ICOT Technical Memorandum: TM-0103

TM-0103

Some Aspects of Generalized Phrase Structure Grammar

ICOT Working Group 3.4

March, 1985

ICOT

Mita Kokusai Bldg. 21F
4-28 Mita 1-Chome
Minato-ku Tokyo 108 Japan

(03) 456-3191~5
Telex ICOT J32964

Institute for New Generation Computer Technology

Some Aspects of Generalized Phrase Structure Grammar

by ICOT Working Group 3.4

Introduction

We at Working Group 3.4 at ICOT started in April 1984 a project on implementing some version of Japanese parser based on the framework of Generalized Phrase Structure Grammar (GPSG). This particular grammatical framework for natural language is chosen because of its apparent feasibility for computational implementation; this framework is both explicit and formal enough to allow for computational interpretation, even though the grammar itself may not be necessarily constructed for a particular application such as computer implementation in mind. For example, most (or perhaps all) of the syntactic rules can be described within the constraint of context-free grammar, which allows efficient parsing by the well-known algorithm. Also, context-free grammar is the basis for the well-known parser in Prolog, i.e., Definite Clause Grammar (DCG). This is another reason for choosing this grammar for the next-generation computer, in which Prolog is assumed to be playing a central role.

Another attractive character of GPSG is that there is already some preliminary results in analyzing Japanese in this framework. Even if such works turn out to be not adequate for the entire system of a Japanese analyzer for some reason, we can start experimenting a system with at least something at hand.

For the first year, we made an initial survey of existing literature on GPSG systems. At the same time, some preliminary research on GPSG, Japanese grammar, parsing, implementation language, and several related topics were conducted by the members of the Working Group. Our initial impression is that most of GPSG's metagrammatical mechanisms, e.g., the Head Feature Convention (HFC), the Control Agreement Principle (CAP), the Foot Feature Principle (FFP), could fit Prolog's unification mechanism and would not cause any technical difficulty in implementation. A potential difficulty would rather be the treatment of metarules: the choice between expanding the ID rules beforehand or dynamically applying the metarules during the time of parsing. Currently, we are not in the position to give a definitive answer to these alternatives, which will be one of the most important topics to investigate in the second year.

This Technical Memo summarizes our effort during the first year of the project, which is expected to continue for several more years. It contains brief notes on the following materials:

*Overall design of the prospective parser (Hirakawa)

*How a parser works (Amano)

*Short-term memory and parsing (Hashida)

*Feature system and unification in Prolog (Mukai)

*Overall grammar for the Japanese language (Gunji)

*Analysis of the Japanese verb phrase system (Udo)

Introduction

*Comparison of GPSG with transformational grammar (Shirai)

*Bibliography of relevant materials

Toward an Implementation of the GPSG System

Hideki Hirakawa
ICOT Research Center

One of the most important characteristics of the GPSG theory is that it is a formal linguistic theory based on Context Free Grammar. Since various principles and conventions adopted in GPSG have concrete formal definitions, we can be sure that a grammar written in GPSG formalism will be handled by computers correctly and efficiently. Implementation of the GPSG system will provide the basis for natural language processing systems as well as the tools for verifying language theories. This section describes some aspects of an implementation of GPSG.

1. Framework of GPSG

Current GPSG theory uses the following tools for describing grammar rules (which corresponds to explanations of linguistic phenomena).

(a) Feature system
    This includes X bar theory, the Feature Coocurrence Restriction (FCR), Feature Specification Default (FSD), Lexical Subcategorization, Head Feature Convention (HFC), Foot Feature Principle (FFP), Control Agreement Principle and Notational Convention (NC).
(b) ID/LP rule
(c) Metarule
(d) Montague's semantic theory

A GPSG system on computer should be able to treat the grammar written in this formalism and parse (or generate) sentences. The above framework constitutes the specification of the GPSG system.

2. Design Principles

Given the specification of the system, the next step is to decide the algorithms, data structures, implementation language and so on. These implementation issues are discussed in this section. The designs require an actual image of system usage, for example, the size of a grammar is crucial for designing the parser. In fact, the existing GPSG systems differs from each other according to their purposes. Basically the following design principles are assumed in our system:

(a) An experimental and neutral system
(b) A clear and modifiable system

The first principle implies that the system has no application-oriented and language-oriented features. This is because one of the

purposes of the development of a GPSG system is to discover the best implementation method of the GPSG system itself. It is assumed that the system will be used for some linguistic experiments and experimental test of some applications. The second principle is very important because it enables the system to follow the real and/or experimental revision of the GPSG theory.

It follows from the above design principles that the programming language for the GPSG system should be a logic programming language. This is reasonable because a logic programming language not only has high descriptive power but also has an affinity with the GPSG system itself.

3. Discussions on implementation methods

This subsection describes some notes on the implementation of the GPSG system in the logic programming language Prolog (description of Prolog is omitted here). This discussion focuses only on parsing sentences. Sentence generation will be considered elsewhere.

3.1 Feature system

Categories of GPSG can be implemented naturally using Prolog's terms. The feature system can be handled by Prolog's unification-based processing. For example, FCR, FSD, NC, and HFC can be manipulated by implementing a preprocessor of the grammar [Evans 84]. Of course, these mechanisms can be handled dynamically during parsing in stead of at preprocess time. However, it is better to deal with them at preprocess time because they can be expanded rule by rule and the preprocess method offers greater parsing efficiency. On the other hand, FFP and CAP have different properties. For example, the expansion of the 'slash' category (FFP) will cause an enlargement of the grammar because new slash-attached categories are added to it. This impairs parsing efficiency. Shirai's system [Shirai 83] handles FFP (slash category) and CAP during parsing. In the Prolog-based parsing system, slash can be dealt with the stack method as shown in XG [Pereira 83]. As for CAP, it may be better to expand agreement features at preprocess time if it is possible. In this case the agreement check is embedded in Prolog's unification.

3.2 ID/LP and Metarules

The ProGram system [Evans 84] expands the ID/LP rules to usual (extended) CFG rules during preprocessing, while Sieber [Sieber 83] and Kilbury [Kilbury 84a] propose direct parsing methods for ID/LP rules. As Kilbury claims, expansion of the ID/LP grammar enlarges the grammar and results in ineffective parsing when a backtrack parsing method is applied. One reason for this inefficiency is the duplication of computation which can be avoided by using any tabular parsing method [Aho 72]. Generally, preprocess (or compile time) processing is more efficient than parsing (or execution) time processing. Therefore, the

following two methods should be compared.

(a) Direct Parsing Method
(b) Preprocessor and Tabular Parsing Method

Metarule manipulation involves the theoretical problem that unrestricted application of metarules produces an infinite number of CFG rules. Without restricting the metarule itself or its application, the system cannot deal with it correctly. The implementation of metarule processing will require the same choices as that for ID/LP rules.

3.3 Semantic Part

Although the GPSG theory adopts Montague's semantic theory as its semantic part, several GPSG systems have different semantic frameworks such as First Order Logic [Gawron 82] and Semantic Representation [Kilbury 84b]. Since the semantic part seems to be more application-dependent than the syntactic part of the GPSG system, it is desirable that the system has high modularity between them. One problem for this approach is that the definition of the 'control' relation depends on the semantic part of a rule. This should be treated correctly when using a semantic framework other than Montague's.

3.4 Parser

Selecting the parsing method is one of the most important issues in designing the GPSG system. The design is influenced by discussions on how to deal with several aspects of GPSG mentioned above. From the viewpoint of the implementation language Prolog, XG and BUP [Matsumoto 83] are preferable because they can compile grammar rules to machine codes. However, direct treatment of ID/LP rules requires the interpretive application of rules. This tradeoff has to be considered when deciding the parsing method.

4. Summary

Some issues involved in an implementation of a GPSG system were described. Even though the details of the system have not been decided, it is clear that a Prolog-based system is preferable because of the language's high descriptive power and its affinity with GPSG. To design an efficient GPSG system it is important to clarify the advantages and disadvantages of the two approaches, i.e., the direct parsing method and the indirect parsing method.

References

[Aho 72] Aho, A.V. and Ullman, J.D.: "The Theory of Parsing, Translation, and Compiling, vol.1 Parsing", Prentice-Hall pub.,

1972.

[Evans 84] Roger Evans and Gerald Gazdar: "The ProGram Manual", University of Sussex, 1984.

[Gowron 82] Jean M. Gowron et. al.: "The GPSG linguistic system", Proc. of the 20th Annual Meeting of The ACL, 1982.

[Kilbury 84a] James Kilbury: "Earley-basierte Algorithmen fuer Direktes Parsen mit ID/LP-Grammatiken", KIT-report 16, Fachbereich Informatik, Technische Universitaet Berlin, 1984.

[Kilbury 84b] James Kilbury: "GPSG-based Parsing and Generation", KIT-report 17, Fachbereich Informatik, Technische universitat Berlin, 1984.

[Matsumoto 83] Yuji Matsumoto, et. al.: "BUP: A Bottom-Up Parser Embedded in Prolog", New Generation Computing, vol.2, OHM-Springer, 1983.

[Pereira 83] Fernando Pereira: "Logic for Natural Language Analysis", Technical Note 275, SRI International, January, 1983.

[Sieber 83] Stuart M. Sieber: "Direct Parsing of ID/LP Grammars", SRI Technical Note 291, 1983.

[Shirai 83] Hidetoshi Shirai: "Deterministic Parser", Proc. of the Workshop on Non-Transformational Grammars, Tokyo: ICOT, 1983

Some Difficulties in the Implementation of a Life-Size Parser

S. Amano

1. Lack of computing power in today's machines

The fact is well known that super computers are urgently desired in the fields of magnetohydrodynamics, meteorology, nuclear physics, and so on. There are no such systematic urgent requests for a super computer in the field of natural language processing as far as the author knows.

The reasons are as follow;

1) There are few large-scale and sophisticated parsing systems.

There are many parsing systems used throughout the world. Most of them are only experimental systems, and have no large-scale grammars and dictionaries. This means they have no time-critical problems.

On the other hand, there are a few large-scale systems which are not sophisticated, especially when used as a machine translation system. Even morphosyntactic error-checks are omitted in some of them. Orthodox computational linguists could hardly imagine such an awkward situation. They are fast, but at the cost of precision, are not concerned with time.

2) Fast parsing-algorithms are themselves a goal.

Computational linguists regard finding fast algorithms for parsing as a goal. They tend to avoid depending on the brute computing power of the machine. Often, they would be embarrassed to talk about the speed of their parser, when it is not fast enough. They might think their parsing algorithm is not sophisticated, even if it is linguistically powerful.

3) There is no typical parallelism for a parser.

Super computers can compute vectors. Grammars for natural languages would not be so well-formed as vectors. Since the design of a parser is an open-ended problem, the designer can hardly decide on a special hardware-dependent formalism which has sufficient linguistic power.

As an actual problem, parsers need a high-speed computer. Design of a large-scale, life-size parser is necessarily confronted with a problem of computational complexity. Designers try to avoid this problem with heuristic or ad hoc methods.
Let's take the problem of a homograph as the simplest example. Homographs make a great amount of strings of categories which must be checked by a parser to determine whether or not they form a sentence.

For example;

Sentence;	I	think	that	I	will	wear	that	red	dress.
Category;		vt	dpn	aux	vt	dpn	adj	vt	
		vi	dadj	n	vi	dadj	n	vi	
			conj	vt	n	conj		n	
			rprn			rprn			
			dadv			dadv			

Possibility; 2 x 5 x 3 x 3 x 5 x 2 x 3 = 2700

The parser not only parses a string of correct categories, but also checks up strings of wrong categories. If it parses one string of categories in 1 millisecond, it will take 2.7 seconds to parse all possibilities. Generally a powerful parser, because of its linguistic power, takes much time. Provided it takes 1 second for one parsing, it will take 2700 seconds totally. To cope with this homograph problem, linguists limit the domain of available sentences and also limit available categories. For example, in the field of information science, designers may be able to throw out the noun and verb meanings of the word 'will' from the lexicon. This remedy is very ad hoc. They can apply a better method for grammars than for lexicons. They may introduce several heuristic methods into the grammar which will be implemented as condition-checks or control mechanisms.

Computers have to be at least 1000 times as fast as they are, in order that linguists can write down grammars only in response to the need of linguistics without heuristic or ad hoc program control.

2. Lack of human-like parsing algorithm

Computational linguists must think of parsing algorithms which do not devour parsing time. If a computer parses sentences with naturally constructed grammar, which does not include heuristic or ad hoc control mechanisms, it will take much time to try out all possibilities. To lessen this difficulty, introduction of the semantic and pragmatic knowledge is thought of as a powerful device. This kind of knowledge, as used today, will only make interpretation of a sentence more precise. The more precise the parsing becomes, the more time that is needed by semantic and pragmatic analysis (together with inference). Semantic and pragmatic analysis, because of its complexity, will spend all of the time which syntactic analysis can save, or even more, by being freed from exhaustive search for all syntactic possibilities by use of semantics and pragmatics.

This sort of use of semantics and pragmatics is not a good simulation of a human way of parsing. Humans do not take much time to parse a sentence using semantics and pragmatics. "Quasi deterministic parsing strategy" (to backtrack only in case of garden path sentences) seems to be done in the human brain. It is reminescent of Marcus' deterministic parser.

As a surface phenomenon, it could be a cue for human-like parsing algorithm, if its ability were not quite limited and if linguists could write a large-scale grammar with its formalism. Marcus' idea is only oriented to determinism, not oriented to a genuine parser.

Faster and more precise (and maybe quasi deterministic) algorithms, which can be actually implemented on a computer, must be studied.

Language Processing on Short-Term Memory

Kôiti Hasida

Department of Information Science,
Faculty of Science, University of Tôkyô

7-3-1 Hongô Bunkyô-ku, Tôkyô 113, JAPAN

The human capacity for holding some sort of information immediately in the mind is severely limited, and has been attributed to **short-term memory** (**STM** hereafter). Combined with the fact that people cannot process some complicated sentences, this would suggest a hypothesis that transient information used by human language faculty is stored in STM. This hypothesis motivates finite-state models of language processing. One criterion as to the reality of such models would be how accurately they predict the extent to which STM is loaded during language processing. Henceforth we call this extent the **transient memory load** (**TML**). TML constitutes part of the difficulty people feel when they produce or comprehend sentences. Several researchers, discussing along the hypothesis above[1] support or assume that the depth of self-embedding (SE) or of center-embedding (CE) is a crucial factor of TML.

The challenge here is to find a more general and principled measure of TML than those ever discussed.[2] Inaccuracies of the two measures of TML, one defined by SE and the other by CE, are shown up in comparison between (1) and (2):

(1) Tom knows of the story that a man who lived in America and his wife were poor but they were happy.

(2) Tom knows that the story on the fact that the rumor that the man killed John was false is funny.

Both measures fail to explain why most of us feel (1) easier than (2), since, as indicated in (3) and (4), they are not differentiated from each other in terms of SE or CE depth. Note that those sentences are equally deep with respect to SE of S and of NP.[3] Observe further that both sentences are three-fold center-embedded.[4] Hopeless too is an attempt to account for the contrast between those sentences by assuming that SE of "minor" categories such as $\overline{S}$ or PP also contributes to difficulty.[5]

---

1 See, e. g., Yngve (1960), Miller and Chomsky (1963), Langendoen (1975), and Church (1980).

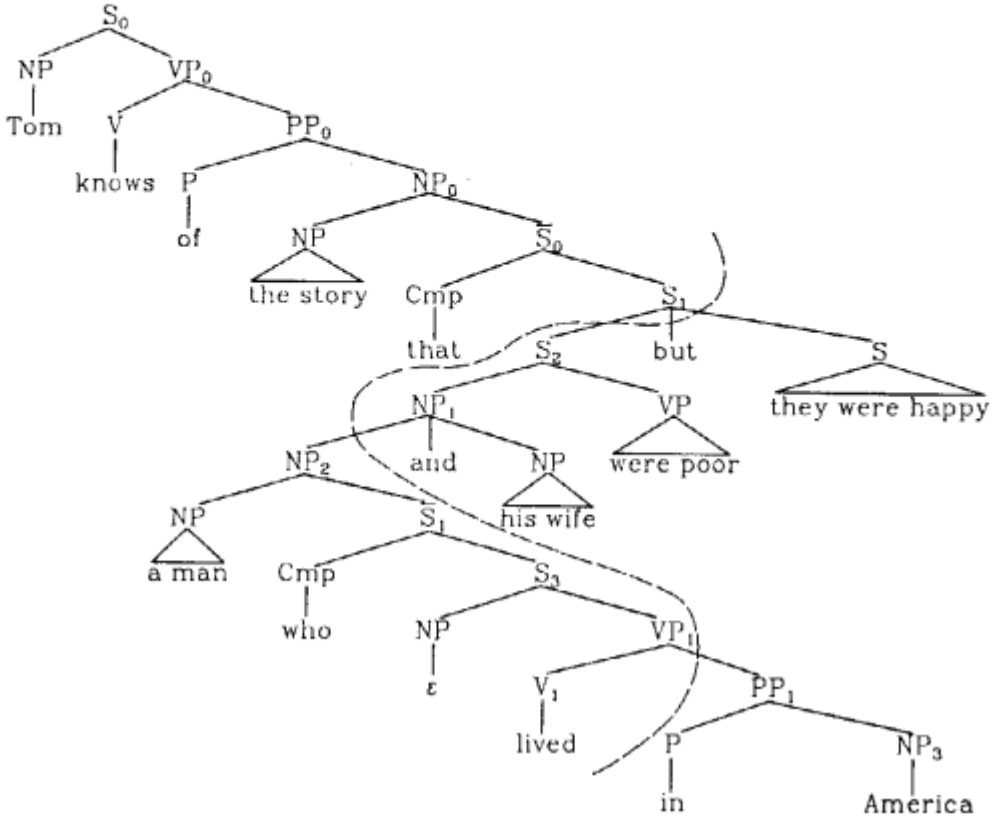
2 Although SE is alleged to cause more difficulty than just CE does, and this difficulty might be argued to relate itself with something other than TML (e. g., Miller and Isard (1964)), such observations turn out more apparent than real.

3 Let us call a list of categories $[X_0, X_1, \cdots, X_n]$ an **embedding chain** (**E-chain**) when X_i center-embeds X_{i+1} for $0 \leq i < n$. $[S_0, S_2, S_3]$ in both sentences, $[NP_0, NP_1, NP_3]$ in (3), and $[NP_0, NP_1, NP_2]$ in (4) are E-chains.

4 In both (3) and (4), $[S_0, S_2, S_3, V_1]$ is an E-chain.

5 The SE depth of $\overline{S}$ appears to explain why (3) is felt easier: $[\overline{S}_0, \overline{S}_1]$ and $[\overline{S}_0, \overline{S}_1, \overline{S}_2]$ are E-chains in (3) and (4), respectively. However, the SE depth of PP predicts to the opposite effect: $[PP_0, PP_1]$ is an E-chain in (3), while there is no SE as to PP in (4).

(3)

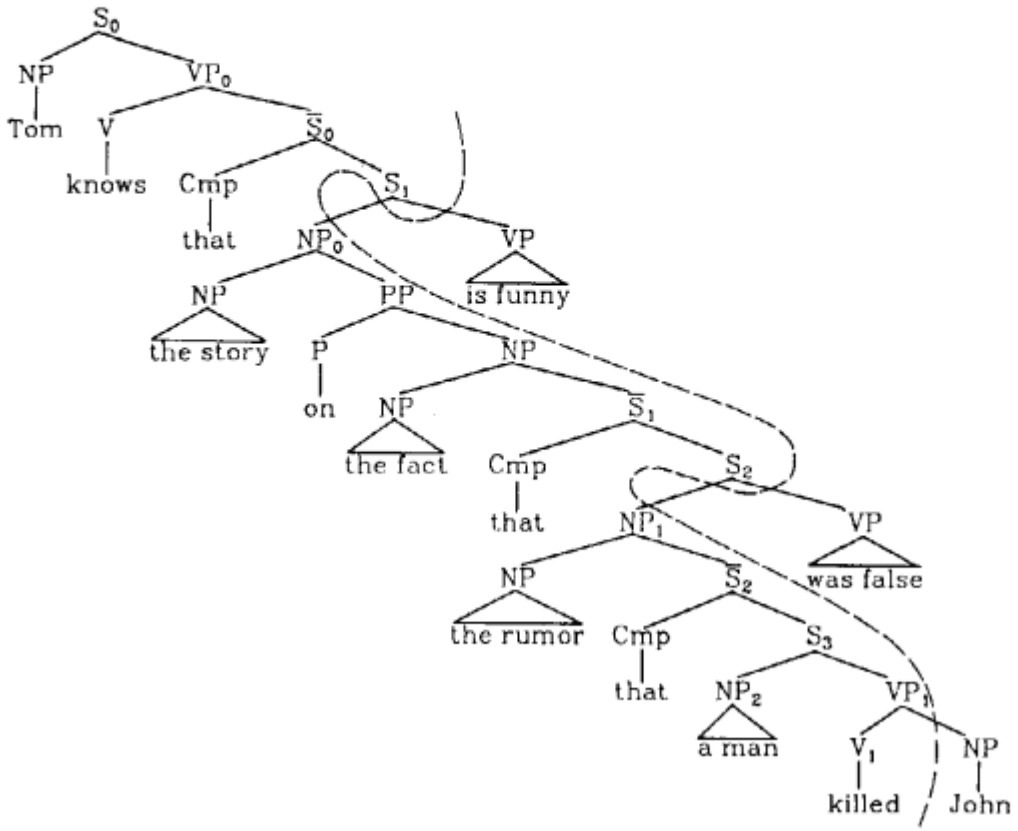


Our explanation shall apply here. Assume that structural descriptions of sentences are phrase-structure trees licensed by some grammar consisting of local tree admissibility rules. The major principle our discussion is based upon is that, while a sentence is being processed, all that is to be retained, perhaps in STM, is the information that specifies the way in which the determined partial structure of that sentence may fit the rest of the context. This information is limited to the neighborhood of the line along which the determined part of the structure confronts the remaining part. Call this line the **frontier**. One must retain potentially every known category that is a part of some branching in which some edges is crossed by the frontier.

As for (1) and (2), the frontier would look like the broken lines depicted in (3) and (4) respectively, when TML is maximum. Observe that, in (3), NP_1 is not known to exist as apart from NP_2, nor is S_2 as apart from S_1. Hence, the categories one must memorize at this moment are S_1, NP_2, VP_1, and V_1 in the case of (3), and S_1, NP_0, S_2, NP_1, VP_1, and V_1 in the case of (4). Thus, the puzzle is resolved by postulating that those categories which are not referred to later are irresponsible for TML.

This explanation might appear to postulate a very special condition as to how the frontier is drawn, but it turns out applicable at least when sentences are locally disambiguated by looking one or two words ahead. The TML measure proposed here is better than others which draw upon SE or CE, in that it explains

(4)



such contrasts as between (1) and (2), and that it is not stipulated separately but derived from functional considerations.

References

Church, K. W. (1980) "On Memory Limitations in Natural Language Processing," MIT/LCS/TR-245, Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, Massachusetts.

Langendoen, D. T. (1975) "Finite State Parsing of Phrase Structure Languages and the Status of Readjustment Rules in Grammar," *Linguistic Inquiry*, **6**: 533-554.

Miller, G. A. and N. Chomsky (1963) "Finitary Models of Language Users," in R. D. Luce, R. R. Bush, and E. Galanter, *Handbook of Mathematical Psychology, Vol. II*, pp. 419-491, John Wiley and Sons, New York.

Miller, G. A. and S. Isard (1964) "Free Recall of Self-embedded English Sentences," *Information and Control*, **7**: 292-303.

Yngve, V. H. (1960) "A Model and an Hypothesis for Language Structure," *Proceedings of the American Psychological Society, Vol. 104*, pp. 444-466,

Feature System and Unification

by

Kuniaki Mukai

Research Center
Institute for New Generation Computer Technology

1. Introduction

Some relation between the two unifications are discussed in the section. One is the unification over features in the GPSG theory [Gazdar 1985]. The other is an extension of the usual first order unification to include the complex indeterminates, which are basic elements of the theory of the situation semantics. The latter one is used as a basic computation mechanism in the programming language, which has been implemented recently on DEC-10 Prolog by the author. The language is called CIL (Complex Indeterminate Language). CIL seems to have a power enough to describe the GPSG feature system in natural way. The main idea is that the feature system is a restricted class of complex inderminates.

2. Complex Indeterminates

The complex indeterminates are abstract elements in situation semantics of J. Barwise & J. Perry [J. Barwise & J. Perry 1983]. Complex indeterminates seems to have a power to make a unified approach to the syntax, semantics, and pragmatics of natural language processing.

Let X and C be an indeterminate and a codition. C may have parameters X, P0, P1, ... ,Pn. The pair of the form <X,C> is called a complex indeterminate. It is written

```
@(X, (R1:P1, ..., Rn:Pn), C)
```

in the programming language CIL. R1, R2, ..., Rn are names for roles of the paremeters. They are given by the user. A complex indeterminate for a discourse situation for instance may be written

```
@(S,

  (speaker:I, hearer:You, disc_loc:Here, expression:Exp ),

 in(S,(Here, (speaking,I), 1))&
 in(S,(Here, (addressing, You),1))&
 in(S,(Here, (utter, Exp),1))),
```

where "in" is the membership relation.

3. Prolog with Complex Indeterminates

An outline of the programming language CIL was given in Mukai [Mukai 1985]. The main motivation behind the language is to describe computational models of discourse understanding based on situation semantics [Barwise & Perry 1983, Barwise 1984]. The language is an extension of Prolog. Parameterized types and complex indeterminates are added to Prolog to represent objects in situation semantics. As a basic control feature to process demons, the language uses the control primitive called "freeze" in Prolog II [Colmerauer 1982], also known as bind hook control.

A few revisions in the CIL system have been made since the first report. First one is the introduction of the notation for complex indeterminates. The second one is the one way unification to compute "anchors". The third one is a restricted version of the compiler.

The following items are the main elements of CIL. The meanings of them should be clear to the reader.

1) clause

<head> <- <condition>

2) term

<indeterminate>
| <complex type>
| <atom>
| <functor>(<term>, ..., <term>)

3) complex type

#(<term>, <role definition list>, <condition>)

4) indeterminate

<basic indeterminate>
| <role indeterminate>
| <complex indeterminate>

5) basic indeterminate

<Prolog variable>

6) comlex indeterminate

@(<term>, <role definition list>, <condition>)

7) role definition

<role symbol> : <term>

8) role indeterminate

<role symbol> ! <basic indeterminate>
| <role symbol> ! <role indeterminate>

9) unification

<term> = <term>

10) freeze

<basic indeterminate> ^ <condition>
| <role indeterminate> ^ <condition>

4. Features and Unification

The following definition of features is equivalent to the one given in [Gazdar etal 1985].

Definition. A feature F is an unordered directed acyclic graph with the following properties.

1) F has the root node.
2) each node in F has a label.
3) F has no branch which has two nodes with the common label.
4) each node has no brother node with the common label.

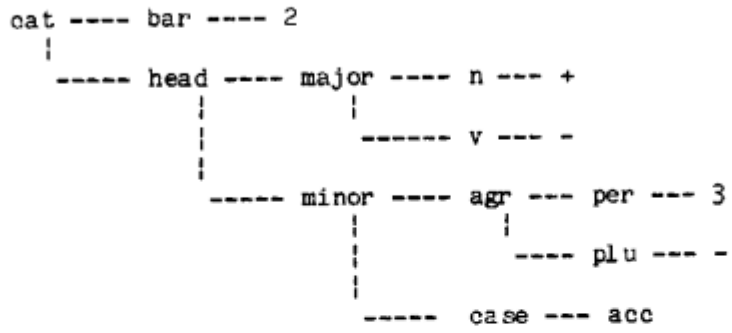
Also the following definition of unification over features is equivalent to the one in [Gazdar etal 1985].

Definition. Given two features x and y, the unification problem is to compute the least common super feature z of x and y if it exists, or to return "fail" otherwise. z is called the result of unification of x and y.

The unification in this sense is one of the primitive functions of CIL.

The complex indeterminate provides a unified view over the feature of GPSG and the functional structure of LFG [Bresnan 1983]. A feature is a complex indeterminate whose condition part is trivial, i.e., "true". A functional structure is a complex indeterminate whose condition is composed of only equality, conjunction, disjunction, and negation. On the other hand, the condition of a complex indeterminate in CIL is able to have any relations as its constituents which are defined by the Horn clauses.

The following is an example use of CIL to describe the GPSG features system. The feature system includes the following category, for instance: (see the next page)



In the follwings, the description, for instance, @(X,_,bar(X)) is written @bar simply. This is one of the syntax sugars used in CIL. CIL can generates the number of 72(=3*2*2*2*3) complete features using these definitions.

```
cat( @(C, (bar: @bar,
           head: @head))
      )<-true.

head( @(H, (major: @major,
            minor: @minor))
      )<-true.

major( @(M, (n: @n,
             v: @v))
      )<-true.

minor( @(M, (agr: @ agr,
             case: @case))
      )<-true.

agr( @(A, (per: @per,
           plu: @plu))
      )<-true.


case(acc)<-true.

bar(1)<-true.
bar(2)<-true.
bar(3)<-true.

n((+))<-true.
n((-))<-true.

v((+))<-true.
v((-))<-true.

plu((+))<-true.
plu((-))<-true.
```

```
per(1)<-true.
per(2)<-true.
per(3)<-true.
```

Given these definitions, an instance of the Head Feature Convention is written:

```
cat(X) & cat(Y) &
agr!minor!head!X = agr!minor!head!Y.
```

This condition works as the constraint of agreement between the features X and Y.

REFERENCE

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

[Barwise 1984] J. Barwise: Lectures on Situation Semantics, Winter Quarter at CSLI, 1984.

[Bresnan 1983] J. Bresnan(eds.): Mental Representation of Grammatical Relations, The MIT Press, 1983.

[Colmerauer 1982] A. Colmerauer: Prolog II: Reference Manual and Theoretical Model, Internal Report, Groupe Intelligence Artificielle, Universite Aix-Marseille II, 1982.

[Gazdar & Klein & Pullum & Sag 1985] G. Gazdar, E. Klein, G.K. Pullum & I.A. Sag: Generalized Phrase Structure Grammar, (to be published), 1985.

[Mukai 1985] K. Mukai : Horn Clause Logic with Parameterized Types for Situation Semantics Programming, ICOT Technical Report, No. TR101, 1985.

Overview of the Japanese Grammar

Takao Gunji

The following is a list of the grammatical rules which appear in the draft of the book in preparaion tentatively entitled *A Phrase Structure Analysis of the Japanese Language*. These rules cover such grammatical constructions as intransitive verbs, transitive verbs, indirect objects, causatives, both transitive (direct) and intransitive (adversity) passives, benefactives (*te-moraw* constructions), "zero" pronominals, reflexive *zibun*, topics (themes), relative clauses, exhaustive *ga*, etc.

Syntactic Features

First, a list of the syntactic features used is exhibited. These features are shown with their possible feature values in the curly brackets.

```
N {+, -}; V {+, -}; BAR {LEX, PHR};
CASE {SBJ {ga, ni}, OBJ {wo, ni, ga}, IOB {ni}};
CON; AUX; CAUS; PASS; BENE; TOP; EXH; SCT;
AGR {set of features};
FOOT {SLASH {set of features}, REFL {SBJ}}; ...
All are HEAD features except for BAR
```

Currently, only two levels of BAR are distinguished: LEXical and PHRasal. The CASE feature takes one of SBJ, OBJ, and IOB as its value. If SBJ is chosen, then this value is another feature which takes *ga* or *ni* as its value. Similarly for the OBJ feature. CON is a feature on verbs which, if present, shows that the verb stem ends with a consonant. This feature is used in determining the form of the verb suffixes *sase* (causative), *rare* (passive), etc. CAUS, PASS, and BENE distinguishes the auxiliary taking a VP complement as the causative, passive, and benefactive suffix, respectively. TOP and EXH identifies the PP with the postposition *wa* (topic) and *ga* (exhaustive), respectively. SCT shows that the behavior denoted by a verb (phrase) is subject-controllable. This feature plays a role in determining the proper postposition of the object in causatives. The AGRecment feature can take a set of features as its value. Thus, any feature in the list can occur as the value of AGR. In this sense, the entire feature system is recursive. Only two kinds of FOOT features are used: SLASH and REFL.

Note that this list is in no ways exhaustive. These are only some of the conceivable features which are useful in describing Japanese.

Notational Conventions for Syntactic Categories and Logical Variables

As is generally assumed in GPSG, sytactic categories are actually a set of features. Here are some of the categories in the more familiar notation with their formal definitions in terms of the above-mentioned features.

```
S:       {[N -], [V +], [AGR {}]}
V(P):    {[N -], [V +], [AGR {SBJ}] [BAR LEX (PHR)]}
TV(P):   {[N -], [V +], [AGR {SBJ, OBJ}], [BAR LEX (PHR)]}
DTV(P):  {[N -], [V +], [AGR {SBJ, OBJ, IOB}], [BAR LEX (PHR)]}
P(P):    {[N -], [V -], [BAR LEX (PHR)]}
N(P):    {[N +], [V -], [BAR LEX (PHR)]}
QNT:     {[N +], [V +], QNT}
```

Note that the verbal categories are disinguished by the AGR feature in this system for Japanese. An S has an empty set as the value of AGR; that is, it doesn't require anything to agree. A V or a VP, on the other hand, has a set consisting of SBJ as the value of AGR; that is, it requires a subject. Likewise, a TV or a TVP requires both a subject and an object, and a DTV or a DTVP requires a suject, an object, and an indirect oject.

The following also introduces shorthand notations for logical variables used for the translations of "zero" pronouns and the reflexive *zibun*.

Notational convention for FOOT variables
- u for r[FOOT [SLASH [PP SBJ]]]
- v for r[FOOT [SLASH [PP OBJ]]]
- z for r[FOOT [REFL SBJ]]

Semantic Types

In semantic translation, most of the forms of the translation are predictable from the semantic types assigned to each syntactic category. The following type assignment is assumed:

```
S:          t
N:          <e,t>
NP:         <<s,<e,t>>,t>
V(P):       <<s,NP>,t>
V[AUX,F]:   <<s,VP>,TVP> where F is CAUS, PASS, or BENE
TV(P):      <<s,NP>,<<s,NP>,t>>
DTV(P):     <<s,NP>,<<s,NP>,<<s,NP>,t>>>
P:          <<s,NP>,NP>
PP:         NP
PP[TOP]:    <<s,t>,t>
QNT:        <<s,N>, NP>
```

ID Rules and Specific Semantic Translations

In the following list of ID rules, if the semantic translation part is in the form of the straightforward functional application (i.e. in the form of *A'(^B')*), the ID rule is shown without mentioning the translation. A specific translation is shown only if the form is not predictable from the semantic types of the constituents as shown above.

Sentences
- <1; S --> PP[SBJ], HP>

Verb Phrases
- <2; VP --> PP[OBJ], HP>

Verb Phrases with a Second Object
⟨3; TVP[wo] --> PP[IOB], HP⟩

Noun Phrases
⟨4; PP --> NP, H⟩

Causativization
⟨10; TVP --> VP[(CON)], H[AUX, CAUS]⟩

VP Embedding
⟨12; TVP[ni] --> VP, H[AUX, F]⟩
where F is CAUS, PASS, or BENE

Causativization with a *wo*-object
⟨13; TVP[wo] --> VP[-wo], H[AUX, CAUS]⟩

Causativization with a *ni*-object
⟨14; TVP[ni] --> VP[SCT], H[AUX, CAUS]⟩

Transitive Passivization (Advancement Passivization)
⟨15; TVP[ni, -PAS] --> TVP, H[AUX, PASS]⟩
(cf. FVD below)

Intransitive Passivization (Adversity Passivization)
⟨16; TVP[ni, -PAS] --> VP, H[AUX, PASS]⟩

Transitive Benefactivization
⟨17; TVP[ni, -PAS] --> TVP, H[AUX, BENE]⟩

Intransitive Benefactivization
⟨18; TVP[ni, -PAS] --> VP, H[AUX, BENE]⟩

VP Embedding with Gaps
⟨12; TVP[SLASH [PP [CASE -c]]]
--> VP[SLASH [PP [CASE c]]], H[AUX]⟩
where, by convention, -SBJ = OBJ and -OBJ = SBJ, and H' in addition to its usual operation, switches any occurrences of *u* and *v* in VP'

Transitive Passivization with Gaps
⟨15; TVP[SLASH [PP [CASE -c]]]
--> TVP[SLASH [PP [CASE c]]], H[AUX, PASS]⟩
where, by convention, -SBJ = OBJ and -OBJ = SBJ, and H', in addition to its usual operation, switches any occurrences of *u* and *v* in TVP'

Topicalization (Type I)
⟨20; S --> PP[TOP, [AGR FOOT]], H⟩

Topics (Type I)
a. ⟨22a; PP[TOP, [AGR [SLASH [PP c]]]] --> PP[c], H⟩
where c is not *ga*, *wo*, or *no*
b. ⟨22b; PP[TOP, [AGR [SLASH [PP c]]]] --> NP, H⟩
where c is *ga*, *wo*, *ni*, or *no*
c. ⟨22c; PP[TOP, [AGR REFL]] --> NP, H⟩

Topicalization (Type II)
⟨21; S --> PP[TOP], H⟩

Topic (Type II)
<22d; PP[TOP] --> NP, H>

N Relativization (Type I)
<23; N --> S[FOOT], H; λr[FOOT][H'(r) & R(r, ^S')]>
where R is a contextual relation between an individual and a proposition whose semantic and pragmatic content is determined depending on the conext

NP Relativization (Type I)
<24; NP --> S[FOOT], HP; λP HP'(Ρ[FOOT][P(r) & R(r, ^S')])>
where R is as in Rule 23

Quantification
<27; NP --> QNT, H>

N Relativization (Type II)
<25, N --> S, H; λx[H'(x) & R(x, ^S')]>
where R is as in Rule 23

NP Relativization (Type II)
<26; NP --> S, HP; λP HP'(Ρ[P(x) & R(x, ^S')])>
where R is as in Rule 23

Exhaustivization
<29; S --> PP[EXH, [AGR FOOT]], H>

Exhaustives
a. <30a; PP[EXH, [AGR [SLASH [PP c]]]] --> PP[c], H>
where c is not *ga, wo, ni*, or *no*
b. <30b; PP[EXH, [AGR [SLASH [PP c]]]] --> NP, H>
where c is either *ga,wo, ni*, or *no*
c. <30c; PP[EXH, [AGR REFL]] --> NP, H>

LP Rule

Along with the above-mentioned ID rules, the following single LP rule is assumed for Japanese.

A < H for any category A

Feature Value Default and Feature Coocurence Restriction

In addition, the following default and restriction on the occurrence of feature values are assumed. These show only some examples and don't exhaust the necessary FVDs and FCRs.

FVD: TVP[PAS]
FCR: *TVP[[SBJ ni] [OBJ {wo, ni}]]

Lexical Rules and Specific Translations

The following is a list of lexical rules and translations specifically required for some particular lexical items.

Case Markers
〈P[c] --> *c*; c'〉

Causative Suffix
a. 〈V[AUX, CAUS] --> *sase*; CAUS'〉
b. 〈V[AUX, CAUS, CON] --> *ase*; CAUS'〉

Semantics of the Causative Suffix
CAUS' = λXλQ λP P{ϑCAUSE(x, ^X{Q})}

Passive Suffixes
a. 〈V[AUX, PASS, n] --> *rare*; PASS[n]'〉
b. 〈V[AUX, PASS, n, CON] --> *are*; PASS[n]'〉

Semantics of the Transitive Passive Suffix
PASS[15]' = λXλQ λPX{Q, P}

Semantics of the Intransitive Passive Suffix
PASS[16]' = λXλQ λP P{ϑR(x, ^X{Q})}

Benefactives
a. 〈V[AUX, BENE, n] --> *temoraw*; BENE[n]'〉
b. 〈V[AUX, BENE, n, CON] --> *i-temoraw*; BENE[n]'〉

Semantics of the Transitive Benefactive Suffix
BENE[17]' = λXλQ λP P{ϑBENEFIT(x, ^X{Q, ^x*})}

Semantics of the Intransitive Benefactive Suffix
BENE[18]' = λXλQ λP P{ϑBENEFIT(x, ^X{Q})}

Reflexive
〈NP[REFL] --> *zibun*; z*〉

Subject Gap
〈PP[SBJ]/PP[SBJ] --> e; u*〉

Object Gap
〈PP[OBJ]/PP[OBJ] --> e; v*〉

Topic Marker (Type I)
〈P[TOP, FOOT] --> *wa*; λPλpP'[FOOT]R(r, p)〉
where R is a contextual relation between an individual and a proposition whose semantic and pragmatic content is determined depending on the context

Topic Marker (Type II)
〈P[TOP] --> *wa*; λPλp P{ϑR(x, p)}〉
where R is as above

Quantifiers
a. 〈QNT --> *aru*; λQλP∃x[Q{x} & P{x}]〉
b. 〈QNT --> *arayuru*; λQλP∀x[Q{x} -> P{x}]〉

Exhaustive Marker
〈P[EXH, FOOT] --> *ga*; λPλp P{ϑ∀x[y=x <-> λr[FOOT][ˇp](x)]}〉

Metarule

Currently, the following general metarule is assumed for "zero" pronouns and the reflexive. This metarule may well be replaced by a principle of translation of some sort in a later version of the grammar.

Control Metarule (CM)
<n; {[N −], [V +], [AGR [CASE c]], FOOT} --> X; T>
==> <n; {-FOOT} --> X; T'>
where c is a case feature coefficient, FOOT is either [SLASH [PP [CASE c]]] or [REFL [CASE c]], and T' is obtained from T by binding any free occurrence of r[FOOT] in accordance with Bach's principle

Bach's Principle
If a controllee is in an XP (X=V or TV), it is controlled by the argument of the XP, i.e., next higher NP outside the XP.

This metarule has more specific forms depending of the value of the FOOT feature taken, as are shown below.

Reflexivization Metarule (RM)
<n; VP[REFL] --> X; T> ==> <n; VP --> X; λp p{∃T(^z*)}>

Subject Control Metarule (SCM)
<n; VP/PP[SBJ] --> X; T> ==> <n; VP --> X; λp p{∀T(^u*)}>

Object Control Metarule (OCM)
<n; TVP/PP[OBJ] --> X; T> ==> <n; TVP --> X;
λQ λp Q{∀T(p, ^v*)}>

Subcategorization and Morphology of the Japanese Verb Syntax

Mariko Udo

The Graduate School of Humanities and Social Sciences,
Kobe University

1. Introduction

This paper is to sketch very briefly the outline of the subcategorization framework in syntax for Japanese verbs especially in relation to its morphology. The rules given below are mostly based on Udo (1982), but it has been considerably revised on the features to be employed and the treatment of NP particle agreement in accordance with the current GPSG framework(Gazdar et al "Generalized Phrase Structure Grammar", in press).

2. Basic assumptions

To incorporate morphological facts into syntactic rules, the following are assumed.

1) Taking one and only one inflectional form is prerequisite for any verb to behave as a syntactic category. The verb form serves as a morphosyntactic feature [VFORM], and in some VP rules it is required onto the VP compliments of its lexical head. The inflection patterns differ according to what work class the verb in question belongs to, and this information is to be stated in lexicon (see Table 1).

2) Syntactic postpositions on NPs are divided into two types according to whether the whole phrases marked by them behave as the arguments of verbs or the modifiers of them. The former are marked as NPs, the cases of which agree with co-occurring verbs, and the latter simply as PPs.

3) There is neither 'adjectives' nor 'auxiliary verbs', which are the terms used in the prevailing Japanese linguistics. Their implicitly understood difference from verbs is that the former have intuitively adjective-like meaning and the latter do not appear independently of so-called main verbs. This, however, is not such a great distinction as to justify postulating two separate categories. On the contrary, that would result in losing various generalizations, e.g. about coordinatability, morphological pattern, finite property (tensability), structural distributions, etc.. Unitary treatment is more desirable: regarding any category with the finite property as [+V,-N] category, 'Adj' is its extension with the feature [+COP](mnemonic for copular), and 'AUX' is another extension of it subcategorized to have VP compliments.

3. Morphology

Table 1 : The Inflection Patterns of Japanese Verbs

patterns / morpho-syntactic features		+ COP		- COP			
				regular		irregular	
		1	2	1	2	1	2
+FIN	-PAST	akai ($-i)	sizukada ($-a)	yomu ($-u)	miru ($-ru)	kuru	suru
	+PAST	akakatta ($-katta)	sizukadatta ($-atta)	yonda ($-ita)	mita ($-ta)	kita	sita
-FIN	M 1	* (by FCR)	* (by FCR)	yoma ($-a)	mira (s) ($-ra sa)	kora (s)	sa
	M 2	akaku ($-ku)	sizukade ($-e)	yoma ($-a)	mi ($-)	ko	si
	M 3	akaku ($-ku)	sizukani sizukade ($-e)	yomi ($-i)	mi ($-)	ki	si
	M 4	akakere ($-kere)	* (by DC)	yome ($-e)	mire ($-re)	kure	sure
	M 5	akakaro ($-karo)	sizukadaro ($-aro)	yomo ($-o)	miyo ($-yo)	koyo	siyo
	+IMP	* (by DC)	* (by DC)	yome ($-e)	miro ($-ro)	koi	siro
	INF	akakute	* (by DC)	yonde	mite	kite	site
stems		aka-	sizukad-	yom-	mi-	k-	s-
members (in [-PAST] form)		kawaii hosii rasii etc.	da genkida yooda etc.	iku naku utsu etc.	taberu reru seru etc.	kuru only	suru only

DC ; decopularization (see below)
$; verb stem

Some word-formation rules (contraction with post-verbal particles)

V^{o}[+PAST] $\longrightarrow$ V^{-1}[M 3] ta
V^{o}[+INF] $\longrightarrow$ V^{-1}[M 3] te
V^{o}[CONJ] $\longrightarrow$ V^{-1}[FIN] shi
V^{o}[Conditional] $\longrightarrow$ V^{-1} [M 4] ba

Decopularization : the relation between copular verbs and 'aru' A copular (+COP) verb is attached to a verb (-COP)'aru' and cliticization takes place, removing the word boundary, which results in forming one word (morphological aru-insertion).

e.g. akaku [+COP M3] + are [-COP M4] ⟶ akakere (ba) [+COP M4]

undergoes some phonological process

This aru-insertion sometimes occurs without cliticization, which enables a copula verb to appear in structures which copula verbs cannot appear in, i.e. imperative constructions. Here the verb is decopularized.

e.g. tadashiku [M3(+COP)] + are [+IMP] ⟶ tadashiku are ('Be fair!)

4. Features employed and restrictions on the application of them

(a) morphosyntactic features on verbs

VFORM: {{+FIN { +PAST, -PAST }}, {-FIN { M1, M2, M3, M4, M5, +IMP, INF }}}

These follow the Feature Cooccurrence Restriction(FCR) 1a

FCR 1a: $[+x] \longrightarrow [-y]$

where $x, y \in \{$ +PAST, -PAST, M1, M2, M3, M4, M5, +IMP, INF $\}$

(b) features to indicate syntactically significant categories

TRN(transitive), DTR(ditransitive), COP(copular)
NEG(negative), CONJ(introducing coordinate constructions)

FCR 2 : $[+COP] \supset \sim [IMP]$

FCR 3 : $[M1] \supset \sim [STAT]$

FSD : $[+STAT] \supset [+COP]$

(c) Features to specify the dynamic properties of verbs: every verb has the value for the following features in lexicon and these features constitute syntactic categories which may influence upon the well-formed conditions of subcategories.

AGNT (mnemonic for agentive, meaning action takes place or some state holds in control of its semantic subject)

STAT (stative; condition described in verbs has aspect-relevance)

To give a few examples, agentive verbs are tsukuru(make), taberu(eat), dekiru(be able to), iku(go), seru(causative make), etc. and non-agentive ones are, aru(exist), sinu (die), reru(passive be), etc. Stative verbs are nai(do not exist), wakaru(understand), akai(be red), hosii(want), etc. and non-stative ones are naku(cry), hosigaru(show interest), siru(get to know), akaramu(become red),etc..

(d) NP case-agreement features :

In the ID rules introducing subject and object NPs, NPs are unspecified for case-markers, and the FCRs stated here would consequently subcategorize transitive VPs. The FCR 4-6 suggest Japanese has two distinct classes of verbs, namely so-called accusative type and ergative type. The FCR 7-9 give some constraints on the distribution of case-particles partially determined by the dynamic properties of the governing verbs.

AGR $\left\{ \begin{array}{l} \text{SAGR } \{\text{ga, ni}\} \text{ (subject agreement)} , \\ \text{OAGR } \{\text{o, ni, ga}\} \text{ (object agreement)} \end{array} \right\}$

e.g. [SAGR, NP [CASE ni]] (or simply [SAGR,ni])
[OAGR, NP [CASE o]] (or simply [OAGR,o])

FCR 1b : $[+x] \rightarrow [-y]$ where $x, y \in \{\text{ga, o, ni}\}$

FCR 4 : [+ TRN] $\supset$ [+ OAGR]

FCR 5 : [OAGR, {ni, o}] $\supset$ [SAGR, ga] (accusative type)

FCR 6 : [OAGR, ga] $\supset$ [SAGR, ni] (ergative type)

FCR 7 : [OAGR, ni] $\supset$ [-AGNT]

FCR 8 : [OAGR, o] $\supset$ [+AGNT]

FCR 9 : [OAGR, ga] $\supset$ [+STAT]

5. The constituent structure

(a) Linear Precedence(LP) rule

$$\alpha \; < \; H[+V, -N]$$

(b) Immediate Dominance(ID) rules

< 1. $V^2[-SUBJ] \longrightarrow H^2$; V^{2}' > e.g. Ameda (Raining!), ...

< 2. $V^2[+SUBJ] \longrightarrow N^2$, $H[-SUBJ]$; $V^{2}'(N^{2}")$ >

< 3. $VP \longrightarrow H$; V' > e.g. iku(go), warau(laugh), ...

< 4. $VP \longrightarrow X^2$, $H[+PRD]$; $\lambda \mathbb{P}[V'(\hat{}X^{2}'(\mathbb{P}))]$ >

e.g. da(be), rasii(seem),...

where $X[+PRD] \in \{V^2, N^2\}$

FCR 10 : $[+PRD] \supset \sim[TRN]$

< 5. $VP \longrightarrow N^2$, $H[+TRN]$; $V'(N^{2}")$ >

< 6. $VP \longrightarrow N^2, N^2$, $H[+DRN]$; $V'(N^{2}")(N^{2}")$ >

e.g. ataeru(give), osieru(teach),...

FCR 11 : $[+DRN] \supset [[OAGR, o], [OAGR, ni]]$

N.B. This, together with FCR 5, guarantees that the V [SUBCAT 6] appears only in accusstive type structures.

< 7. $VP \longrightarrow VP[-COP, M3]$, $H[-AGNT]$; $\lambda \mathbb{P}[V'(\hat{}VP'(\mathbb{P}))]$ >

e.g. sooda(belikely to), yasui(be easy to),..

< 8. $VP \longrightarrow VP[-COP, M3]$, $H[+AGNT]$; $V'(VP")$ >

e.g. hajimeru(begin), owaru(finish),...

< 9. $VP[+NEG] \longrightarrow VP[M2]$, H ; $\lambda \mathbb{P}[\neg VP'(\mathbb{P})]$ >

e.g. nai, zuda.

< 10. $VP \longrightarrow V^2[M1]$, H ; $\lambda \mathbb{P}[V'(V^{2}")(\mathbb{P})(N^{2}")]$ >

V[10] = reru(passive), seru(causative).

FCR 12 : $[M1] \supset [SAGR, ni]$

FCR 13 : $[M1, +TRN] \supset [OAGR, \{o, ni\}]$ (See also FCR3)

Passive < 10a. $VP \longrightarrow V^2[M1]$, $H[-AGNT]$ >

Causative < 10b. $VP \longrightarrow V^2[M1]$, $H[+AGNT]$ >

N.B. In this analysis so-called 'o-causative' constructions are treated as the case where verb heads lexically-inherently have causative meaning. Those particular class of verbs are formed via the word formation rule $\langle V^0 \longrightarrow V^{-1}$ [M1] s $\rangle$ (morphologically belonging to the type '-COP regular 1'); e.g. tokasu(melt), nigasu(free), warawasu(make laugh), akasu(reveal), etc. Having both [+TRN] and [+AGNT] properties, these get the NP objects marked with 'o' by FCR 8.

Imperative

Positive imperative : the extension of ⟨1⟩

$\langle V^2[+IMP] \rightarrow N^2,\ H \rangle$

Negative imperative

$\langle V^2[+IMP, +NEG] \longrightarrow H\ [+na] \rangle$

$\langle V^0[+na]\ V^{-1}[-PAST, -NEG, -COP]\ na \rangle$

Verb Coordination Schemata

$\langle V^n \longrightarrow V^n [CONJ, c]^+ ,\ H \quad (n = 1, 2) \rangle$

where $c \in \{M3, INF, shi\}$

Transformational Rules and Generalized Phrase Structure Grammar

Hidetosi Sirai (Tamagawa Univ.)

1. Introduction

GPSG(Generalized Phrase Structure Grammar) is the linguistic theory that attempts to explain linguistic phenomena without 'transformations.' The major concern here is whether every linguistic phenomena can be fully described in GPSG framework. In other words, how does GPSG explain such phenomena that transformational grammarians needed transformational rules to describe? In this paper, we discuss about the GPSG theory that corresponds to so-called movement transformational rules.

2. Transformational Rules and GPSG

2.1 Subject-Auxiliary Inversion

This is the rule to invert auxiliary verbs and subject NPs in interrogative sentences.

(2.1) Kim can go. => Can Kim go?

[GPSG] Subject-Auxiliary Inversion Metarule.

(2.2) VP[-SUBJ] -> W => VP[+INV,+SUBJ] -> W, NP

2.2 Tag Formation

This is the rule to attach tag questions at the end of declarative sentences.

(2.3) John is a boy. => John is a boy, isn't he?

[GPSG] Not yet given.

2.3 Topicalization

This is the rule that moves NP to the front of sentences.

(2.4) I love Mary. => Mary, I love.

[GPSG] Generally handles with the following ID rule schema.

(2.5) S -> XP, H/XP.

2.4 WH Fronting

This rule moves interrogative phrases to the front of sentences.

(2.6) John saw somebody. => Who did John see?

[GPSG] ID rules.

(2.7) S -> NP, H[-SUBJ]

(2.8) S -> XP, H/XP

As interrogative pronouns have FOOT feature WHs, we have the following ID rules generated from the above two rules.

(2.9) S[+Q] -> NP[+Q], VP

(2.10) S[+Q] -> NP[+Q], S/NP

(2.11) S[+Q] -> PP[+Q], S/PP

(2.12) S[+Q] -> AP[+Q], S/AP.

For interrogative root sentences, we need the following ID rule.

(2.13) S[+Q] -> XP[+Q], H[+INV,+SUBJ,SLASH XP].

2.5 Relativization

This rule makes relative phrases.

(2.14) This is the book. I bought the book at an auction.
=> This is the book (that) I bought at an auction.

[GPSG] ID rules.

(2.15) N' -> H, S[+R]
(2.16) S -> NP, H[-SUBJ]
(2.17) S -> XP, H[SLASH XP]

Note that these last two rules are the same in WH-Fronting because R is also a FOOT feature.

2.6 Dative Movement

This rule changes dative NP to PP and moves it to the end of sentences.

(2.18) John gives Mary a book. => John gives a book to Mary.

[GPSG] Deals with different ID rules as follows preserving meaning equivalence by meaning postulates.

(2.19) VP -> H[SUBCAT 5], NP, NP
(2.20) VP -> H[SUBCAT 3], NP, PP[PFORM to].

2.7 There Insertion

This rule operates on sentences containing occurrences of auxiliary 'be', it will have the effect of inserting the subject NP into Aux, in between be and its affix.

(2.21) Some boys have been running down the road.
=> There have been some boys running down the road.

[GPSG] ID rules.

(2.22) VP[AGR NP[there, αPLU]] -> H[SUBCAT 22], NP[αPLU].
(2.23) NP -> H, VP[+PRD].

Here, 'there' is assigned to the category indicated by the lexical entry:

(2.24) <there, NP[PRO, NFORM there], {}, Δ>

2.8 Extraposition

This rule moves the embedded sentence to the end of the main sentence.

(2.25) That Robin was chosen bothered Lou.
=> It bothered Lou that Robin was chosen.

[GPSG] Extraposition Metarule.

(2.26) XP[AGR S] -> W => XP[AGR NP[it]] -> W, S.

For example, 'bother' is assigned the subcategorization corresponding to the following ID rule:

(2.27) VP[AGR S] -> H[SUBCAT 20], NP.

However, it would be a problem that the S included in the produced new ID rule is considered as a complement.

2.9 Passivization

This rule produces passive sentences.

(2.28) John ate the apple. => The apple was eaten by John.

[GPSG] Passive Metarule and ID rules.

(2.29) VP -> W, NP => VP[VFORM PAS] -> W, (PP[PFORM by]).
(2.30) VP -> H[SUBCAT 7], XP[+PRD].
(2.31) VP -> H[SUBCAT 49], NP, VP[VFORM PAS]

Here, V[7] = {be} and V[49] = {get, have, see, ...}.

3. Consideration on Tag Questions

Tag questions have such patterns as follows

(3.1) V[+AUX,(+n't)] NP[+PRO] or
V[+AUX] NP[+PRO] NOT.

They also have such syntactic features that the auxiliary verb and NP correspond to the main verb and the subject NP of the main sentence, respectively. Furthermore, if the main sentence is negative, then the verb in the tag question is positive, and vice versa.

We propose such metarule, ID rules and FCR as follows.

(3.2) Tag Metarule:
VP -> W, VP => S[TAG,AGR α] -> W, NP[PRO, α]

(3.3) S -> NP[α], H[-SUBJ,~AUX], S[TAG,+NEG,AUX do,AGR α]

(3.4) S -> NP[α], H[-SUBJ,βNEG,AUX γ], S[TAG,-βNEG,AUX γ,AGR α]

(3.5) VP[+NEG,AUX α] -> H[SUBCAT α,+n't],VP

(3.6) VP[+NEG,AUX α] -> H[SUBCAT α], NOT, VP

(3.7) Feature Cooccurrence Restriction:
[NEG] ⊃ [+AUX]
[TAG] ⊃ [+INV]

Here, we have the following assumptions:

1. Tag questions don't have slash category in the syntactic sense,
2. NEG is a HEAD and boolean feature,
3. AUX is not a boolean feature but has values as shown {do, be, have, shall, will, must, should, would, ...} and it corresponds to some appropriate SUBCAT feature value.
4. 'not' belongs to the cagegory NOT and the [+n't] feature is realized in the following words:isn't, aren't, don't, etc.

4. Conclusion

We have described small parts of linguistic phenomena, which transformational grammarians needed transformations to explain. And we have also shown that GPSG theory deals with them in the context free grammar framework without transformations.

References

Gazdar, G.:"Unbounded Dependencies and Coordinate Structure," Linguistic Inquiry, 12 (1981) pp.155-184.

Gazdar, G.:"Phrase Structure Grammar," In P. Jacobson and G. K. Pullum (eds.)The Nature of Syntactic Representation, D. Reidel Pub. (1982) pp.131-186.

Gazdar, G., Pullum, G. and Sag, I.:"Auxiliaries and Related Phenomena in a Restrictive Theory of Grammar," Language 58 (1982) pp.591-638.

Gazdar, G., Klein, E., Pullum, G. and Sag, I.:"Coordinate Structure and Unbounded Dependencies," In M. Barlow, D. Flickinger and I. Sag (eds.)Developments in Generalized Phrase Structure Grammar:Stanford Working Papers in Grammatical Theory, Vol. 2, Indiana Univ. Ling. Club (1982) pp.38-68.

Gazdar, G., Klein, E., Pullum, G. and Sag, I.:Generalized Phrase Structure Grammar, Blackwell (to be published).

Bibliography

The following is a bibliography of works on computational implementation of GPSG discussed at the Working Group during the first year, along with some of the fundamental works in GPSG general.

Bear, J. and Karttunen, L. (1979), "PSG: a simple phrase structure parser," *Texas Linguistic Forum, 15*, (1979), 1-46.

Evans, R. and Gazdar, G. (1984), *The ProGram Manual*, Technical Report, Cognitive Studies Programme, University of Sussex, April 1984.

Farkas, D., Flickinger, D., Gazdar, G., Ladusaw, W., Ojeda, A., Pinkham, J., Pullum, G., and Sells, P. (1983), "Some revisions to the theory of features and feature instantiation," *Proceedings of the ICOT Workshop on Non-Transformational Grammars*, 11-13.

Flickinger, D. (1983), "Lexical heads and phrasal gaps," in M. Barlow, D. Flickinger, and M. Westcoat (eds.), *Proceedings of the Second West Coast Conference on Formal Linguistics*, Stanford Linguistics Department, 1983.

Gazdar, G. (1981) "Unbounded dependencies and coordinate structure," *Linguistic Inquiry, 12*, (1981), 155-184.

Gazdar, G. (1982), "Phrase structure grammar," in P. Jacobson and G.K. Pullum (eds.), *The Nature of Syntactic Representation*, Dordrecht, D. Reidel, 1982, pp. 131-186.

Gazdar, G. (1983), "Recent computer implementations of phrase structure grammars," unpublished manuscript, Cognitive Studies Programme, University of Sussex, October 1983.

Gazdar, G. and Pullum, G. (1982), *Generalized Phrase Structure Grammar: A Theoretical Synopsis*, Indiana University Linguistics Club, August 1982.

Gazdar, G., Klein, E., Pullum, G., and Sag, I. (1982), "Coordinate structure and unbounded dependencies," in M. Barlow, D. Flickinger, and I.A. Sag (eds.), *Developments in Generalized Phrase Structure Grammar: Stanford Working Papers in Grammatical Theory, Volume 2*, Indiana University Linguistics Club, November 1982.

Gunji, T. (1983), "Generalized phrase structure grammar and Japanese reflexivization," *Linguistics and Philosophy, 6*, (1983), 115-156.

Gunji, T. (1982-4), *A Phrase Structure Analysis of the Japanese Language*, unpublished book draft.

Joshi, A.K. "Factoring recursion and dependencies: an aspect of tree adjoining grammars (TAG) and a comparison of some formal properties of TAGs, GPSGs, PLGs, and LFGs," *Proceedings of the 21st Annual Meeting of the ACL*, (1983), 7-15.

Kilbury, J. *GPSG Based Parsing and Generation*, KIT-Report 17, Project KIT,

Technische Universität Berlin, May 1984.

Klein, E. and Sag, I. (1982), "Semantic type and control," in M. Barlow, D. Flickinger, and I.A. Sag (eds.), *Developments in Generalized Phrase Structure Grammar: Stanford Working Papers in Grammatical Theory, Volume 2*, Indiana University Linguistics Club, November 1982.

Klein, E. and Sag, I. (to appear), "Type driven translation," to appear in *Linguistics and Philosophy.*

Koskenniemi, K. (1983), "Two-level model for morphological analysis," *Proceedings of IJCAI '83*, (1983), 683-685.

Pollard, C.J. (1984), *Generalized Phrase Sructure Grammars, Head Grammars, and Natural Language*, Ph.D. dissertation, Stanford University, 1984.

Pullum, G. (1982), "Free word order and phrase structure rules," in J. Pustejovsky and P. Sells (eds.), *Proceedings of the Twelfth Annual Meeting of the North Eastern Linguistic Society*, Graduate Linugistics Student Association, University of Massachusetts, 1982, pp. 209-220.

Pulman, S.G. (1982), "Generalised phrase structure grammar, Earley's algorithm and the minimisation of recursion," in K. Sparck-Jones and Y. Wilks (eds.) *Automatic Natural Language Parsing*, Cognitive Studies Centre, University of Essex, 1982.

Sag, I.A., Gazdar, G., Wasow, T., and Weisler, S. (1984), *Coordination and How to Distinguish Categories*, Report No. CSLI-84-3, Center for the Study of Language and Information, Stanford University, March 1984.

Shieber, S.M. (1983), "Sentence disambiguation by a shift-reduce parsing technique," *Proceedings of IJCAI '83*, (1983), 699-703.

Shieber, S.M. (1984), "Direct parsing of ID/LP grammars," *Linguistics and Philosophy, 7*, (1984), 135-154.

Shieber, S., Stucky, S., Uszkoreit, H., and Robinson, J. (1983), "Formal constraints on metarules," Technical Note 283, SRI International, 1983.

Stucky, S. (1983), *Metarules as Meta-Node-Admissibility Conditions*, Technical Note 304, SRI International, September 1983.

Udo, M. (1982), *Syntax and Morphology of the Japanese Verb: A Phrase Structural Approach*, MA thesis, University College London, 1982.

LIST OF CONTRIBUTORS

Shinya	Amano	Toshiba Corporation
Takao	Gunji	Osaka University
Koiti	Hashida	Tokyo University
Hideki	Hirakawa	3rd Laboratory, ICOT
Kuniaki	Mukai	3rd Laboratory, ICOT
Hidetoshi	Shirai	Tamagawa University
Mariko	Udo	Kobe University