PROCESSING OF SYNTAX AND SEMANTICS OF NATURAL LANGUAGE BY PREDICATE LOGIC

Hiroyuki Yamauchi

Institute of Space and Aeronautical Science,
University of Tokyo
4-6-1 Komaba, Meguro-ku, Tokyo 153, Japan

Summary

The syntax and semantic analyses of natural language are described from the standpoint of man-machine communication. The knowledge based system KAUS(Knowledge Acquisition and Utilization System) which has capabilities of deductive inference and automatic program generation of database access is utilized for that purpose. We try to perform syntax and semantic analyses of English sentences more or less conccurently by defining the correspondence between the basic patterns of English and the extended atomic formula in the framework of KAUS. Knowledge representation based on sets and logic, the sentence analysis utilizing this knowledge are given with some examples.

1. Introduction

This paper presents natural language understanding in man-machine invironments. The syntax and semantic analyses program is given almost in logical forms of the knowledge based system KAUS (knowledge Acquisition and Utilization System). KAUS is a logic machine based on the axiomatic set theory and it has capabilities of deductive inference and automatic program generation of database access.

In natural language understanding, it should be required that the syntax analysis is performed with association of word semantics. The descriptions of word semantics are fundamental as well as syntax features in language analysis. When using natural language to communicate with a machine, the understanding of meanings of sentences is presupposed to the machine.

We think of words as representing concept sets or property sets, and formalize them into a structure called SKELETON STRUCTURE using their element-set relationships. In this, we can represent semantic categories of words in hierarchical order. The correspondence between natural language expressions and system's logical formulas is given in the straightforward manner such as

"X give Y to Z" --- (GIVE X Z Y P)

where the variables X, Y and Z are usually bounded by their semantic categories. We call this set of representations ATOM FORMAT DEFINITION SET. Furthermore, causality relations of general facts and paraphrases of sentences are given as general axioms. Individual facts are also given as axioms. We call these sets of representations AXIOM SET. Conceptual schemas of databases can also be given in the axiom set. The KAUS's knowledge base comprises the above three components.

Utilizing this knowledge base, we try to perform the syntax and semantic analyses of natural language more or less conccurently in the sense that all these processes will be carried out through the deductive process in KAUS. At the time, the logic program of the analyses written in KAUS language is executed by the deductive process.

In the chapter 2, some considerations on natural language(NL) processing are given as preliminaries. In the chapter 3, the outline of KAUS is described, where the knowledge representation, the system language and the deductive inference rule are presented. In the chapter 4, a rather intuitive and straightforward approach to the analyses of English sentences is described with some examples. A part of the logic program of the analyses is also given there.

2. Some Considerations on NL Processing

When we are going to construct a natural language understanding system, the following four points must be clarified:

1). the main motivation and goal for NL processing
2). knowledge representation suitable for NL processing
3). the aspect of programming language and methodology by which the modification, updating and extension of the program of NL processing can be easily made.
4). the efficiency of processing

In the sequel, we will clarify our standpoint about the above matters.

2.1 Motivation and Goal

At present, the details of the mechanism of human language comprehension and knowledge memorization have not yet been clarified. These are still left uncertain. Now, let us consider the human process of natural language understanding.

When we read a sentence, we make an image (meaning) of each word and match the pattern of the sentence predicting what sorts of words will be coming to the next. The imagination, prediction and pattern matching are usually made involuntarily. Furthermore, if several sentences are following, we settle the scene of the discourse and extract the meaning of the current sentence with relation to the discourse. That is, the word meanings (semantics), the sentence pattern (syntax) and the world knowledge are fundamental components for the sentence comprehension.

Our main motivation and goal are to deal with the NL processing from the standpoint of man-machine communication, and with the above consideration, to write a syntax and semantics analysis program under the knowledge-based system KAUS which has capabilities of deductive inference and automatic program generation of database access.

2.2 Knowledge Representation

In natural language analysis, knowledge representation to parse the sentences and extract their meanings is most crucial. The various kinds of parsers have been revised by experts; among them, the ATN parser by W. Woods is widespread and used elsewhere. It is essentially a context-free grammar based one. Further, to represent the semantic structure of the sentences, the frame theory revised by M. Minsky is widespread. The system using the ATN or ATN-like parser and the frame or frame-like structure, present them in semantic networks. On the other hand, predicate calculus oriented knowledge representation is also available to language processing. We will adopt predicate calculus oriented representation of knowledge for NL processing and database definitions. In database applications, predicate logic is of greate advantage to define intensional and extensional data to which the deductive inference rule can be applied. Moreover, it has been pointed out that there exist similarities between natural language and predicate logic. But as the expressive power of first order logic is rather restricted, we will extend the formalism of first order logic. This formalism will be given in the next chapter.

2.3 Programming Language and Methodology

To make a computer understand natural language, we must program "the understanding mechanism" and give "the state of the world" to the machine. It should be noticed here that natural language contains ambiguous expressions in itself. It is usually very difficult how to process ambiguities of sentences. In general, "ambiguity" may be a property of subjects but should not be a property of the processing mechanism; the solution of the subject and the subject itself may be given plausibly or uncertainly but not the solution mechanism itself. With this consideration, we adopt the logic programming method which can be involved in KAUS's system language. By this method, we can design a sentence analysis program in the top-down style. The logic programming with use of the KAUS's formal logic is perspicuous, by which the modification, updating and extension can be made easily. In particular, the correspondences between the form of natural language expressions and the system's own formulas can be defined fairly in the straightforward manner, which are referred to in turn by the deductive inference and retrieval mechanism to translate the former into the latter.

2.4 Efficiency of Processing

With respect to the efficiency of processing, the program written in KAUS's logic programming style may be slightly less comfortable than the program written altogether in the procedural form becuase of the necessity of the execution of deductive retrieval program. However, the efficiency would be slightly sacrificed at the moment for the sake of clarity of the program.

3. Outline of KAUS

In this chapter, we describe briefly about an knowledge based system called KAUS(Knowledge Acquisition and Utilization System) which was realized in accordance with the design philosophy presented in the paper [6] and [7]. KAUS is constructed on the basis of the axiomatic set theory which has capabilities of deductive inference and automatic program generation for database access. It can be also applied to the logic programming of the semantic processing of natural language. In the following, we focus our discussions to the characteristics of knowledge representation formalism, the deductive inference rule and the system language features.

3.1 Knowledge Representation Formalism

We think of words as representing concept or property sets of things, and organize them into a structure on the basis of their set-theoretic implications. We will call such a structure SKELETON STRUCTURE. Using this formalism, we can represent semantic categories of words in hierarchical order. For example, the following set relations are structurized into Figure 1.

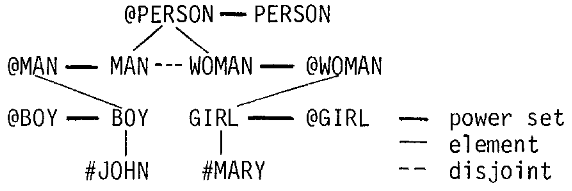


Figure 1. An Example of Skeleton Structure

$$\text{PERSON} \supset \text{MAN}\ ,\ \ \text{PERSON} \supset \text{WOMAN}$$
$$\text{MAN} \supset \text{BOY}\ ,\ \ \text{WOMAN} \supset \text{GIRL}\ ,\ \ \text{MAN} \cap \text{WOMAN} = \emptyset$$
$$\text{MAN} \ni \#\text{JOHN}\ ,\ \ \text{WOMAN} \ni \#\text{MARY} \qquad (1)$$

In the figure, the relation of set inclusion is uniformly represented as the element-set relations by using the concept of power sets. Then the power set of PERSON (we denote it by @PERSON) comprises all the subsets of PERSON:

$$@\text{PERSON} = [\ \text{MAN, WOMAN, BOY, GIRL},\ldots,\ \#\text{JOHN},\ \#\text{MARY},\ldots] \qquad (2)$$

Hereupon, let us consider the ordinary first order predicate logic at a moment. In the first order logic, domains of variables are described by using predicates. For example, "every man walks" is represented by

$$(\forall x)[\text{MAN}(x) \text{ ------> } \text{WALK}(x)] \qquad (3)$$

where MAN(x) is used for the domain restriction of the variable X. Thus, this restriction can be interpreted as "X is an element of MAN". Moreover, if it were required to answer the question "does every boy walks?", the following axiom would have to be given.

$$(\forall x)[\text{BOY}(x) \text{ ------> } \text{MAN}(x)] \qquad (4)$$

Then, using (3) and (4), the above question can be evaluated by Resolution Principle. But such descriptions as (3) and (4) are rather cumbersome. In place of them, we give the following (5) and (6) which have the same interpretation as (3) and (4) respectively.

$$(\forall x/\text{MAN})[\text{WALK}(x)] \qquad (5)$$
$$\text{MAN} \supset \text{BOY} \qquad (6)$$

where the prefix $(\forall x/\text{MAN})$ in (5) which can be included in our axiom set described later denotes "for every $x \in \text{MAN}$". The set relation (6) can be also given in our skeleton structure. Using both (5) and (6), we can derive the answer of the above question faster than the ordinary first order logic where such representations as (3) and (4) are used. This is the merit of the skeleton structure and will be much more clarified in the section of Deductive Inference Rule.

Atom Format Definition Set In the first order logic, constants, variables and functions are recognized as terms. Then, an atom is defined as $P(t_1, t_2, \ldots, t_n)$ where P is a n-place predicate symbol and t_i $(i = 1,\ldots,n)$ is a term. A formula is defined by using such atoms, logical connectives and quantifiers. However, when we wish to translate a natural laguage expression into a predicate form in the above formalism, we cannot directly handle phrases and clauses both of which are used as verb objects or verb complements. Therefore, we extend the atom definition as follows:

1). a formula can be permitted as a term besides constants, variables and functions
2). a function can be used as an atom --- we call this type of an atom PROCEDURAL TYPE ATOM (PTA), while the other type atom is called NONPROCEDURAL TYPE ATOM (NTA). (note: because of permitting a PTA as an atom, we can perceive that our logical formulas afford a sort of logic programming facilities.)

The atom format definition set provide us with a definition of a correspondence between some syntactic features of natural language and our logical formulas. In addition, PTA definitions are also given in the set. In the figure 2, some examples are illustrated , where all of the used atom formats conform to the following standard format definitions:

$$\text{NTA1}:\quad (\text{V}\ \ \text{S}\ \ X_1\ \ X_2\ \ \text{P}) \qquad (7)$$
$$\text{NTA2}:\quad (\text{A}\ \ \text{O}\ \ \text{V}) \qquad (8)$$
$$\text{PTA}:\quad '(\text{F}\ \ Y_1 \text{---} Y_m;\ X_1 \text{---} X_n) \qquad (9)$$

NTA1 is usually used for representing a simple sentence, while NTA2 is used for representing an Adj.X Noun phrase. In the NTA1 definition, V denotes a predicate symbol. S, X_1, X_2 and P are terms, of which S is usually used as the subject of V, and P is a formula which denotes the modifiers concerning time, locus, goal, reason and manner of V. Some of the terms may be omitted according to the syntactic feature of a sentence. In the NTA2 definition, (A O V) denotes "the Attribute of Object is Value". Finally, in the PTA definition, F denotes a function name which takes input variables $X_1,\ldots, X_n$ and output variables $Y_1,\ldots, Y_m$. It must be remarked here that some of the variables in (7), (8) and (9) may be bounded by their domains or semantic categories, but they were omitted for the sake of simplicity.

"X give Y to Z"	---	(GIVE X Z Y P)
"red X"	---	(COLOR X RED)
"two X"	---	(CARDINAL X 2)
"X on Y"	---	(ON X Y P)
"X be father of Y"	---	(FATHER Y X)
"X + Y = Z"	---	'(SUM Z; X Y)
"X / Y = Z"	---	'(DIV Z; X Y)
"sin(X) = Y"	---	'(SIN Y; X)

Figure 2. Examples of Atom Definition

Axiom Set The axiom set comprises descriptions of the world which specify general facts such as causality relations of things and sentence paraphrasing, and individual facts such as "John gave a red car to Mary". These facts are represented by formulas in the standard form

$$Q\ (\sim F \vee G\ ) \qquad (10)$$

Where Q represents quantification of variables, taking the form $(q_1 v_1/d_1)(q_2 v_2/d_2)\ldots(q_k v_k/d_k)$, in which q_i denotes a quantifier symbol $\forall$ or $\exists$, v_i is a variable name and d_i indicates the domain of the variable v_i. F and G represent premise and conclusion respectively in the formula; namely, F ---> G. More clearly, F is constituted from NTAs and PTAs by using logical connectives $\wedge$, $\vee$ and $\sim$, and also the same is G except that no PTAs are permitted. The internal representation of a formula in KAUS is an AND-OR tree to whose root node the quantification part Q is attached (see Figure 3).

Now, we have here several examples* :

Example 1. We can represent a causality relation of "give" as follows:

$$(\forall x/\mathrm{PS})(\forall y/\mathrm{PS})(\forall z/\mathrm{PO})[\sim(\mathrm{GIVE}\ x\ y\ z\ (\mathrm{TIME\ past}))\vee((\mathrm{HAVE}\ y\ z\ (\mathrm{TIME\ present}))\wedge\ \sim(\mathrm{HAVE}\ x\ z\ (\mathrm{TIME\ present})))]$$

This says that "if x ($\in$PerSon) gave z ($\in$Physical-Object) to y ($\in$PerSon), then y has z and x has not z at the moment".

Example 2. We can paraphrase "x give z to y" to "y receive z from x" :

$$(\forall x/\mathrm{PS})(\forall y/\mathrm{PS})(\forall z/\mathrm{PO})(\forall p/\mathrm{VMOD})[\sim(\mathrm{GIVE}\ x\ y\ z\ p)\vee(\mathrm{RECEIVE}\ y\ z\ x\ p)]$$

where p/VMOD specifies that p is any formula which qualifies the predicates.

Example 3. We can express the meaning of "float" as follows :

$$(\forall x/\mathrm{PO})(\forall y/\mathrm{LQ})(\forall u/\mathrm{RNUM})(\forall v/\mathrm{RNUM})(\forall p/\mathrm{VMOD})[\sim((\mathrm{SPEC\text{-}GR}\ x\ u)\wedge(\mathrm{SPEC\text{-}GR}\ y\ v)\wedge(\mathrm{LT}\ u\ v))\vee(\mathrm{FLOAT}\ x\ y\ p)]$$

This says that "if the specific gravity of x is u, the specific gravity of y is v and u is less than v , then x float on y".

Example 4. The fact, "John gave a red car to Mary" is represented as follows :

$$[(\mathrm{GIVE}\ \#\mathrm{JOHN}\ \#\mathrm{MARY}\ \mathrm{CAR}\ (\mathrm{TIME\ PAST}))\wedge(\mathrm{COLOR\ CAR\ RED})\wedge(\mathrm{QF\ CAR\ A})]$$

Example 5. The fact, "John drinks coffee after John plays tennis" is represented as follows :

$$(\mathrm{DRINK}\ \#\mathrm{JOHN}\ \mathrm{COFFEE}\ (\mathrm{TIME}\ (\mathrm{AFTER}\ (\mathrm{PLAY}\ \#\mathrm{JOHN}\ \mathrm{TENNIS}\ (\mathrm{TIME\ PRESENT})))))$$

All of these general/individual facts can be put into the axiom set and they will be referred to through the deductive retrieval mechanism in KAUS.

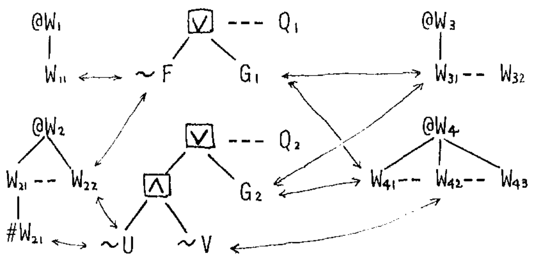


Figure 3. Interconnection between Axioms and Skeleton Structure

3.2 Deductive Inference Rule

In the previous section, we have given the formalism of knowledge representation, where the concepts and properties of words can be partially ordered according to their set-theoretic implications. Moreover, we have defined the correspondence between the syntax features of natural language and logical expressions somewhat in the straightforward manner. We have also described how to represent universal/individual facts in our axiom set. However, it must be stressed here that these three types of knowledge are not independently presented in the knowledge base but interrelated each other with respect to domains of variables of atoms; that is, the bi-directed access paths to each constituent of the knowledge base are mutually defined (see Figure 3). By making much use of these access paths, the knowledge only necessary for deduction can be retrieved efficiently. The deduction is performed by the inference rule comprising the following four components:

1). S R: Selection Rule for a literal resolved upon
2). TIC: Test for Implicative Condition
3). R R: Replacement Rule
4). T T: Test for Termination of deduction

SR(Selection Rule) A literal resolved upon is selected from the query tree. The selection criterions are as follows. First, the left-to-right search for a non-evaluated NTA(non-procedural type atom) is made in the tree. When it is found, it is checked whether a constant or $\exists$-variable is contained in the NTA. If it does the case, this NTA is took as a candidate resolved upon. If no such a literal is presented in the tree, certain counterplans are executed. The details would be omitted here.

TIC(Test for Implicative Condition) After the candidate literal to be resolved upon is decided, axioms which contain the literals with the same predicate symbol as the candidate literal's but with the opposite sign are retrieved in the knowledge base. At that occasion, the search domain is narrowed to the subset of the axiom set by indexing the variable domains of the

---

*). The real notation implemented by KAUS is partly modified (see 3.3 System Language).

literal resolved upon. The skeleton structure is referred to for this purpose.

Now, let us denote P as such a literal searched in the knowledge base and C as a candidate literal resolved upon. TIC checks the implicative conditions of P ---> C. One of the conditions to be checked is the set relation of the P's variable domains to the corresponding C's variable domains. The Table 1 shows this condition. Let us consider an example:

$$\text{P:}\quad (\exists y/\text{GIRL})(\forall x/\text{MAN})(\forall p/\text{VMOD})[(\text{LOVE } x\ y\ p)] \qquad (11)$$

$$\text{C:}\quad (\forall u/\text{BOY})(\exists v/\text{WOMAN})(\forall q/\text{VMOD})[(\text{LOVE } u\ v\ q)] \qquad (12)$$

P says that "a certain particular girl is loved by every man". On the other hand, C says that "every boy loves some woman" in the ordinary sense. In this case, P ---> C can be established because all the corresponding variable pairs (x, u), (y, v) and (p, q) satisfy the condition in Table 1. However, if P and C are

$$\text{P:}\quad (\forall x/\text{BOY})(\exists y/\text{WOMAN})(\forall p/\text{VMOD})[(\text{LOVE } x\ y\ p)] \qquad (13)$$

$$\text{C:}\quad (\exists v/\text{WOMAN})(\forall u/\text{BOY})(\forall q/\text{VMOD})[(\text{LOVE } u\ v\ q)] \qquad (14)$$

then, P ---> C can never established in spite of satisfaction of Table 1, because the most general significant unifier does not exist. TIC tests this condition, too.

RR(Replacement Rule) Let F be an axiom containing P and let G be the query clause containing C. RR generates a new query clause R after substitution σ has been done. That is, $C\sigma$ is replaced with $(F\sigma - P\sigma)$, resulting in

$$\text{R:}\quad Q_R\ (G\sigma - C\sigma) \bullet (F\sigma - P\sigma) \qquad (15)$$

where • denotes that $(F\sigma - P\sigma)$ should be attached to the place where C was resided. Substitution σ is defined in Table 1.

TT(Test for Termination) When all the nodes in the query tree have been evaluated, deduction is terminated. Though we have said that all NTA nodes in the query tree are evaluated by means of the above SR, TIC and RR, we have not said almost anything about PTAs. We must now mention about them. PTAs in the query tree are in general, ready to be evaluated just at the time when all its input variables have been filled up with their values. Then, the evaluation is performed by calling the subroutines attached to PTAs.

3.3 System Language

The system language provide us with fucalties of constructing knowledge base and relational database. Besides that, it can be used as substitute for logic programming and query representations. The syntax of the system language has already been supposed in the section 3.1. At this section, we present the really implemented features of the language briefly. Details are excluded because this is not the purpose of this paper.

Table 1. Implicative Condition

Q_{P_i}	$Q_{\bar{C}_i}$	Q_{R_i}	X_{R_i}	CONDITION
$\forall$	$\forall$	$\forall$	$X_{P_i} \cap X_{\bar{C}_i}$	$X_{P_i} \cap X_{\bar{C}_i} \neq \emptyset$
$\forall$	$\exists$	$\exists$	$X_{\bar{C}_i}$	$X_{P_i} \supset X_{\bar{C}_i}$
$\exists$	$\forall$	$\exists$	X_{P_i}	$X_{P_i} \subset X_{\bar{C}_i}$
---	$\forall$	$\forall$	$X_{\bar{C}_i}$	---------
---	$\exists$	$\exists$	$X_{\bar{C}_i}$	---------
$\forall$	---	$\forall$	X_{P_i}	---------
$\exists$	---	$\exists$	X_{P_i}	---------
$\exists$	$\exists$		(non-implicative)	

P: $Q_P(X_{P_1}\ X_{P_2}\ X_{P_3})$ $\bar{C}$: negated form of query C

$\bar{C}$: $Q_{\bar{C}}(X_{\bar{C}_1}\ X_{\bar{C}_2}\ X_{\bar{C}_3})$

R: $Q_R(X_{R_1}\ X_{R_2}\ X_{R_3})$ R: replaced form of $\bar{C}$

The syntax of the system language is based on the tuple $(V_0, V_1, \ldots, V_n)$ where V_0 is either a system command name, a predicate symbol or a variable whose domain is a PREDICATE; $V_i (i \neq 0)$ is a term as which a string and a numerical constant and a formula are permitted. The system language has the following characteristics that are not included in the first order logic:

1). Variables can be explicitly specified. For example, [AX/MAN][EY*/WOMAN] where A and E denote the universal and existential quantifier respectively, and the symbol * attached to Y denotes that Y is a query variable.
2). A predicate symbol itself can be a variable. For example,
    [EX*/PRED]($X, #JOHN, #MARY)?
    where the symbol $ attached to X denotes that X is a variable and the symbol ? denotes that the expression is a query.
3). A recursive expression can be permitted.
4). A procedural type atom PTA --- a function --- can be permitted as an atom. For this reason, an aspect of logic programming is obtained.

4. Syntax and Semantics Analyses

In syntax and semantics analyses, word meanings, sentence patterns and world knowledge are fundamental. Characteristics of the sentence analysis partly depend on knowledge representation used. As described in the chapter 3, we represent knowledge in the framework of sets and logic in KAUS. The characteristics of our sentence analysis program are that, during the analysis, we use the rather direct correspondence between the basic sentence patterns (syntax) of natural language and the extended atomic formulas in KAUS, and that the pattern matching method can be used together with the deductive inference rule. A more characteristic is that words in a clause are put into four groups preserving the word order, each of which contains subjects, the main verb, direct objects/complements and indirect objects/complements respectively. In this chapter we present the English sentence analysis program using the above method.

4.1 Descriptive Presentation of the analysis

The analysis takes the following steps:

1. The parts of speech of each word in the input sentence and atom format definitions attached to each predicative/attributive word such as verbs, adjectives, prepositions and the others are fetched from the knowledge base.
2. The input sentence is partitioned into a set of clauses by considering the positions of conjunctions, relative pronouns and punctuation marks.
3. Words in a clause are put into four groups preserving the word order in the clause, each of which may contain subjects, the main verb, direct objects/complements respectively. For this purpose, the correspondence relation between the NL syntax and the atomic formula syntax is utilized.
4. Each phrase in the four groups is decided whether it indicates qualification of nouns or of predicates.
5. If there are AND/OR conjunctions in a clause it is decided whether they are in the scope of a preposition or not.
6. After all of the words in a clause have been interrelated each other, the remaining clauses are processed by repeating 3 to 5.
7. After all of the clauses have been processed, substantiation of each personal and demonstrative pronoun is established.
8. In consequence, the extended formula deduced from the input sentence is obtained.

To get comprehension of the above description, let us consider the next example:

IN HIBIYA-PARK, JOHN MET JACK AND MARY. (16)

This is the case that the sentence comprises only one clause. On this sentence, the basic sentence pattern of "meet", that is, "PERSON1 meet PERSON2", is fetched from the atom format definition set. The atom format definition of the preposition, "in" is also fetched; that is, "THING in PLACE". But this will not be used in this case. Then, grouping of words in the sentence is made according to the pattern "PERSON1 meet PERSON2", resulting in

group 1 : [IN HIBIYA-PARK, JOHN*]
group 2 : [MET]
group 3 : [JACK* AND MARY*]
group 4 : [ϕ] ; an empty group

where the words marked with the star symbol * denote the instances of PERSON1 and PERSON2. This can be established by using the skeleton structure in which JOHN, JACK and MARY are defined as elements of PERSON. In the next place, the phrase "IN HIBIYA-PARK" is decided as indicating qualification of the verb "MET". The conjunction "AND" is then determined as to be in the scope of the conjunction of direct objects of "MET". As the final result, we obtain the following formula:

$$[(\text{MEET JOHN JACK (TIME PAST)} \wedge (\text{PLACE (IN HIBIYA-PARK)})) \wedge (\text{MEET JOHN MARY (TIME PAST)} \wedge (\text{PLACE (IN HIBIYA-PARK)}))] \quad (17)$$

Let us consider a more complex sentence which contains a relative clause and a personal pronoun in it:

JOHN GAVE A RED CAR TO MARY WHO HE LOVES. (18a)
JOHN GAVE A RED CAR TO MARY WHO LOVES HIM. (18b)

In this case, both of the sentence (18a) and (18b) are split into the two clauses respectively; among which, each nucleus clause of (18a) and (18b) is the same, that is, "John gave a red car to Mary"; but the relative clause in (18a) means that "John loves Mary" while that in (18b) means that "Mary loves John". On the nucleus clause, the basic sentence pattern "PERSON1 gave THING to PERSON2" for "give" and "red THING" for the attributive adjective "red" are fetched from the knowledge base. Then, grouping of words in the clause is made according to the basic sentence pattern, resulting in

group 1 : [JOHN*]
group 2 : [GAVE]
group 3 : [A RED CAR*]
group 4 : [TO MARY*]

In the next place, the pattern "red THING" is applied to "RED CAR" in the group 3, and in consequence, the formula (COLOR CAR RED) is derived. We translate the indefinite article "a" into the formula (QF CAR A) to denote that "car" is qualified by "a" in the clause. Thus, all the semantic relations among words in the nucleus clause "John gave a red car to Mary" have been derived:

$$(\text{GIVE \#JOHN \#MARY CAR (TIME PAST)}) \wedge (\text{COLOR CAR RED}) \wedge (\text{QF CAR A}) \quad (19)$$

The relative clause in (18a), "who he loves", is transformed to (LOVE HE MARY (TIME PRESENT)), and "who loves him" in (18b) is transformed to (LOVE MARY HE (TIME PRESENT)). This can be attained by making use of the basic sentence pattern, "PERSON1 love PERSON2". In case of "who he loves", the following four word groups are initially generated by using this pattern.

--- (WHO) HE LOVES ---
group 1 : [(WHO) HE*]
group 2 : [LOVES]

```
group 3 :    [ ∅ ]    ; an empty group
group 4 :    [   ]    ; not to be used
```

Then, taking account of the state of the group 1 and the group 2, the antecedent of "who" is decided to be the direct object of "love". In the last place, the personal pronoun "he" is substantiated by "John". The similar discussion can be given to the case, "who loves him", and therefore, it is omitted here. The final results of the analysis of (18a) and (18b) are

```
(GIVE #JOHN #MARY CAR (TIME PAST))
  ∧(COLOR CAR RED)∧(QF CAR A)
     ∧(LOVE #JOHN #MARY (TIME PRESENT))  (20a)
(GIVE #JOHN #MARY CAR (TIME PAST))
  ∧(COLOR CAR RED)∧(QF CAR A)
     ∧(LOVE #MARY #JOHN (TIME PRESENT))  (20b)
```

So far, we have been concerned with declarative sentences. With regard to interrogative and imperative sentences, we transform them to declarative sentences prior to the successive analysis of them. Further, passive sentences have not been taken into account of hitherto. The passive voice is especially used in sentences in which it is unnecessary or undesirable to mention the agent, though the agent may be expressed by means of an adjunct with "by". The verbal meaning of the passive voice may also be brought out by adjuncts expressing other adverbial relations, such as time, manner, cause or instrument. A passive sentence may be analyzed with adaptation of the special treatment of the verb "be" followed by a past participle. For example,

```
JOHN IS LOVED BY MARY.
 ---> [JOHN] IS [LOVED] [BY MARY].
 ---> MARY LOVES JOHN.
 ---> (LOVE #MARY #JOHN (TIME PRESENT))  (21)

THE BOOK IS WRITTEN IN ENGLISH.
 ---> [THE BOOK] IS [WRITTEN] [IN ENGLISH].
 ---> [$X] WRITE THE BOOK IN ENGLISH.
 ---> (∃x/PS)[(WRITE x BOOK (MANNER
          (IN ENGLISH))∧(TIME PRESENT))
            ∧(QF BOOK THE)]              (22)
```

where the [ ] is used to denote special attention to readings. The special treatment of the verb "be" is not only introduced to passive sentences but also to the other fragments shown in Table 2. For example, a sentence pattern with the formal subject of "be" is treated as

```
IT IS EASY TO PLEASE JOHN.
 ---> [TO PLEASE JOHN] IS [EASY].
 ---> (GRADE-DIFFI (TOINF (TO PLEASE #JOHN))
                   EASY)                 (23)

IT IS SNOOPY THAT HAS STOLEN THE FISH.
 ---> [THAT HAS STOLEN THE FISH] IS [SNOOPY].
 ---> (STRESS-VAL (STEAL #SNOOPY FISH
        (TIME PRES.PERFECT)) #SNOOPY)
           ∧(QF FISH THE)               (24)
```

It should be denoted here that both of (23) and (24) have been transformed to a similar form in terms of (A O V) atoms.

Table 2. Treatment of the Fragment of " X be Y "

X be Y	Atomic Formula
N be Adj.	(ATTR N Adj.)
N_1 be Prep. N_2	(PREP N_1 N_2)
N_1 be Rel.N of N_2	(REL.N N_2 N_1)
Rel.N of N_1 be N_2	(REL.N N_1 N_2)
N_1 be N_2	(ATTR N_1 N_2)
THERE be N_1 Prep. N_2	(PREP N_1 N_2)
IT be Adj. TO-INF	(ATTR TO-INF Adj.)
IT be Z THAT-CLAUSE	(ATTR THAT-CLAUSE Z)
N be P.P. Adjunct	(PRED S U V P)

note. N : Noun
Adj. : Adjective
Prep. : Preposition
Rel.N : Attributive noun
TO-INF: To-infinitive
P.P. : Past Participle

4.2 Logic Program

By using a logic programming method, we can clearly show what should be done in the program depending on an approach taken for the sentence analysis, in which modification, updating and extension can easily be made. In the previous section, we have shown our approach by which a sentence may be analyzed rather intuitively and straightforwardly in the framework of KAUS. According to this method, we present, in this section, a part of the program by which an input clause is analyzed yielding an extended formula in KAUS. The execution of the program is undertaken by the deductive retrieval process with use of the merits of our knowledge representation. In the following representation of the program, we made convention that the top formula is concluded if all of the successive formulas indented are evaluated (proofed).

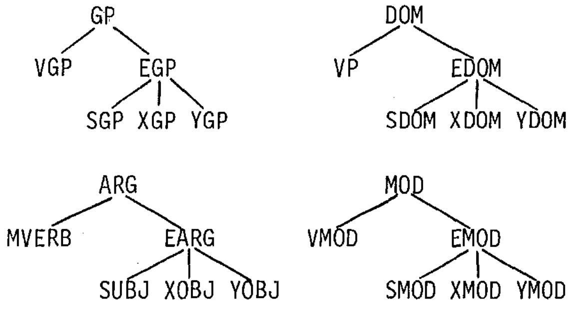
Program

```
 (LOGICAL-FORM EF S)
   '(GET-VINF VP MVERB ; S)
   '(GET-DOM SDOM XDOM YDOM ; VP S)
   '(CREATE-GROUP SGP VGP XGP YGP ;
                          SDOM VP XDOM YDOM S)
    (ARGUMENT SUBJ SDOM SGP)
    (ARGUMENT XOBJ XDOM XGP)
    (ARGUMENT YOBJ YDOM YGP)
    (MOD-ATOM SMOD SUBJ SGP)
    (MOD-ATOM XMOD XOBJ XGP)
    (MOD-ATOM YMOD YOBJ YGP)
    (VMOD-ATOM VMOD VGP SGP XGP YGP)
    (KERNEL-S KS VP MVERB SUBJ XOBJ YOBJ VMOD)
   '(CNCT-LOGICAL EF ; KS SMOD XMOD YMOD)
   '(ALL-MARKED ; S)

---
```

```
(ARGUMENT EARG EDOM EGP)
  '(MARK EGP ; EDOM EGP)
  '(UNMARK-PREP-OBJ EGP ; EGP)
  '(MAKE-SET EARG ; EGP)
---
(ARGUMENT EARG EDOM empty)
  '(SEARCH REL ; INPUT)
  '(NON-EQ REL WHERE)
  '(ANTEC X ; REL INPUT)
  '(PRE-W Z ; REL)
  '(SYN-CAT K ; Z)
  '(NON-ELM K PREP)
  '(PUT-ELM EARG ; X)
---
(MOD-ATOM EMOD EARG EGP)
  '(DET-MOD DET ; EARG EGP)
  '(ADJ-MOD ADJ ; EARG EGP)
  '(NPREP-MOD NPREP ; EARG EGP)
  '(MAKE-EF EMOD ; DET ADJ NPREP)
---
(VMOD-ATOM VMOD VGP SGP XGP YGP)
  '(TENSE-MOD TENSE ; VGP)
  '(AUX-MOD AUX ; VGP)
  '(VPREP-MOD VPREP ; VGP SGP XGP YGP)
  '(ADV-MOD ADV ; VGP SGP XGP YGP)
   (TO-INF-MOD INF VGP SGP XGP YGP)
  '(MAKE-EF VMOD ; TENSE AUX VPREP ADV INF)
---
..............
..............
```

```
*********************************************
```



```
*********************************************

  IN HIBIYA-PARK, JOHN MET JACK AND MARY.

  VGP = [MET]
  SGP = [IN HIBIYA-PARK, JOHN]
  XGP = [JACK AND MARY]
  YGP = [ φ ]  ; not used
  VP  = [VP1]   cf. VP1: Verb × Direct Object
  SDOM = [PERSON]
  XDOM = [PERSON]
  YDOM = not used
  MVERB = [MEET]
  SUBJ  = [JOHN]
  XOBJ  = [JACK, MARY]
  YOBJ  = not used
    .      EF = [(MEET #JOHN #JACK (TIME PAST)
    .             ∧(PLACE (IN HIBIYA-PARK)))
    .            ∧ (MEET #JOHN #MARY .......)))]
```

Figure 4. A Sample Process by the Program

The first program denotes that an input clause S is translated into an extended formula EF by evaluating the successive thirteen indented formulas, of which the ARGUMENT formulas are in turn to be replaced with the premises of the second or the third ARGUMENT program, or the other ARGUMENT programs that are not exhibited here, by the deductive inference rule. Then, these premises are in turn to be evaluated. The third ARGUMENT program exhibited above is concerned with a relative clause except where- and prep.-relative-noun clauses. The other NTAs in the first program may be evaluated in the similar way. The figure 4 shows a sample process of the program, where the main terms used in the program are also shown by categorizing them hierarchically in the skeleton structure.

Conclusion

We have described the syntax and semantics analyses of natural language (English) within the framework of the intelligent man-machine system KAUS. The more the volume of data related to the sentence analysis is enlarged in the knowledge base and the sentence analysis program itself is extended, the more the class of acceptable sentences will be broadened. This may be ensured by the method described hitherto.

Acknowledgement

The author is indebted to associate professor Setsuo Ohsuga of University of Tokyo, who gave the author helpful suggestions in the presentation.

REFERENCE

[1]. Bertram Bruce, Case Systems for Natural Language, Artificial Intelligence 6, 1975.
[2]. C.L. Chang & R.C.T. Lee, Symbolic Logic and Mechanical Theorem Proving, Academic Press, 1973.
[3]. Hendrix G.G. et al., Developing a Natural Language Interface to Complex Data, Proc. 3rd VLDB, Part II, pp.37-58, 1977.
[4]. Herve Gallaire & Jack Minker (edited), Logic and Databases, Plenum Press, New York, 1978.
[5]. Robert F. Simmons, Some Relations between Predicate Calculus and Semantic Net Representations of Discourse, 2nd IJCAI, 1971.
[6]. S. Ohsuga, Semantic Information Processing in Man-Machine Systems, Proc. IEEE, 1977.
[7]. S. Ohsuga, Toward Intelligent Interactive Systems, Proc. The IFIP W.G. 5.2 Workshop Seilac II on Methodology of Interaction, North Holand, 1979.