Ngoc Thanh Nguyen · Richard Chbeir ·
Ernesto Exposito · Philippe Aniorté ·
Bogdan Trawiński (Eds.)

LNAI 11683

Computational Collective Intelligence

11th International Conference, ICCCI 2019
Hendaye, France, September 4–6, 2019
Proceedings, Part I

Springer

Lecture Notes in Artificial Intelligence 11683

Subseries of Lecture Notes in Computer Science

Organization

Organizing Committee

Honorary Chairs

Mohamad Amara	President of University of Pau and Pays de l'Adour, France
Cezary Madryas	Rector of Wrocław University of Science and Technology, Poland

General Chairs

Richard Chbeir	University of Pau and Pays de l'Adour, France
Ngoc Thanh Nguyen	Wrocław University of Science and Technology, Poland

General Vice-chair

Philippe Aniorté	University of Pau and Pays de l'Adour, France

Program Chairs

Ernesto Exposito	University of Pau and Pays de l'Adour, France
Nick Bassiliades	Aristotle University of Thessaloniki, Greece
Costin Bădică	University of Craiova, Romania
Gottfried Vossen	University of Münster, Germany
Edward Szczerbicki	University of Newcastle, Australia

Organizing Chair

Khouloud Salameh	American University of Ras Al Khaimah, UAE

Steering Committee

Ngoc Thanh Nguyen	Wrocław University of Science and Technology, Poland
Shyi-Ming Chen	National Taiwan University of Science and Technology, Taiwan
Toyoaki Nishida	Kyoto University, Japan
Geun-Sik Jo	Inha University, South Korea
Kiem Hoang	University of Information Technology, VNU-HCM, Vietnam
Ryszard Kowalczyk	Swinburne University of Technology, Australia
Lakhmi C. Jain	University of South Australia, Australia

Contents – Part I

The Stable Model Semantics of Normal Fuzzy Linguistic Logic Programs

Van Hung Le(✉)

Faculty of Information Technology, Hanoi University of Mining and Geology, Duc Thang, Bac Tu Liem, Hanoi, Vietnam
levanhung@humg.edu.vn

Abstract. Fuzzy linguistic logic programming is a framework for representing and reasoning with linguistically-expressed human knowledge. It is well known that allowing the representation and the manipulation of negation is an important feature for many real-world applications. In this work, we extend the framework by allowing negation connectives to occur in rule bodies, resulting in normal fuzzy linguistic logic programs, and study the stable model semantics of such logic programs.

Keywords: Fuzzy logic programming · Stable model · Hedge algebra

1 Introduction

Humans mainly use words, which are essentially qualitative and vague, to describe real world phenomena, to reason, and to decide. Also, humans often express knowledge linguistically and hence make use of fuzzy predicates. Moreover, in linguistically-expressed human knowledge, various linguistic hedges are usually used simultaneously to express many levels of emphasis. Thus, it is necessary to have formalisms that can directly deal with linguistic terms and hedges since those systems can represent and reason with linguistically-expressed human knowledge to some extent.

Fuzzy linguistic logic programming (FLLP), introduced in [1], is a logic programming (LP) framework without negation for dealing with vagueness in linguistically-expressed human knowledge, where truth values of vague sentences are linguistic terms, and linguistic hedges can be utilized to state various levels of emphasis. In FLLP, every fact or rule is evaluated to some degree stated by a linguistic truth value, and hedges can be utilized as unary connectives in rule bodies. For instance, the statement "(A worker is good if he is *rather* skillful and *very* hard-working) is *highly true*" can be represented using the following rule:

$$good(X) \leftarrow \wedge(Rather\ skillful(X), Very\ hard\_working(X)).HighlyTrue$$

The framework can have a counterpart for most of the concepts and results of traditional definite LP [1,2]. The procedural semantics [1] and tabulation proof procedures [3] of FLLP can directly compute on linguistic terms to give answers

N. T. Nguyen et al. (Eds.): ICCCI 2019, LNAI 11683, pp. 53–65, 2019.
https://doi.org/10.1007/978-3-030-28377-3_5

to queries. Hence, in the presence of vagueness, FLLP can provide to human reasoning a computational approach.

It is well known that allowing the representation and the manipulation of *default negation* is an important feature for many real-world applications. In this paper, we extend FLLP by allowing negation connectives to occur in rule bodies, resulting in *normal fuzzy linguistic logic programs*, and present the stable model semantics of such programs.

The paper is organized as follows: Sect. 2 gives an overview of FLLP without negation while Sect. 3 presents FLLP with negation and its stable model semantics. Section 4 discusses related work and Sect. 5 gives several directions for future work and concludes the paper.

2 Preliminaries

2.1 Linguistic Truth Domains

In hedge algebra (HA) theory [4,5], values of the linguistic variable *Truth*, e.g., *Very True, Very Slightly False*, can be regarded as being generated from the set of primary terms $\mathcal{G} = \{False, True\}$ using hedges from a set $\mathcal{H} = \{Very, Slightly, \dots\}$ as unary operators. There is a natural ordering among such values, for instance, *Very False* $<$ *False*, where $x \leq y$ states that x represents a truth degree less than or equal to y. Hence, a set $\mathcal{X}$ of such values of *Truth* is a partially ordered set and can be characterized by an HA $\underline{X} = (\mathcal{X}, \mathcal{G}, \mathcal{H}, \leq)$.

Linguistic hedges either decrease or increase the meaning of terms, thus they can be considered as *order operations*, i.e., for all $h \in \mathcal{H}$ and for all $x \in \mathcal{X}$, we have either $hx \geq x$ or $hx \leq x$. We denote by $h \geq k$ if h modifies terms more than or equal to k, i.e., for all $x \in \mathcal{X}$, either $hx \leq kx \leq x$ or $x \leq kx \leq hx$. Note that $\mathcal{H}$ and $\mathcal{X}$ are disjoint, so the same notation $\leq$ can be used without confusion for the order relations on $\mathcal{H}$ and $\mathcal{X}$. Let *V, R, H, S, A, M,* c^+, and c^- stand respectively for *Very, Rather, Highly, Slightly, Approximately, More or less, True*, and *False*. We have $S > R$ ($h > k$ iff $h \geq k$ and $h \neq k$) since, e.g., $Sc^+ < Rc^+ < c^+$ and $c^- < Rc^- < Sc^-$, and $V > H$ since, e.g., $c^+ < Hc^+ < Vc^+$ and $Vc^- < Hc^- < c^-$.

$\mathcal{H}$ can be split into disjoint subsets $\mathcal{H}^+$ and $\mathcal{H}^-$ characterized by $\mathcal{H}^+ = \{h | hc^+ > c^+\} = \{h | hc^- < c^-\}$ and $\mathcal{H}^- = \{h | hc^+ < c^+\} = \{h | hc^- > c^-\}$. For instance, given the set $\mathcal{H} = \{V, H, A, M, R, S\}$, $\mathcal{H}$ can be divided into $\mathcal{H}^+ = \{V, H\}$ and $\mathcal{H}^- = \{A, M, R, S\}$. Two hedges in each of $\mathcal{H}^+$ and $\mathcal{H}^-$ might be comparable, e.g., V and H, or incomparable, e.g., A and M. An HA $\underline{X} = (\mathcal{X}, \mathcal{G}, \mathcal{H}, \leq)$ is called a *linear* HA (lin-HA) if both $\mathcal{H}^+$ and $\mathcal{H}^-$ are linearly ordered. For instance, the HA $\underline{X} = (\mathcal{X}, \mathcal{G}, \{V, H, R, S\}, \leq)$ is a lin-HA since $\mathcal{H}$ is split into $\mathcal{H}^+ = \{V, H\}$ with $V > H$ and $\mathcal{H}^- = \{R, S\}$ with $S > R$. For any lin-HA $\underline{X} = (\mathcal{X}, \mathcal{G}, \mathcal{H}, \leq)$, we have $\mathcal{X}$ is linearly ordered.

A *linguistic truth domain* (LTD) $\overline{X}$ taken from a lin-HA $\underline{X} = (\mathcal{X}, \mathcal{G}, \mathcal{H}, \leq)$ is the linearly ordered set $\overline{X} = \mathcal{X} \cup \{0, W, 1\}$, where 0 (*AbsolutelyFalse*), W (the *middle truth value*), and 1 (*AbsolutelyTrue*) are respectively the least, the neutral and the greatest elements of $\overline{X}$, and for all $x \in \{0, W, 1\}$ and for all

$h \in \mathcal{H}$, we have $hx = x$ [1,3]. To have well-defined Łukasiewicz operations, we consider only finitely many truth values. An *l-limit* HA, where l is a positive integer, is a lin-HA in which all terms have a length of at most $l+1$, i.e., at most l hedges. An LTD taken from an l-limit HA is finite.

Example 1. The 1-limit HA $\underline{X} = (\mathcal{X}, \{c^-, c^+\}, \{V, H, R, S\}, \leq)$ gives the LTD $\overline{X} = \{v_0 = 0, v_1 = Vc^-, v_2 = Hc^-, v_3 = c^-, v_4 = Rc^-, v_5 = Sc^-, v_6 = W, v_7 = Sc^+, v_8 = Rc^+, v_9 = c^+, v_{10} = Hc^+, v_{11} = Vc^+, v_{12} = 1\}$, in which truth values are listed in ascending order.

2.2 Truth Functions of Hedge Connectives

Let $I \notin \mathcal{H}$ be an artificial hedge, called the *identity*, defined by $\forall x \in \mathcal{X}$, $Ix = x$. I is the least element in each of $\mathcal{H}^+ \cup \{I\}$ and $\mathcal{H}^- \cup \{I\}$ [1,3]. An *extended order relation* $\leq_e$ on $\mathcal{H} \cup \{I\}$ is defined by: $\forall h, k \in \mathcal{H} \cup \{I\}$, $h \leq_e k$ if one of the following conditions is satisfied: (i) $h \in \mathcal{H}^-, k \in \mathcal{H}^+$; (ii) $h, k \in \mathcal{H}^+ \cup \{I\}$ and $h \leq k$; (iii) $h, k \in \mathcal{H}^- \cup \{I\}$ and $h \geq k$. We denote $h <_e k$ if $h \leq_e k$ and $h \neq k$.

Let $\underline{X} = (\mathcal{X}, \{c^+, c^-\}, \mathcal{H}, \leq)$ be a lin-HA. *Truth functions* $h^\bullet : \overline{X} \to \overline{X}$ of all hedges $h \in \mathcal{H} \cup \{I\}$ satisfy the following conditions [3,6,7]:

$$\forall x \in \{0, W, 1\}, h^\bullet(x) = x \tag{1}$$

$$\forall x \in \overline{X}, I^\bullet(x) = x \tag{2}$$

$$h^\bullet(hc^+) = c^+ \tag{3}$$

$$\text{if } x \geq y, h^\bullet(x) \geq h^\bullet(y) \tag{4}$$

$$\forall k \in \mathcal{H} \cup \{I\} \text{ such that } h \leq_e k, h^\bullet(x) \geq k^\bullet(x) \tag{5}$$

Truth functions of hedges are non-decreasing and preserve 0 and 1. Moreover, truth functions of all truth-stressing hedges $h \in \mathcal{H}^+$ are subdiagonal ($h^\bullet(x) \leq x$), and those of all truth-depressing hedges $h \in \mathcal{H}^-$ are superdiagonal ($h^\bullet(x) \geq x$). Condition (3) ensures that if the truth value of the sentence "Lucia is *young*" is *very true*, then that of the sentence "Lucia is *very young*" is *true* [8,9]. It is shown in [1] that truth functions of hedges always exist.

2.3 Operations on Linguistic Truth Domains

Gödel t-norm, its residuum, and Gödel t-conorm can be defined as [1,3]:

$$\mathcal{C}_G(v_i, v_j) = \min(v_i, v_j),$$

$$\leftarrow^\bullet_G (v_j, v_i) = \begin{cases} v_n \text{ if } i \leq j \\ v_j \text{ otherwise,} \end{cases}$$

$$\vee^\bullet_G(v_i, v_j) = \max(v_i, v_j).$$

Łukasiewicz t-norm, its residuum, and Łukasiewicz t-conorm can be defined on a finite LTD $\overline{X} = \{v_0, \ldots, v_n\}$ with $v_0 \leq v_1 \leq \cdots \leq v_n$ as [1,3]:

$$\mathcal{C}_L(v_i, v_j) = \begin{cases} v_{i+j-n} \text{ if } i+j-n > 0 \\ v_0 \quad \text{otherwise,} \end{cases}$$

$$\leftarrow_L^{\bullet}(v_j, v_i) = \begin{cases} v_n \quad \text{if } i \leq j \\ v_{n+j-i} \text{ otherwise,} \end{cases}$$

$$\vee_L^{\bullet}(v_i, v_j) = \begin{cases} v_{i+j} \text{ if } i+j < n \\ v_n \quad \text{otherwise.} \end{cases}$$

All the residua are non-decreasing in the first argument and non-increasing in the second, and all the other operations are non-decreasing in all arguments.

The negation is defined as follows [1]: given $x = \sigma c$, where σ is a string of hedges (including the empty one) and $c \in \{c^+, c^-\}$, then $y = -x$, if $y = \sigma c'$ and $\{c, c'\} = \{c^+, c^-\}$, e.g., $Vc^+ = -Vc^-$ and $Vc^- = -Vc^+$.

Each of the t-norms and its residuum satisfy the following properties [10]:

$$\mathcal{C}(b, r) \leq h \text{ iff } r \leq \leftarrow^{\bullet}(h, b) \tag{6}$$

$$(\forall b)(\forall h)\ \mathcal{C}(b, \leftarrow^{\bullet}(h, b)) \leq h \tag{7}$$

$$(\forall b)(\forall r)\ \leftarrow^{\bullet}(\mathcal{C}(b, r), b) \geq r \tag{8}$$

2.4 Fuzzy Linguistic Logic Programming Without Negation

The language is a many-sorted (or typed) predicate language without function symbols. Clauses in logic programs are without negation. The fact that function symbols do not appear in the language makes FLLP implementable. Without function symbols, Herbrand universes of all sorts of variables of a finite logic program are finite, and so is the Herbrand base (consisting of all ground atoms) of the program. This, together with a finite LTD, allows obtaining the least Herbrand model of a logic program by finitely iterating an immediate consequence operator from the least Herbrand interpretation. As usual the underlying language of a program P is assumed to be defined by constants (if no such constant exists, we add some constant a to form ground terms) and predicate symbols occurring in P.

Connectives consist of the following: $\wedge_G, \wedge_L$ (Gödel and Łukasiewicz conjunctions); $\vee_G, \vee_L$ (Gödel and Łukasiewicz disjunctions); $\leftarrow_G, \leftarrow_L$ (Gödel and Łukasiewicz implications); and hedges as unary connectives. For a connective c, its truth function is denoted by $c^{\bullet}$. Moreover, it is shown in [2] that linguistic aggregation operators can be used in rule bodies.

A *term* is either a constant or a variable. An *atom* (or *atomic formula*) is of the form $p(t_1, ..., t_n)$, where p is an n-ary predicate symbol, and $t_1, ..., t_n$ are terms. A *fact* is a graded atom $(A.a)$, where A is an atom called the logical part of the fact, and a is a truth value apart from 0. A *body formula* is defined inductively as follows: (i) an atom is a body formula; (ii) if B_1 and B_2 are body formulae, then so are $\wedge(B_1, B_2)$, $\vee(B_1, B_2)$, and hB_1, where h is a hedge

connective. A *rule* is a graded implication $(A \leftarrow B.r)$, where A is an atom called *rule head*, B is a body formula called *rule body*, and r is a truth value apart from 0; $(A \leftarrow B)$ is called the *logical part* of the rule. In a graded formula $(\varphi.t)$, t is understood as a lower bound to the exact truth value of φ. A *fuzzy linguistic logic program* (*positive program*, for short) is a finite set of facts and rules, and there are no two facts (rules) having the same logical part, but different truth values. Thus, a program P can be represented as the following partial mapping:

$$P : Formulae \rightarrow \overline{X} \setminus \{0\},$$

where the domain of P, denoted $dom(P)$, is finite and composed only of logical parts of facts and rules. For each formula $(\varphi.t) \in P$, $P(\varphi) = t$.

Let P be a program, and $\overline{X}$ the LTD; a *fuzzy linguistic Herbrand interpretation* (interpretation, for short) f of P is a mapping from the Herbrand base B_P to $\overline{X}$, associating with each ground atom in B_P a truth value in $\overline{X}$.

The ordering $\leq$ in $\overline{X}$ is extended to interpretations pointwise as follows: for any interpretations f_1 and f_2 of a program P, $f_1 \sqsubseteq f_2$ iff $f_1(A) \leq f_2(A), \forall A \in B_P$. Let $\otimes$ and $\oplus$ denote the meet (or infimum, greatest lower bound) and join (or supremum, least upper bound) operators, respectively; for all interpretations f_1 and f_2 of P and for all $A \in B_P$, we have: (i) $(f_1 \otimes f_2)(A) = f_1(A) \otimes f_2(A)$, and (ii) $(f_1 \oplus f_2)(A) = f_1(A) \oplus f_2(A)$.

Every interpretation f can be extended to all ground formulae, denoted $\overline{f}$, as follows: (i) $\overline{f}(A) = f(A)$, if A is a ground atom; (ii) $\overline{f}(c(B_1, B_2)) = c^{\bullet}(\overline{f}(B_1), \overline{f}(B_2))$, where B_1, B_2 are ground formulae, and c is a binary connective; and (iii) $\overline{f}(hB) = h^{\bullet}(\overline{f}(B))$, where B is a ground body formula, and h is a hedge. For non-ground formulae, since all variables in formulae are assumed to be universally quantified, the interpretation $\overline{f}$ is defined as

$$\overline{f}(\varphi) = \overline{f}(\forall\varphi) = \otimes\{\overline{f}(\varphi\vartheta) | \varphi\vartheta \text{ is a ground instance of } \varphi\},$$

where $\forall\varphi$ denotes the *universal closure* of φ, which is obtained from φ by adding a universal quantifier for every variable having a free occurrence in φ.

An interpretation f is an *Herbrand model* (model, for short) of a program P if for all formulae $\varphi \in dom(P)$, we have $\overline{f}(\varphi) \geq P(\varphi)$.

Theorem 1. *[2] Let P be a positive program.*

(i) Let $\mathcal{F}_P$ be the set of all interpretations of P. Then $\langle \mathcal{F}_P, \otimes, \oplus \rangle$ is a complete lattice.
(ii) Let F be a non-empty set of models of P. Then $\otimes F$ is a model of P.
(iii) $M_P = \otimes\{f | f$ is a model of $P\}$ is the least model of P.

Definition 1 (Immediate consequence operator). *Let P be a positive program. The operator T_P that maps from interpretations to interpretations is defined as follows: for any interpretation f and each ground atom $A \in B_P$,*

$T_P(f)(A) = max\{\oplus\{\mathcal{C}_i(\overline{f}(B), r) : (A \leftarrow_i B.r)$ is a ground instance of a rule in $P\}$, $\oplus\{a : (A.a)$ is a ground instance of a fact in $P\}\}$.

Since the Herbrand base B_P is finite, for each $A \in B_P$, there are a finite number of ground instances of rule heads and logical parts of facts which match A. Therefore, both $\oplus$ operators in the definition of T_P are, in deed, maxima. T_P is shown to be monotone and continuous [1].

The bottom-up iteration of T_P is defined as follows:

$$T_P^k(\bot) = \begin{cases} \bot & \text{if } k = 0 \\ T_P(T_P^{k-1}(\bot)) & \text{if } k \text{ is a successor ordinal} \\ \oplus\{T_P^n(\bot) | n < k\} & \text{if } k \text{ is a limit ordinal,} \end{cases}$$

where $\bot$ denotes the least interpretation, mapping every ground atom to 0.

The least model M_P is exactly the least fixpoint of T_P, denoted $lfp(T_P)$, and can be obtained by finitely bottom-up iterating T_P.

Theorem 2. *[1] Let P be a positive program. Then there exists a finite number α such that $n \geq \alpha$ implies $T_P^n(\bot) = lfp(T_P) = M_P$.*

3 Normal Fuzzy Linguistic Logic Programs

We extend FLLP with negation by allowing negative literals to occur in rule bodies, resulting in *normal fuzzy linguistic logic programs* (*normal programs*, for short). An *extended positive fuzzy linguistic logic program* (*extended positive program*, for short) is a (positive) fuzzy linguistic logic program in which elements of the truth domain can appear in the places of atoms in rule bodies, and the value of such an element under any interpretation of the program is itself.

The notions of an interpretation, a model and the immediate consequence operator of a normal or extended positive program can be defined similarly to those of a positive program.

Definition 2 (Stable model). *Let P be a normal program and I an interpretation of P. Let P^* denote a program consisting of all ground instances of rules and facts in P. I is a* stable model *of P iff $I = I'$, where the interpretation I' is obtained according to the Gelfond-Lifschitz transformation [11] as follows:*

(i) first, substitute (fix) in P^* all negative literals by their values w.r.t. I, which are computed as $\overline{I}(\neg A) = -I(A)$ for all ground atoms A. Let P^I denote the resulting extended positive program, called the reduct *of P w.r.t. I;*
(ii) then, compute the least model I' of P^I.

This approach defines a whole family of models of a normal program P, called the *stable model semantics* of P. In the subsequent proofs, it can be easily seen that when proving a model of P being a model of P^I or vice versa, since the fact parts of the programs have the same ground instances, we only need to deal with the rule parts. For each rule $(A \leftarrow B.r)$ in P, B can be denoted by $B[B_1, \ldots, B_m, \neg B_{m+1}, \ldots, \neg B_n]$, where $1 \leq m \leq n$ and $B_1, \ldots, B_n$ are all atoms appearing in B. Each ground instance of the rule in P^* is of the form $(A\vartheta \leftarrow B[B_1\vartheta, \ldots, B_m\vartheta, \neg B_{m+1}\vartheta, \ldots, \neg B_n\vartheta].r)$, and the corresponding one in

P^I can be denoted as $(A\vartheta \leftarrow B[B_1\vartheta, \ldots, B_m\vartheta, -I(B_{m+1}\vartheta), \ldots, -I(B_n\vartheta)].r)$, where every negative literal $\neg B_i\vartheta$, $m+1 \leq i \leq n$, is replaced by its value under I. Let $\mathcal{B}$ be the expression obtained from B by replacing every connective apart from the negation with its truth function.

Theorem 3. *Any stable model of a normal program P is a model of P.*

Proof. Let $(A \leftarrow B[B_1, \ldots, B_m, \neg B_{m+1}, \ldots, \neg B_n].r)$ be any rule in P and I be a stable model of P. By definition, I is the least model of P^I, so for all ground instances $(A\vartheta \leftarrow B[B_1\vartheta, \ldots, B_m\vartheta, \neg B_{m+1}\vartheta, \ldots, \neg B_n\vartheta].r)$ in P^* of the rule and the corresponding one $(A\vartheta \leftarrow B[B_1\vartheta, \ldots, B_m\vartheta, -I(B_{m+1}\vartheta), \ldots, -I(B_n\vartheta)].r)$ in P^I, we have:

$$\begin{aligned} r &\leq \overline{I}(A\vartheta \leftarrow B[B_1\vartheta, \ldots, B_m\vartheta, -I(B_{m+1}\vartheta), \ldots, -I(B_n\vartheta)]) \\ &= \overline{I}(A\vartheta \leftarrow B[B_1\vartheta, \ldots, B_m\vartheta, \neg B_{m+1}\vartheta, \ldots, \neg B_n\vartheta]) \end{aligned}$$

Thus, for all ground instances $(A\vartheta \leftarrow B[B_1\vartheta, \ldots, B_m\vartheta, \neg B_{m+1}\vartheta, \ldots, \neg B_n\vartheta])$ of $(A \leftarrow B[B_1, \ldots, B_m, \neg B_{m+1}, \ldots, \neg B_n])$, we have:

$$\begin{aligned} r &\leq \otimes\{\overline{I}(A\vartheta \leftarrow B[B_1\vartheta, \ldots, B_m\vartheta, \neg B_{m+1}\vartheta, \ldots, \neg B_n\vartheta])\} \\ &= \overline{I}(A \leftarrow B[B_1, \ldots, B_m, \neg B_{m+1}, \ldots, \neg B_n]) \end{aligned}$$

Hence, I is a model of the rule $(A \leftarrow B[B_1, \ldots, B_m, \neg B_{m+1}, \ldots, \neg B_n].r)$. Since the rule is arbitrary, I is a model of P. □

A normal program may have several stable models as in the following example.

Example 2. Given the LTD in Example 1, consider the program P consisting of the following rules:

$$\begin{aligned} (good(X) &\leftarrow_G \neg bad(X).Vc^+) \\ (bad(X) &\leftarrow_G \neg good(X).Vc^+) \end{aligned}$$

We will determine stable models of P. Let $I = \{(good(a), x), (bad(a), y)\}$ be an interpretation of P, where the constant a is added to form ground terms. The reduct P^I consists of the following rules:

$$\begin{aligned} (good(a) &\leftarrow_G -y.Vc^+) \\ (bad(a) &\leftarrow_G -x.Vc^+) \end{aligned}$$

The least model of P^I is $M_{P^I} = \{(good(a), \mathcal{C}_G(-y, Vc^+)), (bad(a), \mathcal{C}_G(-x, Vc^+))\}$. Thus, I is a stable model of P iff $x = \mathcal{C}_G(-y, Vc^+) = \min(-y, Vc^+)$ and $y = \mathcal{C}_G(-x, Vc^+) = \min(-x, Vc^+)$. It can be seen that all $Vc^- \leq x = -y \leq Vc^+$ (so $Vc^- \leq y = -x \leq Vc^+$) satisfying the conditions. That is, we have 11 stable models $I = \{(good(a), x), (bad(a), -x)\}$ such that $v_1 = Vc^- \leq x \leq Vc^+ = v_{11}$.

Furthermore, we have the following result. Note that as shown in Example 2, a normal program may not have a least model, but may have several minimal models.

Theorem 4. *Any stable model of a normal program P is a minimal model of P.*

Proof. Let I be a stable model of P. By definition, I is the least model of P^I. We will prove that I is a minimal model of P. Suppose that there exists a model J of P such that $J \sqsubset I$. We will show that J is a model of P^I. Since I is the least model of P^I, we have $I \sqsubseteq J$, which is a contradiction to the hypothesis.

Let $(A \leftarrow B[B_1, \ldots, B_m, \neg B_{m+1}, \ldots, \neg B_n].r)$ be any rule in P. Each of its ground instances in P^* is of the form $(A\vartheta \leftarrow B[B_1\vartheta, \ldots, B_m\vartheta, \neg B_{m+1}\vartheta, \ldots, \neg B_n\vartheta].r)$, and the counterpart in P^I is $(A\vartheta \leftarrow B[B_1\vartheta, \ldots, B_m\vartheta, -I(B_{m+1}\vartheta), \ldots, -I(B_n\vartheta)].r)$. Since $J \sqsubset I$, we have, for all $m+1 \le i \le n$, $J(B_i\vartheta) \le I(B_i\vartheta)$, hence $-J(B_i\vartheta) \ge -I(B_i\vartheta)$. J is a model of P, so we have:

$$
\begin{aligned}
r &\le \overline{J}(A \leftarrow B[B_1, \ldots, B_m, \neg B_{m+1}, \ldots, \neg B_n]) \\
&\le \overline{J}(A\vartheta \leftarrow B[B_1\vartheta, \ldots, B_m\vartheta, \neg B_{m+1}\vartheta, \ldots, \neg B_n\vartheta]) \\
&= \leftarrow^{\bullet}(J(A\vartheta), \mathcal{B}(J(B_1\vartheta), \ldots, J(B_m\vartheta), -J(B_{m+1}\vartheta), \ldots, -J(B_n\vartheta))) \\
&\le \leftarrow^{\bullet}(J(A\vartheta), \mathcal{B}(J(B_1\vartheta), \ldots, J(B_m\vartheta), -I(B_{m+1}\vartheta), \ldots, -I(B_n\vartheta))) \\
&= \overline{J}(A\vartheta \leftarrow B[B_1\vartheta, \ldots, B_m\vartheta, -I(B_{m+1}\vartheta), \ldots, -I(B_n\vartheta)])
\end{aligned}
$$

The last inequality holds due to the fact that every truth function in $\mathcal{B}$ (apart from the negation) is non-decreasing, and $\leftarrow^{\bullet}$ is non-increasing in the second argument. Thus, J is a model of $(A\vartheta \leftarrow B[B_1\vartheta, \ldots, B_m\vartheta, -I(B_{m+1}\vartheta), \ldots, -I(B_n\vartheta)].r)$ in P^I. Since the rule in P is arbitrary, J is a model of P^I. □

The notion of the immediate consequence operator T_P for a normal program P is defined analogously to the case of positive programs. Similar to the case of positive programs, a model of P is a pre-fixpoint of T_P and vice versa [12] as follows.

Theorem 5. *Let P be a normal program. Then an interpretation f is a model of P iff $T_P(f) \sqsubseteq f$.*

Proof. First, let f be a model of P; we prove that $T_P(f) \sqsubseteq f$. Let A be any ground atom; consider the following cases:

(i) If A is neither a ground instance of the logical part of a fact nor a ground instance of a rule head in P, then $T_P(f)(A) = 0 \le f(A)$.
(ii) For each ground instance $(A.a)$ of a fact, say $(C.a)$, in P, since f is a model of P, and A is a ground instance of C, we have $a = P(C) \le \overline{f}(C) \le f(A)$. Hence, $f(A) \ge \oplus\{a | (A.a)$ is a ground instance of a fact in $P\}$.

(iii) For each ground instance $(A \leftarrow_i B.r)$, where $i \in \{G, L\}$, of a rule, say $(C.r)$, in P, we have: $\mathcal{C}_i(\overline{f}(B), r) = \mathcal{C}_i(\overline{f}(B), P(C)) \leq^{(*)} \mathcal{C}_i(\overline{f}(B), \overline{f}(A \leftarrow_i B)) = \mathcal{C}_i(\overline{f}(B), \leftarrow_i^{\bullet} (f(A), \overline{f}(B))) \leq^{(**)} f(A)$, where (*) holds since $(A \leftarrow_i B)$ is a ground instance of C, and (**) follows from (7). Therefore, $f(A) \geq \oplus\{\mathcal{C}_i(\overline{f}(B), r) | (A \leftarrow_i B.r)$ is a ground instance of a rule in $P\}$.

By definition, $T_P(f)(A) \leq f(A)$. Since A is arbitrary, we have $T_P(f) \sqsubseteq f$.

Finally, let us show that if $T_P(f) \sqsubseteq f$, then f is a model of P. Suppose there is an interpretation f such that $T_P(f) \sqsubseteq f$. Let C be any formula in $dom(P)$. There are two cases:

(i) $(C.c)$, where c is a truth value, is a fact in P. For each ground instance A of C, by the assumption $T_P(f) \sqsubseteq f$, we have $f(A) \geq T_P(f)(A) \geq \oplus\{a | (A.a)$ is a ground instance of a fact in $P\} \geq c = P(C)$. Therefore, $\overline{f}(C) = \otimes\{f(A) | A$ is a ground instance of $C\} \geq P(C)$.

(ii) $(C.c)$ is a rule in P. For each ground instance $A \leftarrow_j D$ of C, where $j \in \{G, L\}$, since $T_P(f) \sqsubseteq f$, we have $f(A) \geq T_P(f)(A) \geq \oplus\{\mathcal{C}_i(\overline{f}(B), r) | (A \leftarrow_i B.r)$ is a ground instance of a rule in $P\} \geq \mathcal{C}_j(\overline{f}(D), c) = \mathcal{C}_j(\overline{f}(D), P(C))$. Hence, $\overline{f}(A \leftarrow_j D) = \leftarrow_j^{\bullet} (f(A), \overline{f}(D)) \geq^{(*)} \leftarrow_j^{\bullet} (\mathcal{C}_j(\overline{f}(D), P(C)), \overline{f}(D)) \geq^{(**)} P(C)$, where (*) holds since $\leftarrow_i^{\bullet}$ is non-decreasing in the first argument, and (**) follows from (8). Consequently, $\overline{f}(C) = \otimes\{\overline{f}(A \leftarrow_j D) | (A \leftarrow_j D)$ is a ground instance of $C\} \geq P(C)$.

Since C is arbitrary, by definition, f is a model of P. □

However, for a normal program P, T_P may not be monotone as in the following example, so it does not necessarily have a least fixpoint.

Example 3. Consider a normal program P consisting of the following rule:

$$(good(X) \leftarrow_G \neg bad(X).1)$$

Given an interpretation $I = \{(good(a), x), (bad(a), y)\}$, where x and y are truth values, $T_P(I) = \{(good(a), -y), (bad(a), 0)\}$. Clearly, $T_P(I)$ is not monotone.

Theorem 6. *Any stable model of a normal program P is a minimal fixpoint of T_P.*

Proof. First, we show that, for every interpretation I, $T_P(I)$ for P coincides with $T_{P^I}(I)$ for the extended positive program P^I. In Definition 1,

(i) The second $\oplus$ operator is obviously the same for both $T_P(I)$ and $T_{P^I}(I)$.

(ii) Concerning the first $\oplus$, let $(A \leftarrow B[B_1, \ldots, B_m, \neg B_{m+1}, \ldots, \neg B_n].r)$ be any rule in P. For each of its ground instances $(A\vartheta \leftarrow B[B_1\vartheta, \ldots, B_m\vartheta, \neg B_{m+1}\vartheta, \ldots, \neg B_n\vartheta].r)$ and the counterpart $(A\vartheta \leftarrow$

$B[B_1\vartheta, \ldots, B_m\vartheta, -I(B_{m+1}\vartheta), \ldots, -I(B_n\vartheta)].r)$ in P^I, we have the following:

$$\begin{aligned}
&\mathcal{C}(\overline{I}(B[B_1\vartheta, \ldots, B_m\vartheta, \neg B_{m+1}\vartheta, \ldots, \neg B_n\vartheta]), r) \\
&= \mathcal{C}(\mathcal{B}(I(B_1\vartheta), \ldots, I(B_m\vartheta), \overline{I}(\neg B_{m+1}\vartheta), \ldots, \overline{I}(\neg B_n\vartheta)), r) \\
&= \mathcal{C}(\mathcal{B}(I(B_1\vartheta), \ldots, I(B_m\vartheta), -I(B_{m+1}\vartheta), \ldots, -I(B_n\vartheta)), r) \\
&= \mathcal{C}(\overline{I}(B[B_1\vartheta, \ldots, B_m\vartheta, -I(B_{m+1}\vartheta), \ldots, -I(B_n\vartheta)]), r)
\end{aligned}$$

By taking suprema for all the ground instances, we have:

$$\begin{aligned}
&\oplus\{\mathcal{C}(\overline{I}(B[B_1\vartheta, \ldots, B_m\vartheta, \neg B_{m+1}\vartheta, \ldots, \neg B_n\vartheta]), r)\} \\
&= \oplus\{\mathcal{C}(\overline{I}(B[B_1\vartheta, \ldots, B_m\vartheta, -I(B_{m+1}\vartheta), \ldots, -I(B_n\vartheta)]), r)\}
\end{aligned}$$

Therefore, $T_P(I) = T_{P^I}(I)$ for every interpretation I.

Now, let M be a stable model of P. By definition, M is the least model of the extended positive program P^M. By Theorem 2, M is the least fixpoint of T_{P^M}, so $M = T_{P^M}(M) = T_P(M)$, i.e., M is a fixpoint of T_P. It remains to show that M is a minimal fixpoint of T_P. Assume that there is a fixpoint N of T_P such that $N \sqsubseteq M$. By Theorem 5, N is a model of P. Moreover, by Theorem 4, M is a minimal model of P, so we have $M = N$. □

As shown in the following example, for a normal program P, T_P may have many minimal fixpoints, which might not be obtained by the bottom-up iteration of T_P. Moreover, a minimal fixpoint of T_P may not be a stable model of P.

Example 4. Consider a normal program P consisting of the following rules:

$$\begin{aligned}
(good(X) &\leftarrow_G good(X).1) \\
(bad(X) &\leftarrow_G \neg good(X).1)
\end{aligned}$$

First, we determine stable models of P. Let $I = \{(good(a), x), (bad(a), y)\}$, where x and y are truth values, be an interpretation of P. The reduct P^I is as follows:

$$\begin{aligned}
(good(a) &\leftarrow_G good(a).1) \\
(bad(a) &\leftarrow_G -x.1)
\end{aligned}$$

The least model of P^I is $M_{P^I} = \{(good(a), 0), (bad(a), -x)\}$. Thus, I is a stable model of P iff $I = M_{P^I}$, i.e., $I = \{(good(a), 0), (bad(a), 1)\}$.

Finally, we determine the set of fixpoints of T_P. Given an interpretation $I = \{(good(a), x), (bad(a), y)\}$, we have $T_P(I) = \{(good(a), x), (bad(a), -x)\}$. Hence, I is a fixpoint of T_P iff $I = \{(good(a), x), (bad(a), -x)\}$, for all truth values x. It can be seen that all such fixpoints are minimal, but only the interpretation $\{(good(a), 0), (bad(a), 1)\}$ is a stable model.

4 Related Work

Of the various approaches to the management of negation in logic programming, the stable model semantics approach, introduced by Gelfond and Lifschitz [11], has become one of the most widely studied and commonly adopted proposal. There are several works on fuzzy logic programming with negation. The authors of [13] and [14] deal with normal *propositional* residuated logic programs and normal *propositional* multi-adjoint logic programs, respectively. In addition to studying the fixpoint characterization of the stable model semantics, they show that there exists a stable model for such finite programs if the truth domain is a convex compact set in an euclidean space and truth functions of all connectives except the implication are continuous. They also give sufficient conditions for the unicity of stable models for several particular truth domains. Nevertheless, the existence of stable models of normal fuzzy linguistic logic programs seems to be more complicated due to the fact that our truth domains are discrete.

Besides the stable model semantics, there are several other approaches to studying the semantics of normal programs as follows:

(i) define the *compromise semantics* [15] based on van Gelder's *alternating fixpoint approach* [16]. The *binary* immediate consequence operator $T_P(I, J)$ for a normal program P is an extension of the one for positive programs in which the interpretation I (resp., J) is used to assign meaning to positive literals (resp., negative literals); $T_P(I, J)$ is similar to the operator $\Psi_P(I, J)$ in [17,18]. T_P is continuous in the first argument and anti-monotone in the second. Then, the operator $S_P(J)$ is defined as the least fixpoint of $T_P(I, J)$, i.e., the least model of P^J, denoted $S_P(J) = T_P^{\infty}(\bot, J)$. Since S_P is anti-monotone, $S_P \circ S_P$ is monotone. It is known that S_P has two *extreme oscillation points* $S_P^{\bot} = (S_P \circ S_P)^{\infty}(\bot)$ and $S_P^{\top} = (S_P \circ S_P)^{\infty}(\top)$, where $\top$ is the greatest interpretation mapping every ground atom to 1, and $S_P^{\bot} \sqsubseteq S_P^{\top}$. As in [16], $S_P^{\bot}$ and $S_P^{\top}$ are an under-estimation and an over-estimation of the semantics of P, respectively. The *compromise semantics* of P is defined as a *partial* interpretation $CS(P) = S_P^{\bot} \cap S_P^{\top}$. This semantics coincides with the *well-founded semantics* for traditional Datalog programs with negation [19].

(ii) define different semantics, e.g., the stable model semantics and the well-founded semantics, of normal programs over bilattices [20], in which the truth domain is a complete lattice under the truth ordering as well as the knowledge ordering. For example, in *interval* bilattices, an element is greater than another under the knowledge ordering if the former is more precise than the latter. Following this idea, it can be seen that all linguistic truth values generated from x using hedges are greater than x in the knowledge ordering since they are more precise values than x; for instance, we have $True \leq_k VeryTrue \leq_k SlightlyVeryTrue$, where $\leq_k$ is the knowledge order relation. Thus, an LTD is also a poset under $\leq_k$. However, it is still not a complete lattice under $\leq_k$ since it does not have a top element and a bottom element.

5 Conclusion and Future Work

In this paper, we extend fuzzy linguistic logic programming with negation and present the stable model semantics of normal fuzzy linguistic logic programs, called *normal programs*. More precisely, a stable model of a normal program is obtained according to the usual Gelfond-Lifschitz transformation. We prove that a stable model of a normal program P is a minimal model of P and a minimal fixpoint of the intermediate consequence operator T_P. We also show that a normal program P can have multiple stable models and a minimal fixpoint of T_P may not be a stable model of P. For future work, we will study the conditions for the existence and unicity of stable models of normal programs and a method to compute such models.

Acknowledgment. This research is funded by Vietnam National Foundation for Science and Technology Development (NAFOSTED) under grant number 105.08-2018.09.

References

1. Le, V.H., Liu, F., Tran, D.K.: Fuzzy linguistic logic programming and its applications. Theor. Pract. Logic Programm. **9**(3), 309–341 (2009)
2. Le, V.H., Tran, D.K.: Further results on fuzzy linguistic logic programming. J. Comput. Sci. Cybern. **30**(2), 139–147 (2014)
3. Le, V.H., Liu, F.: Tabulation proof procedures for fuzzy linguistic logic programming. Int. J. Approximate Reasoning **63**, 62–88 (2015)
4. Nguyen, C.H., Wechler, W.: Hedge algebras: an algebraic approach to structure of sets of linguistic truth values. Fuzzy Sets Syst. **35**, 281–293 (1990)
5. Nguyen, C.H., Wechler, W.: Extended hedge algebras and their application to fuzzy logic. Fuzzy Sets Syst. **52**, 259–281 (1992)
6. Le, V.H., Tran, D.K.: Extending fuzzy logics with many hedges. Fuzzy Sets Syst. **345**, 126–138 (2018)
7. Le, V.H., Liu, F., Tran, D.K.: Mathematical fuzzy logic with many dual hedges. In: The Fifth Symposium on Information and Communication Technology (SoICT), pp. 7–13 (2014)
8. Zadeh, L.A.: A theory of approximate reasoning. In: Hayes, J.E., Michie, D., Mikulich, L.I., (eds.) Machine Intelligence, vol. 9, pp. 149–194. Wiley (1979)
9. Bellman, R.E., Zadeh, L.A.: Local and fuzzy logics. In: Fuzzy Sets, Fuzzy Logic, and Fuzzy Systems: Selected Papers by Lotfi A. Zadeh, pp. 283–335. World Scientific Publishing Co., Inc, River Edge (1996)
10. Hájek, P.: Metamathematics of Fuzzy Logic. Kluwer, Dordrecht (1998)
11. Gelfond, M., Lifschitz, V.: The stable model semantics for logic programming. In: Proceedings of the 5th International Conference on Logic Programming, pp. 1070–1080. MIT Press, Cambridge (1988)
12. Davey, B.A., Priestley, H.A.: Introduction to Lattices and Order. Cambridge University Press, Cambridge (2002)
13. Madrid, N., Ojeda-Aciego, M.: On the existence and unicity of stable models in normal residuated logic programs. Int. J. Comput. Math. **89**(3), 310–324 (2012)
14. Cornejo, M.E., Lobo, D., Medina, J.: Syntax and semantics of multi-adjoint normal logic programming. Fuzzy Sets Syst. **345**, 41–62 (2018)

15. Loyer, Y., Straccia, U.: The well-founded semantics in normal logic programs with uncertainty. In: Hu, Z., Rodríguez-Artalejo, M. (eds.) FLOPS 2002. LNCS, vol. 2441, pp. 152–166. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-45788-7_9
16. van Gelder, A.: The alternating fixpoint of logic programs with negation. In: PODS 1989: Proceedings of the 8th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, pp. 1–10. ACM, New York (1989)
17. Fitting, M.: Fixpoint semantics for logic programming: a survey. Theor. Comput. Sci. **278**(1–2), 25–51 (2002)
18. Fitting, M.: The family of stable models. J. Logic Programm. **17**(2/3&4), 197–225 (1993)
19. van Gelder, A., Ross, K.A., Schlipf, J.S.: The well-founded semantics for general logic programs. J. ACM **38**(3), 619–649 (1991)
20. Fitting, M.: Bilattices and the semantics of logic programming. J. Logic Programm. **11**(2), 91–116 (1991)