs.

Keywords: multi-adjoint logic programming; core fuzzy answer set programming; non-monotonic logic programming; negation operator

1. Introduction

Multi-adjoint logic programming (MALP) was introduced in [1] in order to generalize different non-classical logic programming approaches [2,3]. A multi-adjoint logic program is characterized by the use of different implications in its rules and general operators in the body of its rules. These features make multi-adjoint logic programming a flexible framework with potential applications. Since its introduction, multi-adjoint logic programming has broadly been studied in order to, for example, improve the computation of the least model with either an efficient unfolding process [4,5] or with the computation of reductants [6,7]; consider propositional symbols of different sorts and termination theorems [8,9]; analyze incoherence and contradiction measures [10,11]; and extend it to a first order logic [12,13]. Later, multi-adjoint normal logic programming (MANLP) was presented as an extension of multi-adjoint logic programming, where the use of a negation operator is allowed in the body of the rules [14]. A complete study on the syntax and semantics of multi-adjoint normal logic programs, containing important results about the existence and the unicity of stable models, was carried out in [14]. Recently, extended multi-adjoint logic programming (EMALP) has been proposed with the purpose of increasing the versatility and the expressive power of the multi-adjoint approach, by means of the inclusion of different negation operators in the body of the rules and the consideration of constraint rules [15]. Besides presenting the syntax and the semantics of extended multi-adjoint logic programs, which is also based on stable models, a procedure to translate extended multi-adjoint logic programs into semantically equivalent multi-adjoint normal logic programs was provided in [15].

Core fuzzy answer set programming (CFASP) was introduced in [16] as a logic programming framework, endowed with a compact simple language, which is capable of accommodating different

fuzzy logic programming formalisms proposed in the literature, such as fuzzy logic programming [3], normal residuated logic programming [17], and fuzzy answer set programming [18]. Specifically, this framework provides a bridge between rich and expressive answer set logic languages, and a small core language that is easy to implement and reason about.

This paper is focused on relating multi-adjoint normal logic programs to core fuzzy answer set programs from a semantical perspective. This relation is carried out in both directions, from MANLP to CFASP and from CFASP to MANLP, and it pursues two main objectives. On the one hand, we aim to combine the great expressivity of EMALPs with the simplicity and compactness of CFASPs. In other words, we aspire to handle a flexible powerful language with a significant potential for modelling problems, such as EMALP, and at the same time work in a framework with high computational efficiency, easier to implement and reason about, as CFASP. That ambition is achieved presenting a method which transforms a MANLP into a CFASP such that the stable models of the former coincide with the answer sets of the later, that is, both programs are semantically equivalent. As a result, the task initiated in [15] will be completed, giving rise to a procedure to translate an EMALP into a semantically equivalent CFASP.

On the other hand, this paper also illustrates how results related to the semantics of a logic programming framework can be applied in other logic programming settings concerning its canonical models. In particular, we focus on MANLP and CFASP. For this purpose, a method for translating an arbitrary CFASP into a semantically equivalent MANLP has been shown. Among others, this method entails the possibility of using current theorems in MALP and MANLP in CFASP, such as the existence theorem for stable models given in the multi-adjoint framework [14], to provide a sufficient condition for the existence of answer sets.

The paper is organized as follows. Section 2 includes preliminary notions associated with the logic programming frameworks considered in this study, multi-adjoint normal logic programming and core fuzzy answer set programming. Section 3 presents a procedure to translate multi-adjoint normal logic programs into semantically equivalent core fuzzy answer set programs. Section 4 carries out a reciprocal study to the one given in the previous section, that is, a method to translate core fuzzy answer set programs into semantically equivalent multi-adjoint normal logic programs is shown. Technical properties and results relating stable models to answer sets of the corresponding logic programming frameworks are proven. Section 5 provides some conclusions and prospects for future work.

2. Preliminaries

In this section, we recall the syntax and the semantics of the two logic programming settings involved in this manuscript: multi-adjoint normal logic programming and core fuzzy answer set programming. We assume that the reader is familiar with the basic notions of lattice theory.

2.1. Multi-Adjoint Normal Logic Programming

Multi-adjoint normal logic programming was presented in [14] as a general non-monotonic logic programming framework, which is characterized by the use of different adjoint pairs and a negation operator. Its syntax is defined from a multi-adjoint lattice with negation.

Definition 1. *The tuple $(L, \preceq, \leftarrow_1, \&_1, \dots, \leftarrow_n, \&_n, \neg)$ is a multi-adjoint lattice with negation if the following properties are verified:*

1. (L, \preceq) is a complete lattice, with a bottom \perp and a top \top .
2. $(\&_i, \leftarrow_i)$ is an adjoint pair in (L, \preceq) , for each $i \in \{1, \dots, n\}$. That is, $\&_i$ is order-preserving in both arguments, \leftarrow_i is order-preserving in the first argument and order-reserving in the second argument, and

$$x \preceq z \leftarrow_i y \text{ if and only if } x \&_i y \preceq z$$

for all $x, y, z \in L$.

3. The boundary conditions with respect to the top element. Specifically, $\top \&_i \vartheta = \vartheta \&_i \top = \vartheta$, for all $\vartheta \in L$ and $i \in \{1, \dots, n\}$.
4. $\neg: L \rightarrow L$ is a negation operator, that is, an order-reversing mapping satisfying $\neg(\perp) = \top$ and $\neg(\top) = \perp$.

Example 1. The most usual adjoint pairs are those formed by the Gödel, product and Łukasiewicz t-norms together with their residuated implications, defined as follows, for all $x, y, z \in [0, 1]$:

$$\begin{aligned}
 x \&_G y &= \min\{x, y\} & z \leftarrow_G y &= \begin{cases} 1 & \text{if } y \leq z \\ z & \text{otherwise} \end{cases} \\
 x \&_P y &= x \cdot y & z \leftarrow_P y &= \begin{cases} 1 & \text{if } y \leq z \\ \frac{z}{y} & \text{otherwise} \end{cases} \\
 x \&_L y &= \max\{0, x + y - 1\} & z \leftarrow_L y &= \min\{1, 1 - y + z\}
 \end{aligned}$$

Remark 1. A notable consequence of the adjoint property and the boundary condition with the top element in the first argument ($\top \&_i \vartheta = \vartheta$, for all $\vartheta \in L$) is the equivalence between $y \preceq z$ and $z \leftarrow_i y = \top$.

A multi-adjoint normal logic program is composed of a set of weighted rules where different implications may appear.

Definition 2. Let $(L, \preceq, \leftarrow_1, \&_1, \dots, \leftarrow_n, \&_n, \neg)$ be a multi-adjoint lattice with negation. A multi-adjoint normal logic program (MANLP) \mathbb{P} is a finite set of weighted rules of the form:

$$\langle p \leftarrow_i @ [p_1, \dots, p_m, \neg p_{m+1}, \dots, \neg p_n]; \vartheta \rangle$$

where $i \in \{1, \dots, n\}$, $@$ is an aggregator operator (note that the notion of aggregator operator considered in [14] is just an order-preserving mapping), ϑ is an element of L and p, p_1, \dots, p_n are propositional symbols such that $p_j \neq p_k$, for all $j, k \in \{1, \dots, n\}$ with $j \neq k$. The propositional symbol p is called head of the rule, $@ [p_1, \dots, p_m, \neg p_{m+1}, \dots, \neg p_n]$ is called body of the rule and the value ϑ is its weight. The set of propositional symbols appearing in \mathbb{P} is denoted as $\Pi_{\mathbb{P}}$.

The following example presents a particular multi-adjoint normal logic program, which will be used for introducing the different notions recalled in this section.

Example 2. Let $([0, 1], \leq, \leftarrow_G, \&_G, \leftarrow_P, \&_P, \neg)$ be the multi-adjoint lattice with the standard negation, defined as $\neg(x) = 1 - x$, for each $x \in [0, 1]$. The following rules form a MANLP:

$$\begin{aligned}
 r_1 : \langle p \leftarrow_G \neg t ; 0.6 \rangle & & r_4 : \langle t \leftarrow_P s ; 1 \rangle \\
 r_2 : \langle q \leftarrow_P \neg s ; 0.8 \rangle & & r_5 : \langle s \leftarrow_P 1 ; 0.5 \rangle \\
 r_3 : \langle p \leftarrow_P q \&_P s ; 0.9 \rangle & & r_6 : \langle t \leftarrow_G \neg q \&_G \neg p ; 0.7 \rangle
 \end{aligned}$$

Similarly to normal residuated logic programs [17,19], the semantics of multi-adjoint normal logic programs follows the philosophy of the stable models semantics [20]. Next, the notion of L -interpretation is recalled.

Definition 3. Given a complete lattice (L, \preceq) , a mapping $I: \Pi_{\mathbb{P}} \rightarrow L$, which assigns a truth-value of L to each propositional symbol, is called L -interpretation. The set of all L -interpretations is denoted as $\mathcal{I}_{\mathbb{P}}$.

We will use the word interpretation instead of the term L -interpretation if there is no room for confusion. The evaluation of a formula \mathcal{F} under an interpretation I , denoted as $\hat{I}(\mathcal{F})$, proceeds inductively, until all propositional symbols in \mathcal{F} are evaluated under I . From now on, we will distinguish between an operator symbol ω and its associated operator. Specifically, the interpretation of the symbol ω will be denoted as $\dot{\omega}$.

Definition 4. Given a MANLP \mathbb{P} and an interpretation $I \in \mathcal{I}_{\Sigma}$, we say that:

- (1) A weighted rule $\langle p \leftarrow_i @ [p_1, \dots, p_m, \neg p_{m+1}, \dots, \neg p_n]; \vartheta \rangle$ is satisfied by I if and only if

$$\vartheta \preceq \hat{I}(p \leftarrow_i @ [p_1, \dots, p_m, \neg p_{m+1}, \dots, \neg p_n])$$

- (2) I is a model of \mathbb{P} if and only if all weighted rules in \mathbb{P} are satisfied by I .

Next, a specific model is given to the multi-adjoint normal logic program in Example 2.

Example 3. Let \mathbb{P} be the program defined in Example 2 and consider the interpretation M given by the pairs $M \equiv \{(p, 0.4), (q, 0.4), (s, 0.5), (t, 0.6)\}$. It is easy to see that M is a model of \mathbb{P} , that is, M satisfies all weighted rules in \mathbb{P} . For instance, the next inequality corresponds to the satisfiability of the rule r_1 :

$$0.6 \preceq \hat{M}(p \leftarrow_G \neg t) = M(p) \leftarrow_G \dot{\neg} M(t) = 0.4 \leftarrow_G 0.4 = 1$$

Similarly, the next inequalities, related to the rules r_2, \dots, r_6 , hold as well:

$$0.8 \preceq \hat{M}(q \leftarrow_P \neg s) = M(q) \leftarrow_P \dot{\neg} M(s) = 0.4 \leftarrow_P 0.5 = 0.8$$

$$0.9 \preceq \hat{M}(p \leftarrow_P q \ \&_P s) = M(p) \leftarrow_P M(q) \ \&_P M(s) = 0.4 \leftarrow_P 0.2 = 1$$

$$1 \preceq \hat{M}(t \leftarrow_P s) = M(t) \leftarrow_P M(s) = 0.6 \leftarrow_P 0.5 = 1$$

$$0.5 \preceq \hat{M}(s \leftarrow_P 1) = M(s) \leftarrow_P 1 = 0.5 \leftarrow_P 1 = 0.5$$

$$0.7 \preceq \hat{M}(t \leftarrow_G \neg q \ \&_G \neg p) = M(t) \leftarrow_G \dot{\neg} M(q) \ \&_G \dot{\neg} M(p) = 0.6 \leftarrow_G 0.6 = 1$$

In order to present the notion of the stable model, we need to recall the concept of reduct of a MANLP with respect to an interpretation. Namely, given a MANLP \mathbb{P} and an interpretation I , the reduct of \mathbb{P} with respect to I , denoted as \mathbb{P}_I , is built by substituting each rule in \mathbb{P} of the form

$$\langle p \leftarrow_i @ [p_1, \dots, p_m, \neg p_{m+1}, \dots, \neg p_n]; \vartheta \rangle$$

by the rule

$$\langle p \leftarrow_i @_I [p_1, \dots, p_m]; \vartheta \rangle$$

where the operator $@_I: L^m \rightarrow L$ is defined as

$$@_I[\vartheta_1, \dots, \vartheta_m] = @[\vartheta_1, \dots, \vartheta_m, \dot{\neg} I(p_{m+1}), \dots, \dot{\neg} I(p_n)]$$

for all $\vartheta_1, \dots, \vartheta_m \in L$.

Example 4. Coming back to the MANLP \mathbb{P} defined in Example 2 and taking into account the interpretation $M \equiv \{(p, 0.4), (q, 0.4), (s, 0.5), (t, 0.6)\}$ employed in Example 3, the reduct of \mathbb{P} with respect to M , denoted \mathbb{P}_M , is defined as:

$$\begin{array}{ll} r_1^M : \langle p \leftarrow_G 0.4 ; 0.6 \rangle & r_4^M : \langle t \leftarrow_P s ; 1 \rangle \\ r_2^M : \langle q \leftarrow_P 0.5 ; 0.8 \rangle & r_5^M : \langle s \leftarrow_P 1 ; 0.5 \rangle \\ r_3^M : \langle p \leftarrow_P q \ \&_P s ; 0.9 \rangle & r_6^M : \langle t \leftarrow_G 0.6 ; 0.7 \rangle \end{array}$$

Definition 5. Given a MANLP \mathbb{P} and an interpretation I , we say that I is a stable model of \mathbb{P} if I is the least model of \mathbb{P}_I .

Notice that, since \mathbb{P}_I is a multi-adjoint logic program without negations, there exists the least model of \mathbb{P}_I [1]. Hence, stable models are well-defined, that is, the reduct \mathbb{P}_I has a least model.

As shown in [14], the continuity of the operators in the rules of a MANLP provides a sufficient condition for the existence of stable models.

Theorem 1. Let $(K, \preceq, \leftarrow_1, \&_1, \dots, \leftarrow_n, \&_n, \neg)$ be a multi-adjoint lattice with negation, where K is a non-empty convex compact subset of an euclidean space, and \mathbb{P} be a finite MANLP defined on this lattice. If $\&_1, \dots, \&_n, \neg$ and the aggregator operators in the body of the rules of \mathbb{P} are continuous operators, then \mathbb{P} has at least a stable model.

This theorem ensures the existence of stable models of the program in Example 2.

Example 5. Clearly, the interval $[0, 1]$ is a convex compact set and $\&_G, \&_P, \neg$ are continuous operators. Hence, applying Theorem 1, we can assert that the multi-adjoint normal logic program \mathbb{P} in Example 2 has at least a stable model. For instance, the interpretation $M \equiv \{(p, 0.4), (q, 0.4), (s, 0.5), (t, 0.6)\}$ given in Example 3 is the least model of the reduct \mathbb{P}_M . Hence, by definition, M is a stable model of \mathbb{P} .

2.2. Core Fuzzy Answer Set Programming

A core language for fuzzy answer set logic programming was introduced in [16], as a simple basic framework in which one can express many of the existing fuzzy answer set programming extensions in the literature, as shown in the aforementioned paper.

In what follows, we recall the syntax and the semantics of core fuzzy answer set programs. Namely, the semantics of core fuzzy answer set programs is stated in terms of the notion of the answer set. In this framework, the elements with the role of propositional symbols are called atoms.

Definition 6 ([16]). Let \mathcal{A} be a set of atoms, (\mathcal{L}, \preceq) a complete lattice and \neg a negation operator. A core literal is either an atom $a \in \mathcal{A}$, a value from \mathcal{L} , or a formula $\neg l$, where l is a core literal.

Although atoms in core fuzzy answer set programs play the role of propositional symbols in multi-adjoint normal logic programs, in contrast to multi-adjoint normal logic programs, core literals may have the form $\neg\neg a$, being a an atom.

From now on, if there is no room for confusion, we will just use literal instead of core literal.

Definition 7. Let \mathcal{A} be a set of atoms, (\mathcal{L}, \preceq) a complete lattice and \neg a negation operator. A core fuzzy answer set program (CFASP) \mathbb{P}^c is a finite set of rules of the form:

$$a \leftarrow f(l_1, \dots, l_n)$$

being a an atom, $f: \mathcal{L}^n \rightarrow \mathcal{L}$ an order-preserving mapping in all arguments, l_1, \dots, l_n literals and \leftarrow a residuated implication. A CFASP is called simple if it only contains non-negated literals.

The element a is usually referred to as the head of the rule r , while $f(l_1, \dots, l_n)$ is called its body. In addition, the set of atoms occurring in a CFASP \mathbb{P}^c is denoted as $\mathcal{A}_{\mathbb{P}^c}$, while the set of literals is denoted as $\text{Lit}_{\mathbb{P}^c}$.

An example of the core fuzzy answer set program is given next.

Example 6. Let $*$ be the usual product operator in \mathbb{R} , \max the maximum operator, \neg the negation operator given by

$$\neg x = \begin{cases} 1 - 2x^2 & \text{if } x \leq 0.5 \\ 1 - x & \text{if } x > 0.5 \end{cases}$$

for each $x \in [0, 1]$, and \leftarrow any residuated implication. Consider the complete lattice $([0, 1], \leq)$, being \leq the usual order in \mathbb{R} , and the set $\mathcal{A}_{\mathbb{P}^c} = \{p, q, s, t, u\}$. The following five rules form a CFASP, which will be denoted as \mathbb{P}^c :

$$\begin{aligned} r_1^c &: p \leftarrow \frac{1+s}{2} \\ r_2^c &: p \leftarrow q * \neg \neg s \\ r_3^c &: q \leftarrow 0.8 \\ r_4^c &: s \leftarrow \max\{\neg q, t/2\} \\ r_5^c &: t \leftarrow \neg u \end{aligned}$$

In fact, given the mappings $f_1, f_5: [0, 1] \rightarrow [0, 1]$ and the mappings $f_2, f_4: [0, 1]^2 \rightarrow [0, 1]$ defined, for each $x, y \in [0, 1]$, as $f_1(x) = \frac{1+x}{2}$, $f_2(x, y) = x * y$, $f_4(x, y) = \max\{x, y/2\}$ and $f_5(x) = x$, we straightforwardly obtain that they all are order-preserving mappings. Therefore, \mathbb{P}^c is well-defined, that is, it is actually a CFASP.

The notion of interpretation is crucial for introducing the semantics of CFASP, as usual.

Definition 8. Given a CFASP \mathbb{P}^c on a complete lattice (\mathcal{L}, \preceq) , an interpretation is any mapping $I: \mathcal{A}_{\mathbb{P}^c} \rightarrow \mathcal{L}$.

An interpretation I is extended as usual to the set of formulas in [16]. Although this extension was also denoted by I , considering a clear abuse of notation, in order to be consistent with the notation we use in this paper, we will denote the extension of an interpretation I in CFASP as \hat{I} .

A literal $\neg l$ is evaluated under an interpretation I as $\hat{I}(\neg l) = \neg \hat{I}(l)$, whilst $\hat{I}(\alpha) = \alpha$ for each $\alpha \in \mathcal{L}$. Furthermore, the evaluation of a rule under an interpretation I is carried out extending in a natural way the interpretation of each literal appearing in the rule. That is, given a rule r of the form

$$a \leftarrow f(l_1, \dots, l_n)$$

we obtain that

$$\hat{I}(r) = I(a) \leftarrow f(\hat{I}(l_1), \dots, \hat{I}(l_n))$$

The semantics of CFASPs is based on the notion of the answer set. The first notions appearing in [16] are the definitions of satisfiability and model, which are given as usual.

Definition 9. Let \mathbb{P}^c be a CFASP. An interpretation I of \mathbb{P}^c satisfies a rule $r \in \mathbb{P}^c$ if $\hat{I}(r) = \top$. A model I of \mathbb{P}^c is any interpretation that satisfies all rules appearing in \mathbb{P}^c .

In the following example, an interpretation is presented, which satisfies different rules but it is not a model of the program in Example 6.

Example 7. Let \mathbb{P}^c be the CFASP described in Example 6 and consider the interpretation defined from the following set of pairs

$$I \equiv \{(p, 0.4), (q, 0.9), (s, 0.3), (t, 0.1), (u, 0.7)\}$$

Clearly, rule $r_3^c \in \mathbb{P}^c$ is trivially satisfied, since

$$\hat{I}(r_3^c) = I(q) \leftarrow 0.8 = 0.9 \leftarrow 0.8 = 1$$

where the last equality follows from Remark 1. Moreover, making the corresponding computations, rule $r_2^c \in \mathbb{P}^c$ is satisfied as well. The computations will be displayed with 2-digit precision. This criterion applies throughout the document.

$$\begin{aligned} \hat{I}(r_2^c) &= I(p) \leftarrow I(q) * \hat{I}(\neg\neg s) = I(p) \leftarrow I(q) * \neg\neg I(s) \\ &= 0.4 \leftarrow 0.9 * \neg\neg 0.3 = 0.4 \leftarrow 0.9 * \neg 0.82 = 0.4 \leftarrow 0.9 * 0.18 = 1 \end{aligned}$$

Nevertheless, rule $r_1^c \in \mathbb{P}^c$ is not satisfied by the interpretation I , as shown below.

$$\hat{I}(r_1^c) = I(p) \leftarrow \frac{1 + I(s)}{2} = 0.4 \leftarrow \frac{1 + 0.3}{2} = 0.4 \leftarrow 0.65 \neq 1$$

Since $\hat{I}(r_1^c) \neq 1$, rule r_1^c is not satisfied, and therefore I is not a model of the CFASP \mathbb{P}^c .

It needs to be stressed that the residuated implications appearing in the rules of a CFASP could be different. Nevertheless, as we will see next, this is meaningless from a semantical point of view. In other words, the satisfiability of a rule in a CFASP does not depend on the residuated implication. More specifically, given a rule

$$r : a \leftarrow f(l_1, \dots, l_n)$$

and an interpretation I , by definition, I satisfies r if and only if

$$I(a) \leftarrow f(\hat{I}(l_1), \dots, \hat{I}(l_n)) = \top$$

As \leftarrow is a residuated implication, the previous expression is equivalent to

$$f(\hat{I}(l_1), \dots, \hat{I}(l_n)) \preceq I(a) \tag{1}$$

Clearly, the operator \leftarrow is not involved in Equation (1), and thus it does not affect to the satisfiability of the rule r . As a consequence, we can actually make use of any residuated implication to define the corresponding CFASP of a MANLP. In particular, we can employ the same residuated implication for all rules. In what follows, the definition of the answer set for a simple CFASP is recalled.

Definition 10. Let \mathbb{P}^c be a simple CFASP. An interpretation I of \mathbb{P}^c is called an answer set of \mathbb{P}^c if I is the least model of \mathbb{P}^c .

Next, the notion of the Gelfond–Lifschitz reduct [20] is adapted into this framework.

Definition 11. Let \mathbb{P}^c be a CFASP and I an interpretation of \mathbb{P}^c . The reduct of a literal l with respect to I is defined as $l^I = l$ if $l \in \mathcal{A}_{\mathbb{P}^c} \cup \mathcal{L}$ and $l^I = \hat{I}(l)$ if l is a negated literal. The reduct of \mathbb{P}^c with respect to I is defined as the simple program \mathbb{P}_I^c built from the set of rules

$$a^I \leftarrow f(l_1^I, \dots, l_n^I)$$

where $a \leftarrow f(l_1, \dots, l_n) \in \mathbb{P}^c$.

Finally, the definition of answer set in a general CFASP is given.

Definition 12. Let \mathbb{P}^c be a CFASP. An interpretation I of \mathbb{P}^c is called an answer set of \mathbb{P}^c if I is the answer set of the reduct \mathbb{P}_I^c .

Example 8. Coming back to Example 7, the interpretation I does not satisfy the rule $r_1^c \in \mathbb{P}^c$, and therefore I is not a model of \mathbb{P}^c . Now, notice that there is no negated literal in rule r_1^c . As a consequence, its corresponding rule in the reduct \mathbb{P}_I^c is rule r_1^c itself. Hence, we can assert that I is not a model of \mathbb{P}_I^c , and thus we can conclude that I is not an answer set of \mathbb{P}^c .

3. From MANLP to CFASP

The translation procedures and results presented in this section will show that it is possible to translate a multi-adjoint normal logic program into a semantically equivalent core fuzzy answer set program.

Notice that, apart from composition of negations in CFASPs, the syntax of MANLPs and CFASPs differ in the presence or absence of weights in the rules and the effective consideration of different residuated implications in the rules. Indeed, to convert a MANLP into a CFASP, an alternative consists of including the weight of each rule in its body by means of the conductor residuated to the implication which defines the rule. This procedure is formalized as follows.

Definition 13. Let \mathbb{P} be a MANLP on $(L, \preceq, \leftarrow_1, \&_1, \dots, \leftarrow_n, \&_n, \neg)$. The corresponding CFASP \mathbb{P}^c of \mathbb{P} is defined as the following set of rules:

$$\mathbb{P}^c = \{p \leftarrow \vartheta \&_i @ [p_1, \dots, p_m, \neg p_{m+1}, \dots, \neg p_n] \mid \langle p \leftarrow_i @ [p_1, \dots, p_m, \neg p_{m+1}, \dots, \neg p_n]; \vartheta \rangle \in \mathbb{P}\}$$

where \leftarrow is any fixed implication of the set $\{\leftarrow_1, \dots, \leftarrow_n\}$.

Notice that, for each rule $p \leftarrow \vartheta \&_i @ [p_1, \dots, p_m, \neg p_{m+1}, \dots, \neg p_n] \in \mathbb{P}^c$, taking into account the syntax of multi-adjoint normal logic programs, the operators $\&_i$ and $@$ are order-preserving. As a consequence, the mapping $f: L^{n+1} \rightarrow L$ defined as

$$f(\vartheta, p_1, \dots, p_n) = \vartheta \&_i @ [p_1, \dots, p_m, \neg p_{m+1}, \dots, \neg p_n]$$

is order-preserving, being $\vartheta, p_1, \dots, p_n$ core literals. Thus, the CFASP \mathbb{P}^c is well-defined.

According to Definition 13, Algorithm 1 details stepwise the translation of a MANLP into its corresponding CFASP, where a residuated implication \leftarrow in $\{\leftarrow_1, \dots, \leftarrow_n\}$ is fixed.

Algorithm 1: Corresponding CFASP of a MANLP

```

1 H
  input :  $\mathbb{P}, \leftarrow$ 
  output: Corresponding CFASP of  $\mathbb{P}$ 
2 Define  $\mathbb{P}^c = \emptyset$ ;
3 for each  $\langle p \leftarrow_i @ [p_1, \dots, p_m, \neg p_{m+1}, \dots, \neg p_n]; \vartheta \rangle \in \mathbb{P}$  do
4    $\lfloor$  add  $p \leftarrow \vartheta \&_i @ [p_1, \dots, p_m, \neg p_{m+1}, \dots, \neg p_n]$  to  $\mathbb{P}^c$ 
5 return  $\mathbb{P}^c$ 

```

As shown next, the corresponding CFASP of a MANLP, given by Definition 13, is semantically equivalent to the original program. In other words, the answer sets of the former coincide with the stable models of the latter.

Theorem 2. Let \mathbb{P} be a MANLP. An interpretation I is a stable model of \mathbb{P} if and only if I is an answer set of the corresponding CFASP \mathbb{P}^c of \mathbb{P} .

Proof. Let I be an interpretation. According to the definitions of stable model and answer set, we need to prove that I is the least model of the reduct \mathbb{P}_I if and only if I is an answer set of the reduct \mathbb{P}_I^c . Notice that, to prove the previous statement, it is sufficient to demonstrate that any interpretation J is a model of \mathbb{P}_I if and only if J is a model of \mathbb{P}_I^c , since this fact implies that the models of \mathbb{P}_I coincide with the models of \mathbb{P}_I^c .

By definition, for each rule in \mathbb{P} of the form

$$\langle p \leftarrow_i @ [p_1, \dots, p_m, \neg p_{m+1}, \dots, \neg p_n]; \vartheta \rangle$$

there exists a rule in the reduct \mathbb{P}_I given by

$$\langle p \leftarrow_i @_I[p_1, \dots, p_m]; \vartheta \rangle$$

with $@_I[\vartheta_1, \dots, \vartheta_m] = \dot{@}[\vartheta_1, \dots, \vartheta_m, \dot{\leftarrow} I(p_{m+1}), \dots, \dot{\leftarrow} I(p_n)]$, and a rule in the reduct \mathbb{P}_I^c of the form

$$p^I \leftarrow \vartheta^I \&_i @_I[p_1^I, \dots, p_m^I, (\neg p_{m+1})^I, \dots, (\neg p_n)^I] \tag{2}$$

that is

$$p \leftarrow \vartheta \&_i @_I[p_1, \dots, p_m, \neg I(p_{m+1}), \dots, \neg I(p_n)] \tag{3}$$

Furthermore, all rules in \mathbb{P}_I and \mathbb{P}_I^c are of that form. Notice that, an interpretation J satisfies a rule in \mathbb{P}_I of the form

$$\langle p \leftarrow_i @_I[p_1, \dots, p_m]; \vartheta \rangle$$

if and only if

$$\vartheta \preceq J(p \leftarrow_i @_I[p_1, \dots, p_m])$$

Equivalently, according to the definition of $@_I$,

$$\vartheta \preceq J(p) \dot{\leftarrow}_i \dot{@}[J(p_1), \dots, J(p_m), \dot{\leftarrow} I(p_{m+1}), \dots, \dot{\leftarrow} I(p_n)]$$

Since $(\&_i, \dot{\leftarrow}_i)$ forms an adjoint pair, the previous inequality can be rewritten as

$$\vartheta \dot{\&}_i \dot{@}[J(p_1), \dots, J(p_m), \dot{\leftarrow} I(p_{m+1}), \dots, \dot{\leftarrow} I(p_n)] \preceq J(p)$$

which is equivalent by Remark 1 to

$$\top = J(p) \dot{\leftarrow} \vartheta \dot{\&}_i \dot{@}[J(p_1), \dots, J(p_m), \dot{\leftarrow} I(p_{m+1}), \dots, \dot{\leftarrow} I(p_n)] \tag{4}$$

In other words, J satisfies the rule

$$\langle p \leftarrow_i @_I[p_1, \dots, p_m]; \vartheta \rangle$$

if and only if it satisfies the rule

$$p^I \leftarrow \vartheta^I \&_i @_I[p_1^I, \dots, p_m^I, (\neg p_{m+1})^I, \dots, (\neg p_n)^I] \tag{5}$$

As a consequence, an interpretation I is the least model of the reduct \mathbb{P}_I if and only if I is an answer set of the reduct \mathbb{P}_I^c . \square

Recently, extended multi-adjoint logic programs (EMALPs) were presented in [15] as an extension of multi-adjoint normal logic programs. In this setting, a special type of aggregator operator, called extended aggregator, is considered in the body of the rules and a new kind of rules, called constraints, have been included in the programs.

As shown in [15], extended aggregators can be used to simulate multiple negation operators or, in general, any kind of order-reversing behaviour for a propositional symbol. Additionally, the consideration of constraints enables a user to impose upper bounds to certain formulae. This shed lights on the flexibility and the expressive power of the extended multi-adjoint logic programming framework. Besides presenting the syntax and the semantics of extended multi-adjoint logic programs, a procedure to translate an EMALP into a semantically equivalent MANLP was provided in [15].

As a result, we can assert that Definition 13 together with Theorem 2 completes the labour initiated in [15]. Namely, we can make use of extended multi-adjoint logic programs in order to model real-world problems, and then translate them into CFASPs to handle compact simple programs with the same meaning. This finished translation provides certain advantageous properties. For

instance, from a computational point of view, CFASPs are easier to implement and to reason about, as highlighted in [16].

The next example illustrates how Definition 13 is employed to translate a MANLP into a CFASP, taking into account Algorithm 1. Specifically, we conclude the transformation started in Examples 16, 21 and 26 in [15].

Example 9. Let $([0, 1], \leq, \leftarrow_G, \&_G, \leftarrow_P, \&_P, \leftarrow_L, \&_L)$ be the multi-adjoint lattice where $(\&_G, \leftarrow_G)$, $(\&_P, \leftarrow_P)$ and $(\&_L, \leftarrow_L)$ are Gödel, product and Łukasiewicz adjoint pairs, respectively, and $\neg_1, \neg_2: [0, 1] \rightarrow [0, 1]$ the negation operators defined as $\neg_1(x) = 1 - x$ and $\neg_2(x) = (1 - x^2)^{1/2}$, for all $x \in [0, 1]$. Consider the MANLP \mathbb{P} given by

- $r_1: \langle p \leftarrow_P \min \left\{ \frac{q}{\neg_1(not_s) + \neg_1(not_t) + 0.1}, 1 \right\}; 0.5 \rangle$
- $r_2: \langle q \leftarrow_P \max \left\{ \neg_1(\neg_1(not_s)), \neg_2(\neg_1(not_t)) \right\}; 0.6 \rangle$
- $r_3: \langle p_0 \leftarrow_L f_0(\neg_1(\neg_1(not_{p_0}))) \&_G f_{0.7}(\neg_1(\neg_1(not_q))) \rangle; 1$
- $r_4: \langle s \leftarrow_G 1; 0.8 \rangle$
- $r_5: \langle t \leftarrow_G \max\{s, 0.7\}; 0.8 \rangle$
- $r_6: \langle not_q \leftarrow_G \neg_1 q; 1 \rangle$
- $r_7: \langle not_s \leftarrow_G \neg_1 s; 1 \rangle$
- $r_8: \langle not_t \leftarrow_G \neg_1 t; 1 \rangle$
- $r_9: \langle not_{p_0} \leftarrow_G \neg_1 p_0; 1 \rangle$

where $f_c: [0, 1] \rightarrow [0, 1]$ is defined, for each $c \in [0, 1]$, as

$$f_c(x) = \begin{cases} 0 & \text{if } x \leq c \\ 1 & \text{otherwise} \end{cases}$$

Applying Algorithm 1, consider fixed the implication \leftarrow_G and let $\mathbb{P}^c = \emptyset$. For the rule $r_1 \in \mathbb{P}$, that is, the rule

$$\langle p \leftarrow_P \min \left\{ \frac{q}{\neg_1(not_s) + \neg_1(not_t) + 0.1}, 1 \right\}; 0.5 \rangle$$

we include in \mathbb{P}^c the rule

$$p \leftarrow_G 0.5 \&_P \min \left\{ \frac{q}{\neg_1(not_s) + \neg_1(not_t) + 0.1}, 1 \right\}$$

Similarly, the rule $r_2 \in \mathbb{P}$ is transformed into the rule

$$q \leftarrow_G 0.6 \&_P \max \left\{ \neg_1(\neg_1(not_s)), \neg_2(\neg_1(not_t)) \right\}$$

Following this process for the rest of rules of \mathbb{P} , we conclude that its corresponding CFASP \mathbb{P}^c is defined as the set of rules:

- $r_1^c: p \leftarrow_G 0.5 \&_P \min \left\{ \frac{q}{\neg_1(not_s) + \neg_1(not_t) + 0.1}, 1 \right\}$
- $r_2^c: q \leftarrow_G 0.6 \&_P \max \left\{ \neg_1(\neg_1(not_s)), \neg_2(\neg_1(not_t)) \right\}$
- $r_3^c: p_0 \leftarrow_G f_0(\neg_1(\neg_1(not_{p_0}))) \&_G f_{0.7}(\neg_1(\neg_1(not_q)))$
- $r_4^c: s \leftarrow_G 0.8$
- $r_5^c: t \leftarrow_G 0.8 \&_G \max\{s, 0.7\}$
- $r_6^c: not_q \leftarrow_G \neg_1 q$
- $r_7^c: not_s \leftarrow_G \neg_1 s$
- $r_8^c: not_t \leftarrow_G \neg_1 t$
- $r_9^c: not_{p_0} \leftarrow_G \neg_1 p_0$

Notice that, the CFASP \mathbb{P}^c is clearly simpler than the MANLP \mathbb{P} , from a syntactical point of view.

Applying Theorem 2, we can assert that \mathbb{P}^c is semantically equivalent to \mathbb{P} . For instance, as shown in Example 26 in [15], $N \equiv \{(p, 9/85), (q, 0.36), (s, 0.8), (t, 0.8), (p_0, 0), (not_q, 0.64), (not_s, 0.2), (not_t, 0.2), (not_{p_0}, 1)\}$ is a stable model of the MANLP \mathbb{P} , from which we conclude that N is also an answer set of \mathbb{P}^c .

4. From Core Fuzzy Answer Set Programs to Multi-Adjoint Normal Logic Programs

The semantics of core fuzzy answer set programs is defined in terms of answer sets. Sufficient conditions to ensure the existence of answer sets are then instrumental in order to define the semantics of a CFASP. In this section, we provide a method to transform a CFASP into a semantically equivalent MANLP. One of the consequences of that procedure is the possibility of applying different results given in MALP and MANLP, such as the termination results introduced in [8,9] or Theorem 1 to guarantee the existence of answer sets.

Notice that, there are two requirements to translate a CFASP into a MANLP:

- (i) The mappings in the body of the rules must be aggregator and/or negation operators. However, this is straightforwardly verified, according to the syntax of CFASPs.
- (ii) Composition of negations, that is, literals of the form $\neg\neg\neg b$, are not allowed in MANLPs. In order to deal with this, in what follows, we devise a procedure to transform composited negations into a single negation.

Consider a rule r of the form $a \leftarrow f(l_1, \dots, l_j, \dots, l_n)$ with $l_j = \neg\neg\neg b$. The idea of the proposed method is introducing three new atoms (or propositional symbols) not_b^1 , not_b^2 and not_b^3 in order to represent the information given by l_j :

- not_b^1 is equivalent to $\neg b$
- not_b^2 is equivalent to $\neg\neg b$, and thus to $\neg not_b^1$
- not_b^3 is equivalent to $\neg\neg\neg b$, and thus to $\neg not_b^2$

Notice that, the three previous statements can be modelled by the rules

$$\begin{aligned} r_1 : not_b^1 &\leftarrow \neg b \\ r_2 : not_b^2 &\leftarrow \neg not_b^1 \\ r_3 : not_b^3 &\leftarrow \neg not_b^2 \end{aligned}$$

respectively. Hence, the rule r could be replaced by the rule

$$a \leftarrow f(l_1, \dots, not_b^3, \dots, l_n)$$

together with rules r_1 , r_2 and r_3 .

In order to formalize the preceding approach, we will fix some notation. First and foremost, notice that we can assume without loss of generality that, for each rule $a \leftarrow f(l_1, \dots, l_n)$ in a CFASP \mathbb{P}^c and $j \in \{1, \dots, n\}$, l_j is either an atom or a negated literal. Otherwise, if $l_j \in \mathcal{L}$, then we consider the rule $a \leftarrow f_{l_j}(l_1, \dots, l_{j-1}, l_{j+1}, \dots, l_n)$ where $f_{l_j}(l_1, \dots, l_{j-1}, l_{j+1}, \dots, l_n) = f(l_1, \dots, l_n)$.

Now, given a CFASP \mathbb{P}^c and $b \in \mathcal{A}_{\mathbb{P}^c}$, we say that the *degree* of b is the highest non-negative integer k such that $\neg^k b$ appears in the body of some rule of \mathbb{P}^c , being \neg^k the k -th composition of the operator \neg . From now on, the set of atoms of \mathbb{P}^c with degree $k \in \mathbb{Z}^*$ will be denoted as $\mathcal{N}_{\mathbb{P}^c}^k$.

Once the required notation has been introduced, the corresponding MANLP of a CFASP can formally be defined.

Definition 14. Let \mathbb{P}^c be a CFASP defined with a residuated implication \leftarrow . The corresponding MANLP \mathbb{P} of \mathbb{P}^c is defined on the multi-adjoint lattice with negation $(\mathcal{L}, \preceq, \leftarrow, \&, \neg)$, being $(\&, \leftarrow)$ an adjoint pair, as the following set of rules:

$$\begin{aligned} \mathbb{P} &= \{ \langle a \leftarrow @ [p_1, \dots, p_n]; \top \rangle \mid a \leftarrow f(l_1, \dots, l_n) \in \mathbb{P}^c \} \\ &\cup \{ \langle \text{not}_b^1 \leftarrow \neg b; \top \rangle \mid b \in \mathcal{N}_{\mathbb{P}^c}^k, k \geq 1 \} \\ &\cup \{ \langle \text{not}_b^h \leftarrow \neg \text{not}_b^{h-1}; \top \rangle \mid b \in \mathcal{N}_{\mathbb{P}^c}^k, k \geq 2, h \in \{2, \dots, k\} \} \end{aligned}$$

where the operator $@$ coincides with f and

$$p_j = \begin{cases} l_j & \text{if } l_j \in \mathcal{A}_{\mathbb{P}^c} \\ \text{not}_b^h & \text{if } l_j = \neg^h b, b \in \mathcal{A}_{\mathbb{P}^c}, h \geq 1 \end{cases}$$

for each $j \in \{1, \dots, n\}$.

It is important to highlight that, given a rule $a \leftarrow f(l_1, \dots, l_n) \in \mathbb{P}^c$, since f is an aggregator operator, the corresponding operator $@$ is also an aggregator. Furthermore, according to the syntax of CFASPs, \leftarrow is a residuated implication, and thus there exists an operator $\&$ such that $(\&, \leftarrow)$ is an adjoint pair. Hence, the program \mathbb{P} is well-defined, that is, \mathbb{P} is a MANLP. Notice that, when a CFASP is simple (Definition 7), then the obtained multi-adjoint program is a MALP.

Algorithm 2 shows how Definition 14 is applied in order to compute the corresponding MANLP of a CFASP.

Algorithm 2: Corresponding MANLP of a CFASP

input : \mathbb{P}^c
output: Corresponding MANLP of \mathbb{P}^c

- 1 Compute the sets $\mathcal{N}_{\mathbb{P}^c}^k, k \geq 1$;
- 2 Define $\mathbb{P} = \emptyset$;
- 3 **for each** $a \leftarrow f(l_1, \dots, l_n) \in \mathbb{P}^c$ **do**
- 4 \lfloor add $\langle a \leftarrow @ [p_1, \dots, p_n]; \top \rangle$ to \mathbb{P} , where $@, p_1, \dots, p_n$ are defined according to Definition 14
- 5 **for each** $b \in \mathcal{N}_{\mathbb{P}^c}^k, k \geq 1$ **do**
- 6 \lfloor add $\langle \text{not}_b^1 \leftarrow \neg b; \top \rangle$ to \mathbb{P}
- 7 **for each** $b \in \mathcal{N}_{\mathbb{P}^c}^k, k \geq 2, h \in \{2, \dots, k\}$ **do**
- 8 \lfloor add $\langle \text{not}_b^h \leftarrow \neg \text{not}_b^{h-1}; \top \rangle$ to \mathbb{P}
- 9 **return** \mathbb{P}

Example 10. Consider the CFASP \mathbb{P}^c given in Example 6, consisting of the rules

$$\begin{aligned} r_1^c &: p \leftarrow \frac{1+s}{2} \\ r_2^c &: p \leftarrow q * \neg s \\ r_3^c &: q \leftarrow 0.8 \\ r_4^c &: s \leftarrow \max\{-q, t/2\} \\ r_5^c &: t \leftarrow \neg u \end{aligned}$$

In what follows, we compute the corresponding MANLP of \mathbb{P}^c by means of Algorithm 2. Notice that, the degree of p and t is 0, the degree of q and u is 1 and the degree of s is 2. Hence, by definition, $\mathcal{N}_{\mathbb{P}^c}^1 = \{q, u\}$ and $\mathcal{N}_{\mathbb{P}^c}^2 = \{s\}$.

Let $\mathbb{P} = \emptyset$. According to lines 3 and 4, the rule $r_1^c \in \mathbb{P}^c$ is included in \mathbb{P} as

$$\langle p \leftarrow \frac{1+s}{2}; 1 \rangle$$

In what regards the rule $r_2^c \in \mathbb{P}^c$, it is included in \mathbb{P} as

$$\langle p \leftarrow q * \text{not}_s^2; 1 \rangle$$

Similarly, the rules $r_3^c, r_4^c, r_5^c \in \mathbb{P}^c$ are adapted to be added to \mathbb{P} .

Concerning lines 5 and 6, as $\mathcal{N}_{\mathbb{P}^c}^1 = \{q, u\}$ and $\mathcal{N}_{\mathbb{P}^c}^2 = \{s\}$, the next rules are added to \mathbb{P} :

$$\begin{aligned} &\langle \text{not}_s^1 \leftarrow \neg s; 1 \rangle \\ &\langle \text{not}_q^1 \leftarrow \neg q; 1 \rangle \\ &\langle \text{not}_u^1 \leftarrow \neg u; 1 \rangle \end{aligned}$$

Finally, applying lines 7 and 8, we conclude adding into \mathbb{P} the rule

$$\langle \text{not}_s^2 \leftarrow \neg \text{not}_s^1; 1 \rangle$$

Therefore, the corresponding MANLP of \mathbb{P}^c is defined on the multi-adjoint lattice with negation $([0, 1], \leq, \leftarrow, \&, \neg)$, and consists of the following rules:

$$\begin{aligned} r_1 : \langle p \leftarrow \frac{1+s}{2}; 1 \rangle & \quad r_s^1 : \langle \text{not}_s^1 \leftarrow \neg s; 1 \rangle \\ r_2 : \langle p \leftarrow q * \text{not}_s^2; 1 \rangle & \quad r_s^2 : \langle \text{not}_s^2 \leftarrow \neg \text{not}_s^1; 1 \rangle \\ r_3 : \langle q \leftarrow 0.8; 1 \rangle & \quad r_q : \langle \text{not}_q^1 \leftarrow \neg q; 1 \rangle \\ r_4 : \langle s \leftarrow \max\{\text{not}_q^1, t/2\}; 1 \rangle & \quad r_u : \langle \text{not}_u^1 \leftarrow \neg u; 1 \rangle \\ r_5 : \langle t \leftarrow \text{not}_u^1; 1 \rangle & \end{aligned}$$

Now, we will present a technical result which will be useful in order to show the relationship between the answer sets of a CFASP \mathbb{P}^c and the stable models of its corresponding MANLP \mathbb{P} .

Lemma 1. Let \mathbb{P}^c be a CFASP, \mathbb{P} the corresponding MANLP of \mathbb{P}^c , $N^c, M^c : \mathcal{A}_{\mathbb{P}^c} \rightarrow \mathcal{L}$ two interpretations and we define $N_M : \Pi_{\mathbb{P}} \rightarrow \mathcal{L}$ as $N_M(b) = N^c(b)$ if $b \in \mathcal{A}_{\mathbb{P}^c}$ and $N_M(\text{not}_b^h) = \neg^h M^c(b)$ for each $b \in \mathcal{N}_{\mathbb{P}^c}^k, k \geq 1, h \in \{1, \dots, k\}$. Then, N^c is a model of the reduct $\mathbb{P}_{M^c}^c$ if and only if N_M is a model of the reduct \mathbb{P}_M , where M denotes the interpretation M_M .

Proof. Taking into account that $(\&, \leftarrow)$ forms an adjoint pair, the following statements hold:

- (i) N_M satisfies the rule $\langle \text{not}_b^1 \leftarrow \neg M(b); \top \rangle$ in the reduct \mathbb{P}_M if and only if $\neg M(b) \preceq N_M(\text{not}_b^1)$.
- (ii) N_M satisfies a rule of the form $\langle \text{not}_b^h \leftarrow \neg M(\text{not}_b^{h-1}); \top \rangle$ in the reduct \mathbb{P}_M if and only if $\neg M(\text{not}_b^{h-1}) \preceq N_M(\text{not}_b^h)$.

Notice that, the equalities $M(\text{not}_b^1) = \neg M(b)$ and $M(\text{not}_b^h) = \neg^h M(b) = \neg(\neg^{h-1} M(b)) = \neg M(\text{not}_b^{h-1})$ are satisfied. As a consequence, by definition of N_M and M , we obtain that N_M straightforwardly satisfies all rules in \mathbb{P}_M with head not_b^h , with $b \in \mathcal{N}_{\mathbb{P}^c}^k, k \geq 1, h \in \{1, \dots, k\}$.

Now, note that a rule $a \leftarrow f(l_1, \dots, l_n)$ belongs to \mathbb{P}^c if and only if the rule $\langle a \leftarrow @[p_1, \dots, p_n]; \top \rangle$ belongs to \mathbb{P} . Since every p_j is a “positive” propositional symbol, with $j \in \{1, \dots, n\}$, we can assert that the rule r^c given by

$$r^c : \quad a \leftarrow f(I_1^{M^c}, \dots, I_n^{M^c})$$

is in the reduct $\mathbb{P}_{M^c}^c$ if and only if the rule r defined as

$$r : \quad \langle a \leftarrow @[p_1, \dots, p_n]; \top \rangle$$

belongs to \mathbb{P}_M . In what follows, we show that N^c satisfies the rule r^c if and only if N_M satisfies the rule r . Clearly, N^c satisfies r^c if and only if $N^c(f(l_1^{M^c}, \dots, l_n^{M^c})) \preceq N^c(a)$, or equivalently

$$f(N^c(l_1^{M^c}), \dots, N^c(l_n^{M^c})) \preceq N^c(a) \tag{6}$$

On the other hand, N_M satisfies r if and only if $N_M(@[p_1, \dots, p_n]) \preceq N_M(a)$, i.e.,

$$\dot{@}[N_M(p_1), \dots, N_M(p_n)] \preceq N_M(a) \tag{7}$$

We will see that Equations (6) and (7) are identical. Indeed, as $N_M(b) = N^c(b)$ for each $b \in \mathcal{A}_{\mathbb{P}^c}$, the right-hand side of both inequalities coincide.

Now, given $j \in \{1, \dots, n\}$, suppose that $l_j \in \mathcal{A}_{\mathbb{P}^c}$. Then $l_j^{M^c} = l_j$ and $p_j = l_j$, from which $N_M(p_j) = N_M(l_j) = N^c(l_j) = N^c(l_j^{M^c})$. On the contrary, assume that $l_j = \neg^h b$ with $b \in \mathcal{A}_{\mathbb{P}^c}$ and $1 \leq h$. In that case, $l_j^{M^c} = M^c(l_j)$ and $p_j = \text{not}_b^h$. Hence, the following chain of equalities hold:

$$\begin{aligned} N_M(p_j) &= N_M(\text{not}_b^h) = \neg^h M(b) = \neg^h M^c(b) \stackrel{(\dagger)}{=} \neg^h N^c(M^c(b)) \\ &= N^c(\neg^h M^c(b)) = N^c(M^c(\neg^h b)) = N^c(M^c(l_j)) = N^c(l_j^{M^c}) \end{aligned}$$

(\dagger) Note that $M^c(b) \in \mathcal{L}$, and thus $N^c(M^c(b)) = M^c(b)$.

As a result, we conclude that $N_M(p_j) = N^c(l_j^{M^c})$, for each $j \in \{1, \dots, n\}$. Since $\dot{@} = f$, Equations (6) and (7) coincide, as we want to demonstrate. Hence, we obtain then that N^c is a model of $\mathbb{P}_{M^c}^c$ if and only if N_M is a model of \mathbb{P}_M . \square

The following result shows that the answer sets of a CFASP \mathbb{P}^c are associated with a family of stable models of its corresponding MANLP \mathbb{P} .

Theorem 3. *Let \mathbb{P}^c be a CFASP, \mathbb{P} the corresponding MANLP of \mathbb{P}^c , $M^c: \mathcal{A}_{\mathbb{P}^c} \rightarrow \mathcal{L}$ an interpretation and $M: \Pi_{\mathbb{P}} \rightarrow \mathcal{L}$ given by $M(b) = M^c(b)$ if $b \in \mathcal{A}_{\mathbb{P}^c}$ and $M(\text{not}_b^h) = \neg^h M(b)$ for each $b \in \mathcal{N}_{\mathbb{P}^c}^k, k \geq 1, h \in \{1, \dots, k\}$. Then, M^c is an answer set of \mathbb{P}^c if and only if M is a stable model of \mathbb{P} .*

Proof. By Lemma 1, we straightforwardly obtain that M^c is a model of $\mathbb{P}_{M^c}^c$ if and only if M is a model of \mathbb{P}_M . It remains to demonstrate that M^c is the least model of $\mathbb{P}_{M^c}^c$ if and only if M is the least model of \mathbb{P}_M . We will proceed by reductio ad absurdum. Suppose that M is the least model of \mathbb{P}_M but there exists a model $N^c: \mathcal{A}_{\mathbb{P}^c} \rightarrow \mathcal{L}$ of $\mathbb{P}_{M^c}^c$ such that $N^c \prec M^c$, that is, $N^c(b) \preceq M^c(b)$ for each $b \in \mathcal{A}_{\mathbb{P}^c}$ and there exists $a \in \mathcal{A}_{\mathbb{P}^c}$ such that $N^c(a) \prec M^c(a)$. According to Statement (1), the interpretation N_M is then a model of \mathbb{P}_M . By definition of N_M and M , we obtain $N_M(b) = N^c(b) \preceq M^c(b) = M(b)$, for each $b \in \mathcal{A}_{\mathbb{P}^c}$, and $N_M(\text{not}_b^h) = \neg^h M(b) = M(\text{not}_b^h)$, for each $b \in \mathcal{N}_{\mathbb{P}^c}^k, k \geq 1, h \in \{1, \dots, k\}$. Furthermore, $N_M(a) = N^c(a) \prec M^c(a) = M(a)$. Therefore $N_M \prec M$, in contradiction with the hypothesis, since M is the least model of \mathbb{P}_M .

Suppose now that M^c is the least model of $\mathbb{P}_{M^c}^c$ but there exists a model $N: \Pi_{\mathbb{P}} \rightarrow \mathcal{L}$ of \mathbb{P}_M such that $N \prec M$. Hence, we can consider the interpretation $N^c: \mathcal{A}_{\mathbb{P}^c} \rightarrow \mathcal{L}$ defined as $N^c(b) = N(b)$, for each $b \in \mathcal{A}_{\mathbb{P}^c}$. Clearly, if $N(\text{not}_b^h) \prec M(\text{not}_b^h)$, for some $b \in \mathcal{N}_{\mathbb{P}^c}^k, k \geq 1, h \in \{1, \dots, k\}$, then N does not satisfy the rule $\langle \text{not}_b^h \leftarrow \neg M(\text{not}_b^{h-1}); \top \rangle$ in the reduct \mathbb{P}_M . Therefore, we can assert that there exists $a \in \mathcal{A}_{\mathbb{P}^c}$ such that $N(a) \prec M(a)$ and so, $N^c \prec M^c$. Now, we consider the interpretation $N_M^c: \Pi_{\mathbb{P}} \rightarrow \mathcal{L}$ defined as $N_M^c(b) = N(b)$, for each $b \in \mathcal{A}_{\mathbb{P}^c}$ and $N_M^c(\text{not}_b^h) = \neg^h M(b)$, for all $b \in \mathcal{N}_{\mathbb{P}^c}^k, k \geq 1, h \in \{1, \dots, k\}$. Since N is a model of \mathbb{P}_M , then N_M^c satisfies all rules in \mathbb{P}_M and so, N_M^c is also a model of \mathbb{P}_M . Thus, by Lemma 1, we obtain that N^c is a model of $\mathbb{P}_{M^c}^c$, contradicting the fact that M^c is the least model of $\mathbb{P}_{M^c}^c$. \square

The subsequent theorem completes the foundations of the equivalence between the semantics of a CFASP \mathbb{P}^c and its corresponding MANLP \mathbb{P} . Specifically, it states that the evaluation of not_b^h under

any stable model M of \mathbb{P} is equal to $\neg^h M(b)$. As a result, we conclude that Theorem 3 covers all stable models of \mathbb{P} , and thus \mathbb{P}^c and \mathbb{P} are equivalent, from a semantical point of view.

Theorem 4. *Let \mathbb{P}^c be a CFASP and \mathbb{P} the corresponding MANLP of \mathbb{P}^c . Any stable model M of \mathbb{P} satisfies $M(\text{not}_b^h) = \neg^h M(b)$, for each $b \in \mathcal{N}_{\mathbb{P}^c}^k, k \geq 1, h \in \{1, \dots, k\}$.*

Proof. Let M be a stable model of \mathbb{P} , i.e., the least model of the reduct \mathbb{P}_M . We will proceed by induction on h .

Base case: We show that $M(\text{not}_b^1) = \neg M(b)$, for each $b \in \mathcal{N}_{\mathbb{P}^c}^k, k \geq 1$.

Since M is a model of the reduct \mathbb{P}_M , M satisfies the rule $\langle \text{not}_b^1 \leftarrow \neg M(b); \top \rangle$ in \mathbb{P}_M , and therefore $\neg M(b) \preceq M(\text{not}_b^1)$. Furthermore, since M is actually the least model of \mathbb{P}_M and $\langle \text{not}_b^1 \leftarrow \neg M(b); \top \rangle$ is the unique rule with head not_b^1 in \mathbb{P}_M , we conclude that $M(\text{not}_b^1) = \neg M(b)$, for each $b \in \mathcal{N}_{\mathbb{P}^c}^k, k \geq 1$.

Inductive step: We assume $M(\text{not}_b^{h-1}) = \neg^{h-1} M(b)$ is satisfied, for some $b \in \mathcal{N}_{\mathbb{P}^c}^k, k \geq 2, h \in \{2, \dots, k\}$, and we show that $M(\text{not}_b^h) = \neg^h M(b)$ holds.

By an analogous reasoning to the base case, M satisfies the rule $\langle \text{not}_b^h \leftarrow \neg M(\text{not}_b^{h-1}); \top \rangle$ in \mathbb{P}_M , from which $\neg M(\text{not}_b^{h-1}) \preceq M(\text{not}_b^h)$. Again, as M is the least model of \mathbb{P}_M and $\langle \text{not}_b^h \leftarrow \neg M(\text{not}_b^{h-1}); \top \rangle$ is the unique rule with head not_b^h in \mathbb{P}_M , we obtain that $\neg M(\text{not}_b^{h-1})$ must be equal to $M(\text{not}_b^h)$. Now, taking into account the induction hypothesis, we deduce the required equality:

$$M(\text{not}_b^h) = \neg M(\text{not}_b^{h-1}) = \neg \left(\neg^{h-1} M(b) \right) = \neg^h M(b)$$

which finishes the proof. \square

As a consequence of Theorems 3 and 4, given a CFASP \mathbb{P}^c and its corresponding MANLP \mathbb{P} , the number of answer sets of \mathbb{P}^c coincides with the number of stable models of \mathbb{P} .

Corollary 1. *Let \mathbb{P}^c be a CFASP and \mathbb{P} its corresponding MANLP. Then, there exists an answer set of \mathbb{P}^c if and only if there exists a stable model of \mathbb{P} .*

Proof. It straightforwardly follows from Theorems 3 and 4. \square

Corollary 1 leads us to assert that, if one guarantees the existence of a stable model of \mathbb{P} , then the existence of an answer set of \mathbb{P}^c is ensured. As a result, Theorem 1 can be used to provide a sufficient condition for the existence of answer sets of a CFASP. In particular, the following result is obtained.

Corollary 2. *Let \mathbb{P}^c be a CFASP. If the order-preserving mappings and the negation operator involved in the rules of \mathbb{P}^c are continuous operators, then there exists at least an answer set of \mathbb{P}^c .*

Proof. Assume that the operators in the rules of \mathbb{P}^c are continuous. Taking into account the definition of the corresponding MANLP \mathbb{P} of \mathbb{P}^c and Theorem 1, we deduce that \mathbb{P} has at least a stable model. By Corollary 1, we conclude that \mathbb{P}^c has at least an answer set. \square

This section concludes with an example in order to illustrate Corollary 2 and Theorems 3 and 4. More precisely, we retrieve the CFASP \mathbb{P}^c introduced in Example 6 and we ensure the existence of at least an answer set of \mathbb{P}^c . Then, we construct a stable model of its corresponding MANLP \mathbb{P} and we translate it into an answer set of \mathbb{P}^c .

Example 11. *In Examples 6 and 10, the negation operator \neg in the CFASP \mathbb{P}^c is clearly continuous. Furthermore, f_1, f_2, f_3 and f_4 are continuous mappings as well. Hence, Corollary 2 leads us to conclude that there exists at least an answer set of \mathbb{P}^c . For instance, consider the interpretation*

$$I \equiv \{(p, 0.4), (q, 0.8), (s, 0.5), (t, 1), (u, 0), (\text{not}_q^1, 0.2), (\text{not}_u^1, 1), (\text{not}_s^1, 0.5), (\text{not}_s^2, 0.5)\}$$

Since the corresponding MANLP \mathbb{P} of \mathbb{P}^c was already shown in Example 10, the reduct of \mathbb{P} with respect to I , denoted as \mathbb{P}_I , is given by the following rules:

$$\begin{aligned}
 r_1^I &: \langle p \leftarrow \frac{1+s}{2}; 1 \rangle & r_q^I &: \langle \text{not}_q^1 \leftarrow 0.2; 1 \rangle \\
 r_2^I &: \langle p \leftarrow q * \text{not}_s^2; 1 \rangle & r_u^I &: \langle \text{not}_u^1 \leftarrow 1; 1 \rangle \\
 r_3^I &: \langle q \leftarrow 0.8; 1 \rangle & r_s^{1I} &: \langle \text{not}_s^1 \leftarrow 0.5; 1 \rangle \\
 r_4^I &: \langle s \leftarrow \max\{\text{not}_q^1, t/2\}; 1 \rangle & r_s^{2I} &: \langle \text{not}_s^2 \leftarrow 0.5; 1 \rangle \\
 r_5^I &: \langle t \leftarrow \text{not}_u^1; 1 \rangle & &
 \end{aligned}$$

As there are no rules in \mathbb{P}_I with head u , we can assert that the least model M of \mathbb{P}_I satisfies $M(u) = 0$. Moreover, as $q, s, t, \text{not}_q^1, \text{not}_u^1, \text{not}_s^1$ and not_s^2 appear in the head of only one rule, specifically in the head of $r_3^I, r_4^I, r_5^I, r_q^I, r_u^I, r_s^{1I}$ and r_s^{2I} , respectively, then:

$$\begin{aligned}
 M(q) &= 0.8 \\
 M(s) &= \max\{M(\text{not}_q^1), M(t)/2\} = \max\{0.2, 0.5\} = 0.5 \\
 M(t) &= M(\text{not}_u^1) = 1 \\
 M(\text{not}_q^1) &= 0.2 \\
 M(\text{not}_u^1) &= 1 \\
 M(\text{not}_s^1) &= 0.5 \\
 M(\text{not}_s^2) &= \neg M(\text{not}_s^1) = -0.5 = 0.5
 \end{aligned}$$

Finally, the value assigned to p by M is computed as follows:

$$\begin{aligned}
 M(p) &= \min \left\{ \frac{1 + M(s)}{2}, M(q) * M(\text{not}_s^2) \right\} \\
 &= \min \left\{ \frac{1 + 0.5}{2}, 0.8 * 0.5 \right\} \\
 &= \min\{0.75, 0.4\} = 0.4
 \end{aligned}$$

As I coincides with M , we conclude that I is the least model of \mathbb{P}_I . In other words, I is a stable model of \mathbb{P} . Hence, applying Theorem 3, I is an answer set of \mathbb{P}^c .

5. Conclusions and Future Work

We have presented a methodology to simulate an arbitrary MANLP by means of a semantically equivalent CFASP. We proposed the inclusion of the weight of each rule appearing in a given MANLP in its body, by using the residuated pair, which defines the rule. This procedure allows us to complete the labour initiated in [15], that is, extended multi-adjoint logic programs can be translated into CFASPs with the goal of handling compact simple programs, which are semantically equivalent. This fact considerably increases the potential of EMALPs to model real-life problems, since modelling the information contained in a text or in a database by decision rules and the interpretation of those rules will be easier through a EMALP and its translation into a CFASP will facilitate the simulation and computation of the consequences/deductions from the program.

We have also studied the opposite translation procedure, that is, a mechanism to translate an arbitrary CFASP into a semantically equivalent MANLP has been given. The proposed translation method increases the number of rules of the original program, since it transforms the composited negations appearing in a given CFASP into a single negation, by including new rules with new propositional symbols. From this mechanism, we have established a one-to-one correspondence between the answer sets of a CFASP and the stable models of its corresponding MANLP. Indeed, we have proven that Theorem 1, introduced in [14], provides sufficient conditions under which the existence of answer sets of a CFASP can be ensured.

As future work, other results given in MALP and MANLP will also be considered, such as the termination results given in MALP [8,9] and the studies on coherent and incoherent information in MANLP [21,22]. In addition, we will study the extension of EMALP with the consideration of disjunctions in the head of the rules. Furthermore, we would like to apply the translation mechanisms and the theoretical advances achieved in this work to solve real-life problems. Specifically, we are interested in investigating the potential of their application to the digital forensics field, where the developed theory can be useful to extract knowledge with the goal of modelling behaviour patterns. For instance, in order to detect and prevent fraud related to transactional credit card databases as well as to manage databases containing information about crimes in a certain city. Due to the large amount of data in digital forensics, the simplicity of CFASPs is very useful to achieve a feasible computational efficiency. On the other hand, these data have a broad variability, for example, it can be given in natural language, and an easy mechanism for translating the data to logic programs is fundamental. In this sense, the high expressiveness potential of EMALPs can support this translation. Thus, both logic frameworks CFASP and EMALP are of great interest for obtaining information from (big) data.

Author Contributions: All authors have contributed equally to this work. All authors have read and agreed to the published version of the manuscript.

Funding: This research was partially supported by the 2014–2020 ERDF Operational Programme in collaboration with the State Research Agency (AEI), in project TIN2016-76653-P, and with the Department of Economy, Knowledge, Business and University of the Regional Government of Andalusia in project FEDER-UCA18-108612, by the European Cooperation in Science & Technology (COST) Action CA17124, and by the research and transfer program of the University of Cádiz.

Acknowledgments: The author would like to thank the four anonymous reviewers for their careful reading of the paper and useful suggestions to clarify this work.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Medina, J.; Ojeda-Aciego, M.; Vojtáš, P. Multi-adjoint logic programming with continuous semantics. In *Logic Programming and Non-Monotonic Reasoning, LPNMR'01. Lecture Notes in Artificial Intelligence 2173*; Springer: Berlin/Heidelberg, Germany, 2001; pp. 351–364.
2. Damásio, C.V.; Pereira, L.M. Monotonic and residuated logic programs. In *Symbolic and Quantitative Approaches to Reasoning with Uncertainty, ECSQARU'01. Lecture Notes in Artificial Intelligence, 2143*; Springer: Berlin/Heidelberg, Germany, 2001; pp. 748–759.
3. Vojtáš, P. Fuzzy logic programming. *Fuzzy Sets Syst.* **2001**, *124*, 361–370. [[CrossRef](#)]
4. Morcillo, P.J.; Moreno, G. Efficient Unfolding of Fuzzy Connectives for Multi-adjoint Logic Programs. In *Interactions Between Computational Intelligence and Mathematics*; Kóczy, L.T., Medina, J., Eds.; Springer International Publishing: Cham, Switzerland, 2018; pp. 57–78.
5. Moreno, G.; Penabad, J.; Rianza, J.A. Symbolic Unfolding of Multi-adjoint Logic Programs. In *Trends in Mathematics and Computational Intelligence*; Cornejo, M.E., Kóczy, L.T., Medina, J., De Barros Ruano, A.E., Eds.; Springer International Publishing: Cham, Switzerland, 2019; pp. 43–51.
6. Julián, P.; Moreno, G.; Penabad, J. An improved reductant calculus using fuzzy partial evaluation techniques. *Fuzzy Sets Syst.* **2009**, *160*, 162–181. [[CrossRef](#)]
7. Julián-Iranzo, P.; Medina, J.; Ojeda-Aciego, M. On reductants in the framework of multi-adjoint logic programming. *Fuzzy Sets Syst.* **2017**, *317*, 27–43. [[CrossRef](#)]
8. Damásio, C.; Medina, J.; Ojeda-Aciego, M. Sorted Multi-adjoint Logic Programs: Termination Results and Applications. *Lect. Notes Artif. Intell.* **2004**, *3229*, 252–265.
9. Damásio, C.; Medina, J.; Ojeda-Aciego, M. Termination of logic programs with imperfect information: Applications and query procedure. *J. Appl. Logic* **2007**, *5*, 435–458. [[CrossRef](#)]
10. Bustince, H.; Madrid, N.; Ojeda-Aciego, M. The Notion of Weak-Contradiction: Definition and Measures. *IEEE Trans. Fuzzy Syst.* **2015**, *23*, 1057–1069. [[CrossRef](#)]

11. Madrid, N.; Ojeda-Aciego, M. On the measure of incoherent information in extended multi-adjoint logic programs. In Proceedings of the 2013 IEEE Symposium on Foundations of Computational Intelligence (FOCI), Singapore, 16–19 April; Institute of Electrical and Electronics Engineers (IEEE): Piscataway, NJ, USA, 2013.
12. Medina, J.; Ojeda-Aciego, M.; Vojtáš, P. Similarity-based unification: A multi-adjoint approach. *Fuzzy Sets Syst.* **2004**, *146*, 43–62. [[CrossRef](#)]
13. Strass, H.; Muñoz-Hernández, S.; Pablos-Ceruelo, V. Operational Semantics for a Fuzzy Logic Programming System with Defaults and Constructive Answers. In Proceedings of the Joint 2009 International Fuzzy Systems Association World Congress and 2009 European Society of Fuzzy Logic and Technology Conference, Lisbon, Portugal, 20–24 July 2009.
14. Cornejo, M.E.; Lobo, D.; Medina, J. Syntax and semantics of multi-adjoint normal logic programming. *Fuzzy Sets Syst.* **2018**, *345*, 41–62. [[CrossRef](#)]
15. Cornejo, M.E.; Lobo, D.; Medina, J. Extended multi-adjoint logic programming. *Fuzzy Sets Syst.* **2020**, *388*, 124–145. [[CrossRef](#)]
16. Janssen, J.; Schockaert, S.; Vermeir, D.; De Cock, M. A core language for fuzzy answer set programming. *Int. J. Approx. Reason.* **2012**, *53*, 660–692. [[CrossRef](#)]
17. Madrid, N.; Ojeda-Aciego, M. Towards a fuzzy answer set semantics for residuated logic programs. In Proceedings of the Web Intelligence/IAT Workshops, Sydney, NSW, Australia, 9–12 December 2008; pp. 260–264.
18. Van Nieuwenborgh, D.; De Cock, M.; Vermeir, D. An introduction to fuzzy answer set programming. *Ann. Math. Artif. Intell.* **2007**, *50*, 363–388. [[CrossRef](#)]
19. Madrid, N.; Ojeda-Aciego, M. On the existence and unicity of stable models in normal residuated logic programs. *Int. J. Comput. Math.* **2012**, *89*, 310–324. [[CrossRef](#)]
20. Gelfond, M.; Lifschitz, V. The stable model semantics for logic programming. *ICLP/SLP* **1988**, *88*, 1070–1080.
21. Cornejo, M.E.; Lobo, D.; Medina, J. Selecting the Coherence Notion in Multi-adjoint Normal Logic Programming. *Lect. Notes Comput. Sci.* **2017**, *10305*, 447–457.
22. Cornejo, M.E.; Lobo, D.; Medina, J. Measuring the Incoherent Information in Multi-adjoint Normal Logic Programs. *Adv. Intell. Syst. Comput.* **2018**, *641*, 521–533.



© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).