Augmented Dependency Grammar : A Simple Interface between the Grammar Rule and the Knowledge

Kazunori MURAKI , Shunji ICHIYAMA
C&C Systems Research Laboratories
NEC Corporation
Kawasaki-city,213 JAPAN and

Yasutomo FUKUMOCHI
Softwear development devision
NSIS Corporation
Kawasaki-city,213 JAPAN

ABSTRACT

This paper describes some operational aspects of a language comprehension model which unifies the linguistic theory and the semantic theory in respect to operations. The computational model, called Augmented Dependency Grammar **(ADG)**, formulates not only the linguistic dependency structure of sentences but also the semantic dependency structure using the extended deep case grammar and field-oriented fact-knowledge based inferences. Fact knowledge base and ADG model clarify the qualitative difference between what we call semantics and logical meaning. From a practrical view point, it provides clear image of syntactic/semantic computation for language processing in analysis and synthesis. It also explains the gap in semantics and logical meaning, and gives a clear computaional image of what we call conceptual analysis.

This grammar is used for analysis of Japanese and synthesis of English, in the Japanese-to-English machine translation system called **VENUS** (Vehicle for Natural Language Understanding and Synthesis) currently developed by NEC.

Basic Idea

The **VENUS** analysis model consists of two components, Legato and Crescendo, as shown in Fig. 1. Legato based on the ADG framework, constructs semantic dependency structure of Japanese input sentences by feature-oriented dependency grammar rules as main control information for syntactic analysis, and by semantic inference mechanism on a object fields' fact knowledge base. Legato maps syntactic dependency directly to meaningful logical dependency if possible, or maps it to language-particular semantic dependency if two kinds of dependencies do not coincide. The second component, Crescendo, extracts a conceptual structure about facts from the semantic dependency structure through logical interpretation on the language-particular semantic dependency using knowledge based inferences.

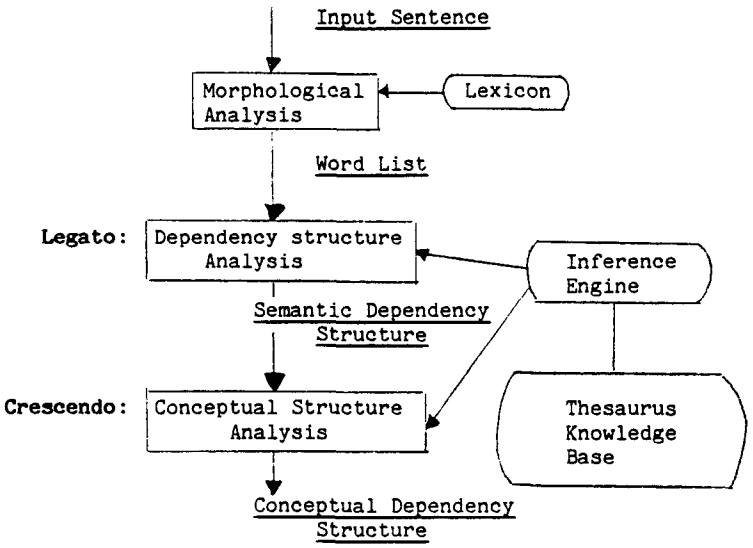


Fig. 1 VENUS Analysis Module

A computational comprehension model for the **ADG** is given in Fig. 2. Three different kinds of information sources other than the lexicon support language comprehension, and two inference functions defined on them extract the interpretation of input sentences. The top level information is a language structure model. The bottom is a logical(factual/conceptual) interpretation model which determine the possible logical relations between "**OBJECT**s and **THING**s".

The semantics located between the above two models, which has not been clarified in any paper. Suppose interpretaion is a process of determining the relation between " **OBJECT**s and **THING**s ", the ordinary notion of semantics allows us to determine words' semantics in particular syntagmatic relations, but not relational interpretation between concepts.

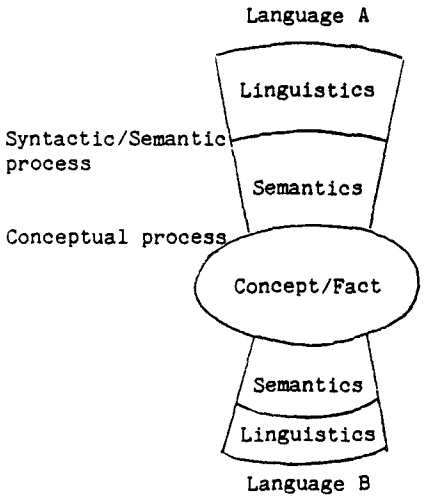


Fig. 2 Comprehension Model

The semantics here is defined as information concerning the denotation of **OBJECT**s and **THING**s. It interprets the (semantic) relations between them, and must be inducible from the raw syntagmatic information. That is to say, it may sometimes inherits such language particular features as syntactic structure, wording, culture. The structure representing semantics may not be interpretable in terms of pure logic, but may be represented linguistically.

1) The ADG defines syntactic dependency structure, semantic dependency structure, and descriminates the semantic dependency from the logical structure.
2) It functions as the interface between syntactic dependency and semantic dependency.

The notion of basically binary "dependency" has a primary role to simplify the above interface, just in the sense that either syntactic or semantic inference recognizes interpretable binary relation. The semantics in the sense used here may not necessarily be shared among languages, while facts are shared among languages.

Legato built on the model is syntactic and semantic analysis module which construct directly semantic dependency structure from surface structure. Crescendo is an engine to eliminate non-logical part in semantic structure and induces logical structure with pragmatic information deduced from semantics.

Semantic/Logical Interpretation

Each word has its own meaning, sometimes plural meanings. In this paper word meaning is represented by a logical symbol called **CONCEPT SYMBOL**. The symbol is a representation primitive for fact knowledge base and internal conceptual representation of sentences. Semantic structure representation is also defined on them, but it borrows syntagmatic function called dummy symbols which never appear in conceptual representation.

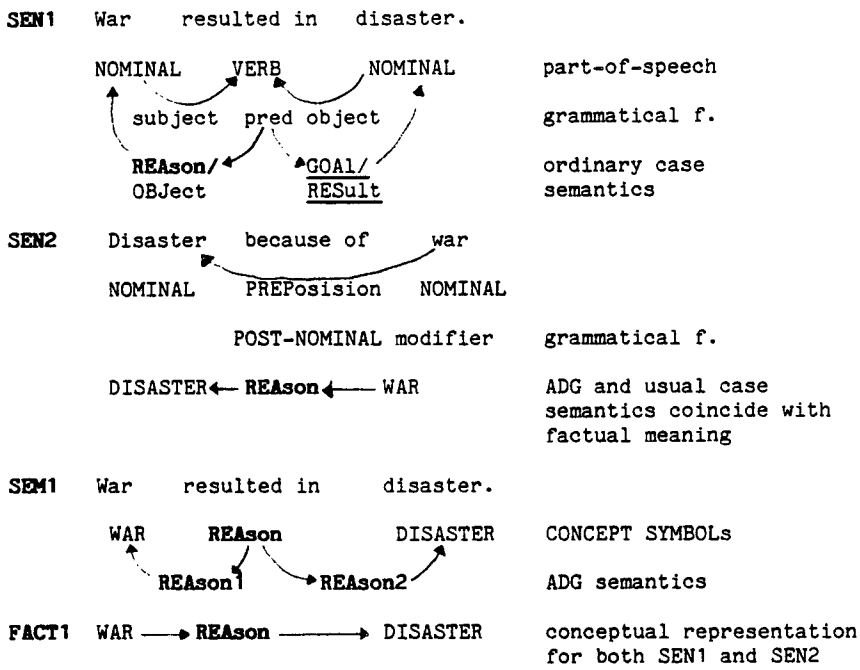
The above examples **SEN1**, **SEN2** share the same meaning as shown in **FACT1**, except pragmatic and temporal information. Ordinary analysis of SEN1 produces subject-predicate-object syntagmatic information, and further case interpretaion of subject-predicate,object-predicate relations. However this kind of case interpretation brings into difficulties to select case marking ambiguities such as GOAl or RESult for the above object-predicate. **SEN2** analysis produces instantly **REAson** interpretation between two nominals in terms of "**REAson**"-marking preposition "because-of". This comparison supports the case even a verb must be interpreted in some case as a logical relation and clarifies the standpoint to specify the ADG.

1. Factual(conceptual) information must be independent of syntagmatic meaning as well as independent of syntax.

2. Ordinary case marking strategy produces anomaly because it dare to interpret syntagmatic relations logically even if those are purely syntagmatic existence.

3. Fillmore's case is not suitable for conceptual representaion primitive for a variety of syntactic and syntagmatic structures.

On the other hand, syntax is a clue to understanding of sentences. Syntagmatic relations, in most cases, can be interpretable as in **FACT1** for **SEN2**, and linguistic information is a sole trigger for human to recognize new notion or new word meaning in a sentence.



Comprehension of constructing factual information is defined by two different levels understanding; 1 **LEGATO** semantic analysis (as shown in **SEM1**, **FACT1** for **SEN1,2** respectively) with direct correspondence to syntagmatic relation, and 2. **CRESCENDO** factual (logical) understanding as in a extraction process of FACT1 from **SEN1** via **SEM1**.

The symbols; **REAson1,2** as in **SEM1**, are called dummy relations in the sense that REAson1(2) has no logical significance because REAson1(2) holds in any combination of REAson and other concept, while REAson in FACT1 holds in the special combination of concepts like WAR with DISASTAR. They play a role to match syntagmatic relation with semantics in terms of syntax. These two processes analize the pragmatic, modal, and temporal information which is added into the factual structure to produce the conceptual structure.

"Dependency" is 2nd idea, to figure out that semantic (dependency) analysis of sentences is executable at the same time of syntactic (dependency) analysis. **ADG** employs dependency framework in a different way from the ordinary one. It deals with prepositions, postpositions, case inflections, grammatical functions, copula etc., as the functional features for relational interpretation. For example, preposition in English may not be a syntactic governor ('head' in this paper) of its object phrase, copula "be" in front of adjective modifies the syntactic feature of the adjective as a syntagmatic head predicate which allows it to have a dependent marked as a subject, while adjective in itself has a function of pre-nominal modifier. Namely, most of the functional words are dealt like case inflections. They add functional features to words or modify their features.

The functional features map word-to-word dependency to concept-to-concept semantic dependency. The figure 3 explains the simple interface mechanism. Functional features such as SUBject, OBJect, BECAUSE-OF corresponds to **REA**son1, **REA**son2, **REA**son respectively. The **ADG** syntactic dependency rules(see *s below) predict those semantic relations using the functional features and word syntax, and at the same time they trigger fact knowledge base inference to interpret Concept-to-Concept relations. A fact(concept) knowledge base is composed of such binary pieces as **S**s or **C**s. In this figure **S** and **C** mean semantic knowledge dependent on languages, and conceptual knowledge respectively.

Word/Concept	Function/Relation	Word/Concept
* war	subject	result-in
S WAR	**REA**son1	**REA**son
* war	object	result-in
S WAR	**REA**son2	**REA**son
* war	BECAUSE-OF	disaster
C WAR	**REA**son	DISASTER

Fig. 3 Syntactic dependency, dummy/conceptual dependency

ADG definition

D1. FEATURE describes morphological, syntactic, semantic, and conceptual information , and is used for describing the lexicon, semantic structure, conceptual structure and ADG rules. Feature is formalized as :

Feature Name . { **Feature Value** } . { **Context** }

Dependency function, one of the syntactic features for a particle , is described as follows.

LD.{NULL}.{A} LH.{NULL}.{A}
no word on the left depends on a particle. it depends on no word on the left

RD.{NOM}. A RH.{NULL}. A
it depends on NOMinal on the right
etc.

D2. CONCEPTUAL SYMBOL(CS) is a large set of intensional symbols standing for meanings conveyed by words. **CONCEPTUAL SYMBOL** includes those symbols such as NOTION, COMPUTER, GIVE, COLOR, BEAUTIFUL, SUP-SUB, PARTOF, AGT and so on. **CS** is one of the features included in **FEATURE.**

D3. THESAURUS is a system defined as a subset of:

CONCEPTUAL SYMBOL^2 **x SUP-SUB(PARTOF) relation**

D4. PTABLE is a system defined as a subset of:

CONCEPTUAL SYMBOL^2 **x CONCEPTUAL/dummy RELATIONs**

Relation symbols in PTABLE consist of 45 **CONCEPTUAL** relations except for SUP-SUB relation, and dummy relations such as REAson1, REAson2, LOC1, etc. **CONCEPTUAL RELATION** is a subset of **CONCEPTUAL SYMBOL: AGT relation , OBJ relation , POSSess relation, LOC relation and the other 41 relations.**

Relations are directed binary relations including logical ones such as REAson, CAUSAL, PARTOF, SUP-SUB, etc. and deep case relations such as AGT, OBJ, LOC, etc., and several language dependent dummy relations such as LOC1, LOC2 ,CNT1, REAson1 etc.

The **THESAURUS** and the **PTABLE**, which is described interms of semantic dependency and conceptual information, compose the fact knowledge base. The former forms directed network called an abstraction hierarchy for concept generalization.

CONCEPT SYMBOL

The CS(CONCEPT SYMBOL) differs from that of Schank's primitives in many respects. The number of **CS**s grows in proportion to the size of vocaburary as human cultivates new ideas and notions. The meaning of each **CS** is intensionally defined by Lambda**CS** COOCURR(**CS**,**CS**i,**CR**j). This model does not require to explain the reason why these **CS**s may be primitives and set up lexical rules for mapping Schank's semantic primitives to the corresponding words. That is to say, human can perceive the word concept only through observing which **CS**s and **CR**s CO-OCURR with logical and pragmatic functions. Each description of COOCURR(**CS**1,**CS**2,**CS**3) in the world model, where one **CS**i can be interpreted as **CR**, specifies the meaning Lambda**CS**i.

ADG rules are defined as feature-oriented.

D5. ADG: dependency rule for Legato.

(FEATURE1) + (FEATURE2) ----------▶(FEATURE3)

- **Head Selection**
- **Feature Inheritance**
- **Conceptual Relation Prediction**
- **Triggering Thesaurus/PTABLE**
- **Semantic Dependency Construction**

D6. contextual rule for Crescendo.

{**PATH**} ----------▶ { **PATH** }
PATH = FEATURE (dep/hed FEATURE)
(dep/hed :a dependency direction)

D7. Network structure is used for **INTERNAL REPRESENTATION:** semantic dependency structure and conceptual structure. **Network Structure** is defined as a subset of:

CONCEPTUAL SYMBOL^2 **x**{**45 conceptual relations, dummy relations** }

D8. Each lexical entry has its **KEY** and **CONTENT.** The **KEY** consists of WORD spelling and **CS.** The **CONTENT** is a set of **FEATURE**s . CS may be one piece of those conceptual FEATUREs .

Atomic formula in PTABLE and THESAURUS

Knowledge Base consists of **LEXICON, THESAURUS** and **PTABLE.**

The case grammar, as a basis of internal representation, which is constructed with the combination of binary case relations, fits the dependency grammar very well, since both dependency and case relation are basically binary. The dependency analysis also correlates to the atomic formula adopted for fact model specification. The formula has the following form, but not the ordinary predicate convention. The formula tells only the fact that three CSs (one may be CR) coocurr logically.

COOCURR (CSi , CSj , CSk)

This convention also implies some order-free calculation. The following example illustrates this kind of flexible function.

S11 An Apple existed on the table.

APPLE LOCation TABLE - - -F1

LOC(APPLE,TABLE).

S12 The location of an apple was the table.

LOC APPLE TABLE

eq (TABLE, LOC of APPLE) - - - F2
TABLE (LOC , APPLE) - - - F3

S22 Tom processed data.

HUMAN PROCESS DATA - - - F4

PROCESS(HUMAN , DATA)

S22 The agent of process was TOM.
(TOM is a process-or).

AGT PROCESS HUMAN

eq (HUMAN, AGT of PROCESS) - - -F5
HUMAN (AGT , PROCESS) - - - F6

Many kinds of formula can be set up for representing the above propositions . In our framework, the following unique representation format resolves the higher order difficulties, such as

F1&F3 = LOC(APPLE,TABLE(LOC,APPLE).
F4&F6 =PROCESS(HUMAN(AGT,PROCESS),DATA).

by using alternatives

COOCURR(APPLE, TABLE).
COOCURR(PROCESS,HUMAN,AGT).
COOCURR(PROCESS,DATA,OBJ).

Dependency grammar framework has been augmented as follows:

ADG functions

1. detects a possible pair of syntactic head and its dependent based on their FEATUREs,
2. predicts a set of permissible conceptual relations between them, using their pre- or post-positional features, phrase structural features, case structural features and so on,
3. triggers the knowledge base inference mechanism using their CSs in their conceptual information and the predicted permissible relations,
4. constructs their dependency structure using their FEATUREs if the knowledge base returns consistent semantic interpretation; in other words, if the consistent conceptual relation between their CSs is found.

Legato Implementation

Legato is a bottom-up dependency analysis engine (a kind of shift-reduce mechanism) based on the non-deterministic push-down automaton 2 , which is extended by devising context holding mechanism (context stack) to deal with exceptional dependencies (to be mentioned later).

The binary (augmented) dependency rule has a structure shown in Fig. 2. If the focused word (called **FOCUS**) and the word on the top of the push-down stack (called **Pd-TOP**) have the FEATUREs specified by the rule, a new HEAD with the derived FEATUREs is created by the action in the rule.

F O C U S + P d T O P ---→ A C T I O N

feature conditions	feature conditions	actions
for the focus word	for the push down stack top word	

Fig. 4 Legato rule form

In the case of Japanese,

1. Japanese sentences satisfy the non-crossing condition in syntactic dependency relation.
2. Moreover, the syntactic dependency relation coincides with the semantic and conceptual dependency relation in most cases.

However, the semantic dependency sometimes doesn't coincide with the syntactic dependency. In a worse case, even the non-crossing condition does not hold. The sample sentences in Fig. 5 exemplify such a linguistic phenomenon.

The non-crossing condition does not hold semantically in Ex. 2 and Ex. 3. Here in this figure, the solid lines indicate a syntactic dependency and the dotted lines indicate a semantic dependency. The arrows run from the head word to the dependent word.

A case of non-correspondence between syntactic and semantic dependency is shown in Ex. 2 (a1 & a2). although, w4 is recognized as w3's syntactic head, the true semantic head of w3 can be found among the words (w1 and w2) syntactically dependent on the word, w3. That is the word, w1. Furthermore, the crossing of a2 and a3 violates the non-crossing condition.

The context stack is a small push-down stack for keeping sub-context associated with the dependent words , and it is attatched to the

newly generated HEAD in order to bridge the gap between both kinds of dependencies. When Legato creates a new HEAD from Pd-TOP and HEAD, the context associated with Pd-TOP is stacked up onto the context stack in the new HEAD. At the same time, the semantic dependency is constructed between Pd-TOP and HEAD if it is permissible. Legato refers to the context in the context stack if needed, and then constructs the semantic dependency if the word which has a semantic dependency relation to the word stored within a context in the context stack can be identified.

This enables the analysis mechanism to easily deal with the sister dependency, which cannot done with in the traditional dependency grammar framework.

Crescendo implementation

The conceptual structure to be extracted as the final result of the comprehension process must be independent of the surface expression, while the semantic structure given by Legato may retain the inherited characteristics from the surface expression in the source language. If

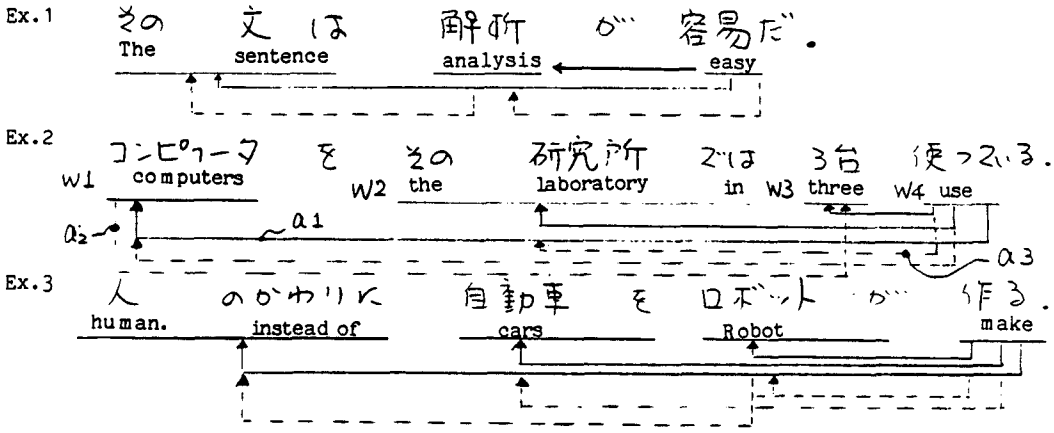
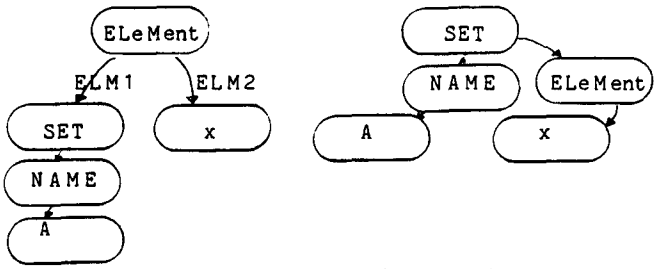


Fig. 5 Examples of the gap between syntactic and semantic dependency

a. Input sentence

X is an element of the set A.

b. Crescendo inference



CSs: 'ELeMent', 'SET', 'NAME', ('A' and 'x')
Conceptual Relations: 'NAME' and 'ELeMent'.
dummy relations: 'ELM1, 'ELM2'

c. Contextual Rule

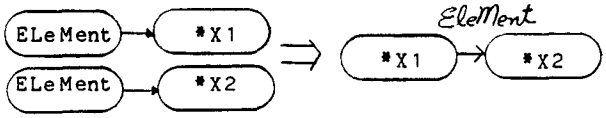


Fig.6 Crescendo diagram

the surface sentences express the same concepts, they must be organized into the same conceptual dependency structure.

In the semantic structure example given on the left in Fig. 6.b, the **CS** "ELeMent", which usually has two meanings (an object concept and a membership relation concept), functions as an object concept. It is reasonable, from a logical point of view, to regard the **CS** as a relation name in the conceptual structure , as shown on the right in Fig.6.b because 'SET -ELeMent - X' is easily deduced from the two propositions of 'ELeMent -ELM2 - X' and 'ELeMent - ELM1 - SET'. That is to say, the two sentences, like "The set A includes X" and "X is an element in the set A," must have the same conceptual structure.

Crescendo controls this kind of logical deduction neccessary for concluding the conceptual structure from the semantic structure. Besides conceptual and logical inference rules, it has causal inference rules among the facts for determing consistent causal chains.

Figure 6.c shows an example of the logical inference rules. It infers the right conceptual structure in Fig. 6.b from the left semantic structure. The knowledge based inference also assures the consistency of the deduced conceptual structures.

Conluding Remark

This paper has introduced a language comprehension model **ADG** to determine linguistic and semantic structures in sentences with a simple binary operation framework. The proposed dependency structure analysis engine (Legato) and the conceptual structure extraction engine (Crescendo) have been implemented. The **ADG** succeeded in constructively formalizing syntactic specification and semantic interpretation, using the knowledge base of a set of conceptual relations and the inference mechanism on it, defined only by simple binary operations.

Legato and Crescendo were incorporated in **VENUS** Japanese-to-English machine translation system. The experiments have proved its operational efficacy, fitness and justification.

The ADG points out anomaly in usual case systems, and resolves it by introducing the concept of dummy relation which can not and must not be interpreted logically. This extension puts the semantics of a linguistic theory in the correct position.

References

1. Gaifman, H., "Dependency System and Phrase Structure Systems, "Information and Control 8,304-337(1965).
2. Aho, A.V., Hopcroft, J.E. and Ullman ,J.D., "The Design and Analysis of Computer Algorithms," Addison-Wesley Publishing Co.(1974).