TR-243

Constraint Analysis on Japanese Modification

by

R. Sugimura and H. Miyoshi

March, 1987

Constraint Analysis on Japanese Modification

Ryoichi Sugimura & Hideo Miyoshi

Institute for New Generation Computer Technology (ICOT)
Mitakokusai Building, 21F
1-4-28, Mita, Minato-ku, Tokyo 108 Japan

ABSTRACT

The Japanese grammatical *constraints* on *modifications* using the *lazy evaluation mechanism*, and its analysis are presented. Lazy evaluation programming enables us to write complicated constraints on grammatical modifying relations (MRs) simply and statically, and to deal with the ambiguity of MRs explicitly. This paper describes simple methods which analyzes the MRs using constraint facilities of the logic programming language, CIL.

This analysis has three stages.

In the first stage, sentences are segmented into a sequence of phrases and analyzed as a right-branching structure. In this stage, no ambiguity exists.

In the second stage, based on the right-branching structure, MRs between phrases are analyzed in unification-based construction mechanism. In this stage, syntactic and semantic features are used to reduce the ambiguities of MRs. All of the MRs are represented as three kinds of logical values:0 (modifies), 1 (does not modify), and Unbound (modifies or does not modify).

In the last stage, contextual analysis is carried out and contextual information is used to disambiguate ambiguous MRs.

1. Introduction

This paper proposes one kind of constraint programming approach [Dincbas 86] to analyzing ambiguity in complicated Japanese grammatical modifying relations (MRs). MRs means a relation- ship in which one phrase modifies other phrases in a sentence [Hashimoto 80][Watanabe 81]. The complexity of Japanese sentence analysis comes from the ambiguity of MRs [Ozeki 86]. The traditional approach to this ambiguity depends on the heuristics applied in sentence analysis [Nakamura 86][Tsujii 84]. These heuristics are applied to reduce the ambiguity basically in two- valued logic, and there is no room to represent ambiguity explicitly. The main feature of this analysis is the treatment of MRs using three kinds of logical values: bound value (1), bound value (0), and unbound value (Unbound). In this framework, ambiguity is expressed explicitly as Unbound. The constraint programming on MRs enables us to handle these ambiguities in the proper stage in contextual analysis [Grotz 86].

Section 2 describes the grammatical structure of a Japanese sentence and various kind of constraints on MRs.

Section 3 refers to, the former approaches [Nakamura 86] based on the heuristics for the disambiguations of MRs, and discusses the basic problems of those approaches.

Section 4 outlines the programming languages, CIL [Mukai 85], and describes its constraint programming aspects.

Section 5 outlines the simple framework for semantic analysis based on the Situation Theory [Barwise 86][Barwise 83].

Section 6 describes the mechanism for analyzing the MRs of Japanese sentences using the lazy evaluation programming framework.

Section 6.1 describes the analysis of the right-branching phrase structure of a Japanese sentence as the first stage of MR analysis. There is no discussion on whether Japanese is right-branching [Gunji 87] or leftbranching [Kuno 73] because of the space limitation. The first stage of this analysis does not generate a ambiguities and right-branching structure is passed to the second stage of the analysis.

Section 6,2 gives syntactic and semantic analysis of one sentence as the second stage of MR analysis. First, constraints on all kinds of MRs according to the right-branching phrase structure are set up. In this stage, unification mechanism constructs the syntactic and semantic structure and reduces ambiguity of MRs. A special feature of this stage is that no heuristics are applied to reduce the ambiguity of MRs. If there are still any ambiguities in MRs. They are left as unbound logical values.

Section 6,3 represents the contextual analysis. It gives some examples and discusses the feature direction towards disambiguation with contextual information. There are many kinds of open problems in this stage which are beyond our research ability. They will be studied in the feature.

2. Modifications in Japanese Sentences

Unlike European languages, which are based on phrase structure, Japanese sentence has a dependency structure among its constituents [Nitta 86]. A Japanese sentence can be segmented into a set of special grammatical phrases (called 'bunsetu' in Japanese). This papers, uses the term '*PHRASE*' to mean Japanese 'bunsetsu'. A *PHRASE* consists of some words and can be regarded as a minimal semantic element to present various kinds of case in a sentence. Any *PHRASE* has a dependent relation with other *PHRASE*s, more exactly, modifying relations (MRs). The inside structures of the *PHRASE* have been well studied in Japanese linguistics [Hashimoto 80][Mizutani 83][Watanabe 81][Tokieda 41], so details are not given here. Section 2.1 describes the grammatical structure of a Japanese sentence. Section 2.2 describes the various kinds of constraints on MRs.

2.1. Grammatical Structure of a Japanese Sentence

The syntactic structure of a Japanese sentence is as follows:

```
<sentence>     -->     <phrases>
<phrases>      -->     <phrase>    <endmark>
<phrases>      -->     <phrase>  <phrases>
<phrase>       -->     <jiritsugo>  <fuzokubu>
<jiritsugo>    -->     <NOUN> | <VERB> | <ADJECTIVE> | <ADVERB> | ...
<fuzokubu>     -->     <fuzokugo> | <fuzokugo>  <fuzokubu>
<fuzokugo>     -->     <nil> | <POSTPOSTION> | <AUXILIARY-VERB> | ...
```

The following are examples of *PHRASE*s:

kinou:	yesterday
Taro ga:	A boy "Taro" with syntactic agent case marker "ga"
houkoku_suru:	A verb "to report"

The grammatical structure of a Japanese sentence is represented as a dependency structure which is a set of MRs between two *PHRASE*s. Fig.2.1 is an example of a dependency structure for 'Ken ga utsukusii Naomi wo aisu (Ken loves beautiful Naomi).' In this example, both the nominative *PHRASE* ('Ken ga') and the accusative *PHRASE* ('Naomi wo') modify the predicative *PHRASE*('aisu'). And adjective *PHRASE* modifies the substantive *PHRASE* ('Naomi wo'). The arrows represents those MRs.

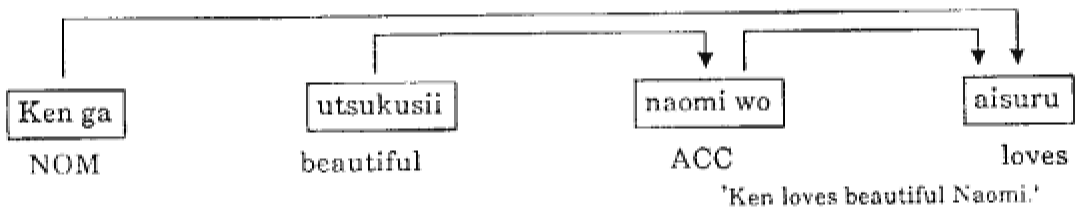


Fig. 2.1.

2.2 Constraints on MRs

There are many levels of constraints on MRs.

(1) Universal Constraints on MRs

UC1) Regardless of its grammatical nature, any *PHRASE* except the last one should modify at least one*PHRASE* which appears in the remainder.

UC2) The last *PHRASE* modifies no *PHRASE*s.
UC3) No two MRs should cross each other.

(2) Syntactic Constraints on MRs

SynC1) A *PHRASE* whose last word is postposition "ga" modifies a predicative*PHRASE*. For example, in Fig 2.1, 'Ken ga' cannot modify 'Naomi wo'.

SynC2) A *PHRASE* consists of an adjective that modifies a substantive *PHRASE*. For example, in Fig.2.1, 'utsukusii' cannot modify 'aisu'.

(3) Semantic Constraints on MRs

This type of constraintscorresponds to a consistency checking of semantic features in ordinary natural language processing systems. The following is an example of a semantic constraint.

SemC1) A non-animate nominative *PHRASE* cannot modify an emotional predicative *PHRASE*. Fig. 2.2 shows this example.

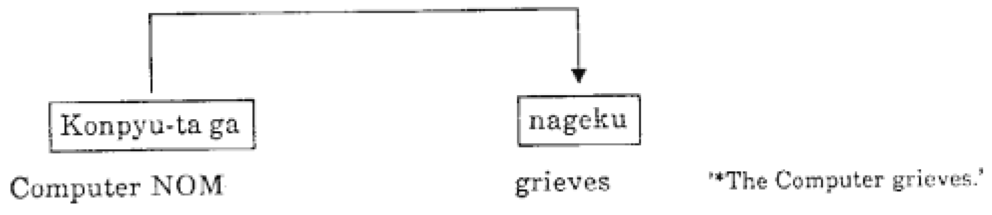


Fig. 2.2 Semantically Illegal Sentence

The process of the syntax and semantic analysis of a Japanese sentence is to compute the dependency structure of an input sentence applying the constraints on MRs.

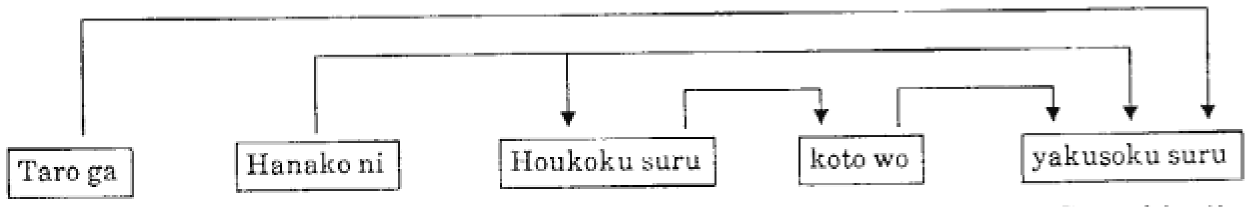
3. Ambiguities of MRs

As a result of syntax and semantic analysis applying the constraints, ambiguities of the MRs usually remain, i.e. there are several possible dependency structures (or interpretations). Besides, the scramblings and ellipses make the treatment of ambiguities more difficult.

For example, 'Taro ga Hanako ni houkoku_suru koto wo yakusoku_suru.' ('Taro promises Hanako to report something.',) can be segmented into the following five *PHRASE*s.

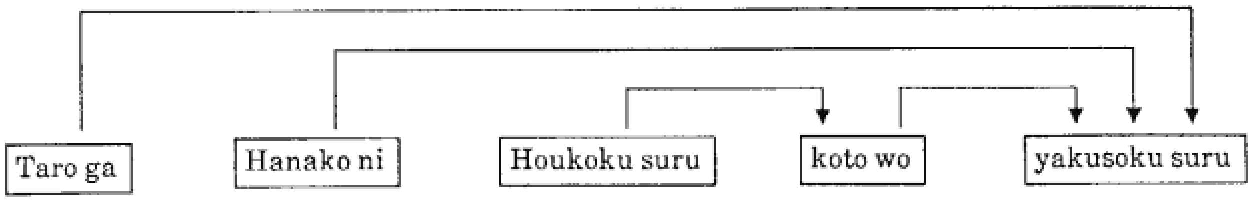
(3.1)	Taro ga	Hanako ni	Houkoku suru	koto wo	Yakusoku suru
	Taro NOM	Hanako Dat	to report	event ACC	to promise

(3.1) is an ambiguous sentence. Even after applying syntactic and semantic constraints, the following five dependency structures still remain.

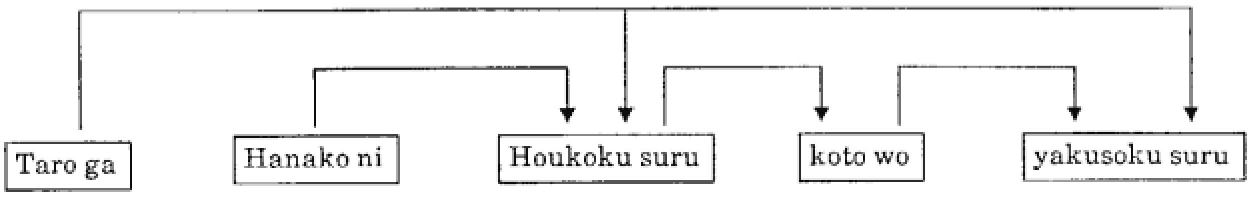


Promise(taro,hanako,report(Agent,hanako,Something))

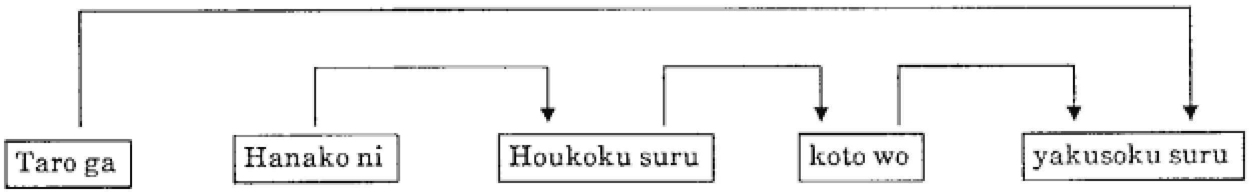
In ordinary natural language processing systems such as machine translation systems, the disambiguation of the MRs is performed applying many heuristics within one sentence analysis to determine only one interpretation. This often generates wrong interpretations. Disambiguations should be done using 'contextual' information in contextual analysis, and syntax and semantic analysis modules should pass the feasible ambiguity structure to the contextual analysis. Section 6 describes this non-heuristic method.



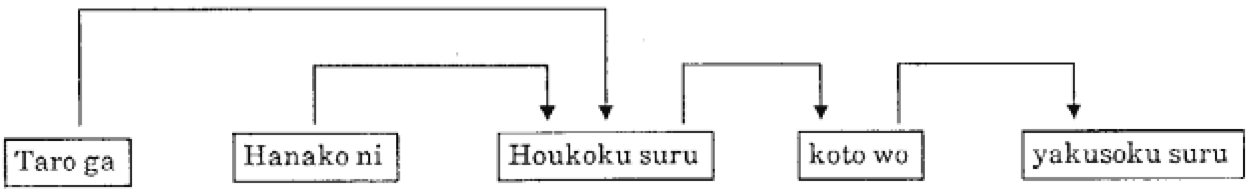
Promise(Taro,Hanako,report(Agent1,Agent2,Something))



Promise(Taro,Agent1,report(Taro,Hanako,Something))



Promise(Taro,Agent1,report(Agent2,Hanako,Something))



Promise(Agent1,Agent2,report(Taro,Hanako,Something))

4. CIL

Before going into the representation of mechanism for analyzing the MRs, CIL which is used to realize the lazy evaluation programming is described. Syntactic and semantic notations are also described using CIL. Accurate accounts of CIL can be found in [Mukai 85]. Here,only the part needed to understand the lazy evaluation programming and syntactic and semantic notation is described.

CIL (Complex Indeterminate Language) can be represented by the following formula.

$$\begin{aligned} \text{CIL} &= \text{Prolog} \\ &+ \text{Partially Specified Term} \\ &+ \text{Freeze} \\ &= \text{Prolog} + \text{Frame} + \text{Freeze}. \end{aligned}$$

CIL has a unique data structure called "partially specified term (PST)." A theoretical study on completeness and soundness of PST has already been completed [Mukai 86], but no details are given here. This data structure can be regarded as a frame and represented as in the following example.

```
(1) term({a/X?, b/Y?)) :- X = Y.
(2) > term(Z),a!Z = abc,b!Z = abc.
  yes.
```

Formula (1) is the specification of the data type and can be used in formula (2). "a/X" means assign value X to slot "a". "?" denotes the freeze mechanism attached to logical variable X. So, "X = Y" in (1) will not be evaluated until X and Y are bound to some values. If this condition is not satisfied, the unification "X = Y" in

(2) will fail when X and Y get their values. Formula (2) utilizes the complex indeterminate and unifies "abc" to each slot value.

5. Semantic Framework Based on the Situation Theory

Before going into explanation of examples, semantic framework is defined based on the Situation Theory [Barwise 86]. The framework for semantics needs to be studied more deeply, so this framework is probably not the last version. This section shows just enough frameworks to understand the frame- work in section 6. [Mukai 87] gives more details of this framework. Section 6 describes the relationship between this framework and syntactic framework. The definition of the framework is given below. For notaitonal convention, single ":" is used in a line to represent repetition. Signs with brackets < > are terminology in the Situation Theory.

```
5.1) <soa> ::= <state of affairs>
5.2) <state of affairs> ::=
            {sort/soa,
             rel/<relation>,
             pol/<polarity>,
             <argument place name>/<object>,
                          :
             <argument place name>/<object>
            }
5.3) <relation> ::=
            {sort/relation,
             name/<ground term as name>,
             args/{<argument place name>/<property>,
                              :
                   <argument place name>/<property>}}
5.4) <property> ::=
            {sort/property,
             rel/<relational expression>}.
5.5) <relational expression> ::=  <ground term>
            | <term> ":" <predicate as condition>
5.6) <polarity> ::= 0 | 1 | UNBOUND.
5.7) <object> ::= <soa> | <parameter> | <conditioned parameters>
5.8) <parameter> ::=
            {sort/parm,
             pname/<string>,
             anc/<object>,
             property/<property>}
5.9) <conditioned parameters> ::=
            {sort/cp,
             anc/<anchor>,
             pl/[<parameter>,            NB) pl means parameter list
                       :
                 <parameter>],
             cond/<soa>}
5.10) <anchor> ::= [ <parameter> '=' <object>,
                                   :
                     <parameter> '=' <object>]
```

5.1) indicates that <soa> means a state of affairs in Situation Theory.

5.2) shows the structure of <soa>. <soa> is constructed from one indicator and four kinds of constituents. "sort/soa" is a indicator, and indicates the type of this object. "rel/<relation>" represents the relation of this <soa>. "pol/<polarity>" indicates polarity of this <soa>. "<argument place name>/<object>" indicates arguments of <soa>. The number of arguments is ideally 0 to infinity.

5.3) represents the structure of <relation>. <relation> is constructed from one indicator and two kinds of slots. "sort/relation" indicates the type of this object. "name/<ground term as name>" represents the name of

<relation>. In the Situation Theory, the name cannot be structurized. "args/.." represents restriction of arguments in <soa>. "<argument place name>" corresponds to "<argument place name>" in <soa>. "<property>" represents the restriction for the <object> at <argument place name> in <soa>.

5.4) indicates <property> as a restriction for <object> at <argument place name> in <soa>. <property> has its indicator <sort/property>, and <rel/..> as a restriction.

5.5) represents the restriction. Restriction can be coded as the general term <term> or a term with condition <term>:<predicate as condition>. This condition can be lazy evaluated.

5.6) indicates polarity.Situation Theory does not use *UNBOUND*. In this framework, *UNBOUND* is used to represent some contextual constraint in lazy evaluation mechanism.

5.7) indicates <object> which should be placed on the argument place of <soa>.

5.8) represents the structure of <parameter> which can be placed on the argument place of <soa>. <parameter> is one kind of minimum data in semantic analysis. <parameter> has one indicator, "sort/parm", and three kind of constituents. "pname/<string>" is the name of a parameter. <parameter> has anchor as "anc/<object>". There should be some kind of ambiguity when anchoring <parameter> to some <object> in context analysis especially in discourse analysis.

5.9) represents <conditioned parameters>. <conditioned parameters> has the indicator, "sort/parm", and three kinds of constituents. <conditioned parameters> gets some <parameter>s in <soa> together. Then, "anc/<anchor>" indicates the correspondence between <parameter> and <object>. "pl/[..]" indicates <parameter>s in <soa> which are indicated by "cond/<soa>".

6. Modifying Relations (MRs) in Japanese Sentences

This section presents MRs in Japanese sentences. First, the basic framework which realizes constraints on MRs using the lazy evaluation mechanism are represented.

To realize grammatical constraints on MRs, a table (MRT) in which MRs are represented as matrix elements is used. This table resembles the well-formed table which is utilized in Earley's algorithm [Earley 70]. Fig.6.1 is an example of this table.

1	U	U	1	U
	2	U	U	0
		3	1	0
			4	1
				5

-Fig 6.1-

In Fig 6.1, numbers from 1 to 5 indicate the number of the line and column. "U" indicates *UNBOUND* of MRs. "1" indicates a connected link and "0" indicates that there is no chance to connect the link. For example, "1" on MRT(1,4) (line 1, column 4) indicates that the *PHRASE* at position 1 connected to the *PHRASE* at position 4. MRT(4,5) always becomes "1" because of constraint UC2) in 2.2.

In the first stage of analysis, the initial value of every element in MRT is *UNBOUND* and attached to grammatical constraints using the lazy evaluation mechanism. This section describes only the execution mechanism using lazy evaluation mechanism for UC1) UC2) UC3). Appendix gives details of the programming method and the whole program used to attatch constraints.

When constraints on MRs are set, MRT(4,5) becomes "1" automatically as shown in Fig 6.2.

If MRT(1,4) becomes 1 because of UC3), then MRT(2,5) and MRT(3,5) become 0 as shown in Fig 6.3. Then, because of UC2), *PHRASE* 3 should modify at least one phrase and there remains only *PHRASE* 4 to modify, so MRT(3,4) becomes "1" as shown in Fig 6.1.

UC3) and UC2) are related to each other, and these relationships are programmed using MRT. In the analysis of one sentence, "1" or "0" is set to "U" and the ambiguity of MRs is reduced.

6.1. Analysis of Right-branching Phrase Structure

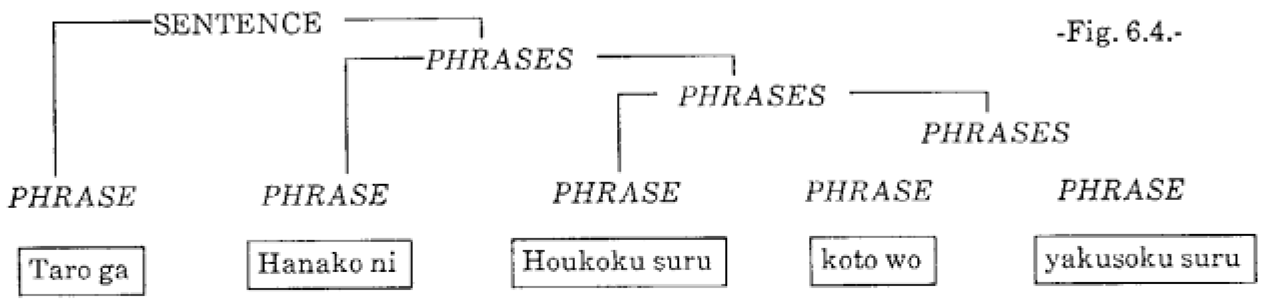
Section 2.1.has already represented the basic phrase structure rules. Using such a rule, right-branching phrase structure are constructed as shown in Fig.6.4..

1	U	U	U	U
	2	U	U	U
		3	U	U
			4	1
				5

-Fig.6.2-

1	U	U	1	U
	2	U	U	0
		3	U	0
			4	1
				5

-Fig. 6.3-



-Fig. 6.4.-

This analysis does not cover MRs. MRs are set in syntactic and semantic analysis.

6.2. Syntactic and Semantic Analysis

In the first stage of syntactic and semantic analysis, MRT is made corresponding to right-branching phrase structure. Grammatical constraints UC1), UC2), and UC3) are set up in MRT using the lazy evaluation mechanism.

For the syntactic rules, a simple mechanism is set to check whether one *PHRASE* can modify another *PHRASE*.

The ordinary main part of the *PHRASE* <jiritsugo> which was represented in 2.1., is classified into two kinds of features such as indeclinable parts of speech (IDPS) and declinable parts of speech (DPS). For example, noun is classified as indeclinable part of speech, and verb, adjective, and adverb are classified as declinable parts of speech.

<fuzokugo> in the *PHRASE* can be classified as a modifying feature of indeclinable parts of speech or a modifying feature of declinable parts of speech.

Using these features, MRs can be easily restricted as shown in the followig example.

SENTENCE:	Taro ga	Hanako ni	Houkoku suru	koto wo	Yakusoku suru
PHRASE No.:	1	2	3	4	5

MRT:

1	U	U	1	U
	2	U	U	0
		3	1	0
			4	1
				5

PHRASE	<jiritsugo>	<fuzokugo>
Taro ga	IDPS	DPS
Hanako ni	IDPS	DPS
houkoku_suru	DPS	IDPS
koto wo	IDPS	DPS
yakusoku_suru	DPS	none

- Fig. 6.5 -

Using the syntactic modifying feature of <fuzokubu> in *PHRASE* and the feature of <jiritsugo>, "0" is set to MRT. If the *PHRASE* can modify more than two *PHRASE*s , the element of MRT remains as *UNBOUND*.

Together with the syntactic analysis shown in Fig.6.5, semantic analysis is carried out. If there is still ambiguity after applying the syntactic rules of MRs, the relationship between the semantic decriptions of *PHRASE*s is checked. This check is based on SemC1) in section 2.2.(3). For example, in Fig.6.5, the semantic checking mechanism is used to reduce ambiguity in MRT(2,5) like following as shown in the following example Fig.6.6.

```
PHRASE           |      semantics
-------------------------------------------------------------------------------------
hanako ni        |      {sort/parm,
                 |      pamc/hanako,
                 |      anc/UNBOUND,
                 |      property/{     sort/property,
                 |                     rel/emotional!human!girl}
                 |      }
                 |
yakusoku_suru    |      {sort/soa,
                 |      rel/{sort/relation,
                 |      name/yakusoku_suru, (promise)
                 |      args/{         agent/A#{      sort/property,  (POINT A)
                 |                                    rel/emotional!human}:X!property = A
                 |                     object/B#{sort/property,
                 |                                    rel/soa}:Y!property = B (POINT B)
                 |                     partner/C#{sort/property,   (POINT C)
                 |                                    rel/emotional!human}:Z!property = C
                 |             }
                 |      agent/X,object/Y,partner/Z}
```

- Fig. 6.6. -

To check semantics, the properties of each *PHRASE* in (POINT A), (POINT B), and (POINT C) are unified. In "rel/" part of property, is_a relations are represented using a kind of inheritance mechanism. In this example, ambiguities in MRT cannot be reduced, so the result of analysis is passed to the contextual analysis.

6.3. Contextual Analysis on MRs

Before describing contextual analysis, the terminology for ambiguity and vagueness must be rearranged. Japanese sentence has ellipsis in which *PHRASE* indicating agent or object can be omitted For example, the following sentence is a complete Japanese sentence.

Hanako_ni yakusoku_sita. (Someone promised Hanako something)

In this case, the agent and object are vague. Vague menas that something is not described. In MRs, there are many alternative solutions and there is no need to think of any *PHRASE* which was not described. In this case, MRs are "ambiguous".

So, even if all ambiguities in MRT are solved, some vagueness may remain in semantics. For example, there are two ambiguities in the following MRT.

SENTENCE:	Taro ga	Hanako ni	Houkoku suru	koto wo	Yakusoku suru
PHRASE No.:	1	2	3	4	5

1	0	1	0	0
	2	1	0	0
		3	1	0
			4	1
				5

MRT

Semantic Interpretation: (details are omitted}

{sort/soa,
rel/yakusoku__suru (promise),
agent/X,partner/Z
object/{sort/soa, rel/houkoku__suru (report)
agent/taro, object/W, partner/V,loc/*UNBOUND*}

- Fig.6.7. -

No ambiguity on MRs remained in Fig. 6.7, but there was still vagueness about the agent and partner of "promise", and the object and partner of "report".

(1) Reduction of Ambiguity on MRs

Sentences have some structure [Grotz 86]. To disambiguate MRs, this paper proposes to utilize a structured world with inference rules. If there are still ambiguities, after a sentence analysis, they should represented as shown in the following example. This example corresponds to the ambiguity of MRs at MRT(1,3) in Fig. 6.5.

{sort/soa,pol/*UNBOUND*,
rel/houkoku__suru,(report)
agent/taro,
partner/*UNBOUND*,
object/*UNBOUND*}

- Fig. 6.8 -

A tipical feature of this interpretation is that polarity remains as *UNBOUND*.

Whether they are ambiguous or not, all the interpretations should be collected together in a structurized world. If one phrase with some sentences has some coherent arguments, all interpretations should be in one world.

For example, if the following sentence exists after analyzing the sentence in Fig. 6.5, the ambiguity in Fig. 6.8. can be easily disambiguated.

Yokujitsu Taro ha houkoku__shita.
(Next day Taro reported something to someone)
Interpretation:
{sort/soa,pol/1,
rel/houkoku__suru, (reported)

agent/taro,
partner/*UNBOUND*,
object/*UNBOUND*}

This sentence has no ambiguity on MRs, but the partner and object are vague. Unifying this interpretation and interpretation on Fig.6.8., the ambiguity on MRs at MRT(1,3) in Fig.6.5. can be reduced. In this case, since MRT(1,3) becomes 1, MRT(2,3) becomes 1 and MRT(2,5) becomes 0 automatically. Then, that partner is Hanako, as shown below.

{sort/soa,pol/1,
rel/houkoku__suru, (reported)
agent/taro,
partner/hanako,
object/*UNBOUND*}

As stated above, some kind of ambiguities can be solved easily by collecting interpretations together. However, there are still some difficult problems.

The following sentence is an example.

Shikashi Taro ha houkoku__sinakatta.
(But Taro did not report something to someone)
Interpretation:
{sort/soa,pol/0,
rel/houkoku__suru, (reported)
agent/taro,
partner/*UNBOUND*,
object/*UNBOUND*}

In this case, this interpretation cannot be unified with the interpretation in Fig. 6.8. There should be some rules to maintain coherency of interpretation.

Moreover, there is another kind of problem on anchoring [Barwise and Perry 81]. For example, when someone says "Taro", there should be many kind of persons to be anchored to "Taro". To disambiguate this, some focusing mechanisms or analysis of topic will be effective [Sidner 83][Hajicova 86].

Anaphora is also a kind of ambiguities. Some heuristics [Kameyama 84] can be used to anchor object to pronouns. We are now researching some algorithms to solve this problem definitely but this research will appear in the future.

(2) Vagueness

A tipical type of vagueness is ellipsis. Some kinds of vagueness can be solved in disambiguation of MRs, such as that shown in the previous example on ambiguity. In a text analysis, some kind of vagueness should be solved by maintaining coherency of text structure. However, some phenomena, like such as zero anaphora is still difficult to solve.

7. Conclusion

As mentioned above, the approach to show ambiguity explicitly by lazy evaluation mechanism opens the door to contextual analysis. Basically, this approach is backtrack free. However, some study about phenomena such as garden path sentences, should be done.

From the viewpoint of computer linguistics, backtracking is very expensive. So, the approach presented in this paper will achieve high performance in execution. Human beings do not backtrack like computers because there are very few people who can think without any materials such as a black- board. However, this should be studied in the field of cognitive science in the future.

We are now researching rules to maintain coherency of discourse. Some other contextual aspects like honorifics [Sugimura 86] should be taken into consideration when we design the whole model of discourse. We believe that the Situation Theory will be a powerful tool in building the model. This should be studied throughly in future research.

[Acknowledgements]

We would like to thank Mr.Mukai of ICOT for his comments on lazy evaluation mechanisms, and Dr.Yokoi, Deputy Director of Research Laboratory and Dr.Uchida, the Director of the 2nd Research Laboratory at ICOT, for their encouragement and for allowing us ample time for this work.

[References]

[Barwise 86] Barwise, J.,: Recent Developments in Situation Semantics, in International Symposium on Language and Artificial Intelligence:Contributed Papers 21, International Institute for Advanced Studies,1986

[Barwise,Perry 83] Barwise,J. and Perry,J.,: Situations and Attitudes,MIT Press,1983

[Dincbas 86] Dincbas,M.: Constraints,Logic Programming and Deductive Databases, Proceedings of France-Japan Artificial Intelligence and Computer Science Symposium 86,ICOT, 1986.

[Earley 70] Earley,J., : An Efficient Context-Free Parsing Algorithm, Comm.ACM,Vol.13, No.2, 1970,pp.94-102

[Grotz 86] Grotz,B.J.: The Structures of Discourse Structure,in International Symposium on Language and Artificial Intelligence: Contributed Papers 2 International Institute for Advanced Studies, 1986

[Gunji 87] Gunji,T.,:Japanese Phrase Structure Grammar,D.Reidel Publishing Company,(to appear)

[Hajicova 86] Hajicova,E.: Linguistic Meaning as Related to Syntax and to Semantic Interpretation, in International Symposium on Language andArtificial Intelligence,Contributed Papers 18,1986

[Hashimoto 80] Hashimoto,S: Kokubunpou Taikeiron (Structure of Japanese),Hasimoto Shinkichi Hakase Chosakushuu,No.7.,Iwanami,.1980,(in Japanese)

[Kuno 73] Kuno,S.,: "The Structure of the Japanese Language", MIT Press,1973.

[Kameyama 84] Kameyama,M.: Centering in Japanese; Basic Observation, Draft of Ph.D. thesis, Dept. of Linguistics, Stanford University, 1984

[Mizutani 83] Mizutani,S.,: Kokubunpou so-byou (Sketch of Japanese Grammar),Bunpo to imi (Grammar and Meaning),Asakura,1983,pp.158-178

[Mukai 85] Mukai,K.,: Unification over Complex Indeterminates in Prolog,ICOT Technical Report No.TR113,1985

[Mukai 86] Mukai,K.,: Anadic Tuples in Prolog, in J. Minker (ed),the Preprint of the Workshp of the Workshop on Foundation of Deductive Database and Logic Programming, Washington DC., 1986.

[Mukai 87] Mukai,K.,: A System of Logic Programming for Linguistic Analaysis based on Situation Semantics, in the Proceedings of Workshop on Semantic Issues in Human and Computer Languages, Half Moon Bay, 1987.

[Nakamura 86] Nakamura,J.,: Solutions for Problems of MT Parser. Methods used in Mu-Machine Translation Project,Coling 86,1986,pp133-135

[Nitta 86] Nitta,Y.,: Idiosyncratic Gap:A Tough Problem to Structure-bound Machine Translation,Coling86',1986,pp107-111

[Ozeki 86] Ozeki,K.,: A Multi-Stage Decision Algorithm for Optimum Bunsetsu Sequence Selection from Bunsetsu Lattice,COMP,86-47,1986 (in Japanese)

[Sidner 83] Sidner, C.L.,: Focusing in the Comprehension of Definite Anaphora, Brandy, M. and Barwick,C.(ed), Computational Models of Discourse, MIT Press, 1983,pp.107-164

[Sugimura 86] Sugimura, R. , : Japanese Honorifics and Situation Semantics,Coling 86, 1986, pp.507-510

[Tokieda 41] Tokieda S.,: Kokugo_gaku Genron (The Principles of Japanese),Iwanami,1941

[Tsujii 84] Tsujii,J., Nakamura,J., and Nagao,M.,: Analysis Grammar of Japanese in the Mu-Project---A Procedural Approachto Analysis Grammar,Coling 84,1984,pp.267-274

[Watanabe 81] Watanabe,Minoru: Nihon_go Koubunron (Theory of Japanese phrase structure,Hakama shobou,1981

[Appendix]
Program for UC1,UC2,UC3 in CIL

```
cons_kakari(N,W) :- consk(1,N,W).
consk(N,N,W).
consk(S,N,W) :- Temp is S + 1,jn(S,N,W),consk1(S,Temp,N,W),!,consk(Temp,N,W).
consk1(K,U,N,W) :-U > N.
consk1(K,U,N,W) :-k(N,K,U,W),Next is U + 1,!,consk1(K,Next,N,W).
jn(S,N,W)  :-afters(S,N,A_List),!,kakeyo(S,A_List,A_List,Flag,W).
kakeyo(_,[],_,_,_) :- !.
kakeyo(S,[A],[A],_,W) :- W!S!A = 1.
kakeyo(S,[A|R],A_List,Flag,W) :-
   freeze(W!S!A,(pv(Flag,((W!S!A = 0,
            get_rest(A,A_List,A_Rest),
            kakeyo(S,A_Rest,A_Rest,FN,W));  true)))),!,kakeyo(S,R,A_List,Flag,W).
get_rest(A,[A],[]) :-!.
get_rest(A,[A|R],R) :- !.
get_rest(A,[B|R],[B|RR]) :- !,get_rest(A,R,RR).
k(N,K,U,W) :- Temp is K + 1,
   (Temp == U    ;
      inter(K,U,InLis),  befores(K,BefLis),afters(U,N,AftLis),
      mkcons(W!K!U,BefLis,InLis,W),mkcons(W!K!U,InLis,AftLis,W)).
inter(K,U,InLis)  :-mk_lis(K,U,InLis).
befores(K,BefLis)  :-mk_lis(0,K,BefLis).
afters(U,U,[]) :- !.
afters(U,N,AftLis) :-T is N + 1,mk_lis(U,T,AftLis).
mk_num_list(U,N,Nlis) :- !,mk_numl(1,U,N,Nlis).
mk_numl(N,N,N,[]) :- !.
mk_numl(N,U,N,[N]) :- !.
mk_numl(U,U,N,Nlis) :-Pn is U + 1,!,mk_numl(Pn,U,N,Nlis).
mk_numl(S,U,N,[S|R]) :-Sn is S + 1,!,mk_numl(Sn,U,N,R).
mk_lis(K,U,[]) :-T is K + 1,U == T.
mk_lis(K,U,[KP|R]) :-KP is K + 1,!,mk_lis(KP,U,R).
mkcons(_,[],_,_).
mkcons(_,_,[],_).
mkcons(F,InLis,OutLis,W) :-
   freeze(F,(F = 1,set_zero(InLis,OutLis,W);true)).
set_zero([],_,_).
set_zero([A|R],O,W)  :-set_zero1(A,O,W),!,set_zero(R,O,W).
set_zero1(_,[],_).
set_zero1(A,[B|R],W) :-W!A!B = 0,!,set_zero1(A,R,W).
```