Resolution Method in Linguistic Propositional Logic

Thi-Minh-Tam Nguyen
Faculty of Information Technology,
Vinh University, VIETNAM

Duc-Khanh Tran
Center of Interdisciplinary Research
Ho Chi Minh University of
Technology (HUTECH), VIETNAM

***Abstract*—In the present paper, the resolution method for a linguistic propositional logic with truth value in a logical algebra - refined hedge algebra, is focused. The preliminaries of refined hedge algebra are given first. Then the syntax and semantic of linguistic propositional are defined. Finally, a resolution method which based on resolution principle in two-valued logic is established. Accordingly, the research in this paper will be helpful support for the application of intelligent reasoning system based on linguistic-valued logic which includes incomparable information.**

Keywords*—*Resolution; Linguistic Truth Value; Linguistic Propositional Logic; Hedge Algebra. .

I. Introduction

Resolution-based automated reasoning theory is an important and active research field in artificial intelligence. Since 1965, automated reasoning based on Robinson's resolution rule [1] has been extensively studied in the context of finding natural and efficient proof systems to support computational tasks. However, the resolution rule only applied in two-valued logic, so it cannot handle uncertain information, especially linguistic information. Many researchers have attent to find resolution principles in non-classical logics as effective as the resolution principles in two-valued logic.

The resolution rule, initially designed for two-valued logics by Robinson in 1965 [1], is the heart of many kinds of deductive systems such as theorem proving and logic programming. But resolution two-valued logics suffers from a drawback, it cannot handle uncertain or fuzzy information. Zadeh [2] introduced the fuzzy set theory and fuzzy logic to deal with uncertain reasoning. In fuzzy logic, a truth value domain is not the classical set $\{$False,True$\}$ or $\{0, 1\}$, but a set of linguistic truth values [3] or the whole unit interval [0,1]. Moreover, in fuzzy logic, linguistic hedges play an essential role in the generation of the values of a linguistic variable and in the modification of fuzzy predicates [4]. Substantial works have been done for finding resolution methods in fuzzy logic as effective as the resolution principle in two-valued logic [5], [6],[7],[8],[9].

The theory of hedge algebras (HAs), introduced in Nguyen and Wechler [10], forms an algebraic approach to a natural qualitative semantics of linguistic terms in a term domain. Hedge algebra is an algebraic approach to linguistic hedges in Zadeh's fuzzy logic [11], [12]. The hedge-algebra-based semantics of linguistic terms is qualitative, relative and dependent on the order-based structure of the term domain. HAs have been shown to have a rich algebraic structure to represent linguistic domains [11], and the theory can be effectively applied to problems such as linguistic reasoning [11] and fuzzy control [13]. Le et al. [14] introduced the fuzzy linguistic logic programming whose truth value domain is based on monotone symmetrical finite hedge algebras, then gave a procedural semantics based on many-valued modus ponens. Nguyen et al. [17],[18] presented linguistic logics with truth-valued domain based on linear symmetrical hedge algebra. Lai and Xu [15] presented a linguistic truth-valued lattice-valued propositional logic system, called *l*P(X)P(X), whose truth value domain is a lattice implication algebra. Liu et al. [16] proposed an automated reasoning algorithm based on the linguistic valued Lukasiewicz propositional logic with truth-value in Łukasiewicz linguistic valued algebras.

Hedge algebra structure is a complete lattice but not distributive [10], and hence composed linguistic terms are not able to be expressed in the disjunction and conjunction normal forms. In addition, the structure of such algebras is rather rough. For example, let us consider the set of all possible truth values T = $\{$true, false, very true, very false, approximately true, possibly true, approximately true or possibly true, approximately true and possibly true, etc.$\}$. It can be shown from [19] that the linguistic term "approximately true OR possibly true" will be expressed by "approximately true" $\cup$ "possibly true" and it equals to "true" in the structure of extended hedge algebra of the linguistic truth variable, where $\cup$ is the join operation of this algebra. This is clearly unsuitable in nature. To overcome this problem, the so-called refined hedge algebras have been developed, based on an extension of the axiomatic system of the hedge algebras. The results in [12] showed that refined hedge algebra (RHA) of a linguistic variable with a chain of the primary terms is a distributive lattice. The lattice operations join and meet can model the semantics of the logical disjunction and conjunction. RHAs with two antonymous primary terms are called symmetrical refined hedge algebras have a rich enough algebraic structure in order to be used as a truth value domain for logical systems. Starting from this observation, we have studied a linguistic proposition logic whose the truth value domain is generated by symmetrical refined hedge algebras.

In this work, we integrate resolution rule and RHAs to construct a logical system that facilitates the representation and reasoning on knowledge expressed in natural languages. In our logical system, the set of truth values is that of linguistic ones taken from an RHA of a linguistic truth variable. Furthermore, people use finitely many degrees of quality or quantity to describe real world applications which are granulated [20], so we consider only finitely many truth values. Since the truth value domain of the logic is generated by symmetrical refined hedge algebra we can express every logical formula in the disjunction and conjunction normal forms [12]. A resolution procedure is presented, simultaneously its soundness and completeness are proved. In addition, we introduce the notion of reliability to capture the approximate nature of resolution inference. A reliability of the conclusion of a resolution inference is a semantics value that is not greater than the reliability of premises of the resolution inference.

The paper is structured as follows: the next section gives some concepts and results about refined hedge algebra. Section III presents syntax and semantics of linguistic-valued propositional logic. Section IV presents the resolution rule and resolution procedure along with soundness and completeness results. The paper is concluded in Section V.

II. Preliminaries

In this section, we will present some elementary concepts, the details can be found in the Ref [10] and [12].

Let $AX = (X, G, LH, \leq)$ be an abstract algebra where X is the term set, G is the set of generators, H is the set of hedges or modifiers, and $\leq$ is partial order on X.

We assume that the set LH is decomposed into two subsets LH^+, LH^- such that $LH^+ + I$ and $LH^- + I$ are finite lattices with zero-element I. Then X and LH are said to be *semantically consistent* [12] if the following conditions hold:

1. X is generated from the generators in G by means of hedges in LH, i.e. elements of X are of the form $h_n \dots h_1 a$ for $h_i \in LH$, $i = 1, \dots, n$, and $a \in G$.
2. For any $h, k \in LH^+ + I$ (respectively $LH^- + I$, $h < k$ in $LH^+ + I$ (respectively $LH^- + I$) iff $\forall\ x \in X((hx > x$ or $kx > x$ implies $hx < kx)$ and $(hx < x$ or $kx < x$ implies $hx > kx))$. And h, k are incomparable in $LH^+ + I$ (respectively $LH^- + I$) iff $\forall\ x \in X)(hx \neq x$ or $kx \neq x)$ implies hx and kx are incomparable.

Example II.1. *Let $AX = (X, G, LH, \leq)$ be an abstract algebra where $G = \{\mathsf{True}, \mathsf{False}\}$, $H = \{Very, More, Approximately, Possibly, Less, Little\}$ is a set of linguistic hedges. Then, the poset of values of the linguistic variable Truth X is represented in Fig. 1. Intuitively, it can be seen that $H^+ = \{Very, More\}$ and $H^- = \{Less, Approximate, Possible, Little\}$; $H^+ + I$ and $H^- + I$ are lattices given in Fig. 2. The result of applying any operation h to an element x can be understood as follows: $hTrue$ and $hFalse$ are defined to be the elements given in Fig. 1; and khx=hx; for all $h, k \in H$ and $x \in X$. It can easily be seen that X and H are semantically consistent.*

AX is called a *refined hedge algebra* (RHA) [12], if X and LH are semantically consistent and the following conditions hold (where $h, k \in LH$):

(1) Every operation in LH^- is converse to each operation in LH^+.
(2) The unit operation V of $H^+ + I$ is either positive or negative w.r.t. any operation in H. In addition, H should satisfy the PN-homogeneous property.
(3) (Semantic independent property) If u and v are independent, i.e. $u \notin LH(v)$ and $v \notin LH(u)$, then $x \notin LH(v)$ for any $x \in LH(u)$ and vice versa. If $x \neq hx$ then $x \notin LH(hx)$. Further, if $hx \neq kx$ then kx and hx are independent.
(4) (Semantic inheritance) If hx and kx are incomparable, then same for elements $u \in LH(hx)$ and $v \in LH(kx)$. Especially, if a, $b \in G$ and $a < b$ then $LH(a) < LH(b)$. And if $hx < kx$ then
   i) In the case that $h, k \in LH_i^c$, for some $i \in SI^c$ the following statements hold:
      - $\delta hx < \delta kx$, for any $\delta \in LH^*$.
      - δhx and y are incomparable, for any $\forall y \in LH(kx)$ such that $y \not\geq \delta kx$.
      - δkx and z are incomparable, for any $\forall y \in LH(hx)$ such that $z \not\leq \delta hx$.
   ii) Otherwise, $h, k \notin LH_i^c$, then $h'hx \leq k'kx$, for every $h', k' \in UOS$.
(5) (Linear order between the graded classes) Assume that $u \in LH(x)$ and $u \notin LH(LH_i^c[x])$, for any $i \in I^c$. If exist $v \in LH(hx)$, for $h \in LH_i^c$ such that $u \geq v$ (or $u \leq v$), then $u \geq h'v$ (or $u \leq h'v$) for any $h' \in UOS$.

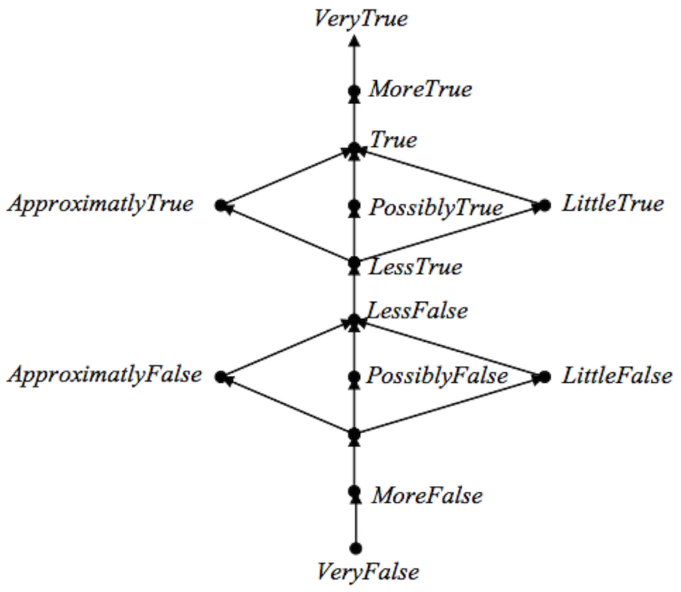

Fig. 1: A poset of values of the linguistic variable Truth.

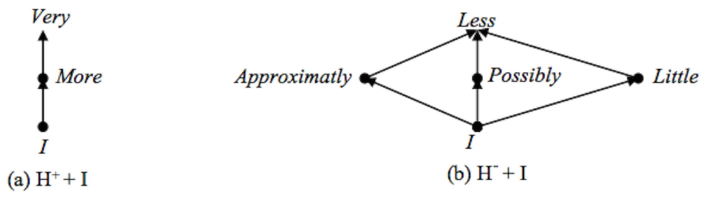

Fig. 2: Lattices of hedges.

In [12] show that if $AX = (X, G, LH, \leq)$ is an RHA where G is a totally ordered set, then AX is a distributive lattice. The condition that G is a totally ordered set is reasonable because on linguistic domain the sets of generators are often simple sets where all element are comparable (e.g. G includes two elements $\mathsf{True}, \mathsf{False}$). Operations of meet and join on AX lattice are determined recursively, based on the corresponding operations on hedge lattice LH.

Denote $LH[x] = \{hx : h \in LH + I\}$, where $x \in X$. It is proved that $LH[x]$ is a distributive sub-lattice of AX and operations of meet and join on it is defined as follows:

$$hx \vee kx = \begin{cases} (h \cup k)x & \text{if } hx \geq x, \\ (h \cap k)x & \text{if } hx \leq x \end{cases}$$

$$\text{and } hx \wedge kx = \begin{cases} (h \cap k)x & \text{if } hx \geq x, \\ (h \cup k)x & \text{if } hx \leq x \end{cases}$$

where $\cup, \cap$ are the join and meet operators of LH, and $\vee, \wedge$ are the join and meet operators of X.

In natural languages, there are many linguistic variables, which have only two distinct primary terms. These terms have intuitive contradictory meaning such as "true" and "false", "old" and "young", "tall" and "short", etc. Therefore, we consider hedge algebras have exactly two primary generators, one of which is called positive generator, and the other is called negative generator. It seems reasonable to consider "true", "old", and "tall" as positive generators and "false", "young", and "short" as negative ones. Note that the set of generators may contain special constants such as $\bot, \top$, and W which are different from the primary terms and understood as *"absolutely false"*, *"absolutely true"* and the *"neutral"*, respectively. These constants can be characterized by the conditions that $hc = c$ for all $h \in LH, c \in \{\bot, W, \top\}$ and $\bot < W < \top$. In this paper, we shall be working with RHA $AX = \{X, G, LH, \leq\}$, where $G = \{\bot, \mathsf{True}, \mathsf{W}, \mathsf{False}, \top\}$.

Let x be an element of the RHA AX, $x = h_n...h_1a$ is called the canonical representation of x where $a \in \{\mathsf{True}, \mathsf{False}\}$. The *contradictory element* of x, denoted by $\overline{x}$, is an element y such that $y = h_n...h_1a'$ where $a' \in \{\mathsf{True}, \mathsf{False}\}$ and $a' \neq a$. The contradictory element of $\top$ is $\bot$ and, conversely, the contradictory element of $\bot$ is $\top$. In the case where $x = W$, we define the contradictory element of W to be just itself. For example, $\overline{x}$ = *"VeryVeryFalse"* is a contradictory element of *x* = *"VeryVeryTrue"*; $\overline{y}$ = *"VeryLittleBad"* is a contradictory element of *u*=*"VeryLittleGood"*. By the definition, it is obvious that the positive generator is a contradictory element of the negative one and vice versa and if y is a contradictory element of x then x is a contradictory element of y.

Let $AX = (X, G, LH, \leq)$ be an RHA where $G = \{\bot, \mathsf{False}, \mathsf{W}, \mathsf{True}, \top\}$. Then, AX is said to be a *symmetrical RHA* provided every element $x \in X$ has a unique contradictory element in X.

It is useful to limit the set of values X only consists of finite length elements. This is entirely suitable with the practical application in natural language, which does not consider an infinite number of hedge of string. Denote $LH_p[G] = \{h_n...h_1a : h_i \in LH + I, a \in G, n \leq p\}$. A symmetrical RHA $AX = (LH_p[G], G, LH, \leq)$ is a complete distributive lattice.

From now on, we consider a symmetrical RHA $AX = (LH_p[G], G, LH, \leq, \neg, \vee, \wedge, \rightarrow)$, where $G = \{\bot, \mathsf{False}, \mathsf{W}, \mathsf{True}, \top\}$, $\bot, \top$ are fixed points, W is the neutral element, and $\bot < \mathsf{False} < \mathsf{W} < \mathsf{True} < \top$.

Let x and y be two elements of the symmetrical RHA AX, then

- the negation operator is an unary operator, which is defined by $\neg x = \overline{x}$, where $\overline{x}$ is the contradictory element of x,
- the implication operator is a binary operator, which is defined through negative and join operators: $x \rightarrow y = \neg x \vee y$.

Theorem II.1. *[12] Let $AX = (LH_p[G], G, LH, \leq)$ be a symmetrical RHA. Then, for all $x, y \in LH_p[G]$, for all $h, k \in LH$, we have:*

1) $\neg(hx) = h\neg x$.
2) $\neg(\neg x) = x$.
3) $\neg(x \vee y) = \neg x \wedge \neg y$ *and* $\neg(x \wedge y) = \neg x \vee \neg y$.
4) $x \wedge \neg x \leq y \vee \neg y$.
5) $x \wedge \neg x \leq \mathsf{W} \leq x \vee \neg x$.
6) $\neg\top = \bot, \neg\bot = \top$ *and* $\neg\mathsf{W} = \mathsf{W}$.
7) $x > y$ *iff* $\neg x < \neg y$.
8) $x \rightarrow y = \neg y \rightarrow \neg x$.
9) $x \rightarrow (y \rightarrow z) = y \rightarrow (x \rightarrow z)$.
10) $x \rightarrow y \leq x' \rightarrow y'$ *if* $x \leq x'$ *and* $y \geq y'$.
11) $x \rightarrow y = \top$ *iff* $x = \bot$ *or* $y = \top$.
12) $\top \rightarrow x = x$ *and* $x \rightarrow \top = \top$; $\bot \rightarrow x = \top$ *and* $x \rightarrow \bot = \neg x$.
13) $x \rightarrow y \geq \mathsf{W}$ *iff* $x \leq \mathsf{W}$ *or* $y \geq \mathsf{W}$, *and* $x \rightarrow y \leq \mathsf{W}$ *iff* $x \geq \mathsf{W}$ *or* $y \leq \mathsf{W}$.

III. Linguistic Propositional Logic

In this section, we shall define the syntax and semantics of the linguistic propositional logic.

Definition III.1. *An alphabet of linguistic propositional logic consists of:*

- *constant symbols:* $\bot, \top, \mathsf{MoreTrue}, \mathsf{VeryFalse}, ...$,
- *propositional variables:* $A, B, C, ...$,
- *logical connectives:* $\vee, \wedge, \rightarrow, \neg, \equiv$, *and*
- *auxiliary symbols:* $\Box, (,), ...$

Definition III.2. *An atom is either a propositional variable or a constant symbol.*

Definition III.3. *Let A be an atom and α be a constant symbol. Then A^α is called a literal.*

Definition III.4. *Formulas are defined recursively as follows:*

- *either a literal or a constant symbol is a formula,*
- *if P is a formula, then $\neg P$ is a formula, and*
- *if P, Q are formulas, then $P \vee Q$, $P \wedge Q$ and $P \rightarrow Q$ are formulas.*

Definition III.5. *A clause is a finite disjunction of literals, which is written as $l_1 \vee l_2 \vee \ldots \vee l_n$, where the symbol l_i is a literal. The empty clause is denoted by $\Box$.*

Definition III.6. *A formula F is said to be in conjunctive normal form if it is a conjunction of clauses.*

Definition III.7. *An interpretation consists of the followings:*

- *a linguistic truth domain, which is a symmetrical RHA,*
- *for each constant in the alphabet, the assignment of an element in $LH_p[G]$,*
- *for each formula, the assignment of a mapping from $LH_p[G]^n$ to $LH_p[G]$.*

Definition III.8. *Let I be an interpretation and A be an atom such that $I(A) = \alpha_1$. Then the truth value of a literal A^α under the interpretation I is determined uniquely as follows:*

- $I(A^{\alpha_2}) = \alpha_1 \wedge \alpha_2$ *if* $\alpha_1, \alpha_2 > \mathsf{W}$,
- $I(A^{\alpha_2}) = \neg(\alpha_1 \vee \alpha_2)$ *if* $\alpha_1, \alpha_2 \leq \mathsf{W}$,
- $I(A^{\alpha_2}) = (\neg\alpha_1) \vee \alpha_2$ *if* $\alpha_1 > \mathsf{W}, \alpha_2 \leq \mathsf{W}$, *and*
- $I(A^{\alpha_2}) = \alpha_1 \vee (\neg\alpha_2)$ *if* $\alpha_1 \leq \mathsf{W}, \alpha_2 > \mathsf{W}$.

Definition III.9. *The truth value of formulas under an interpretation is determined recursively as follows:*

- $I(P \vee Q) = I(P) \vee I(Q)$,
- $I(P \wedge Q) = I(P) \wedge I(Q)$,
- $I(\neg P) = \neg I(P)$,
- $I(P \to Q) = I(P) \to I(Q)$

The following result follows from the properties of the $\wedge$ and $\vee$ operators.

Theorem III.1. *Let A, B and C are formulas, and I be an arbitrary interpretation. Then,*

- *Commutative:*
  - $I(A \vee B) = I(B \vee A)$
  - $I(A \wedge B) = I(B \wedge A)$
- *Associative:*
  - $I((A \vee B) \vee C) = I(A \vee (B \vee C))$
  - $I((A \wedge B) \wedge C) = I(A \wedge (B \wedge C))$
- *Distributive:*
  - $I(A \vee (B \wedge C)) = I((A \wedge B) \vee (A \wedge C))$
  - $I(A \wedge (B \vee C)) = I((A \vee B) \wedge (A \vee C))$

Proof: The proof of the Theorem is straightforward. ■

Definition III.10. *Let F be a formula and I be an interpretation. Then*

- *F is said to be true under interpretation I iff $I(F) > \mathsf{W}$, I is also said to satisfy formula F, F is said to be satisfiable iff there is an interpretation I such that I satisfies F, F is said to be tautology iff it is satisfied by all interpretations;*
- *F is said to be false under interpretation I iff $I(F) < \mathsf{W}$, I is also said to falsify formula F, F is said to be unsatisfiable iff it is falsified by all interpretations.*

According to the definition, if $I(F) = \mathsf{W}$ then I both satisfies and falsifies F. Further, *not satisfying* is different from *falsifying* and *not falsifying* is different from *satisfying*.

Definition III.11. *Formula B is said to be a logical consequence of formula A, denoted by $A \models B$, if for all interpretation I, $I(A) > \mathsf{W}$ implies that $I(B) > \mathsf{W}$.*

Theorem III.2. *Let A and B be formulas. Then, $A \models B$ iff $\models (A \to B)$.*

Proof: Assume that $A \models B$, for any interpretation I, then if $I(A) < \mathsf{W}$, $I(\neg A) > \mathsf{W}$; otherwise if $I(A) > \mathsf{W}$, we recall that $A \models B$, so $I(B) > \mathsf{W}$. Hence, $I(A \to B) = \neg I(A) \vee I(B) > \mathsf{W}$. In other words, $\models (A \to B)$. Conversely, by a similar way, we can also show that $\models (A \to B)$ implies $A \models B$. ■

Definition III.12. *Two formulas A and B are logically equivalent, denoted by $A \equiv B$, if and only if $A \models B$ and $B \models A$.*

The following theorem follows from the properties of the $\wedge$ and $\vee$ operators and Definition III.12.

Theorem III.3. *Let A, B and C be formulas. Then the following properties hold*

1) *Idempotency:*
   - $A \vee A \equiv A$
   - $A \wedge A \equiv A$
2) *Implication:*
   - $A \to B \equiv (\neg A) \vee B$
   - $(A \equiv B) \equiv (A \to B) \wedge (B \to A)$
3) *Double negation:*
   - $\neg\neg A \equiv A$
4) *De Morgan:*
   - $\neg(A \vee B) \equiv (\neg A) \wedge (\neg B)$
   - $\neg(A \wedge B) \equiv (\neg A) \vee (\neg B)$
5) *Commutativity:*
   - $A \vee B \equiv B \vee A$
   - $A \wedge B \equiv B \wedge A$
6) *Associativity:*
   - $A \vee (B \vee C) \equiv (A \vee B) \vee C$
   - $A \wedge (B \wedge C) \equiv (A \wedge B) \wedge C$
7) *Distributivity:*
   - $A \vee (B \wedge C) \equiv (A \vee B) \wedge (A \vee C)$
   - $A \wedge (B \vee C) \equiv (A \wedge B) \vee (A \wedge C)$

Proof: The proof is straightforward. ■

As mentioned previously, we will be working with resolution as the inference system of our logic. Therefore formulas need to be converted into conjunctive normal form. The following theorem, which follows from the equivalence properties in Theorem III.3, ensures that the transformation is always feasible.

Theorem III.4. *Let F be a formula of arbitrary form. Then F can be converted into an equivalent formula in conjunctive normal form.*

Proof: It is easy to prove this theorem based on the properties in the Theorem III.3 ■

Let S be a set containing exactly n atoms $A_1, A_2, \dots, A_n$. A *semantic tree* of S is an n-level complete binary tree, each level corresponds to an atom. The left edge of each node at the level i is assigned the label $A_i \leq \mathsf{W}$, and the right edge of each node at the level i is assigned the label $A_i > \mathsf{W}$ (cf. Fig 3).

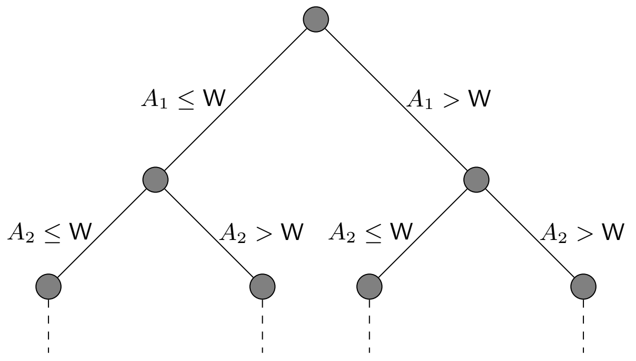

Fig. 3: Semantic tree

A set of clauses S *is failed at the node* t of a semantic tree T iff there exist an interpretation I corresponding to a branch in T which contains t, such that S is false under I. A node t is called a *failure node* of S iff S fails at t and does not fail at any node above t. A node t in a semantic tree T is called an *inference node* iff both successor nodes of t are failure nodes. If there are failure nodes for S on every branch of the corresponding semantic tree T, removing all child nodes of each failure node, we receive a *failure tree* FT.

It is important to underline that if a set of clauses S is unsatisfiable then S has a corresponding failure tree.

Lemma III.1. *There always exists an inference node on the failure tree.*

Proof: Assume that we have a failure tree FT. Because FT has a finite level, so there exists one (or more) leaf node on FT at the highest level, let say this node is called j. Let i be the parent node of j. By definition of failure tree, i cannot be failure node. Therefore, i has another child node, named k (Figure 4). If k is a failure node then i is inference node, the theorem is proved. If k is not a failure node then it has two child nodes: l, m. Clearly l, m are at higher level than j. This contradicts with the assumption that j is at the highest level. Therefore, k is a failure node and i is an inference node. This completes the proof of the lemma. ■

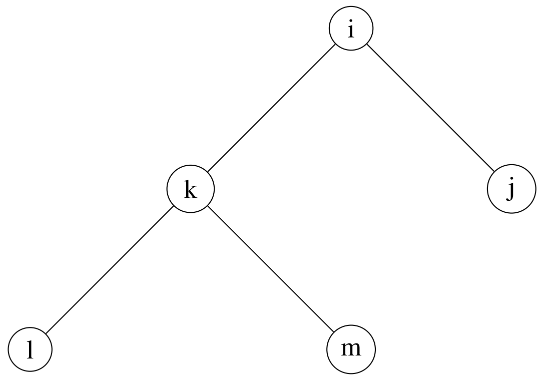

Fig. 4: Inference node on failure tree

Let FT_1 (respectively FT_2) be a failure tree of the set of clauses S_1 (respectively S_2). We denote $FT_1 \supset FT_2$ iff there exists an inference node i of FT_1 such that removing two successor nodes of i on FT_1 we receive FT_2.

IV. Resolution Method

We shall introduce resolution rule for automated reasoning in our linguistic propositional logic. Firstly, we recall basic notions inference rules.

An *inference* has the form:

$$\frac{C_1, C_2 \dots, C_n}{C}$$

where the clauses $C_1, C_2 \dots, C_n$ are the premises and C is the conclusion.

In two-valued logic, when we have a set of formulas $\{A, \neg A\}$ (written as $\{A^{\mathsf{True}}, A^{\mathsf{False}}\}$ in our logic) then the set is said to be contradictory. However, in our logic, the degree of contradiction can vary because the truth domain contains more than two elements. Let us consider the two sets of formulas $\{A^{\mathsf{VeryTrue}}, A^{\mathsf{VeryFalse}}\}$ and $\{A^{\mathsf{LittleTrue}}, A^{\mathsf{LittleFalse}}\}$. It is obvious that the first set of formulas is "more contradictory" than the second one. The notion of reliability is hence introduced to capture the fuzziness of resolution inferences.

Let α be an element of X such that $\alpha > \mathsf{W}$, and C be a clause. The a clause C with reliability α is the pair (C, α). The same clauses with different reliabilities are called variants. That is (C, α) and (C, α') are called a variant of each other.

If $S = \{C_1, C_2, ..., C_n\}$ is a set of clauses, then the reliability α of S is defined as: $\alpha = \alpha_1 \wedge \alpha_2 \wedge ... \wedge \alpha_n$, where α_i is the reliability of the clause C_i (for $i = 1, 2, \dots, n$).

An inference rule R that works with clauses with reliability is represented as:

$$\frac{(C_1, \alpha_1), (C_2, \alpha_2), \dots, (C_n, \alpha_n)}{(C, \alpha)}$$

We call α the reliability of R, provided that $\alpha \leq \alpha_i$ for $i = 1..n$.

Definition IV.1. *Define the fuzzy linguistic resolution rule as follows:*

$$\frac{(A^a \vee B^{b_1}, \alpha_1), (B^{b_2} \vee C^c, \alpha_2)}{(A^a \vee C^c, \alpha_3)}$$

where b_1, b_2 and α_3 satisfy the following conditions:

$$\begin{cases} b_1 \wedge b_2 \leq \mathsf{W}, \\ b_1 \vee b_2 > \mathsf{W}, \\ \alpha_3 = f(\alpha_1, \alpha_2, b_1, b_2) \end{cases}$$

with f is a function ensuring that $\alpha_3 \leq \alpha_1$ and $\alpha_3 \leq \alpha_2$.

In Definition IV.1 α_3 is defined so as to be smaller or equal to both α_1 and α_2. This makes sure that the more inferences we need to deduce a clause the less reliable the obtained clause is. There are different ways to define α_3. Below we define α_3 in three ways based on $\wedge$ and $\vee$ operators.

The reliability of the resolution rule IV.1 based on $\wedge$ operator is given by:

$$\alpha_3 = f(\alpha_1, \alpha_2, b_1, b_2) = \alpha_1 \wedge \alpha_2 \wedge \neg(b_1 \wedge b_2) \qquad (1)$$

The reliability of the resolution rule IV.1 based on $\vee$ operator is given by:

$$\alpha_3 = f(\alpha_1, \alpha_2, b_1, b_2) = \alpha_1 \wedge \alpha_2 \wedge (b_1 \vee b_2) \qquad (2)$$

The reliability of the resolution rule IV.1 based on the combination of $\wedge$ and $\vee$ operators is given by:

$$\alpha_3 = f(\alpha_1, \alpha_2, b_1, b_2) = \alpha_1 \wedge \alpha_2 \wedge (\neg(b_1 \wedge b_2)) \wedge (b_1 \vee b_2) \qquad (3)$$

Proposition IV.1. *The reliability based on $\wedge$ operator (respectively $\vee$ operator or the combination of $\wedge$ and $\vee$ operators) satisfies the conditions on α_3 in Definition IV.1.*

Proof: It is clear that that $\alpha_3 \leq \alpha_1$ and $\alpha_3 \leq \alpha_2$, as well as α_3 depends on b_1, b_2. Additionally, it is clear that $\alpha_1, \alpha_2 \in X^+$. Moreover, $b_1 \wedge b_2 \leq \mathsf{W}$ implies $\neg(b_1 \wedge b_2) > \mathsf{W}$. Then, by Formula (1), we have $\alpha_3 > \mathsf{W}$.

The proof for $\vee$ operator is similar.

For the combination of $\wedge$ and $\vee$ operators, we have

$$\begin{aligned} \alpha_3 &= \alpha_1 \wedge \alpha_2 \wedge (\neg(b_1 \wedge b_2)) \wedge (b_1 \vee b_2) \\ &= (\alpha_1 \wedge \alpha_2 \wedge \neg(b_1 \wedge b_2)) \wedge (\alpha_1 \wedge \alpha_2 \wedge (b_1 \vee b_2)) \end{aligned}$$

Then applying the results of $\wedge$ operator and $\vee$ operator we have $\alpha_3 > \mathsf{W}$. ■

Theorem IV.1. *The fuzzy linguistic resolution rule IV.1 is sound.*

Proof: We need to prove that for any interpretation I, if $I((A^a \vee B^{b_1}) \wedge (B^{b_2} \vee C^c)) > \mathsf{W}$ then $I(A^a \vee C^c) > \mathsf{W}$. We have that

$$\begin{aligned} &I((A^a \vee B^{b_1}) \wedge (B^{b_2} \vee C^c)) \\ &= I((A^a \wedge B^{b_2}) \vee (A^a \wedge C^c) \vee (B^{b_1} \wedge B^{b_2}) \vee (B^{b_1} \wedge C^c)) \\ &= I(A^a \wedge B^{b_2}) \vee I(A^a \wedge C^c) \vee I(B^{b_1} \wedge B^{b_2}) \vee I(B^{b_1} \wedge C^c) \end{aligned}$$

It is easy to show that:

- $I(A^a \wedge B^{b_2}) \leq I(A^a) \leq I(A^a \vee C^c)$,
- $I(A^a \wedge C^c) \leq I(A^a \vee C^c)$,
- $I(B^{b_1} \wedge C^c) \leq I(C^c) \leq I(A^a \vee C^c)$, and
- $I(B^{b_1} \wedge B^{b_2}) \leq \mathsf{W}$

So, if $I(A^a \vee C^c) \leq \mathsf{W}$ then we must have that

$$I(A^a \wedge B^{b_2}) \vee I(A^a \wedge C^c) \vee I(B^{b_1} \wedge B^{b_2}) \vee I(B^{b_1} \wedge C^c) \leq \mathsf{W}$$

which contradicts with the initial assumption. This completes the proof of the theorem. ■

Definition IV.2. *A linguistic resolution* derivation *as a sequence of the form $S_0, \ldots, S_i, \ldots$, where*

- *each S_i is a set of clauses with reliability, and*
- *S_{i+1} is obtained by adding the conclusion of a fuzzy linguistic resolution inference with premises from S_i, that is $S_{i+1} = S_i \cup \{(C, \alpha)\}$, where (C, α) is the conclusion of the fuzzy linguistic resolution*

$$\frac{(C_1, \alpha_1), (C_2, \alpha_2)}{(C, \alpha)}$$

and $(C_1, \alpha_1), (C_2, \alpha_2) \in S_i$.

The following theorem shows that the resolution procedure in the Def.IV.2 is sound and complete.

Theorem IV.2. *Let $S_0, \ldots, S_i, \ldots$ be a fuzzy linguistic resolution derivation. S_n contains the empty clause (for some $n = 0, 1, \ldots$) iff S_0 is unsatisfiable.*

Proof: ($\Rightarrow$) If S_n contains the empty clause, then S_n is false under any interpretation. By Theorem IV.1, we have S_{n-1} is false under any interpretation, too. Similarly, $S_{n-1}, \ldots, S_1, S_0$ are also false under any interpretation.

($\Leftarrow$)Because S_0 is unsatisfiable, there is a corresponding failure tree FT. By Lemma III.1, there exists an inference node i on FT with two child nodes j, k. Assuming that the label of edge $i - j$ is $A \leq \mathsf{W}$ and the label of edge $i - k$ is $A > \mathsf{W}$. The interpretation corresponding to the branch contains the edge $i - j$ and terminating at j makes S_0 satisfiable. So, there is at least one clause in S_0 containing the literal A^{α_1} where $\alpha_1 \leq \mathsf{W}$, let say C_1. Similarly, there exists at least one clause in S_0 containing the literal A^{α_2} where $\alpha_2 > \mathsf{W}$, we name it C_2. Applying the resolution rule in the Def. IV.1:

$$\frac{C_1, C_2}{C_3}$$

C_3 does not contain atom A, so that C_3 is false under all interpretations containing A. Thus, failure tree FT_1 of the clause set $S_1 = S_0 \cup C_3$ does not contain node j, k, this means $FT \supset FT_1$.

By applying resolution procedure, there exist failure trees $FT_2, FT_3, \ldots$ of the sets of clauses $S_2, S_3, \ldots$ such that $FT \supset FT_1 \supset FT_2 \supset FT_3 \supset \ldots$. Because there are only a finite number of nodes in FT, then exists some n satisfying: $FT_n = \Box$ (i.e. $FT \supset FT_1 \supset FT_2 \supset FT_3 \supset \ldots \supset FT_n = \Box$). Only the empty clause is false under the empty interpretation. This means that the set of clauses S_n (S_n corresponds to FT_n) contains the empty clause. ■

A *resolution proof* is of a clause C from a set of clauses S consists of repeated application of the resolution rule to derive the clause C from the set S. If C is the empty clause then the proof is called a *resolution refutation*. We will represent resolution proofs as *resolution trees*. Each tree node is labeled with a clause. There must be a single node that has no child node, labeled with the conclusion clause, we call it the root node. All nodes with no parent node are labeled with clauses from the initial set S. All other nodes must have two parents and are labeled with a clause C such that

$$\frac{C_1, C_2}{C}$$

C where C_1, C_2 are the labels of the two parent nodes. If RT is a resolution tree representing the proof of a clause with reliability (C, α), then we say that RT has the reliability α.

Example IV.1. *Let* $AX = (LH_p[G], G, LH, \leq, \neg, \vee, \wedge, \rightarrow)$ *be a symmetrical RHA, where*

- $LH^+ = \{\mathsf{Very}, \mathsf{More}\}$,

  $LH^- = \{\mathsf{Approximately}, \mathsf{Little}\}$,
- $G = \{\bot, \mathsf{False}, \mathsf{W}, \mathsf{True}, \top\}$,
- $\bot, \top$ *are the smallest and biggest elements,* W *is the neutral element, and* $\bot < \mathsf{False} < \mathsf{W} < \mathsf{True} < \top$.

Assume that we have the set of clauses:

1) $A^{\mathsf{ApproximatelyTrue}} \vee B^{\mathsf{MoreTrue}}$
2) A^{False}
3) $B^{\mathsf{VeryFalse}} \vee C^{\mathsf{True}}$
4) $C^{\mathsf{LittleFalse}}$

At the beginning, we shall assign each clause to the highest reliability $\top$. *We consider the reliability based on* $\vee$ *operator. The resolution refutation is found as following:*

$$\cfrac{\cfrac{\cfrac{(A^{\mathsf{ApproximatelyTrue}} \vee B^{\mathsf{MoreTrue}}, \top) \quad (A^{\mathsf{False}}, \top)}{(B^{\mathsf{VeryTrue}}, \mathsf{ApproximatelyTrue}) \qquad (B^{\mathsf{VeryFalse}} \vee C^{\mathsf{True}}, \top)}}{(C^{\mathsf{True}}, \mathsf{ApproximatelyTrue}) \qquad (C^{\mathsf{LittleFalse}}, \top)}}{(\square, \mathsf{ApproximatelyTrue})}$$

V. Conclusion

We have presented the resolution method in linguistic truth-valued propositional logic based on linguistic truth-valued symmetrical RHA. The syntax and semantics of our logic are defined. The resolution rule with the reliability is given. The soundness and completeness are proved. The presented work provided a key theoretical support for a resolution-based automated reasoning approaches in linguistic truth-valued logic based on RHA. Accordingly, this research work supports linguistic information processing which can directly focus on linguistic values. Further research can be carried out on the practical implementation of the proposed resolution-based automated reasoning procedure as well as its application in intelligent reasoning and decisions making.

References

[1] J. A. Robinson, "A machine-oriented logic based on the resolution principle," *J. ACM*, vol. 12, no. 1, pp. 23–41, 1965.

[2] L. A. Zadeh, "Fuzzy sets," *Information and Control*, vol. 8, no. 3, pp. 338–353, 1965.

[3] ——, "The concept of a linguistic variable and its application to approximate reasoning - ii," *Inf. Sci.*, vol. 8, no. 4, pp. 301–357, 1975.

[4] ——, "Knowledge representation in fuzzy logic," *IEEE Trans. on Knowl. and Data Eng.*, vol. 1, no. 1, pp. 89–100, Mar. 1989.

[5] R. C. T. Lee, "Fuzzy logic and the resolution principle," in *IJCAI*, 1971, pp. 560–567.

[6] M. M. Z. Shen, L. Ding, "Fuzzy resolution principle," in *Proc. 18th Internat. Symp. on Multiple-valued Logic*, 1989, pp. 210–215.

[7] R. Ebrahim, "Fuzzy logic programming," *Fuzzy Sets and Systems*, vol. 117, no. 2, pp. 215–230, 2001.

[8] P. Vojtás, "Fuzzy logic programming," *Fuzzy Sets and Systems*, vol. 124, no. 3, pp. 361–370, 2001.

[9] D. Smutná and P. Vojtás, "Graded many-valued resolution with aggregation," *Fuzzy Sets and Systems*, vol. 143, no. 1, pp. 157–168, 2004.

[10] C. Nguyen and W. Wechler, *Hedge Algebras: An Algebraic Approach in Struture of Sets of Linguistic Truth Values*. Fuzzy Sets and Syst. 35, 1990, pp. 281–293.

[11] C.-H. Nguyen, D.-K. Tran, V.-N. Huynh, and H.-C. Nguyen, "Hedge algebras, linguistic-valued logic and their application to fuzzy reasoning," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 7, no. 4, pp. 347–361, 1999.

[12] C.-H. Nguyen and V.-N. Huynh, "An algebraic approach to linguistic hedges in zadeh's fuzzy logic," *Fuzzy Sets and Systems*, vol. 129, no. 2, pp. 229–254, 2002.

[13] N. C. Ho, V. N. Lan, and L. X. Viet, "Optimal hedge-algebras-based controller: Design and application," *Fuzzy Sets and Systems*, vol. 159, no. 8, pp. 968 – 989, 2008.

[14] V. H. Le, F. Liu, and D. K. Tran, "Fuzzy linguistic logic programming and its applications," *TPLP*, vol. 9, no. 3, pp. 309–341, 2009.

[15] J. Lai and Y. Xu, "Linguistic truth-valued lattice-valued propositional logic system *l*p(x) based on linguistic truth-valued lattice implication algebra," *Inf. Sci.*, vol. 180, no. 10, pp. 1990–2002, 2010.

[16] J. Liu, L. M. Lopez, Y. Xu, and Z. Lu, "Automated reasoning algorithm for linguistic valued lukasiewicz propositional logic," in *ISMVL*, 2007, p. 29.

[17] T. Nguyen, V. Vu, T. Doan, and D. Tran, "Resolution in linguistic first order logic based on linear symmetrical hedge algebra," in *Information Processing and Management of Uncertainty in Knowledge-Based Systems - 15th International Conference, IPMU 2014, Montpellier, France, July 15-19, 2014, Proceedings, Part I*, 2014, pp. 345–354.

[18] ——, "Resolution in linguistic propositional logic based on linear symmetrical hedge algebra," in *Knowledge and Systems Engineering - Proceedings of the Fifth International Conference, KSE 2013, Volume 1, Hanoi, Vietnam, 17-19 October, 2013*, 2013, pp. 327–338.

[19] N. C. Ho and W. Wechler, "Extended hedge algebras and their application to fuzzy logic," *Fuzzy Sets and Systems*, vol. 52, no. 3, pp. 259 – 281, 1992.

[20] L. A. Zadeh, "Toward a theory of fuzzy information granulation and its centrality in human reasoning and fuzzy logic," *Fuzzy Sets and Systems*, vol. 90, no. 2, pp. 111 – 127, 1997.