# UNIT-TO-UNIT INTERACTION AS A BASIS FOR SEMANTIC INTERPRETATION OF JAPANESE SENTENCES

Hozumi TANAKA

ELECTROTECHNICAL LABORATORY, 1-1-4 UMEZONO, SAKURA-MURA, NIIHARI-GUN, IBARAKI-KEN, JAPAN

ABSTRACT: The notion of UNIT-to-UNIT interaction is introduced to analyse dependency relations between words in a sentence. A UNIT is a basic framework for concept representation and is composed of many slots. After generating a parsed tree from an input sentence, our semantic interpretation begins traversing the tree from right to left to discern the case frame in a stage as early as possible, since Japanese is a language in which verb is in the sentence-final and has a case frame. UNIT-to-UNIT interaction, which is performed at each node of the parsed tree, follows a bottom-up progression. There are UNIT descriptions at terminal (bottom) nodes and the UNIT descriptions are modified or merged into other UNITs in the course of the interaction. The results of the interaction will be transferred to upper nodes. The interaction process continues on upward until the top node; at this point, the semantic structure of the input sentence is finally obtained. The notion of UNIT-to-UNIT interaction is feasibly applicable to semantic interpretation of English.

## 1. INTRODUCTION

Semantic processing is very important for us to build a natural language (NL) understanding system. It will be true that semantics takes precedence over syntax when human beings understand language. Based on this assumption, some NL understanding system designers have totally abandoned the traditional use of grammars for linguistic analysis. They are based on special procedures of semantic interpretation to build up semantic structures, and the result of syntactic parsing is not used. Such systems without grammar often lack formalism.

We should not totally abandon a traditional use of grammars for linguistic analysis, since results of syntactic parsing fill the gap between an input sentence and its semantic structure. We have developed Extended LINGOL [13,12] that is the extended version of Pratt's LINGOL [8,9]. Pratt's LINGOL has a very good formalism to merge syntactic and semantic information. The idea is that the result of syntactic parsing, a parsed tree, is considered as a program tree which is evaluated at the time of semantic interpretation. In the course of the interpretation, UNIT-to-UNIT interactions are performed. Thus a parsed tree of LINGOL corresponds to an analysis tree of Montague grammar and the evaluation phase of the parsed tree is analogous to the translation phase of Montague grammar [3] (See the Sec.6).

Our Extended LINGOL inherits the semantic interpretation method from the original Pratt's LINGOL. After generating a parsed tree from an input sentence, semantic interpretation is set out. The parsed tree is composed of context-free rules to each of which a LISP program is attached. In other words, at each node of the tree, there is a program for making semantic interpretation. As will be explained in the Sec. 5., UNIT-to-UNIT interaction will take place at each node of the program tree. The interaction process continues on upward until the top node, at which it stops getting the results of the semantic interpretation.

## 2. <SEM>-PROGRAM TREE AND PARSED TREE

Our Extended LINGOL produces a parsed tree using both grammar and dictionary. The format of our grammatical rule is:

{<left> <right> {<advice> <cog>} <sem>}.

The left-right pair represents a context-free rule in the form of A ⟶ B or A ⟶ B C. The <advice>, which is introduced into our Extended LINGOL, is an arbitrary LISP program for controlling parsing process [13].

The role of <cog> and <sem> is the same as that of Pratt's LINGOL [9]. The <sem> is any LISP program to perform semantic interpretation. The Pratt's LINGOL offers us a flexible method of semantic interpretation.

In order to understand UNIT-to-UNIT interaction, we will briefly illustrate the interpretation method. By means of <sem> attached to each (augmented) context-free rule, we can obtain a <sem>-program tree from a parsed tree.

Consider the following very simple example. The input sentence is "10KGNOOMOSANOMIZU (water of 10 kg)", grammatical rules are:

```
(NP    (NLNK    NP)    (...)  <S-expr1>)
(NLNK  (NOUN    NO)    (...)  <S-expr2>)
(NP    NOUN            (...)  <S-expr3>)
(NOUN  (NPOS2  NOUN)   (...)  <S-expr4>)
(NPOS2 (QUANT   NO)    (...)  <S-expr5>)
(QUANT (NUMB   UNIT)   (...)  <S-expr6>)
```

and dictionary entries are:

Fig.1

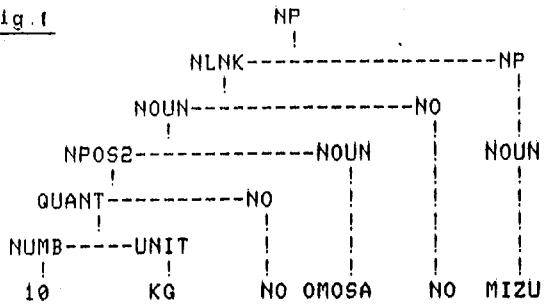



Fig.2(a)

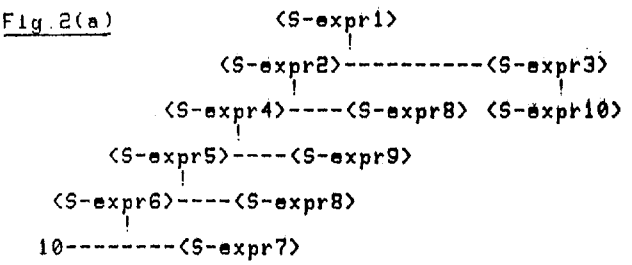
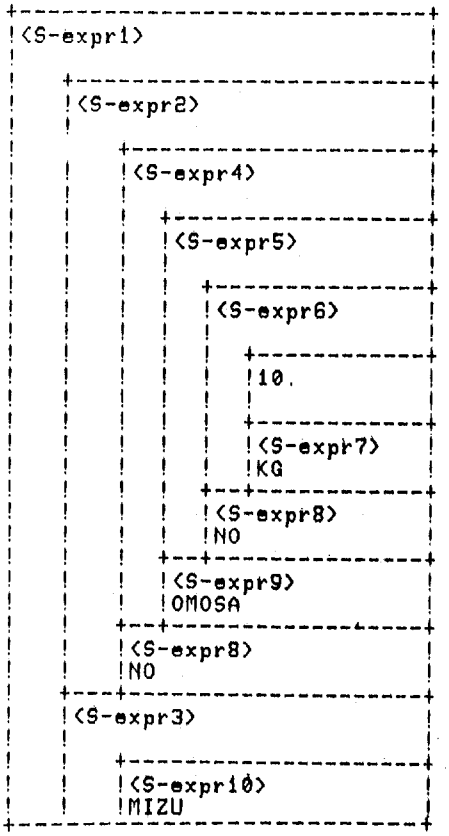



Fig.2(b)




```
(KG     UNIT          (...)  <S-expr7>)
(NO     NO            (...)  <S-expr8>)
(OMOSA  NOUN          (...)  <S-expr9>)
(MIZU   NOUN          (...)  <S-expr10>).
```

Fig.1 is a parsed tree of "10KGNOOMOSANOMIZU." Fig.2(a) is a <sem>-program tree obtained from Fig.1. Fig.2(b) is the nesting structure of the <sem>-program tree of Fig.2(a) which defines the scope of variables. At lower nodes, we see values of variables at upper nodes. For instance, from <S-expr4>, one can refer to the value of variables in <S-expr2> and <S-expr1>.

Semantic interpretation begins with the evaluation of S-expression at the top node. There are several built-in functions two of which are LG and RG, which are evaluated at each <sem>-tree node with left and right branches. The evaluation sequence of LG and RG determines the evaluation sequence of S-expressions at one-level-lower nodes. For example, in <S-expr1>, if the evaluation of RG precedes that of LG, <S-expr3> is evaluated and then <S-expr2>. The result of RG evaluation becomes equal to that of <S-expr3> evaluation. Usually, at each node of the <sem>-program tree, UNIT-to-UNIT interaction takes place and the results of the interaction are transferred to one-level-upper node. As will be explained before, the role of a parsed tree is similar to that of an analysis tree of Montague grammar.

## 3. UNIT DESCRIPTION

A UNIT is a basic framework for concept representation and is composed of many slots. Our UNIT description incorporates some useful features from KRL [1] which was developed by Bobrow and Winograd. Fig.3(a) is an example of our UNIT descriptions.

```
(MIZU unit ...                    (KG unit ...
     (self (a EKITAI))              (self (a OMOSA))
     (sf +natural)                  (sf %unit) ......)
     ..................)
(EKITAI unit ...
     (self (a BUSSITU))
     (sf +natural)
     ..................)
(BUSSITU  unit ...
     (self ...)
     (sf)
     ..........
     (OMOSA ((%value (a OMOSA)) - -NO))
     ............)
(OMOSA    unit ...
     (self ...)
     (sf ...)
     .........
     (VALUE ((%value (a OMOSA)) - -NO)
            <action-1>)
     ..................)
```

Fig.3(a)

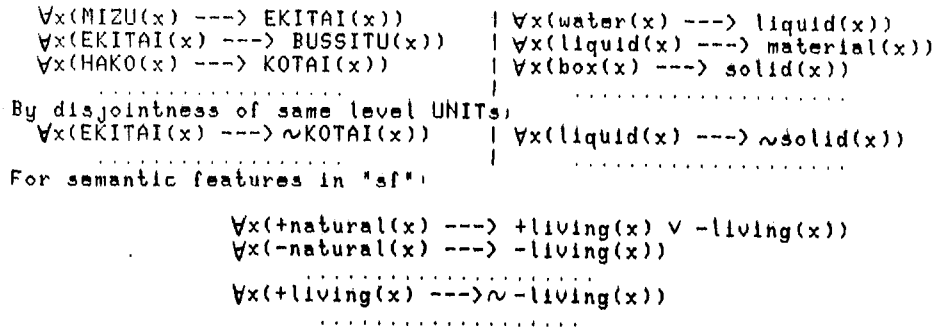



Fig.3(b)

The "self" slot which is present in each UNIT description enables us to understand the UNIT framework as a whole. As in KRL, the "self" slot is used for the hierarchical organization of UNITs and enables all information (slots) to transfer from superordinate UNIT to subordinate UNITs. For example, the "self" slot in MIZU (water) UNIT indicates that the superordinate UNIT of MIZU (water) is EKITAI (liquid).

Our UNIT descriptions are slightly different from KRL descriptions. Semantic features are incorporated into each special slot named "sf". The "sf" slot in MIZU (water) indicates that the semantic feature of MIZU is [+natural].

In order to express gross semantics of a UNIT description, we can use logical expressions of first-order predicate calculus. For example, gross semantics of MIZU UNIT is expressed as:

```
——> MIZU(UNIT);
——> +natural(UNIT).
```

Hierarchical organization of UNITs is expressed as a set of logical entailments [10]. For example, from Fig.3(a) we will have Fig.3(b).

We can regard Fig.3(b) as a set of axioms which is used in performing UNIT-to-UNIT interactions. The details will be explained in the Sec.5.

## 4. ORDINARY SLOT

Most UNITs include a block of ordinary slots which are classified into two categories, <satisfied> and <unsatisfied>:

```
<ordinary-slot>::= <satisfied>|<unsatisfied>.
```

The format of two ordinary slots is:

```
<satisfied>::=[<slot-name>=<value>]
<unsatisfied>::=[<slot-name>
        <precondition> (<action>)].
```

As a procedual attachment [1], we use a production rule [2,7]. A pair of <precondition> and <action> expresses a production rule in the form of:

```
<precondition> ⟹ <action>.
```

As will be explained later, <precondition> and <action> act as though they are to-fill and when-filled method of KRL [1].

Two UNITs, which relate to each other in UNIT-to-UNIT interaction, are called FILLER and ORIGIN. During the interaction, FILLER must satisfy some slot of ORIGIN. <Precondition> specifies conditions of FILLER filled in the slot.

[A] <PRECONDITION>

In order to satisfy some slot of ORIGIN, FILLER has to satisfy the <percondition>, which specifies not only semantics of FILLER but also Japanese surface cases that can follow FILLER in the sentence. <Precondition> is divided into two parts, <f-constraint> and <J-case>:

```
<precondition>::= [<f-constraint> • <J-case>].
```

Semantics of FILLER is expressed by <f-constraint>. On the other hand, Japanese surface cases, which can follow FILLER in a sentence, are specified in <J-case>.

For example, in BUSSITU (material) UNIT of Fig.3(a), there is an <unsatisfied> slot:

```
(OMOSA ((%value (a OMOSA)) - -NO)).
```

The <slot-name> and <precondition> are OMOSA (weight) and ((%value (a OMOSA)) - -NO), respectively. The <f-constraint> and <J-case> are (%value (a OMOSA)) and (- -NO), respectively. The <f-constraint> is expressed as follows:

```
——> %value(FILLER) ∧ OMOSA(FILLER).
```

(Note that logical "and" is always omitted in the description of <f-constraint>.) It is possible to describe any well-formed formula by using @OR and @NOT in <f-constraint>. For example, (%value (@OR (a WEIGHT) (a VOLUME))) is expressed as:

```
——> %value(FILLER) ∧ (WEIGHT(FILLER) ∨
                      VOLUME(FILLER)).
```

The <J-case> of (- -NO) describes what a Japanese surface case is allowed to follow FILLER in a sentence. "-" indicates none of Japanese surface cases should follow FILLER, and "-NO" indicates that Japanese surface case NO ("of") should follow FILLER.

[B] <ACTION>

After FILLER satisfies the <precondition> of some ordinary slot, the <action> which is any LISP program is activated. Typical effects of <action> are:

(1) Modification of UNITs and slots
(2) Creation of new UNITs and new slots
(3) Deletion of UNITs and slots.

If no <action> is specified, the <unsatisfied> slot becomes <satisfied> slot, whose <value> becomes FILLER's name but the <slot-name> remains unchanged.

## 5. UNIT-to-UNIT INTERACTION

As explained before, UNIT-to-UNIT interaction usually occurs at each node of <sem>-program tree. In other word, the structure of <sem>-program tree determines what UNITs should be interacted to each other. For example, at <S-expr4> of Fig.2(b), both UNITs of "10KG" and "OMOSA (weight)" are related by UNIT-to-UNIT interaction.

Two UNITs, which relate to each other in UNIT-to-UNIT interaction, are called FILLER and ORIGIN. During the interaction, FILLER must satisfy some <unsatisfied> slot of ORIGIN. If it is impossible to find out any satisfiable slot in ORIGIN, superordinate UNITs of ORIGIN will be retrieved through "self" until some satisfiable slot will be found. The satisfiability is determined by FILLER and <precondition> in an ordinary slot of ORIGIN.

At first, a surface case which follows FILLER is checked by using <J-case> in <precondition>. If this checking succeeds, then the semantics of FILLER is checked by using <f-constraint> in <precondition>. These checkings are expressed as follows:

> Given the semantics of FILLER and a set of axioms as shown in Fig.3(b), then examine whether <f-constraint> hold or not.

Let us consider two simple examples. As explained before, at <S-expr4> of Fig.2(b), the following two UNITs are interacted to each other:

```
FILLER:
   ((10 KG) unit ...
        (self (a OMOSA)) ...)

ORIGIN:
   (OMOSA unit ...
        .............
        (VALUE ((%value (a OMOSA)) - -NO)
               <action-1>)
        .................).
```

In this case, if a Japanese surface case of NO ("of") follows FILLER, then FILLER can satisfy VALUE-slot of (VALUE ((%value (a OMOSA)) - -NO) <action-1>), since the semantics of FILLER is:

$$\longrightarrow \text{OMOSA(FILLER)} \ ;$$
$$\longrightarrow \text{\%value(FILLER)}.$$

and it is easy to show that the following <f-constraint> holds:

$$\longrightarrow \text{OMOSA(FILLER)} \wedge \text{\%value(FILLER)}.$$

If the VALUE-slot is satisfied by FILLER, <action-1> will be activated to make further semantic interpretation if necessary. Let us consider another example:

```
FILLER:
   (MIZU unit ...
        (self (a EKITAI))
        (sf +natural)
        .................)

ORIGIN:
   (SOSOGU  unit ...
        (self ...)
        .........
        (THEME ((a EKITAI) -WO))
        .......................)
```

where the words, SOSOGU, EKITAI and MIZU in Japanese are POUR, LIQUID and WATER in English, respectively.

If a Japanese surface case of WO follows FILLER, the slot (THEME ((a EKITAI) -WO)) is satisfied by FILLER, because the semantics of FILLER is:

$$\longrightarrow \text{MIZU(FILLER)} \ ;$$
$$\longrightarrow \text{+natural(FILLER)}$$

and from a set of axioms as shown in Fig.3(b), we get

$$\forall x(\text{MIZU}(x) \longrightarrow \text{EKITAI}(x)).$$

It is easy to show that the following <f-constraint> holds:

$$\longrightarrow \text{EKITAI(FILLER)}.$$

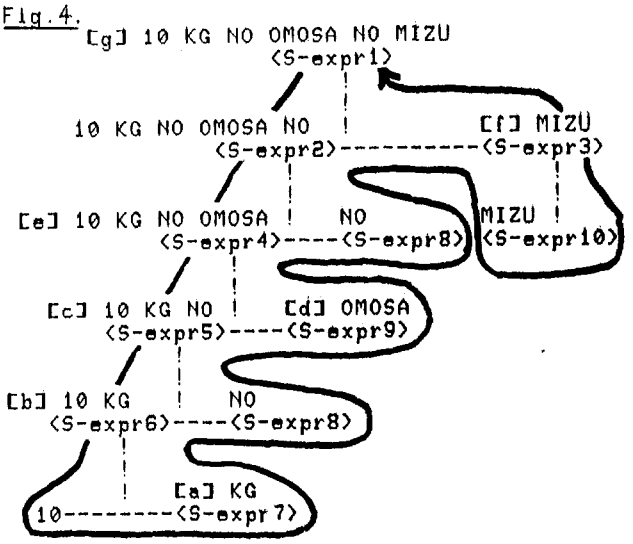
As the result of the interactions, the slot of (THEME ((a EKITAI) -WO)) becomes a <satisfied> slot of (THEME = MIZU).

## 6. SIMPLE EXAMPLE OF SEMANTIC INTERPRETATION BY UNIT-to-UNIT INTERACTION

Let us trace semantic interpretation

process by UNIT-to-UNIT interaction, provided that the input sentence is "10KGNOOMOSANOMIZU." The parsed tree and its <sem>-program tree are shown in Fig.1, Fig.2(a) and Fig.2(b). Depending on the evaluation sequence of LG and RG, we can traverse <sem>-program tree in any order (see the Sec.2). Suppose <sem>-program of Fig.2(a) is traversed as shown by the arrow of Fig.4.

Fig.4.




Trace of UNIT-to-UNIT interaction becomes as follows:

[a] KG ⟹
(KG unit ...
(self (a OMOSA))
(sf %unit) .....)

Create KG UNIT at <S-expr7>.

[b] 10 KG ⟹
((10 KG) unit ...
(self (a OMOSA))
(sf %value) .....)

At <S-expr6>, 10 and [a] are merged.

[c] 10 KG NO ⟹ [b] and NO

At <S-expr5>.

[d] OMOSA ⟹
((N000001 . OMOSA)
unit ...
(self ...)
(sf ...)
.....
(VALUE ((%value (a OMOSA)) - -NO)
<action-1>)
.....)

Create OMOSA UNIT at <S-expr9>.

[e] 10 KG NO OMOSA ⟹
((10 KG) unit ...
(self (a OMOSA))
(sf %value) ...)

At <S-expr4>, UNIT-to-UNIT interaction occurs between FILLER [b] and ORIGIN [d]. VALUE-slot of [d] is satisfied by FILLER and <action-1> is activated to remove UNIT [d], since OMOSA is redundant in this case.

[f] MIZU ⟹
((N000002 . MIZU)
unit ...
(self (a EKITAI))
(sf +natural)
.....)

Create MIZU UNIT at <S-expr10> and send it to <S-expr3>.

[g] 10 KG NO OMOSA NO MIZU
⟹ ((N000002 . MIZU)
unit ...
(self
(a EKITAI with
(OMOSA =
(10 KG))))
(sf +natural)
.....)

At <S-expr1>, UNIT-to-UNIT interaction occurs between FILLER [b] and ORIGIN [f]. There is no <unsatisfied> slot in MIZU UNIT, so superordinate UNITs are retrieved and a OMOSA-slot is found in BUSSITU UNIT (see Fig.3(a)).

## 7. CONCLUSION

In order to explain a basic notion of UNIT-to-UNIT interaction, we showed a very simple example in the Sec.6. Based on the idea, we have implemented a semantic interpretation system called EXPLUS [12,14]. Our experiments by EXPLUS proved that EXPLUS can extract semantic strucures from rather complicated Japanese sentences. Sato uses in turn the semantic structures as a source to generate Japanese sentences[11]. However, we have needed more refinements for UNIT description. For example, we have augmented following features to the UNIT description described in the Sec.3.

(a) Incorporation of arbitrary LISP programs in <precondition> through which we can specify FILLER's semantics in any level of details;
(b) A special slot "part-of" to organize part-whole relations [5];
(c) "Without" and "selector" descriptions to exclude undesirable slots in the superordinate UNITs.

They are related to the problems of knowledge representation. The details of (a)-(c) will be explained in [14]. In [14], more complex examples of UNIT-to-UNIT interaction will be explained.

From our experience, we believe that UNIT-to-UNIT interaction gives us a reasonable framework for semantic interpretation, and will be feasibly applicable to other kinds of languages such as Egnlish.

ACKNOWLEDGEMENT: The author is grateful to Dr. Fuchi, Head of the Pattern Information Division of Electrotechnical Laboratory, for his patient encouragement of this study. Thanks are also due to all members of the Machine Inference Section of Electrotechnical Laboratory, for their valuable discussions.

REFERENCES:

[1] Bobrow,D.G. and Winograd,T.: "An Overview of KRL, a Knowledge Representation Language", Cognitive Science, Vol.1, No.1, 1977.
[2] Davis,R. and King,J.: "An Overview of Production Systems", Stanford AIM-271, Oct.1975.
[3] Dowty,D.R.: "A Guide to Montague's PTQ", Indiana University Linguistic Club, Dec. 1978.
[4] Fillmore,C.J.: "The Case for Case", in Bach and Harms (Eds): "Universals in Linguistic Theory", Holt, Rinehart and Winston, 1968.
[5] Miller,G.A. and Johnson-Laird,P.N.: "Language and Perception", Harvard Univ. Press, 1976.
[6] Minsky,M.: "Framework for Representing Knowledge", in Winston (Ed.): "The Psychology of Computer Vision", McGrow-Hill, 1975.
[7] Newell,A.: "Productions Systems: Models of Control Structures", in Chase,W.G. (Ed.): "Visual Information Processing", Academic Press, 1973.
[8] Pratt,V.R.: "A Linguistic Oriented Programming Language", IJCAI3, 1973, 372-381.
[9] Pratt,V.R.: "LINGOL-A Progress Report", IJCAI4, 1975, 422-428.
[10] Reiter,R.: "On Reasoning by Default", in Waltz (Ed.): TINLAP2, 1978, 210-218.
[11] Sato,T.: "SGS: A System for Mechanical Generation of Japanese Sentences", Proc. of COLING80, 1980, (in this volume).
[12] Tanaka,H., Sato,T. and Motoyoshi,F.: "EXPLUS-A Semantic Parsing System for Japanese Sentences", 3rd USA-JAPAN Computer Conference, 1977, 236-240.
[13] Tanaka,H., Sato,T. and Motoyoshi,F.: "Predictive Control Parser: Extended LINGOL", IJCAI-79, 1979, 868-870.
[14] Tanaka,H.: "A Semantic Processing System for Natural Language Understanding", Research No.797, Electrotechnical Laboratory, July, 1979 (in Japanese).
[15] Wilks,Y.: "An Artificail Intelligence Approach to Machine Translation", in Schank and Colby (Eds.): "Computer Models of Thought and Language", Freeman and Company, 1973.
[16] Winograd,T.: "Understanding Natural Language", Academic Press, 1972.