Journal of Computer Science and Cybernetics, V.30, N.2 (2014), 139–147

FURTHER RESULTS ON FUZZY LINGUISTIC LOGIC PROGRAMMING*

VAN HUNG LE[1], DINH KHANG TRAN[2]

[1]*Faculty of Information Technology, Hanoi University of Mining and Geology, Vietnam; levanhung@humg.edu.vn*

[2]*School of Information and Communication Technology, Hanoi University of Science and Technology, Vietnam; khangtd@soict.hust.edu.vn*

Tóm tắt. Lập trình logic mờ ngôn ngữ được đề xuất cho việc biểu diễn và suy luận với tri thức con người phát biểu bằng ngôn ngữ, trong đó giá trị chân lý của các phát biểu mờ được cho bằng các từ ngôn ngữ và các gia tử có thể được dùng để thể hiện các mức độ nhấn mạnh khác nhau. Lập trình logic mờ ngôn ngữ có các khái niệm và kết quả căn bản như ngữ nghĩa mô tả, ngữ nghĩa thủ tục và ngữ nghĩa điểm bất động. Ngữ nghĩa thủ tục của nó là đúng đắn, đầy đủ và có thể tính toán trực tiếp trên ngôn ngữ để tìm trả lời cho các truy vấn. Trong bài báo này, chúng tôi sẽ chứng minh một số kết quả bổ sung của lập trình logic mờ ngôn ngữ tương ứng với các kết quả quan trọng trong lập trình logic truyền thống. Chúng tôi cũng chỉ ra rằng nó có tính đầy đủ dạng Pavelka mở rộng. Ngoài ra, khả năng sử dụng các toán tử kết hợp ở thân luật cũng được thảo luận.

Từ khóa. Lập trình logic, logic mờ, đại số gia tử, tính toán với từ, tính đầy đủ.

Abstract. Fuzzy linguistic logic programming is introduced to represent and reason with linguistically-expressed human knowledge, where the truth of vague sentences is given in linguistic terms, and linguistic hedges can be used to indicate different levels of emphasis. Fuzzy linguistic logic programming has been shown to have fundamental notions and results of a logic programming framework, especially of the declarative semantics, procedural semantics, and fixpoint semantics. The procedural semantics are sound, complete and directly manipulates linguistic terms in order to compute answers to queries. In this paper, we prove some additional results of fuzzy linguistic logic programming, which can be considered as a counterpart of those of traditional definite logic programming. We also show that it has a generalised Pavelka-style completeness. Moreover, the possibility that aggregation operators can occur in rule bodies is also discussed.

Key words. Logic programming, fuzzy logic, hedge algebra, computing with words, completeness.

1. INTRODUCTION

Fuzzy linguistic logic programming (FLLP) [1], developed from fuzzy logic programming [2], is introduced for representing and reasoning with linguistically-expressed human knowledge. FLLP is a many-valued logic programming framework without negation. In FLLP, each

*This paper is sponsored by Vietnam National Foundation for Science and Technology Development (NAFOSTED) under Grant Number 102.04-2013.21.

fact or rule is graded to a certain degree specified by a linguistic truth value, and hedges can be used as unary connectives in rule bodies. Many fundamental notions and results of traditional definite logic programming (TDLP) [3] can have a counterpart in the framework. FLLP can be applied to deductive databases [4]. Other logic programming frameworks developed in a similar approach include multi-adjoint logic programming [5].

Fuzzy logic in the narrow sense (FLn) [6, 7] is a branch of many-valued logic developed for a paradigm of *inference under vagueness*. Almost all systems of FLn are truth functional. Rational Pavelka logic (RPL) [6] is a simplified version of Pavelka logic [8]. RPL is a system of FLn where truth functions of conjunction and implication are Łukasiewicz t-norm and its residuum. Each evaluation e of propositional variables by truth values in [0,1] uniquely extends to an evaluation $e(\varphi)$ of all formulae φ using the truth functions. Formula φ is called an *1-tautology* if $e(\varphi) = 1$ for each evaluation e. Several 1-tautology formulae are taken as *axioms*. A *theory* is a set of formulae. Evaluation e is called a *model* of a theory T if $e(\varphi) = 1$ for all φ in T. The *deduction rule* of RPL is modus ponens. A *proof* in a theory T is a sequence $\varphi_1, \ldots, \varphi_n$ of formulae whose each member is either an axiom of RPL or a member of T or follows from some preceding members of the sequence; φ_n is called a *provable* formula, denoted $T \vdash \varphi_n$. In the graded approach to syntax, a *graded formula* (φ, r), which is just another notation for the formula $\bar{r} \to \varphi$, states that the truth value of φ is at least r. The deduction rule, called *many-valued modus ponens*, is as follows: if $T \vdash (\varphi, r)$ and $T \vdash (\varphi \to \psi, s)$, then $T \vdash (\psi, r * s)$, where * is Łukasiewicz t-norm. The *truth degree* of a formula φ over a theory T is defined as $||\varphi||_T = \inf\{e(\varphi) | e$ is a model of $T\}$, and the *provability degree* of φ is $|\varphi|_T = \sup\{r | T \vdash (\varphi, r)\}$. It is proved that for each theory T and each formula φ, the truth degree and the provability degree of φ coincide. This result is usually referred to as *Pavelka-style completeness*, one of the most important completeness results in FLn [9, 10].

In addition to the results proved in [1], this paper will show that FLLP has a counterpart of a number of important results of TDLP, e.g., the model intersection property. Also, the completeness of the procedural semantics of FLLP can be seen as a generalised Pavelka-style completeness if one considers FLLP as an FLn system and its computation as a proof. Moreover, aggregation operators can occur in rule bodies, enabling us to describe increased fulfillment of user requirements. The remainder of this paper is organised as follows. Section 2 gives an overview of FLLP. Section 3 proves a number of additional results. Section 4 discusses the possibility of using aggregation operators in rule bodies. Section 5 concludes the paper.

2. FUZZY LINGUISTIC LOGIC PROGRAMMING

2.1. Linguistic truth domains and operations

Values of the linguistic variable *Truth*, e.g., *True* and *VeryLittleFalse*, can be characterised by a hedge algebra (HA) $\underline{X} = (X, G, H, \leq)$, where X is a term set and $\leq$ is its *semantic order relation*[11, 12]. An *l-limit* HA is a linear HA in which every term has a length of at most $l + 1$. A *linguistic truth domain* is a finite and linearly ordered set $\overline{X} = X \cup \{0, W, 1\}$, where X is the term set of an l-limit HA, and W is the *middle truth value*[1]. Operations are defined on $\overline{X} = \{v_0, \ldots, v_n\}$ with $v_0 \leq v_1 \leq \cdots \leq v_n$ as follows: (i) *conjunction*: $x \wedge y = \min(x, y)$; (ii) *disjunction*: $x \vee y = \max(x, y)$; (iii) A non-decreasing *inverse mapping* h^- for each hedge h; (iv) Łukasiewicz t-norm and its residuum are respectively defined as:

$$\mathcal{C}_L(v_i, v_j) = v_{\max(i+j-n, 0)}, \quad \overset{\bullet}{\leftarrow}_L (v_j, v_i) = v_{\min(n, n+j-i)};$$

and (v) Gödel t-norm and its residuum are respectively defined as:

$$\mathcal{C}_G(v_i, v_j) = \min(v_i, v_j), \quad \leftarrow^{\bullet}_G (v_j, v_i) \quad = \quad \begin{cases} v_n & \text{if } i \leq j \\ v_j & \text{otherwise.} \end{cases}$$

Each t-norm and its residuum satisfy the *residuation property* [6]:

$$\mathcal{C}(b, r) \leq h \text{ iff } r \leq \leftarrow^{\bullet} (h, b) \tag{1}$$

2.2. Language

The language is a predicate language without function symbols. Connectives can be conjunctions $\wedge$ (Gödel) and $\wedge_L$ (Łukasiewicz); the disjunction $\vee$; implications $\leftarrow_L$ (Łukasiewicz) and $\leftarrow_G$ (Gödel); and hedges. For a binary connective c, its truth function is denoted by $c^{\bullet}$, and for a hedge connective h, its truth function is its inverse mapping h^{-}.

A *term* is either a constant or a variable. An *atom* is of the form $p(t_1, ..., t_n)$, where p is an n-ary predicate symbol, and $t_1, ..., t_n$ are terms of corresponding attributes. 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 different from 0; $(A \leftarrow B)$ is called the *logical part* of the rule. A *fact* is a graded atom $(A.t)$, where A is an atom called the logical part of the fact, and t is a truth value different from 0. All variables are assumed to be universally quantified. A *fuzzy linguistic logic program* (program, for short) is a finite set of rules and facts. The truth value t in $(\varphi.t)$ is understood as a lower bound to the exact truth value of φ. A program P can be represented as a partial mapping $P : Formulae \rightarrow \overline{X} \setminus \{0\}$, where the domain of P, denoted $dom(P)$, is finite and consists only of logical parts, and $\overline{X}$ is the linguistic truth domain. For each $(\varphi.t) \in P$, $P(\varphi) = t$. We refer to the *Herbrand base* of P by B_P [3].

2.3. Declarative semantics

Let P be a program, and $\overline{X}$ the linguistic truth domain; an *Herbrand interpretation* f of P is a mapping from B_P to $\overline{X}$; 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) $\overline{f}(c(B_1, B_2)) = c^{\bullet}(\overline{f}(B_1), \overline{f}(B_2))$ and $\overline{f}(hB) = h^{-}(\overline{f}(B))$, where B_1, B_2, B are ground formulae, c is a binary connective, and h is a hedge connective. For non-ground formulae, $\overline{f}$ is defined as $\overline{f}(\varphi) = \overline{f}(\forall\varphi) = \inf\{\overline{f}(\varphi\vartheta) | \varphi\vartheta \text{ is a ground instance of } \varphi\}$, where $\forall\varphi$ denotes the *universal closure* of φ. An interpretation f is an *Herbrand model* of P if for all $\varphi \in dom(P)$, $\overline{f}(\varphi) \geq P(\varphi)$. A *query* is an atom used as a question $?A$. A pair $(x; \theta)$, where $x \in \overline{X}$, and θ is a substitution, is called a *correct answer* for P and a query $?A$ if for every model f of P, we have $\overline{f}(A\theta) \geq x$.

2.4. Procedural semantics

Admissible rules are defined as follows:
Rule 1. From $((XA_mY); \vartheta)$ infer $((X\mathcal{C}(B, r)Y)\theta; \vartheta\theta)$ if (1) A_m is an atom; (2) θ is an mgu of A_m and A; and (3) $(A \leftarrow B.r)$ is a rule in the program.
Rule 2. From (XA_mY) infer $(X0Y)$. This rule is usually used for situations where A_m does not unify with any rule head or logical part of facts.

Rule 3. From $(XhBY)$ infer $(Xh^-(B)Y)$ if B is a body formula, h is a hedge connective.
Rule 4. From $((XA_mY);\vartheta)$ infer $((XrY)\theta;\vartheta\theta)$ if (1) A_m is an atom; (2) θ is an mgu of A_m and A; and (3) $(A.r)$ is a fact in the program.
Rule 5. If there are no more predicate symbols and hedge connectives in the expression, replace all connectives $\wedge$'s, and $\vee$'s with $\wedge^\bullet$, and $\vee^\bullet$, respectively, and then evaluate it to obtain a truth value. The substitution remains unchanged.

A pair $(r;\theta)$, where r is a truth value, and θ is a substitution, is said to be a *computed answer* for a program P and a query $?A$ if there is a sequence $G_0,\ldots,G_n$ such that (1) every G_i is a pair consisting of an expression and a substitution; (2) $G_0=(A;id)$ (id is the *identity* (*empty*) substitution); (3) every G_{i+1} is inferred from G_i by one of the admissible rules; and (4) $G_n=(r;\theta')$ and $\theta=\theta'$ restricted to variables of A.

Example 2.1. Assume that we use the linguistic truth domain taken from the 2-limit HA $\underline{X}=(X,\{F,T\},\{V,M,R,L\},\leq)$, where F, T, V, M, R, and L stand for *False, True, Very, More, Rather*, and *Little*, respectively, and there is a piece of knowledge as follows: (i) "If a student studies *very hard*, and his/her university is *rather high-ranking*, then he/she will be a *good* employee" is *Very More True*; (ii) "The university where Ann is studying is *high-ranking*" is *Very True*; and (iii) "Ann is studying *hard*" is *More True*. Let *gd_ em, st_ hd*, and *hira_ un* stand for "good employee", "study hard", and "high-ranking university", respectively. The piece of knowledge can be represented by the following program:

$$\begin{aligned}(gd\_em(X) \leftarrow_G \wedge(V\ st\_hd(X), R\ hira\_un(X)).VMT)\\ (hira\_un(ann).VT)\\ (st\_hd(ann).MT)\end{aligned}$$

Given a query $?gd\_em(ann)$, we have the following computation (the substitution is id):

$$\begin{aligned}?gd\_em(ann)\\ \mathcal{C}_G(\wedge(V\ st\_hd(ann), R\ hira\_un(ann)), VMT)\\ \mathcal{C}_G(\wedge(V^-(st\_hd(ann)), R\ hira\_un(ann)), VMT)\\ \mathcal{C}_G(\wedge(V^-(st\_hd(ann)), R^-(hira\_un(ann))), VMT)\\ \mathcal{C}_G(\wedge(V^-(MT), R^-(hira\_un(ann))), VMT)\\ \mathcal{C}_G(\wedge(V^-(MT), R^-(VT)), VMT)\\ \mathcal{C}_G(\wedge^\bullet(V^-(MT), R^-(VT)), VMT)\\ RT\end{aligned}$$

That is, "Ann will be a good employee" is at least *Rather True*.

Theorem 2.1 (Soundness of the procedural semantics) *[1] Every computed answer for a program P and a query $?A$ is a correct answer for P and $?A$.*

Theorem 2.2. *[1] For every correct answer $(x;id)$ of a program P and a ground query $?A$, there exists a computed answer $(r;id)$ for P and $?A$ such that $r\geq x$.*

Theorem 2.3 (Completeness of the procedural semantics) *[1] Let P be a program, and $?A$ a query. For every correct answer $(x;\theta)$ for P and $?A$, there exists a computed answer $(r;\sigma)$ for P and $?A$, and a substitution γ such that $r\geq x$ and $\theta=\sigma\gamma$.*

The completeness of the procedural semantics states that given a correct answer for a query, we always have a computed answer which is *more general* than the correct answer.

2.5. Fixpoint semantics

Let P be a program. The immediate consequence operator T_P mapping from interpretations to interpretations is defined as follows: for an interpretation f and every ground atom $A \in B_P$, $T_P(f)(A) = \max\{\sup\{\mathcal{C}_i(\overline{f}(B), r)|\ (A \leftarrow_i B.r)$ is a ground instance of a rule in $P\}$, $\sup\{t|(A.t)$ is a ground instance of a fact in $P\}\}$. It is shown in [1] that the least Herbrand model of the program P is exactly the least fixpoint of T_P and can be obtained by finitely iterating T_P from the *bottom interpretation* $\perp$, mapping every ground atom into 0.

3. ADDITIONAL RESULTS OF FUZZY LINGUISTIC LOGIC PROGRAMMING

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) $(f_1 \oplus f_2)(A) = f_1(A) \oplus f_2(A)$.

Proposition 3.1. *Let $\mathcal{F}_P$ be the set of all interpretations of a program P. Then $\langle \mathcal{F}_P, \otimes, \oplus \rangle$ is a complete lattice.*

Proof. We show that for any subset F of $\mathcal{F}_P$, $\otimes F$ and $\oplus F$ exist. For all $A \in B_P$, $(\otimes F)(A) = \otimes\{f(A)|f \in F\}$. It suffices that the set of all truth values is a complete lattice for $\otimes\{f(A)|f \in F\}$ to exist, and a finite and linearly ordered linguistic truth domain is obviously a complete lattice. The case of $\oplus$ is similar.

Lemma 3.1. *Let f_1 and f_2 be two interpretations of a program P such that $f_1 \sqsubseteq f_2$. For any ground body formula B, we have $\overline{f_1}(B) \leq \overline{f_2}(B)$.*

Proof. The lemma is proved by induction on the structure of B. In the base case, where B is a ground atom, we have $\overline{f_1}(B) = f_1(B) \leq f_2(B) = \overline{f_2}(B)$. For the inductive case, by case analysis and induction hypothesis, we have $B = \wedge(B_1, B_2)$, or $B = \vee(B_1, B_2)$, or $B = hB_1$ such that $\overline{f_1}(B_1) \leq \overline{f_2}(B_1)$ and $\overline{f_1}(B_2) \leq \overline{f_2}(B_2)$. By definition, $\overline{f_1}(B) = \wedge^\bullet(\overline{f_1}(B_1), \overline{f_1}(B_2)) \leq \wedge^\bullet(\overline{f_2}(B_1), \overline{f_2}(B_2)) = \overline{f_2}(B)$, or $\overline{f_1}(B) = \vee^\bullet(\overline{f_1}(B_1), \overline{f_1}(B_2)) \leq \vee^\bullet(\overline{f_2}(B_1), \overline{f_2}(B_2)) = \overline{f_2}(B)$, or $\overline{f_1}(B) = h^-(\overline{f_1}(B_1)) \leq h^-(\overline{f_2}(B_1)) = \overline{f_2}(B)$, respectively, since truth functions of all connectives in rule bodies are monotone in all arguments. This completes the proof of the lemma.

The following theorem is the counterpart of the model intersection property in TDLP [3].

Theorem 3.1. *Let P be a program, and F a non-empty set of Herbrand models of P. Then $\otimes F$ is an Herbrand model of P.*

Proof. Since $\otimes F$ always exists, we put $g = \otimes F$. Let φ be any formula in $dom(P)$. There are two cases: (i) $(\varphi.t)$, where t is a truth value, is a fact in P. For each ground instance A of φ and each model $f \in F$ of P, by hypothesis, we have $f(A) \geq t$. Therefore, $g(A) = \otimes\{f(A)|f \in F\} \geq t$. Then $\overline{g}(\varphi) = \otimes\{g(A)|A$ is a ground instance of $\varphi\} \geq t = P(\varphi)$; or (ii) $(\varphi.t)$ is a rule in P. For each ground instance $A \leftarrow_i B$ of φ and each model $f \in F$, by hypothesis, we have $\overline{f}(A \leftarrow_i B) = \leftarrow_i^\bullet (f(A), \overline{f}(B)) \geq t$. By the residuation property

(1), we have $f(A) \geq \mathcal{C}_i(\overline{f}(B), t) \geq^{(*)} \mathcal{C}_i(\overline{g}(B), t)$, where (*) follows from Lemma 3.1. Thus $g(A) = \otimes\{f(A)|f \in F\} \geq \mathcal{C}_i(\overline{g}(B), t)$. By the residuation property again, we have $\overline{g}(A \leftarrow_i B) = \leftarrow_i^{\bullet} (g(A), \overline{g}(B)) \geq t$. Therefore, $\overline{g}(\varphi) = \otimes\{\overline{g}(A \leftarrow_i B)|A \leftarrow_i B$ is a ground instance of $\varphi\} \geq t = P(\varphi)$.

By definition, g is a model of P.

Consider the *top interpretation* $\top$ of a program P which maps every $A \in B_P$ to 1. For any $\varphi \in dom(P)$, it is easily verified that $\overline{\top}(\varphi) = 1 \geq P(\varphi)$. Thus, $\top$ is an Herbrand model of P, and the set of all Herbrand models of P is non-empty.

The following theorem follows immediately from Theorem 3.1 and the definition of the least model.

Theorem 3.2. *Let P be a program. Then $M_P = \otimes\{f|f$ is an Herbrand model of $P\}$ is the least Herbrand model of P.*

Hence, the least Herbrand model of a logic program can be characterised by the greatest lower bound of the set of all its Herbrand models. As in TDLP, M_P can be regarded as the natural interpretation of P, which gives intuitive description of the meaning of P.

The following proposition shows that if we consider FLLP as a system of FLn and the program P as a fuzzy theory, then for each body formula φ, $\overline{M_P}(\varphi)$ is the *truth degree* of φ over P in the sense of Pavelka [8, 6].

Proposition 3.2. *Let P be a program. For every body formula φ, $\overline{M_P}(\varphi) = \otimes\{\overline{f}(\varphi)|f$ is an Herbrand model of $P\}$.*

Proof. Let f be any Herbrand model of P. For each ground instance $\varphi\vartheta$ of φ, by Lemma 3.1, we have $\overline{M_P}(\varphi\vartheta) \leq \overline{f}(\varphi\vartheta)$. Thus, $\overline{M_P}(\varphi) = \otimes\{\overline{M_P}(\varphi\vartheta)|\varphi\vartheta$ is a ground instance of $\varphi\} \leq \otimes\{\overline{f}(\varphi\vartheta)|\varphi\vartheta$ is a ground instance of $\varphi\} = \overline{f}(\varphi)$. Since f is arbitrary, we have $\overline{M_P}(\varphi) \leq \otimes\{\overline{f}(\varphi)|f$ is an Herbrand model of $P\}$, and thus $\overline{M_P}(\varphi) = \otimes\{\overline{f}(\varphi)|f$ is an Herbrand model of $P\}$. ■

On one hand, Theorem 2.3 can be considered as a Pavelka-style completeness in a general sense. On the other hand, we can also have a strict Pavelka-style completeness [8, 6] for ground atoms as follows.

Proposition 3.3. *Let P be a program. For every ground atom A, $M_P(A) = \oplus\{r|(r; id)$ is a computed answer for P and $?A\}$.*

Proof. Since A is ground, every computed answer for $?A$ is of the form $(r; id)$, and the set $\{(r; id)$ is a computed answer for $?A\}$ consists of all computed answers for $?A$. For each computed answer $(r; id)$ for $?A$, by Theorem 2.1, $(r; id)$ is also a correct answer for $?A$, thus $M_P(A) \geq r$. Therefore, $M_P(A) \geq \oplus\{r|(r; id)$ is a computed answer for $?A\}$. On the other hand, since $(M_P(A); id)$ is a correct answer for $?A$, by Theorem 2.2, there exists a computed answer $(r'; id)$ for $?A$ such that $r' \geq M_P(A)$. Hence $M_P(A) = \oplus\{r|(r; id)$ is a computed answer for $?A\}$. ■

Therefore, if one considers FLLP as an FLn system and a computation as a proof, Proposition 3.3 states that $M_P(A)$ is the *provability degree* of atom A in the sense of Pavelka, which, together with Proposition 3.2, establishes a strict Pavelka-style completeness for ground atoms. However, we do not have a similar result for non-ground atoms as shown in the following example.

Example 3.1. Consider a simple program P consisting of two ground facts: $(p(a).VT)$ and $(p(b).T)$. It is easily verified that $\overline{M_P}(p(X)) = \otimes\{M_P(p(a)), M_P(p(b))\} = T$, where X is a variable. However, since $?p(X)$ has a computed answer $(VT; \{X/a\})$, $\oplus\{r|(r;\sigma)$ is a computed answer for P and $?p(X)\} = VT \neq \overline{M_P}(p(X))$.

The reason is that in logic programming, besides the truth values, one also considers the substitutions, which are an important part of computations.

It can be seen that an arbitrary body formula may occur as a query in our framework, and if so, the same result can be obtained for ground ones.

4. USING AGGREGATION OPERATORS IN RULE BODIES

Recall that in fuzzy logic programming [2], body formulae can be built using aggregation operators, which subsume all kinds of fuzzy conjunctions and disjunctions. Aggregation operators are very useful since they enable us to describe increased fulfillment of user requirements. A conjunction is one extreme case where one desires that all the criteria be satisfied, and a disjunction is the other extreme where the satisfaction of any of the criteria is all one needs. In this section, we discuss the possibility of extending our rule bodies with aggregation connectives whose truth functions can directly act on linguistic truth values.

In the literature, there are several kinds of aggregation operators which can directly compute with linguistic labels. The most well-known example is the LOWA (Linguistic Ordered Weighted Averaging) operator [13]. The LOWA operator is developed based on the ordered weighted averaging (OWA) operator defined in [14] and the convex combination of linguistic labels defined in [15].

Definition 4.1 (LOWA operator) [13] Let $S = \{s_1, \dots, s_m\}$ be a set of linguistic terms to be aggregated, the LOWA operator $\phi(s_1, \dots, s_m) = C^m\{w_k, t_k, k = 1 \dots m\}$ is defined inductively as follows.

For $m = 2$,

$$C^2\{\{w_1, 1 - w_1\}, \{t_1, t_2\}\} = (w_1 \odot v_j) \oplus ((1 - w_1) \odot v_i) = v_k$$

where $t_1 = v_j, t_2 = v_i \in \overline{X}, j \geq i$, and $k = min\{n, i + round(w_1.(j - i))\}$, in which $n + 1$ is the cardinality of $\overline{X}$, $round(.)$ is the usual round operation.

For $m > 2$,

$$C^m\{w_k, t_k, k = 1 \dots m\} = C^2\{\{w_1, 1 - w_1\}, \{t_1, C^{m-1}\{\eta_h, t_h, h = 2 \dots m\}\}\}$$

where $W = [w_1, \dots, w_m]$ is a weighting vector associated with S such that: (i) $w_i \in [0, 1]$, and (ii) $\sum_{i=1}^m w_i = 1$; $T = [t_1, \dots, t_m]$ is a vector such that t_i is the ith largest element in the collection $s_1, \dots, s_m$; $\eta_h = w_h / \sum_2^m w_k, h = 2, \dots, m$.

A natural question arising is how to obtain the associated weighting vector. Yager [14] proposed an interesting way to compute the weights of the OWA operator using linguistic quantifiers. More precisely, if Q is a relative or proportional quantifier such as "*Most*", Q can be expressed by a fuzzy subset of $[0, 1]$ such that for each $r \in [0, 1]$, $Q(r)$ indicates the degree to which r portion of objects satisfies the concept denoted by Q. Then, the weights can be obtained by:

$$w_i = Q(i/n) - Q((i - 1)/n), i = 1, \dots, n$$

The membership function of such a quantifier Q can be:

$$Q(r)=\begin{cases} 0 & \text{if } 0 \leq r < a \\ \frac{r-a}{b-a} & \text{if } a \leq r \leq b \\ 1 & \text{if } 1 \geq r > b \end{cases}$$

where $0 \leq a \leq b \leq 1$.

Because of the non-decreasing nature of Q, it follows that $w_i \geq 0$. Furthermore, since $Q(1) = 1$ and $Q(0) = 0$, we have $\sum_{i=1}^{n} w_i = 1$. The use of such quantifiers to generate the weighting vector for the LOWA operator essentially implies that the more criteria are satisfied, the better the solution is.

Example 4.1. Assume that the truth domain in Example 2.1 is taken, and the quantifier "*Most*" with a=0.3 and b=0.8 is used to generate the weighting vector for the LOWA operator. The weighting vectors of dimension 3 and of dimension 2 are $[w_1 = 1/15, w_2 = 2/3, w_3 = 4/15]$ and $[w_1 = 2/5, w_2 = 3/5]$, respectively; we have $\phi(AT, LT, AMT) = \phi(v_{35}, v_{30}, v_{25}) = v_{29} = AAT$.

The LOWA operator has the following properties: (i) it is commutative, i.e., $\phi(s_1, \ldots, s_m) = \phi(\pi(s_1), \ldots, \pi(s_m))$, where π is a permutation over the set of arguments; (ii) it is non-decreasing in all arguments, i.e., given $S = [s_1, \ldots, s_m]$ and $T = [t_1, \ldots, t_m]$ being two vectors such that for all $i, s_i \geq t_i$, we have $\phi(S) \geq \phi(T)$; and (iii) it is an *or-and* operator, i.e., $\min(s_i) \leq \phi(s_1, \ldots, s_m) \leq \max(s_i)$.

Since in all the proofs, we only require that the truth function of a connective in body formulae be non-decreasing in all arguments, allowing body formulae to be built using a LOWA operator (i.e., we can have a rule $A \leftarrow @(B_1, \ldots, B_n)$, where the truth function $@^\bullet$ is a LOWA operator) does not affect any results of our framework at all.

5. CONCLUSION

In this paper, we have presented a number of additional results of FLLP including the counterpart of the model intersection property of TDLP and the characterisation of the least Herbrand model of a logic program by the greatest lower bound of the set of all its Herbrand models. We have also shown that the completeness of the procedural semantics of FLLP can be seen as a generalised Pavelka-style completeness if one considers FLLP as an FLn system and its computation as a proof; in fact, FLLP has a strict Pavelka-style completeness for ground queries, but not for non-ground ones. Moreover, aggregation operators can occur in rule bodies, which enables us to describe increased fulfillment of user requirements. Such results, together with the notions and results presented in [1], show that FLLP can be seen as an elegant and natural generalisation of TDLP for inference under vagueness since (1) it allows one to explicitly represent and reason with partial truth expressed in linguistic terms; (2) it has a counterpart of most fundamental notions and results of TDLP; and (3) it enjoys a generalised Pavelka-style completeness.

REFERENCES

[1] Le, V.H., Liu, F., Tran, D.K., Fuzzy linguistic logic programming and its applications, *Theory and Practice of Logic Programming* **9** (3) (2009) 309–341.

[2] Vojtáš, P., Fuzzy logic programming, *Fuzzy Sets and Systems* **124** (2001) 361–370.

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

[4] Le, V.H., Liu, F., Lu, H., A data model for fuzzy linguistic databases with flexible querying, In Nicholson, A.E., Li, X., eds., *Australasian Conference on Artificial Intelligence*, Volume **5866** of Lecture Notes in Computer Science, Springer (2009) 495–505.

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

[6] Hájek, P., *Metamathematics of Fuzzy Logic*, Kluwer, Dordrecht, The Netherlands, 1998.

[7] Novák, V., Perfilieva, I., Mockor, J., *Mathematical Principles of Fuzzy Logic,* Kluwer, Dordrecht, 2000.

[8] Pavelka, J., On fuzzy logic i, ii, iii, *Zeitschrift fur Mathematische Logik Grundlagen Math* **25** (1979) 45–52, 119–134, 447–464.

[9] Novák, V., Which logic is the real fuzzy logic?, *Fuzzy Sets and Systems* **157** (5) (2006) 635–641.

[10] Esteva, F., Godo, L., Noguera, C., First-order t-norm based fuzzy logics with truth-constants: Distinguished semantics and completeness properties, *Annals of Pure and Applied Logic* **161** (2) (2009) 185–202.

[11] Nguyen, C.H., Wechler, W., Hedge algebras: An algebraic approach to structure of sets of linguistic truth values, *Fuzzy Sets and Systems* **35** (1990) 281–293.

[12] Nguyen, C.H., Wechler, W., Extended hedge algebras and their application to fuzzy logic, *Fuzzy Sets and Systems* **52** (1992) 259–281.

[13] Herrera, F., Verdegay, J.L., Linguistic assessments in group decision, *Proceedings of the 1st European Congress on Fuzzy and Intelligent Technologies*, Aachen, Germany (1993) 941–948.

[14] Yager, R., On ordered weighted averaging aggregation operators in multicriteria decision making, *IEEE Transactions on Systems, Man and Cybernetics* **18** (1988) 183–190.

[15] Delgado, M., Verdegay, J., Vila, M., On aggregation operations of linguistic labels, *International Journal of Intelligent Systems* **8** (1993) 351–370.

Received on March 21, 2013
Revised on March 15, 2014