An English Analysis System of KMITT's MT Project

Ms. Monthika Boriboon
Mr. Booncharoen Sirinaokul
Asst.Prof.Woranut Kerdsinchai
Asst.Prof. Nuantip Tantisawetrat
Artificial Intelligence Center
King Mongkut's Institute of Technology Thonburi
Bangkok 10140, Thailand

1 Introduction

One important area of natural language processing which was a dream of human in 1950s is machine translation. At present, several universities and software houses have established natural language processing laboratories to develop machine translation programs. Some of them are available in the market and are widely used in many private and government sectors in many parts of the world.

In Thailand, research and development on machine translation is still in the early stage. In 1988, National Electronics and Computer Technology of Thailand (NECTEC) and Center of International Cooperation for Computerization of Japan (CICC) have signed an agreement to work on an international cooperative project on multilingual machine translation system. Other countries joining in this project are The People's Republic of China, Malaysia and Indonesia. The project will be ended in 1994.

The KMITT's MT project reported in this paper is one of the products of NECTEC-CICC Machine Translation Project.The background knowledge of the working process of the multilingual machine translation system is developed in order to respond to the aim of NECTEC that the machine translation technology should be expanded to increase the capability of researchers in computer field.

This project is only one part of a machine translation research in KMITT. It aims to build an English analysis system for analyzing the meaning of source language text input and represent its meaning in the form of internal structure, called interlingua. Some ideas on analysis dictionary, parsing techniques and internal semantic representation are adopted from PIVOT machine translation approach. The result of this study shows the possibility of using the artificial intelligence technique in constructing the practical machine aids translation system in the future.

2 Analysis Grammar

The grammar used in this analysis system includes:

2.1 Morphology

The morphology is needed in morphological analysis. In general, the scheme of English word is in the form

$$WORD = PREFIX^{*} + ROOT + SUFFIX^{*}$$

(* known as Kleen star which indicates that the item can be repeated zero or more times)

This given rule generates quite a number of pattern, such as

Ø + ROOT + SUFFIX
Ø + ROOT + 0
Ø + ROOT + SUFFIX + SUFFIX

and so on (The zero indicates a null items). Consequently, the morphology used in morphological analysis is grammar on suffixes and prefixes such as

-Plural Ending : -s, -es, -ies, ives, ss
-Tense Ending : -d, -ed, -ing, -en
-Comparative Ending : -er, -est, -ier
-Adverbial Ending : -ly

2.2 Phrase Structure Grammar

The phrase structure grammar consists of a list of possible construction stated in terms of the constituents that a sentence can have. The phrase structure rules used in syntactic-semantic parsing process are as follows:

Rule	1.	S	-->	NP VP NP
Rule	2.	S	-->	NP VP (PP)
Rule	3.	NP	-->	(<DET>) N* ¦PRON
Rule	4.	NP	-->	(<DET>) (ADJP) NP
Rule	5.	NP	-->	NP PP
Rule	6.	PP	-->	<PREP> NP
Rule	7.	VP	-->	(<AUX>)* (<NEG>) VP1
Rule	8.	VP1	-->	(ADJP) V (ADV)
Rule	9.	ADJP	-->	(ADV) ADJ

* means that the constituent can occur recursively.
<> means that the inside element is treated as a feature of the head of the phrase.
() means that the constituent is optional.

2.3 Dependency Grammar

The dependency grammar is used in the construction of syntactic-semantic representaion to specify the controller term, called head and the controlled term, called depender. This grammar can express as the following expressions :

1.^(V)	verb can appear with no controller thus verb can locate at the top of the hierarchy.
2. V(NP,^,NP)	verb controls the left and right noun phrase of the verb.
3. N(<DET>,^,PP)	noun controls the left determiner and the following preposition phrase of noun by keeping the left determiner as a feature of the noun.
4. N(ADJ,^)	noun controls the left adjective of noun.
5. V(<AUX>,^)	verb controls the left auxiliary verb by keeping it as a feature of the verb.
6. V(^,PP)	verb controls the following prepositional phrase.
7. V(ADV,^ADV)	verb controls the left and right adverb of the verb.
8. V(ADJ,^)	verb controls the adjective phrase in front of the verb.
9. ADJ(ADV,^)	adjective controls left adverb.

2.4 Case Grammar

A case expresses the semantic role that a noun phrase plays with respect to the verbs, adjectives or other nouns around it. In this paper, a list of case has been defined for expressing the semantic relation between words such as AGENT, OBJECT, CAPACITY, NUMBER, LOCATION etc. Base on the Case Grammar, a main verb is treated to be the main focus of the other phrases.

3 Parsing Technique

This syntactic-semantic parser is designed to process a sentence input by using both deterministic top-down and bottom-up parsing. The deterministic bottom-up parsing is used in the constructions of noun phrases, adjective phrases and verb phrases. The constructions is started by considering the surface words and then grouping them into phrase structures. On the other hand, the deterministic top-down parsing is used in the sentence construction by starting from the main verb and find its surrounding arguments. Both cases are carried out with no backtracking since the the parsing is deterministic.

4 Procedural Components and System Design

The procedural components of the system are morphological analyzer, syntactic-semantic parser and conceptual interpreter which function respectively.

The system design is based on an artificial intelligence technique. The processor of the system is divided into two main parts which are knowledge base and inference engine. The knowledge base describes the operations corresponding to the associative subsystem. The inference engine controls the knowledge based processing. This engine composes of two parts. The first is knowledge based compiler which functions as rule command interpreter and the second is rule inferencing system which controls the sequence of rule usages. The advantage of this system design approach is that it is easy to maintenance the the knowledge.

Before the operation in the system is carried out, the knowledge engineer has to put the linguistic knowledge, written in a designed language, into the knowledge base. This knowledge is compiled by the knowledge based compiler and stored in the linked list structure.

In addition to the knowledge base, the analysis system also utilizes the knowledge contained in analysis dictionary. The information from the analysis dictionary will be loaded and attached to each entry of the linear structure in morphological stage.

The block diagram of the designed system is shown below:

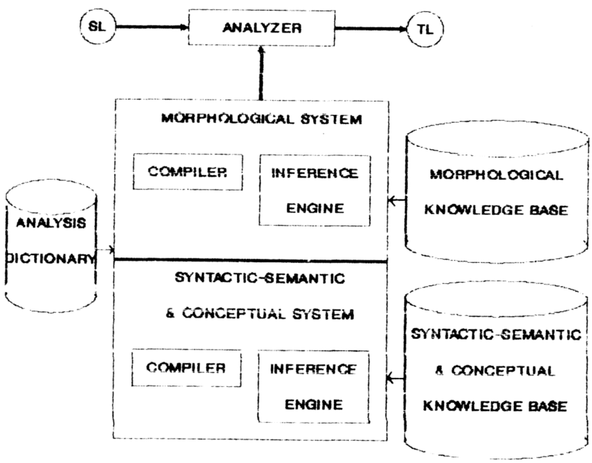

Fig 1. The block diagram of the designed system

5 Analysis Dictionary

5.1 Dictionary Information

The necessary information for analysis process is contained in the analysis dictionary. This information is classified into three parts :

(1) **Lexical Part.** This part contains

Lexical Entry : The lexical entry refers to the root of the word.

Adverb Particle: If the encountered English verb exists with prepositional form and the meaning of the verb is changed, it is classified as an adverb particle.

(2) **Syntactic Part.** This part contains

Part of Speech called Category : noun,verb, preposition, adjective etc.

Subcategory : common noun, proper noun etc.

Verb Pattern: (SUB V DOB), (SUB V DOB ADP) etc.

(3) **Semantic Part.** This part contains

Syntactic-Semantic Mapping : (SUB = AGT, DOB=OBJ) etc.

Word Hierarchy:It is abbreviatedly called AKO

(a kind of semantic class): It takes an account of disambiguation of homonyms. AKO used in this paper adopts from PIVOT system.

Lexicon Concept: The lexicon concept is the symbolic name created to represent the meaning of lexical entry.

5.2 Data Structure of Analysis Dictionary

The analysis dictionary is created by using program DBASE. The entries are arranged in alphabetical order. The created dictionary is compiled and kept in the linked list structure which can improve search time. The data structure of dictionary can be shown below:

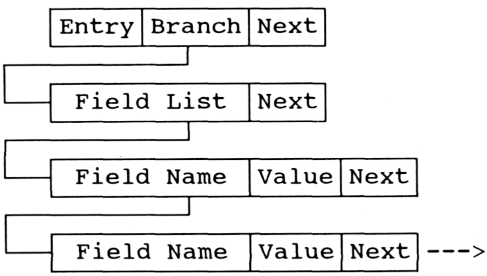

Fig. 2 The data structure of analysis dictionary

6 Knowledge Based System Architecture

The type of architecture of the knowledge based *system is a production system divided into two parts:* rule-base and inference engine.

6.1 Rule-Base

In the morphological knowledge base, the rule-base *file contains the knowledge about morphology. In the* syntactic-semantic knowledge base,the rule-base file contains the knowledge about premodifiers and postmodifiers of nouns, verb phrases, obligatory cases, and the optional cases. In the conceptual knowledge base, the rule-base file contains the knowledge for the mapping the syntactic cases into semantic cases. These kinds of knowledge are encoded in the form of production rules which are typically based on the implication : IF-THEN based representation. These rules are ordered according to the sequence of rule usages provided by analysis rule writer.The rule traverse is "the depth first search". The representation of each rule in the production sets is analogous to LISP language.

6.1.1 Types of Rule Structure

The types of rules are classified as the followings:

Single Structure

```
   Type :
      (Cond  Act)
   Description :
      If  Cond  then  Act
```

And Structure

```
   Type :
      (Cond1(Cond2...(CondN  Act))...)
   Description :
      If Cond1 and Cond2 and Cond3
            ...and CondN then Act
```

Case Structure

```
  Type :
     (Cond (cond1  Act1)
           (Cond2  Act2)
               ...
            (CondN  ActN)
     )
   Description:
     If Cond and Cond1 then Act1
     If Cond and Cond1 then Act2
          ...
     If Cond and CondN then ActN
```

Mixed Structure

```
    Type :
      (Cond(Cond1(Cond11
                      (Cond111...(Cond111...1 Act1)...)))
            (Cond2(Cond21
                      (Cond211...(Cond211...1 Act2)...)))
             ...
            (CondN(CondN1
                      (CondN11...(CondN11...1 ActN)...)))
      )
    Description :
      If Cond  and Cond1 and Cond11...
                                 and Cond111...1 then Act1
      If Cond  and Cond2 and Cond21...
                                 and Cond211...1 then Act2
             ...
      If Cond  and CondN and CondN1...
                                 and CondN11...1 then ActN
```

6.1.2 Syntax of Functions in Rule-Base Writing

The examples of functions in rule-base writing are as the followings :

Morphological Knowledge Base :

Synopsis : (#function.n = v)
Description: test whether the n characters from the left or the right (depending on the specified function) of considering word has value v (In case of negation, the sign "=" is replaced by <>).

synopsis : (#funtion.n)
Description: do the action at the n position from the left or the right (depending on the specified func tion) of the considering word.

synopsis : (#funtion.d, v)
Description: do the action at the field "d" with value v

Syntactic-Semantic and Conceptual Knowledge Base:

Synopsis : (#function.n,v)
Description: test or do the action according to the specified function at the node n with value v

Synopsis : (#function.n)
Description: do the action at the next n node

6.2 Inference Engine:

The inference engine functions as the rule interpreter, called compiler. It also controls the sequence of knowledge usage provided in the rule-base file. This controlling is accomplished by constructing the chains of inference.

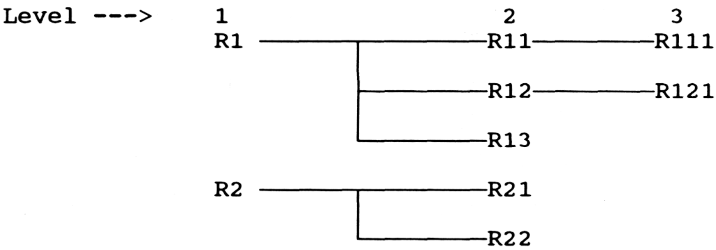

Fig.3 Example of the chains of inference.

6.3 Data Structure of Rule-Base

The rules in the designed knowledge base which are interpreted and ordered are in the form of linked list structure with the following components :

Function:storing the required testing action
Value :storing the required truth value
Result :storing the logical result of the testing
Action :storing the action which must be accomplished if the condition is true
Level :storing the level of the rule traversing
Next :storing the pointer to the next rule

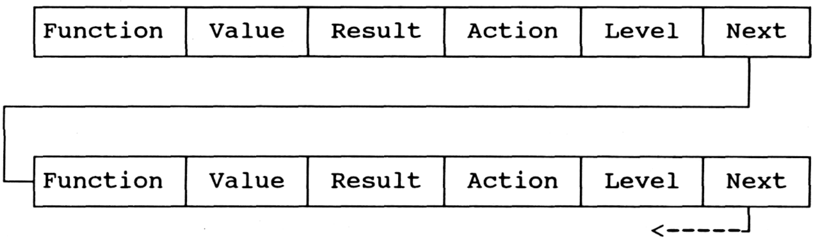

Fig.4 The data structure of Rules

7 Interlingua

An interlingua is a symbolic representaion defined to represent the meaning of an input source language text. The components of the interlingual structure are the concept symbols and relation symbols. These symbols may exist with their features such as progressive aspect,tense etc.The representation is in the following form :

*CP{Fea}([CASE{CFea}]>CP{Fea},[CASE{CFea}]>CP{Fea}).

*	=	root node or entry for traversing
CP{Fea}	=	a word concept with the feature FEA
(	=	the marker of the next level depender
)	=	the marker of the ending depender with the same level
[Case{Fea}]	=	the relation of two concepts, the {} indicates the relaion feature such as LOC{at},EAT{past,prog} etc.
>	=	node in front of this sign is the head of the following node
,	=	the branches conjuncted by this sign is in the same level of the tree
.	=	the end of the tree

8 Morphological Analyzer

The morphological analyzer accepts the input sentence as a string of characters and breaks it into lexical items. Then the analyzer will represent these items as a linear structure. The morphological analyzer also identifies the regular forms of word inflection. Finally, the information from the analysis dictionary is retrieved and attached to each word.

8.1 Procedures in Morphological Analyzer

The process can be divided into three steps as follows :

8.1.1 Input Loading

The morphological analyzer accepts the English sentence string in the following forms

$$w_{11}w_{12}w_{13}...w_{1p}\ w_{21}w_{22}w_{23}...w_{2q}\ ...\ w_{m1}w_{m2}w_{m3}...w_{mr}$$

where w_{ij} = any jth character in word i

8.1.2 Word Boundary Determination

The English sentence input string is initially broken by null character or special characters such as ";", ",", ".", ":", etc.

8.1.3 Lexical Units Determination

The output from the word boundary determination may be in the inflectional forms. The goal of this step is to isolate the possible lexical entries of the words by consulting the analysis dictionary. Then all sets of information associated to these lexical entries are retrieved from the dictionary and attached to the words. The lexical entries of the input sentence are ordered as a linear list structure.

$$(w_{11}w_{12}...w_{1n},\ w_{21}w_{22}...w_{2n},\ ...,\ w_{m1}w_{m2}...w_{mn})$$

The algorithm for isolating the lexical entries is basically follows the algorithm of Marry Dee Harris[3]

```
        repeat
             look for word in dictionary;
             if not found
                then modify the word;
          until  word is found or no futher possible
                modification.
```

The outline below illustrates some examples of multi-ple-step suffix removals. After each step the analysis dictionary would be checked again for the resulting string of characters. (The sharp sign ¦#¦ identifies the final form located in the analysis dictionary.)

1. -s ending (bears, dishes, stories)
   a) take off -s (#bears, dishe, storie)
   b) take off -e (#dish, stori)
   c) change -i to -y (story)
2. -ed endings (lived, opened, tried)
   a) take off -d (#live, opene, trie)
   b) take off -e (#open,tri)
   c) change -i to y (#try)
3. -ing endings (hoping, talking)
   a) take off -ing and add -e (#hope,talke)
   b) take off -e (#talk)

Other frequent endings , namely -er, -est, -ly,en are also isolated. The output of this step can easily be described by graphical representation below:

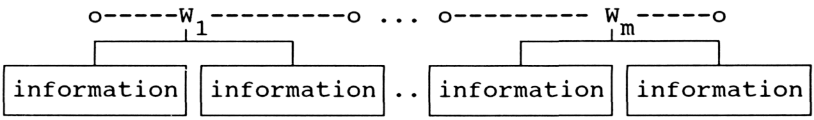

Fig.5 Output of Morphological Analyzer

8.2 Data Structure of Output of Morphological Analyzer

The computer representation of the linear list structure from lexical item determination is a linked list structure as shown below :

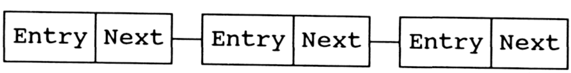

Fig.6 The data structure of the morphological output

Entry = storing the lexical item
Next = storing the pointer to the next entry

The data structure of each entry contains sublinked list to the data structure retrieved from analysis dictionary as shown in Fig.2

9 Syntactic-Semantic Parser

The syntactic-semantic parser accepts the linear list structure of lexical entries from morphological analyzer to construct the syntactic-semantic tree structure. The construction is accomplished by using Dependency Grammar, Phrase Structure Grammar, Case Grammar and information from the analysis dicionary. This tree structure composes of nodes storing information of lexical entries and semantic relations. The semantic relations are immediately assigned between the two nodes of a noun and a noun or a verb and its free arguments. However, a verb and its obligatory arguments will be passed through the process of assigning syntactic relations first. The syntactic disambiguaion is also handled in this step.

9.1 Procedures in Syntactic-Semantic Parser

The procedures in this syntactic-semantic parser are explained respectively as follows:

9.1.1 Noun Phrase Construction

A noun in English language may exist with premodifiers or postmodifiers. Hence, the noun phrase construction is divided into two steps. The first is premodifier noun construction and the second is postmodifier noun construction. In this noun phrase construction, adjective construction will be handled first by using rule 9 (see the section of analysis grammar). The noun phrase construction follows the rule 3, 4 and 5.

9.1.2 Verb Phrase Construction

A verb phrase construction in this paper is conducted by using rule 7 and 8 that is, a verb phrase is composed of a main verb and its surroundings which may be an auxiliary verb, negation,an adjective phrase or an adverb. In case of the auxiliary verb or a negation, it is treated as a feature of a main verb.

9.1.3 Sentence Construction In this step, the *remaining lexical entries in the linear structure* composes only of the head of a verb phrase and the head of a noun phrases. The noun phrase may appear in the position in front of the verb or after the verb or both.The sentence construction is accomplished by using the rule 1,2 and 6. The parser will take a verb pattern to construct a sentence and assign syntactic relation between the head verb and its obligatory arguments. The parts which are not the obligatory arguments of verb will be handled as free arguments and the semantic relations between the head verb and its free arguments are assigned.

The output from the syntactic-semantic parser is shown below:

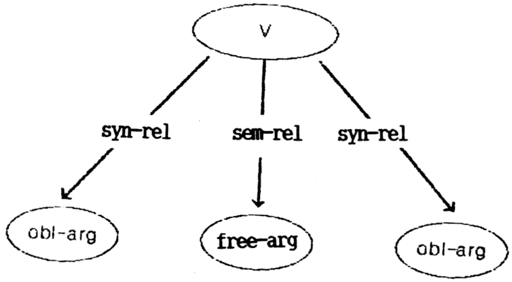

Fig.7 Output of Syntactic-Semantic Parser

9.2 Data Structure of Output of Syntactic-Semantic Parser

The data structure of output from this parser is analogous to the data structure of output from morphological analyzer. But there is a special node, called a case node, in this data structure. This node keeps information about a case relation, a feature, an arrow representing the direction of a depender, a pointer to the head node and a pointer to the depender node.

CASE	FEATURE	ARROW	HEAD	DEPENDER

Fig.8 Case node Structure

10 Conceptual Interpreter

The conceptual interpreter is designed to convert a syntactic-semantic structure into a conceptual structure. It also functions in disambiguation of the concept of lexical entry that still remains.

10.1 Procedures in Conceptual Interpreter

The conversion of syntactic-semantic structure into conceptual structure is carried out by using rules of mapping syntactic-semantic relation to change the relation between a verb and its obligataory arguments to semantic relation. The rules of mapping syntactic-semantic relation is in the analysis dictionary and exists in the verb entry. The conceptual interpreter also functions in disambiguation of the concept of lexical entry that still remains.

10.2 Data Structure of Output of Conceptual Interpreter

The data structure of this interpreter is the same as data structure of syntactic-semantic parser.

11 Disambiguation

Disambiguity is a significant problem in analysis process. The disambiguation in this paper is handled in two levels

11.1 Syntactic Disambiguation

The grammatical rule involved a phrase construction is used in this level. The syntactic disambiguation is managed in parallel to the syntactic-semantic parsing.

11.2 Semantic Disambiguation

If the syntactic disambiguation can not disambiguate the homonymes, the semantic information such as AKO will be used. This level will be processed both in syntactic-semantic parsing and conceptual interpreting.

12 Results and Conclusions

The designed analysis system can handle the English simple sentence input in the domain of 130 lexicons with 250 concepts and 7 structure patterns. The domain of analysis is expandable by expanding the dictionary and knowledge base. The programme uses approximately 250 kbytes of memory units.

13 Acknowledgements

The writers gratefully thank to the researchers of Machine translation Laboratory of King Mongkut's Institue of Technology Thonburi for all aspects of assistance given to the study. We also thank to Miss Wantanee Panthachat for her advice. Finally, we would like to thank The Bureau of the Budget of Thailand for the contribution.

14 Reference

1. Allen James (1987) **Natural Language Understanding.** The Benjamin Cummings Publisihng Inc. California, 24-34,41-73.
2. Gazdar Gerald and Chris Mellish (1986) **Natural Processing in LISP.** Addison-Wesley publishing Company. New York, 149-176,215-219.
3. Harris Mary Dee **Introduction to Natural Language Understanding**, 93-113.
4. Rieinsdijk, Henk Van and Edwin Williams (1978) **Introduction to The Theory of Grammar.**

The MIT Press.Cambridge, 6-12,34-57,225-257.
5. Nirenburg Sergei (1987) **Machine Translation (Theoretical and Methodological Issues.** Cambridge University Press. London, 1-13, 42-58, 90-110, 278-292.
6. Samad Tarq (1986) **A Natural Language Interface for Computer-Aided Design.** Kluwer Academic Publishers. Boston,8-10, 13-20, 39-61.