経営論集　第 26 巻第 1 号　2016 年　1 ～ 11 頁

The Strict Semantics of System FS

Koji KOMATSU

1 Introduction

Frame-structure logic is capable of structural knowledge representation and has strict denotational semantics[1]. This paper provides several lines of evidence that the formula of Frame-structure logic is simpler than that of predicate logic. In this paper, the strict semantics of Frame-structure logic is defined as FS which is an extended system of basic system[1]. The descriptive power of FS is shown by examples of an inference in this paper. The descriptive power of FS is stronger than propositional logic, however, weaker than predicate logic. The formula of FS has a high affinity to the sentence of natural language.

There are several factors that make a mechanical inference difficult in natural language. For example, a high degree of freedom of description, an ambiguity and a dependency on contexts. Natural language is symbolic systems for human being to describe knowledge. In computer science, therefore, predicate logic and its extensions have been frameworks for knowledge representation and inference.

Predicate logic is a system which describes knowledge as relations between individual sets. For example, a compound notion “students who like fruits” is interpreted to the product set of two sets. One is the set of “students”. The other is the set of “everyone who likes fruits”. In natural language the clause “who likes fruits” modifies a noun “students”. In other words, “like fruit” can be interpreted as an attribute which “students” have. There is a gap between the description of predicate logic and that of natural language. There is no such notion as an individual variable in natural language. The representational power of predicate logic is strong. A logical expression of predicate logic, however, is too complex because of the existence of an individual variable and quantifier.

On the other hand, in the system such as a semantic network, knowledge is described as direct relations between words by using nodes and links. When regarding a node as a word, the links as modifications, such a system may fit for the structure of natural language. These systems, however, do not have strict semantics and inference rules which are guaranteed by mathematical theory. Especially, there has been no semantic network in which inference about knowledge including negation and quantification can be strictly executed.

Previously, various logic systems have been proposed besides propositional logic and predicate logic. These non-standard logics are categorized into two types. In logics such an intuitionistic logic[2],

many-valued logic [3] and relevant logic [4], the interpretation of logical operators is different from those of standard logic. Meanwhile, in logics such as belief logic [5] and temporal logic [6], new logical operators are introduced. In each non-standard logic, however, the structure of a proposition is same as those of standard logic.

There is no internal structure in the proposition of propositional logic. And the proposition of predicate logic is constructed from predicates, functions, individual variables and quantifiers. In contrast to these logics, frame-structure logic is a system that has a structural proposition specified for a sentence of natural language. It has following features.

1. Compound notations are represented structurally as an object.
2. Relations between notions are represented by attribute pairs and ordering.
3. There is no individual variable in its axiomatic system.

The name “frame-structure” is derived from frame theory [8]. Frame-structure logic adopts parts of the description methods of F-Logic [9] and Quixote [10]. The difference from these systems is discussed in section 5.

2 Syntax

In this section, the syntax of frame-structure logic is defined and some examples are shown.

Definition 2.1 (Syntax of system FS)

1. Primitive Symbols

   Attribute label: $l_1, l_2, \ldots$

   Object symbol: $\pi, a, b, c, \ldots$

   Object operator: $\cdot$

   Relational operator: $\leq, En$

   Logical operator: $\Rightarrow, \sim$

   The object symbol “π” is called a universal object and intuitively means “everything”.

2. Objects

   (a) If t is an object symbol, “t” is an object.

   (b) If α and β are objects, “$(\alpha \cdot \beta)$” is an object. This object represents the conjunction of α and β.

   (c) If t is an object symbol, “$t\ [ap_1, ap_2, \ldots, ap_n]$ $(n \geq 1)$” is an object. This object represents “t

that has [ap_1, ap_2, … ,ap_n] as the attribute": Each ap_i is either of the following:

$$ap_i = \begin{cases} Lb \to \alpha & \cdots \quad (i) \\ Lb \to \theta & \cdots \quad (ii) \\ Lb \leftarrow \theta & \cdots \quad (iii) \end{cases}$$

Where *Lb* is an attribute label, α is an object and θ is a set of objects (may be an empty set). The ap_i of type (i) is called a single-valued existential attribute pair, (ii) called a multiple-valued existential attribute pair and (iii) is called a multiple-valued universal attribute pair, or simply, each ap_i is called an attribute pair. An attribute pair means that the value of attribute *Lb* is α (or θ). [ap_1, ap_2, … , ap_n] is called a list of attribute pairs.

(d) If *Lb* is an attribute label and α is an object, then "*Lb* (α)" is an object. This object represents "The value of *Lb* which α has as an attribute label", and such a *Lb* is called an attribute function.

An Object obtained by (b) and (c), in particular, is called a compound objects.

3. Object Relations

If α and β are objects, then the object relation is as follows:

(a) $\alpha \leq \beta$

(b) *En* (α)

Where "$\alpha \leq \beta$" means "α is a β" and "*En* (α)" means "α is an actual being".

4. Logical Formulas

(a) If *P* is an object relation, then *P* is a logical formula (or simply a formula)

(b) If *P* and *Q* are logical formulas, then ($P \Rightarrow Q$) and $\sim(P)$ are logical formulas.

Logical operators $\wedge$, $\vee$, $\Leftrightarrow$ may be used, and each can be defined from $\Rightarrow$ and $\sim$ in the same way as standard propositional logic. ■

Definition 2.2 (Extended notations)

Let α and β be objects, and […] be a list of attribute pairs.

1. α / […] $\stackrel{\text{def}}{=}$ $\alpha \leq \pi$ […]
2. $\alpha =_o \beta \stackrel{\text{def}}{=} \alpha \leq \beta \wedge \beta \leq \alpha$
3. α / $\overline{\beta} \stackrel{\text{def}}{=} \alpha \leq \sim En(\alpha \cdot \beta)$
4. α / $\overline{[\ldots]} \stackrel{\text{def}}{=} \alpha \leq \overline{\pi[\ldots]}$ ■

Example 2.1 (Attribute pairs)

1. [Favorite→fruit] … someone's favorites are only fruits (or a fruit)
2. [Favorite→{fruit}] … fruits (or a fruit) are included in someone's favorites
3. [Favorite←{fruit}] … all fruits are included in someone's favorites

Example 2.2 (Objects)

1. man [Pet→{dog, cat}] … men who keep dogs (or a dog) and cats (or a cat)
2. (male・student) … school-boys
3. Pet(student) … pets kept by students (or a student)
4. (Pet(male・student)・dog[Sex→male, Color→black]) … black male dogs kept by school-boys

Example 2.3 (Object relations)

1. car-freak $\leq$ man[Favorite→{exotic-car}] … All car freaks are men who love exotic cars at least.
2. car-freak / [Favorite→{exotic-car}] … All car freaks love exotic cars at least.
3. vegetarian / $\overline{[\mathrm{Favorite} \rightarrow \{\mathrm{meat}\}]}$ … All Vegetarians do not like meats.
4. *En* (student・vegetarian) … Students who are vegetarian exist (Some students are vegetarian).
5. school-boy $=_o$ (male・student) … school-boys are equivalent to students whose sex is male.

Example 2.4 (Logical Formulas)

1. (male・student) $\leq$ student … school-boys are students.
2. school-boy $\leq$ student $\wedge$ student $\leq$ human $\Rightarrow$ school-boy $\leq$ human … If school-boys are students and students are humans then school-boys are human.

As shown in 1. of Example 2.4, an object relation becomes a logical formula by itself. As shown in 2. of Example 2.4, a conjunction of logical formulas is a logical formula and an implication between two logical formulas also becomes a logical formula.

3 Semantics

In this section, the semantics of frame-structure logic is defined strictly. Let α and β be objects and *Lb* be an attribute label.

Definition 3.1 (Semantics of system FS)

Let a set *O* be a set of all object symbols, a set *N* be that of all attribute labels and a set *U* be a countably non-empty set. An interpretation *I* is a tuple (U, ϕ, ψ), where ϕ and ψ are functions defined as

follows.

1. $\phi : O \rightarrow 2^U$ is a function that assigns object symbols to a non-empty subset of U, where $\phi(\pi) = U$.

2. $\psi : N \rightarrow (U \rightarrow 2^U)$ is a function that assigns attribute labels to the function $U \rightarrow 2^U$. For $Lb \in N$, the function $\psi(Lb)$ is simply written as ψ_{Lb}. In general, $\psi_{Lb}(x) (= \psi(Lb)(x))$ represents the value of attribute Lb of x.

   By $I(U, \phi, \psi)$, any attribute pair, any list of attribute pairs or any object is assigned to a subset of U as follows. And based on these assignments, the truth values of object relations and logical formulas under I are defined.

3. Assignment for a single-valued existential attribute pair

   $I(Lb \rightarrow \alpha) = \{x \mid \psi_{Lb}(x) \subseteq I(\alpha) \text{ and } \psi_{Lb}(x) \neq \phi\}$

4. Assignment for a multiple-valued existential attribute pair ($k \geq 1$)
   (a) $I(Lb \rightarrow \{\}) = U$
   (b) $I(Lb \rightarrow \{\alpha\}) = \{x \mid \psi_{Lb}(x) \cap I(\alpha) \neq \phi\}$
   (c) $I(Lb \rightarrow \{\alpha_1, \ldots, \alpha_k\}) = I(Lb \rightarrow \{\alpha_1\}) \cap \ldots \cap I(Lb \rightarrow \{\alpha_k\})$

5. Assignment for a multiple-valued universal attribute pair ($k \geq 1$)
   (a) $I(Lb \leftarrow \{\}) = U$
   (b) $I(Lb \leftarrow \{\alpha\}) = \{x \mid I(\alpha) \subseteq \psi_{Lb}(x)\}$
   (c) $I(Lb \leftarrow \{\alpha_1, \ldots, \alpha_k\}) = I(Lb \leftarrow \{\alpha_1\}) \cap \ldots \cap I(Lb \leftarrow \{\alpha_k\})$

6. Assignment for lists of attribute pairs ($n \geq 1$)

   $I([ap_1, ap_2, \ldots, ap_n]) = I(ap_1) \cap I(ap_2) \cap \ldots \cap I(ap_n)$

7. Assignment for objects ($n \geq 1$)
   (a) $I(t) = \phi(t)$
   (b) $I((\alpha \cdot \beta)) = I(\alpha) \cap I(\beta)$
   (c) $I(t\,[ap_1, ap_2, \ldots, ap_n]) = I(t) \cap I([ap_1, ap_2, \ldots, ap_n])$
   (d) $I(Lb(\alpha)) = \bigcup_{x \in I(\alpha)} \psi_{Lb}(x)$

8. Truth value assignment for object relations
   (a) $I(\alpha \leq \beta) = \mathbf{T}$ iff $I(\alpha) \subseteq I(\beta)$

(b) $En(\alpha) = \mathbf{T}$ iff $I(\alpha) \neq \phi$

If the above conditions are not satisfied, truth value **F** (false) is assigned.

9. Truth value assignment for logical formulas

This is defined in the same way as in standard propositional logic. ■

Definition 3.2 (Logical consequence)

Let S be a set of formulas and P be a formula. P is said to be a logical consequence of S iff P is **T** for every interpretation under which all formulas in S are **T**. When P is a logical consequence of S, it is written as $S \models P$ ■

3.1 Explanation of semantics

Here, explanations about the interpretation of an attribute pair, a list of attribute pairs, an object and an object relation are given.

[Attribute pair]

Attribute pairs "$Lb\rightarrow\alpha$", "$Lb\rightarrow\{\alpha\}$" and "$Lb\leftarrow\{\alpha\}$" have different character from each other. According to the definition of interpretations of each attribute pairs, the following relations are hold.

1. $\models$ (man[Friend→ male])･(man[Friend→ student]) $=_o$ man[Friend→ (male･student)]
2. apple ≤ fruit $\models$ man[Favorite→ apple] ≤ man[Favorite→ fruit]
3. apple ≤ fruit $\models$ man[Favorite→ {apple}] ≤ man[Favorite→ {fruit}]
4. apple ≤ fruit $\models$ man[Favorite← {fruit}] ≤ man[Favorite← {apple}]

The relation 1 represents a potential composition of objects by a single-valued existential attribute. This shows that "male" and "student" are identical for "man".

[List of attribute pairs]

An list of attribute pairs $[Lb_1\rightarrow\alpha_1, \ldots, Lb_n\rightarrow\alpha_n]$ is interpreted ad $I(Lb_1\rightarrow\alpha_1) \cap \ldots \cap I(Lb_n\rightarrow\alpha_n)$. That is, it is regarded as the conjunction of "one whose value of Lb_1 is α_1"and … and "one whose value of Lb_n is α_n". A list including multiple valued existential attribute pairs or multiple valued universal attribute pairs is regarded similarly.

[Object]

An object symbol is assigned to a subset of U by a function ϕ. Compound objects such as $(\alpha\cdot\beta)$, t

[$ap_1, ap_2, \ldots, ap_n$] is interpreted as conjunction of every objects and attribute pairs.

[Object relation]

An object relation "$\leq$", which represents a "is-a" relation, corresponds to a class inclusion of a set "$\subseteq$". Then, a class hierarchy of objects is realized by a transitive of a class inclusion as follows.

$$\text{dog} \leq \text{animal}, \text{animal} \leq \text{creatures} \models \text{dog} \leq \text{creatures}$$

In addition to a class hierarchy, an attribute inheritance is realized as one kind of a class hierarchy. Because "α / […]" (α has-property-of "…") corresponds to "$I(\alpha) \subseteq I(\pi$ […])" in semantics.(1)

$$\text{dog} \leq \text{animal}, \text{animal} / [\text{Feature} \rightarrow \{\text{mortal}\}] \models \text{dog} / [\text{Feature} \rightarrow \{\text{mortal}\}]$$

3.2 Comparison of system FS with basic system

System FS is extended system of basic system[1]. The semantics of basic system is defined based on lattice. Atomic elements,(2) however, do not always exist for any elements of a lattice. Therefore, in basic system, it is quite difficult to introduce quantifiers and to describe a relation between an existential attribute pair and a universal attribute pair. For this reason, the semantics of FS is constructed on a set. To reveal the extended point of FS, let us consider the following two compound notions.

1. $\{x \mid x \leq \text{car-freak}, x / [\text{Favorite} \rightarrow \{\text{car}\}]\}$
2. $\{x \mid x \leq \text{car}, \text{car-freak} / [\text{Favorite} \rightarrow \{x\}]\}$

The set 1 corresponds to the compound notion which is constructed from the subjects of two sentences, "x are car freaks" and "x love cars". The set 2 represents the one from the subject and the predicate of two sentences "x are cars" and "car freaks love x", respectively. In basic system, while the former can be represented as car-freak [Favorite→{car}], the latter can't be. Therefore, this paper introduces a functional aspect to an attribute label to attribute value as an object. For example, the attribute label "Favorite" is used as an attribute function. The description "Favorite (car-freak)" represents "the things that are loved by car freaks". Thus, in system FS, an attribute label has two functions as follows.

1. α / [$Lb \rightarrow \{\beta\}$] … an attribute relation between objects
2. $Lb(\alpha)$ … an attribute function that returns an attribute value

By using an attribute function and an object operator "・", the set 2 becomes to be described as "(car・Favorite(car-freak))". Here, for example, the following relation holds.

$$\text{car-freak} \leq \text{freak} \models \text{Favorite(car-freak)} \leq \text{Favorite(freak)}$$

An attribute function not only enhances expressiveness but simplify the description of a compound object. Let us see the following relations.

1. next(even) ≤ odd, next(odd) ≤ even ⊨ next(next(even)) ≤ even
2. number[next→even] ≤ odd, number[next→odd] ≤ even ⊨ number[next→ number[next → even]] ≤ even

Without an attribute function, the unessential object "number" is necessary as in above relation 2. Next let us compare FS and basic system about the expressiveness of attribute relations. In basic system, only an existentially quantified attribute value such as "certain cars" can be represented. In FS, a universally quantified attribute value such as "all cars" can be represented. Here, the following relation which represents "if busses are cars then car freaks who love all cars are ones who love all busses" holds in FS.

$$\text{buss} \leq \text{car} \models \text{man[Favorite} \leftarrow \{\text{car}\}] \leq \text{man[Favorite} \leftarrow \{\text{buss}\}]$$

Moreover, in basic system, there is another problem that the relation between a single-valued attribute pair and a multi-valued one can't be represented. This problem is ascribable to the difficulty of distinguish "certain" and "all" elements on lattice. In FS, the interpretations of each attribute pairs are defined by an attribute function. The following relations, for example, between each of attribute pairs are hold.

1. car-freak / [Favorite← {car}] ⊨ car-freak / [Favorite→{car}]
2. car-freak / [Favorite→car] ⊨ car-freak / [Favorite→{car}]
3. car-freak / [Favorite→car], car ≤ $\overline{\text{train}}$ ⊨ car-freak / $\overline{\text{[Favorite} \rightarrow \{\text{train}\}]}$

The relation 3, which represents "If car freaks love only cars and cars aren't trains then car freaks don't love trains", includes the exclusiveness of objects by a single-valued existential attribute. This feature is identical to the composition of objects by a single-valued existential attribute.

4 Comparison with predicate logic

In system FS, the knowledge written in natural language can be described in a format reflecting structures of a natural language. In this section, features of description of FS is observed in contrast to that of predicate logic.

In predicate logic a notion is described as a set of all individuals which satisfy a certain character, and a compound notion is as a set in which individuals satisfy certain relations with individuals in other sets. For example, “students who love all fruits” is described as follows.

$$\lambda x\ (\text{student}(x) \wedge \forall y\ (\text{fruit}(y) \rightarrow \text{Favorite}(x, y)))$$

The compound notion is described as individuals in the set “student” who share the relation “favorite” between all individuals which belongs to the set “fruit”[(3)]. Thus, in predicate logic, a compound notion is constructed by representing relations between individual variables with predicates and logical symbols and structures of modification, that is “who like fruits” modify students, can’t be represented.

To the contrary, in Frame-structure logic, a compound notion is described structurally. For example, the above notion is described as follows.

$$\text{student}[\text{Favorite} \leftarrow \{\text{fruit}\}]$$

The notion corresponding to “student” and “someone who like fruits” are assigned to a set individually, and the compound notion “student who like fruits” is assigned to the product set of these sets as a whole. Thus, in frame-structure logic, a notion corresponding to the noun phrase of a natural language is described and interpreted as one object without individual variables and logical operators.

An Integration of deep language understanding and computer algebra is proposed in “Todai Robot Project-Can a Robot Pass the University of Tokyo Entrance Exam?” [8]. In proposed system, all mathematical problems are translated in a term in ZF (Zermelo-Fraenkel). The term in ZF is transferred to the term in which there is no qualifier by QE (quantifier elimination) algorithm. The term in ZF is translated from an examination sentence by natural language processing. An examination sentence of a math problem is relevantly easy to translate into a formula of predicate logic. In general, the translation of a sentence of natural language into a formula of predicate logic is difficult. Predicate logic is certainly useful to solve math problems by a machine. This difficulty of translation become a bottleneck. In “Todai Robot Project”, human power intervenes in this translation. And markup language like LaTeX is used as a pretreatment of natural language.

5 Comparison with other similar systems

In this section, frame-structure logic is compared with other systems which have an object and object operators. The notation of frame-structure logic is based on those of F-logic [9] and Quixote [10]. F-logic is a system of which semantics based on a lattice not on a set. Aiming at the expressiveness of predicate logic, however, individual variables are included in the description of a compound notion. F-logic has been developed as a description language for an object oriented database not as a mathematical logic system [11].

In Quixote, a compound notion can be described without individual variables. Procedural steps, however, are included in its definition of semantics, same as in a semantic network.[(4)]

Also in DOT system[12], a compound notion can be described without individual variables by DOT-notations. However, the compound notion as “male・student” can’t be described, because there is no object operator in DOT system.

Observing the area of logic programming, the notion of ϕ term including the ability of attribute inheritance is proposed in LOGIN [13]. ϕ term is described and interpreted as same as a compound object of FS. The attribute relation, however, is limited to the relation corresponding to the multi-valued existential attribute of FS. And the same in DOT system, there is no object operators. In contrast to these systems, FS have the following features.

1. There are no individual variables in logical formulas
2. Attribute relations can be treated as objects
3. The existence of strict semantics

By virtue of feature 1, knowledge representation in format similar to the structure of a natural language is possible. By feature 2, various compound objects can be synthesized structurally by object operators. By feature 3, the features and relations of each attribute pairs, a class hierarchy and an attribute inheritance are examined based on truth-value.

6 Concluding Remarks

Frame-structure logic is expanded to represents various attribute relations by introducing an attribute function. It is true that frame-structure logic is inferior to predicate logic in the expressive power. A significant feature of frame-structure logic is, however, that a compound notion can be represented structurally without individual variables. Owing to this feature, in frame-structure logic a sentence of a natural language is easier to be translated to a logical formula than in predicate logic. When considering practical application to natural language processing, however, the expressive power of frame-structure logic is still poor. Especially it is an important topic of future study to describe a sentence that contains a verb phrase.

7 Annotation

(1) Refer to the definition 2.2

(2) An atomic element corresponds to an element of a set.

(3) Here, lambda notation is used expediently, because it is impossible to extract a notion corresponding to noun phrase in predicate logic.

(4) This fact isn't a defect, because Quixote is proposed not as a logical system but as a programming language.

8 References

1) T. Sato, N. Nishihara, T. Kamata and S. Yokoyama, "Semantic Analysis for "A no B" Based on the Frame-structure Logic", IEICE Technical Report (In Japanese), NLC97-3, pp.17—24, 1997.

2) A. Heyting, "Intuitionism: An Introduction", North-Holland, Amsterdam, 1956.

3) N. Rescher, "Many-valued Logic", McGraw-Hill, 1969.

4) A. R. Anderson and N. D. Belnap, "Entailment: The Logic of Relevance and Necessity", Vol. 1, Princeton University Press, 1975.

5) J. Hintikka, "Knowledge and Belief", Cornell University Press, Ithaca, 1962.

6) A. N. Prior, "Past, Present and Future", Clarendon Press, Oxford, 1967.

7) M. Minsky, "A Framework for Representing Knowledge", P. H. Winston (ed.), The Psychology of Computer Vision, McGraw-Hill, pp.211—277, 1975.

8) T. Matsuzaki, H. Iwane, H. Anai, A. Aizawa and N. Arai, "Solving University Entrance Exam Math Problems through the Integration of Deep Language Understanding and Computer Algebra", The 27th Annual Conference of the Japanese Society for Artificial Intelligence (In Japanese), 2013.

9) M. Kifer and G. Lausen, "F-Logic: A Higher-Order Language for Reasoning about objects, Inheritance, and Scheme", Proc. ACM SIGMOD. pp.134—146, 1989.

10) Y. Morita, H. Hanyuuda and K. Yokota, "Object Identity in Quixote", IPSJ Technical Report (In Japanese), DBS 80-12, AI 73-12, pp.109—117.

11) M. Kifer, G. Lausen and J. Wu., "Logical Foundations of Object-Oriented and Frame-Based Languages", Journal of ACM, Vol. 42, No. 4, pp.741—843, 1995.

12) M. Tsukamoto and S. Nishio, "An Inheritance System Using Regular Expressions", Journal of JSAI (In Japanese), Vol. 9, No. 3, pp.447—454, 1994.

13) H. Ait-Kaci and R. Nasr, "LOGIN: A Logic Programming Language with Built-In Inheritance", Journal of Logic Programming, Vol. 3, pp.185—215, 1986.

（2016.9.28 受理）