ATLAS: Automatic Translation System

• Hiroshi Uchida • Tatsuya Hayashi • Hiroshi Kushima

(Manuscript received March 20, 1985)

Due to the rapid advancement of both computer technology and linguistic theory, machine translation systems are now coming into practical use.
Fujitsu has two machine translation systems, ALTLAS-I is a syntax-based machine translation system which translates English into Japanese. ATLAS II is a semantic-based system which aims at high quality multilingual translation. In this paper, both the ATLAS-I and ATLAS II translation mechanisms are explained.

1. Introduction

In 1984 Fujitsu marketed the automatic machine translation systems, ATLAS-I and ATLAS II.

ATLAS-I is the world's first commercial English-Japanese translation system, the prototype of which was completed in 1982. Since then, it has been used experimentally by Fujitsu and selected users. The acquired experience has served to improve the ATLAS system's dictionaries and grammatical rules.

A primary aim in the development phase has been to achieve harmonious interaction between man and machine. For this reason, we rejected the "all or nothing" approach, and adopted a "second best" approach. Here, a translation, although rarely perfect is produced swiftly. The text can then be edited up to native speaker quality. This method is both fast and cost-effective.

ATLAS II aims at multilingual translation of three or more languages. At present ATLAS II translates only Japanese-to-English.

This system treats source text analysis and target generation as separate tasks. This feature can be used for translating other languages. Another characteristic is the use of conceptual structure as a bridge between source sentence area and target sentence. Conceptual structure is language-independent, and is expressed as a semantic network. When the conceptual structure is too dependent on the source sentence, it is converted to another structure which will be more likely to produce a target sentence. Knowledge, including common sense, is expressed as a semantic relationship between concepts in the same way as conceptual structure.

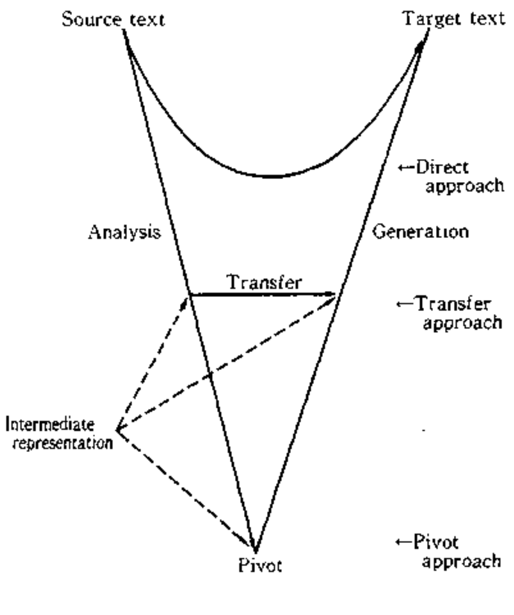

Fig. 1–Translation approach.

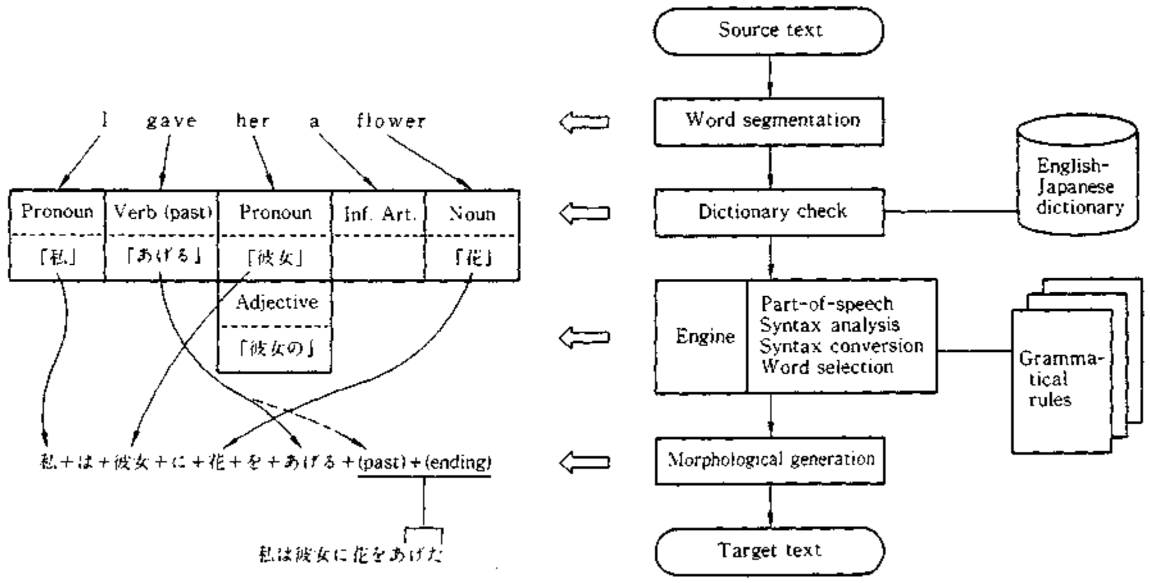

Fig. 2–Translation process of ATLAS-I.

2. Translation Approach

There are three approaches to translation: transfer, direct, and pivot (see Fig. 1). The transfer approach is the most widely used of the three. In this approach, each language to be translated has its own intermediate representation. The translation process consists of three steps: intra-language conversion (analysis) from source text to intermediate representation; inter-language conversion (transfer), intra-language conversion (generation) from intermediate representation to target text.

The direct approach converts directly from the source language to the target language without any intermediate representation. In this approach, three conversion steps are performed simultaneously. In other words, only intra-language conversion is performed.

The pivot approach, uses only intra-language conversion. This approach features intermediate representation for every language. The source language text is converted to the intermediate representation (pivot) which is then converted to the target language. Inter-language conversion is not performed.

The translation quality depends on the amount of information available, not on the translation approach. The translation approach is selected according to the purpose of the system. The direct approach is appropriate for a specific language pair: the pivot approach for multi-language pairs.

3. ATLAS-I

3.1 Outline

ATLAS-I uses a syntactic direct approach with some semantic analysis. Its characteristics are:

1) Processing is swift.
2) Grammatical rules can be written easily.
3) The syntax trace of a source text can be reproduced in the translation.

The ATLAS-I translation process is as follows (see Fig. 2):

First, an input sentence is segmented into words. At this stage, designated characters such as blanks hyphens and virgules separate the segments.

Next, each word is found in the dictionary, which contains parts of speech, equivalents, and other features. Each word with its corresponding features is concatenated into a string of nodes.

Then, rewriting rules are applied to this node string for part of speech decisions, syntactic analysis, syntactic conversion, and equivalent selection. These operations are performed simultaneously from the bottom until all adjacent nodes have been concatenated into one.

Finally, inflections are generated using the inflection table of declinable parts of speech in Japanese; punctuation is generated or deleted, and spaces are inserted.

3.2 Rewriting rules

The rewriting rules consist of condition, generation and exit routines.

In the condition field, up to nine node conditions can be specified. Node conditions may be main categories or subcategories. A main category indicates parts of speech and related features (number for a noun; tense for a verb). A subcategory indicates features not classified in the main categories. For example, semantic and syntactic features for a noun, surface-structure pattern for a verb. Both main categories and subcategories specified in the dictionary can be added to or deleted from the rewriting rules.

In the generation field, an arbitrary number of nodes can be specified. Main categories for a node to be generated are defined in this field. Subcategories specified in the condition field can be inherited, or more subcategories can be specified here.

When the conditions for a rewriting rule have been satisfied, the node string specified in the condition field is replaced by a new node string specified in the generation field. Here, words corresponding to each node are rearranged according to the specification in the generation field.

Specific rules control the execution sequence of the exit routine.

3.3 Backtracking

When two or more rules satisfy certain conditions at the same time, the rule with the highest priority is executed, and the current node string is reserved. Should execution fail, the reserved node string is recovered from the stack and the rule with the next highest priority is executed. Backtracking can be activated by a rule for a specific condition.

3.4 Dictionary

The dictionary is bilingual English-Japanese where each English word is keyed to its Japanese equivalents. The basic dictionary contains about fifty three thousand English words.

Each entry contains several parts of speech; each part of speech has one main category, one standard equivalent word, and an arbitrary number of subcategories. A subcategory indicates syntactic and semantic features of the entry word, and has a subequivalent word which is used for word selection if necessary (see Fig. 2).

In addition to the basic dictionary, a technical dictionary and a user-oriented dictionary are available.

3.5 Translation Failure

Translation fails when a string of several nodes cannot be concatenated into one node by applying rewriting rules. In this case, however, an interim result is output based on the final state of the node string. This result can then be edited to produce a correct translation.

3.6 Editor

Since perfect translation is rarely achieved, pre-editing, post-editing and manual use of the dictionary are often required to use the system effectively.

ATLAS-I provides users with an exclusive translation editor and a dictionary editor.

3.6.1 Translation editor

The translation editor provides an editing screen which is divided vertically into two parts. The left half of the screen displays the source text (English); the right half displays the target text (Japanese).

Both texts are displayed sentence by sentence; corresponding sentences are displayed on the same line.

Sentences can be translated one by one, or a specified number of sentences can be translated at the same time. The source text and target text can be edited independently for effective pre-editing and post-editing.

To assist the system, the user can specify separation or conjoining of sentences and the sentence type on the screen. Users can also call the dictionary editor to modify the contents and translate to confirm the result.

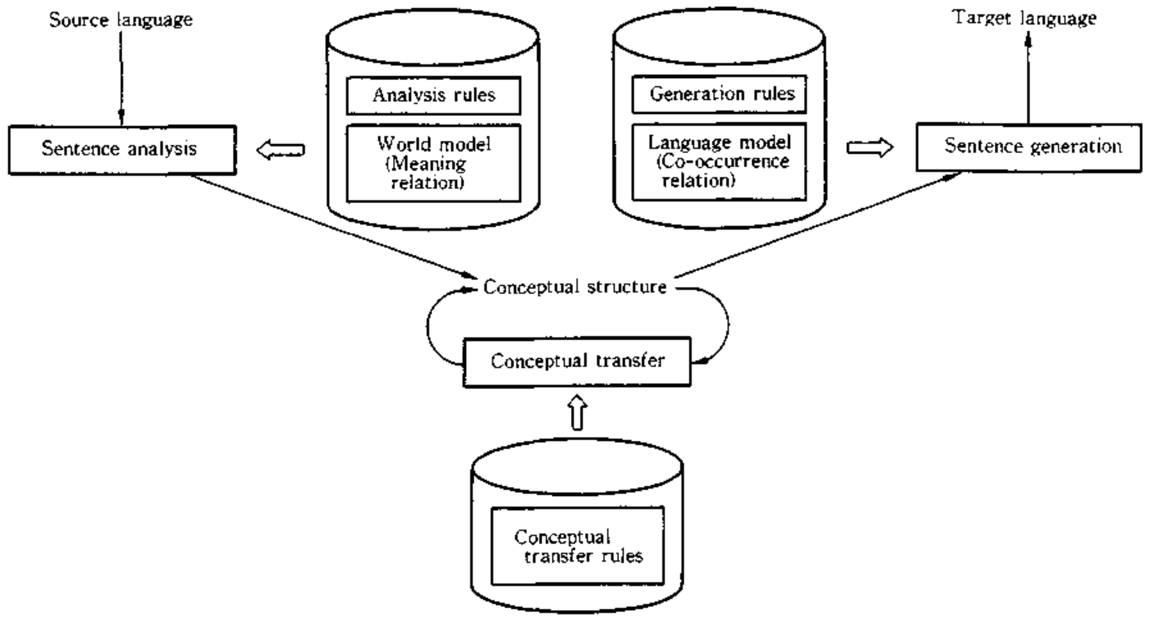

Fig. 3–Translation process of ATLASII.

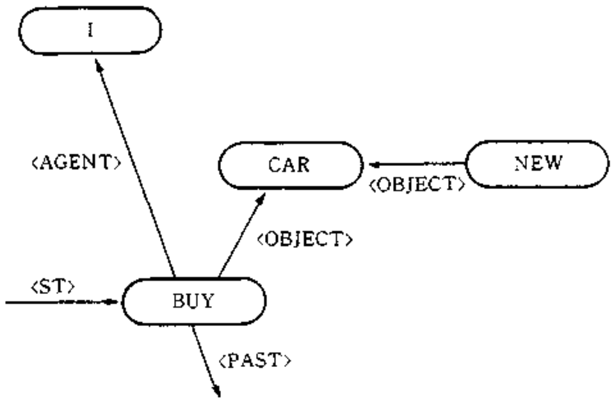

Fig. 4–Conceptual structure.

3.6.2 Dictionary editor

The dictionary editor provides four screens in this order:

1) Dictionary selection
2) Entry selection
3) Part-of-speech selection
4) Definitions classified by part-of-speech.

4. ATLAS II

ATLAS II aims to imitate human translation, understanding a sentence written in one language, then expressing it in another. Any natural language is created on the assumption that every person is able to understand a sentence from the context and the meaning of the component words. Language's syntactic regulations are also made on this assumption. To be able to translate naturally a computer should be able to do this.

Humans have their own world models, formed from linguistic knowledge, common sense, cause-effect relation, and human characteristics. This is why humans can perform both semantic and contextual analysis with ease. The world model can be extended by inference, or specialized according to the context. Humans also have a language model which guides our use of each word.

ATLAS II is equipped with both a world model and a language model (see Fig. 3).

The world model is expressed as a semantic relation between concepts; the language model as a co-occurrence relation between words. Grammatical rules for analysis and generation, and transfer rules are provided for modeling the human translation process.

The conceptual structure is a semantic network representation of an input sentence. Figure 4 shows the conceptual structure which is equivalent to, "I bought a new car". The network consists of nodes and arcs: a node denotes a

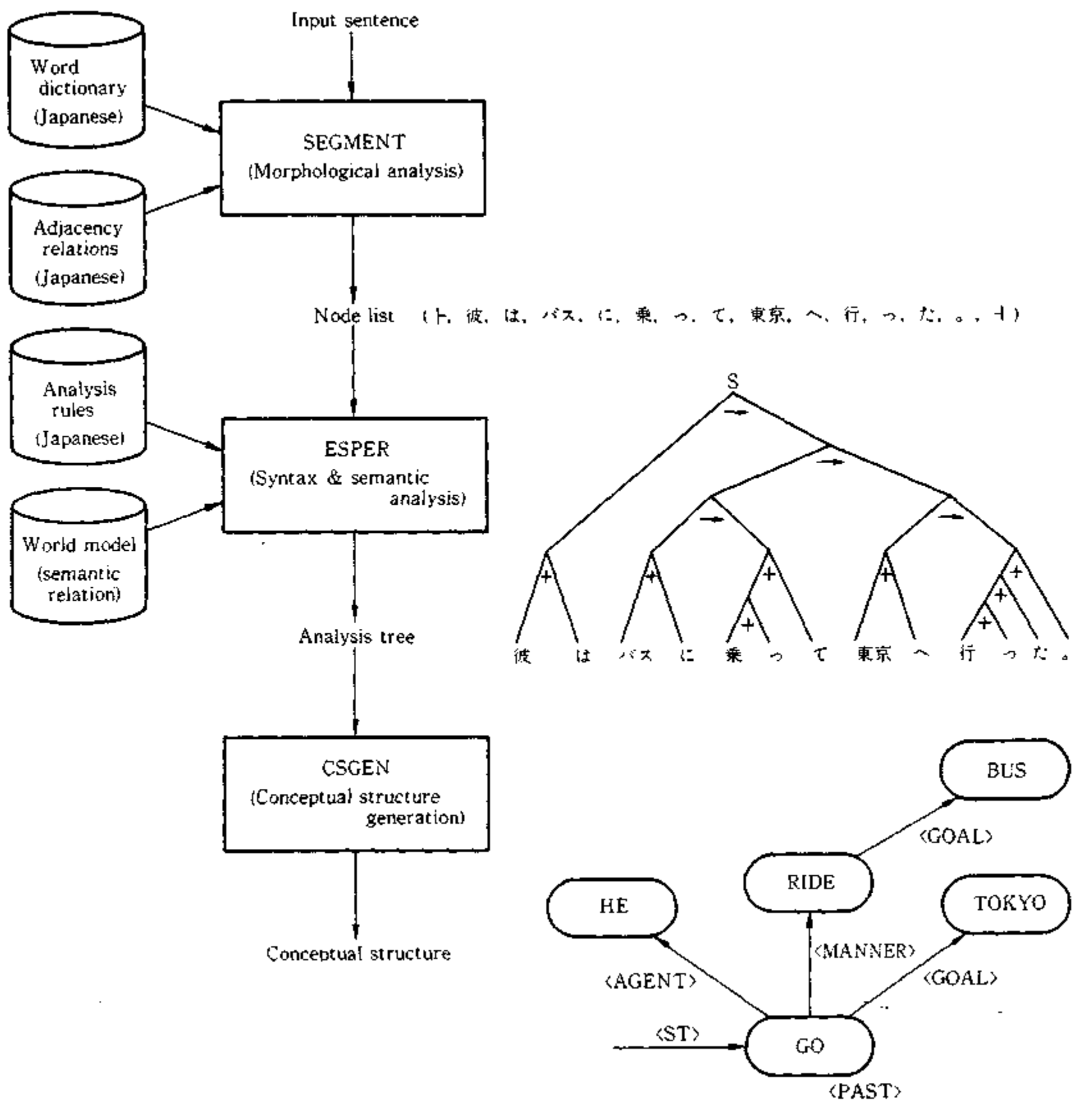
Input sentence
Word dictionary (Japanese)
Adjacency relations (Japanese)
SEGMENT (Morphological analysis)
Node list (ト, 彼, は, バス, に, 乗, っ, て, 東京, へ, 行, っ, た, 。, ┤)
Analysis rules (Japanese)
World model (semantic relation)
ESPER (Syntax & semantic analysis)
S
彼 は バス に 乗 っ て 東京 へ 行 っ た 。
Analysis tree
CSGEN (Conceptual structure generation)
Conceptual structure
BUS
<GOAL>
RIDE
HE
TOKYO
<MANNER>
<AGENT>
<GOAL>
<ST>
GO
<PAST>

Fig. 5–Analysis process of ATLASII.

concept representing the meaning of the words "I", "BUY", "CAR", "NEW"; an arc denotes the deep case relation such as < AGENT >, < OBJECT >, and the junction relation such as < CAUSE >, < SEQUENCE >. In addition to the above binary arcs there are unary arcs which indicate additional information such as tense, aspect and style. In Fig. 4, < PAST > indicates tense and < ST > indicates focus.

The system understands an input sentence in the form of conceptual structure. Humans understand a sentence by using their knowledge. ATLAS II refers to its world model in the same way as humans. The world model defines every relation between concepts. For example, the knowledge, "Birds fly." is expressed in the world model as follows.

(BIRD, FLY, < AGENT >) = TRUE

The left-hand side indicates a conceptual structure where an arc < AGENT > conjoins node "BIRD" and node "FLY". This is found to be true by referring to the world model. The system checks whether the conceptual structure is included in the world model. If it is, the system accepts it; if it is not, the system rejects it and asks for another result of sentence analysis.

Relationships between concepts should be as universal as possible. But it is not possible to apply this to all concepts, because each language is to some degree, unique. As a result, a con-

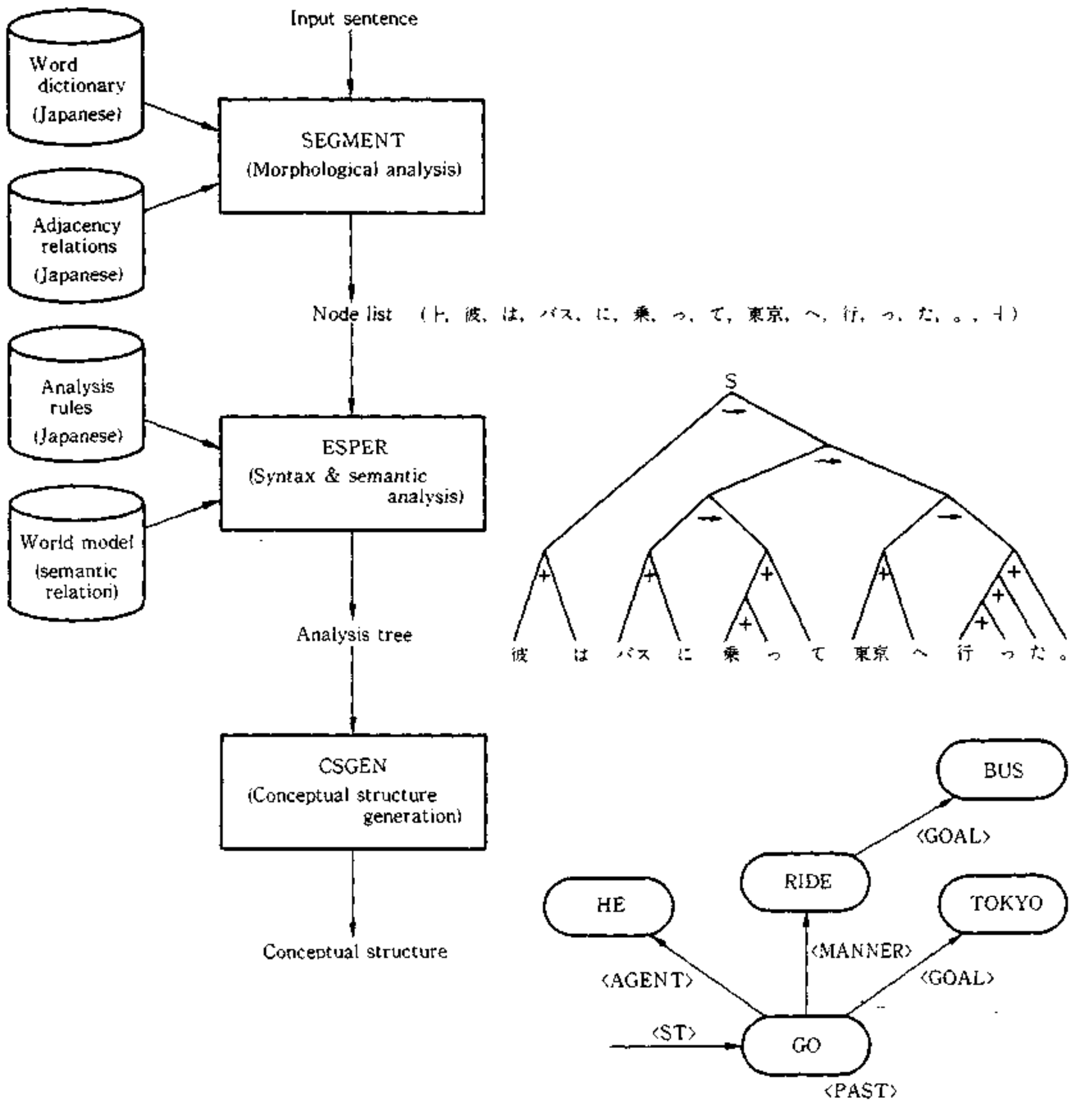
Input sentence
Word dictionary (Japanese)
Adjacency relations (Japanese)
SEGMENT (Morphological analysis)
Node list (⊢, 彼, は, バス, に, 乗, っ, て, 東京, へ, 行, っ, た, 。, ⊣)
Analysis rules (Japanese)
World model (semantic relation)
ESPER (Syntax & semantic analysis)
S
彼 は バス に 乗 っ て 東京 へ 行 っ た 。
Analysis tree
CSGEN (Conceptual structure generation)
Conceptual structure
BUS
<GOAL>
RIDE
HE
TOKYO
<MANNER>
<AGENT>
<GOAL>
<ST>
GO
<PAST>

Fig. 5—Analysis process of ATLASII.

concept representing the meaning of the words "I", "BUY", "CAR", "NEW"; an arc denotes the deep case relation such as < AGENT >, < OBJECT >, and the junction relation such as < CAUSE >, < SEQUENCE >. In addition to the above binary arcs there are unary arcs which indicate additional information such as tense, aspect and style. In Fig. 4, < PAST > indicates tense and < ST > indicates focus.

The system understands an input sentence in the form of conceptual structure. Humans understand a sentence by using their knowledge. ATLAS II refers to its world model in the same way as humans. The world model defines every relation between concepts. For example, the knowledge, "Birds fly." is expressed in the world model as follows.

(BIRD, FLY, < AGENT >) = TRUE

The left-hand side indicates a conceptual structure where an arc < AGENT > conjoins node "BIRD" and node "FLY". This is found to be true by referring to the world model. The system checks whether the conceptual structure is included in the world model. If it is, the system accepts it; if it is not, the system rejects it and asks for another result of sentence analysis.

Relationships between concepts should be as universal as possible. But it is not possible to apply this to all concepts, because each language is to some degree, unique. As a result, a conceptual structure produced by analyzing a Japanese sentence may remain Japanese to some extent; consequently, this structure may not be appropriate for English translation. For example, the sentence "Ningen niwa zunou ga aru." would ideally be translated into "Man has a brain". To do this, conceptual transfer is required; if not, the literal translation "There is a brain in man." will be produced.

Conceptual transfer is performed between conceptual structures: from the source language-dependent one to the target language-dependent one. The conceptual structure interface guarantees complete separation between analysis and generation. The pivot approach serves for almost all translations and the transfer approach is used only for specialized ones, allowing a minimum number of transfer rules. As a result, this system is appropriate for multi-language translation.

4.1 Analysis Process

The sentence analysis section analyzes an input sentence and expresses its meaning as a conceptual structure in the form of semantic network. This section consists of three modules; SEGMENT for morphological analysis; ESPER for syntactic and semantic analysis; CSGEN for conceptual structure generation. This section uses the word dictionary, word adjacency relations, analysis rules, and semantic relations. Figure 5 shows how each module uses the dictionaries and rules and the forms of processing results.

SEGMENT extracts words from the input sentence and produces a node list for analysis. ESPER receives the node list and performs syntactic analysis. The result is expressed as an analysis tree. CSGEN receives the analysis tree and checks it to see whether it satisfies the world model using semantic relations. If not included, CSGEN returns it to ESPER. The result is expressed as a conceptual structure.

4.1.1 Morphological analysis

An input sentence is first divided into morphemes. This is a morphological analysis.

Engish sentences have spaces between words: in Japanese there is no clear word boundary. SEGMENT performs a morphological analysis using the word dictionary and adjacency relations.

Generally, morphological analysis and synthesis are highly language-dependent. This system, however, adopts a language-independent method for multilingual translation. This method uses an adjacency matrix which defines the adjacency possibility between morphemes.

Starting at the left of the input word string, every corresponding morpheme is taken from the word dictionary, and its adjacency number is compared with that of the next morpheme by referring to the adjacency matrix. This matching is based on the length of the morpheme and the frequency of its appearance. The longest, most frequently occurring morpheme will be chosen. The selected morpheme is removed from the input character string, and the next matching is performed until no further morphemes are found. If some morphemes remain unmatched, the system backtracks to construct an acceptable morpheme string.

Morphemes extracted by morphological analysis are output in an analysis node list.

ESPER receives this node list and each morpheme is treated as a terminal node. The sequence of these nodes is the same as that of the input morphemes. Each node obtains grammatical and semantic information from the word dictionary. Grammatical information is a set of grammatical attributes. This allows each grammatical rule to cover a wide range of linguistic phenomena, thus reducing the number of rules. Each terminal node contains the most probable word of several candidates.

4.1.2 Syntax and semantic analysis

Syntactic structure must be analyzed to understand an input sentence. Syntactic analysis requires determing the connections between elements of the sentence and the role of each element.

ESPER receives node lists from SEGMENT and performs simultaneous syntactic and semantic analysis using analysis rules which are based mainly on context-free grammar. ESPER consists of a status stack, analysis window, and control section. The status stack monitors the status during analysis; the analysis window views

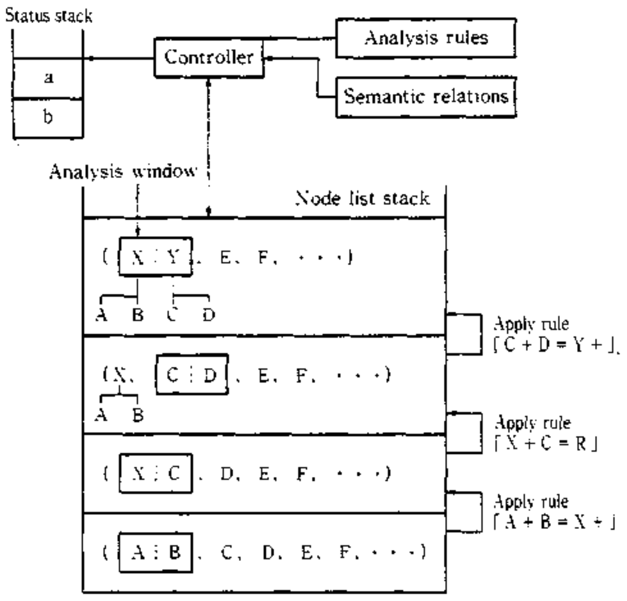

Fig. 6—Configuration of ESPER.

two adjacent nodes.

Analysis rules are based mainly on context-free grammar. A general expression of the analysis rules is:

< CONDITION > < GRAM 1 > + < GRAM 2 > = < GRAM 3> < TYPE > < RELATION > < ACTION > < PRIORITY >.

< CONDITION > indicates the condition under which this rule is applied. < GRAM 1 >, < GRAM 2 >, < GRAM 3 > indicate a set of grammatical attributes. When this rule is applied, < GRAM 1 > and < GRAM 2 > are combined into < GRAM 3 >. < TYPE > indicates one of twelve rules. < RELATION > indictes a modified relation between two nodes. < ACTION > indicates < CONDITION > for the other rules. < PRIORITY > indicates which will be applied first.

When an analysis rule is applied, ESPER operates as follows: First, the two nodes in the analysis window are combined into one node. This newly created node is set as a root node to make a partial tree. Next, the analysis window moves forward down the node list to apply another rule until the analysis tree is completed. If CSGEN rejects the resultant analysis tree. ESPER is called again to create another one.

At first, the analysis window is set on the first and second nodes with the status stack empty as shown in Fig. 6. ESPER finds an appropriate rule by referring to the status stack and the two nodes. If no rule is found, ESPER backtracks. When backtracking, ESPER returns to the most recently applied rule to find an alternative.

An appropriate analysis rule is selected by referring to the status stack and the two nodes which appear in the analysis window.

First, ESPER checks to see if the contents of the status stack match the conditions in < CONDITON > of the rule to be applied. When they match, ESPER further compares the contents of the two nodes with < GRAM 1 > and < GRAM 2 > of the rule. When both are matched, the rule is selected.

The contents of the status stack is specified in < ACTION > of the rule. When this rule is applied, the contens in < ACTION > are written into the status stack. Subsequently the status stack is checked to see if the next rule has the same contents.

The combination of < CONDITION > and < ACTION > makes the analysis rule context sensitive.

When < CONDITION > is satisfied, grammatical attributes reduce the number of applicable rules. Rules with other attributes specified are omitted. When more than one applicable rule remains, the rule with the highest priority is selected.

< GRAM 3 > indicates the new grammatical attributes for the new node, where some new attributes can be added and previous attributes can be inherited.

There are twelve types of analysis rules as shown in Fig. 7.

ESPER performs semantic processing and syntactic processing simultaneously. The suitability of syntactic processing is verified semantically.

Semantic processing is performed with a series of semantic symbols which correspond to the conceptual structure. The applied rule attaches a semantic symbol to the new node and

Before process : A $\boxed{B \vdots C}$ D

Analysis window

Types of rules	Generated tree	Semantics of generated nodes	Reference semantic relation	Position of processed windows
+ Composition (+)	E B C b ⟨c⟩	b , ⟨ c ⟩	(b , * , ⟨ c ⟩)	($\boxed{A \vdots E}$, D) B C
− Composition (−)	E B C ⟨b⟩ c	c , ⟨ b ⟩	(c , * , ⟨ d ⟩)	($\boxed{A \vdots E}$, D) B C
Right modification (⟶)	E B C b, ⟨r⟩ c	c	(b , c , ⟨ r ⟩)	($\boxed{A \vdots E}$, D) B C
Left modification (⟶)	E B C b ⟨r⟩, c	b	(c , b , ⟨ r ⟩)	($\boxed{A \vdots E}$, D) B C
Forward read (F)	E B b	b	——	($\boxed{A \vdots E}$, C , D) B C
Backward read (B)	E C c	c	——	(A , $\boxed{B \vdots E}$, D) C
Right shift (R)	——	——	——	(A , B , $\boxed{C \vdots D}$)
Left shift (L)	——	——	——	($\boxed{A \vdots B}$, C , D)
Exchange (X)	C B	——	——	($\boxed{A \vdots C}$, B , D)
Copy (C)	B B C	——	——	($\boxed{A \vdots B}$, B , C)
Back track (?)	——	——	——	——
Word change (!)	B C	——	——	(A , $\boxed{B \vdots C}$, D)

Fig. 7–Types of analysis pules.

determines the semantic relation between two nodes in the analysis window.

For example, when the rule specifies "synthesis", the two semantic symbols are concatenated. The semantic processing checks to find if the processing is consistent with common sense and linguistics. Finally, the analysis node lists are converted into one analysis tree.

4.1.3 Conceptual structure generation (CSGEN)

CSGEN creates a conceptual structure by extracting relations between concepts from the analysis tree. This analysis tree contains all the information necessary for generation. All CSGEN has to do is to convert the form of expression. The analysis tree is converted into the conceptual structure as shown in Fig. 5. Each

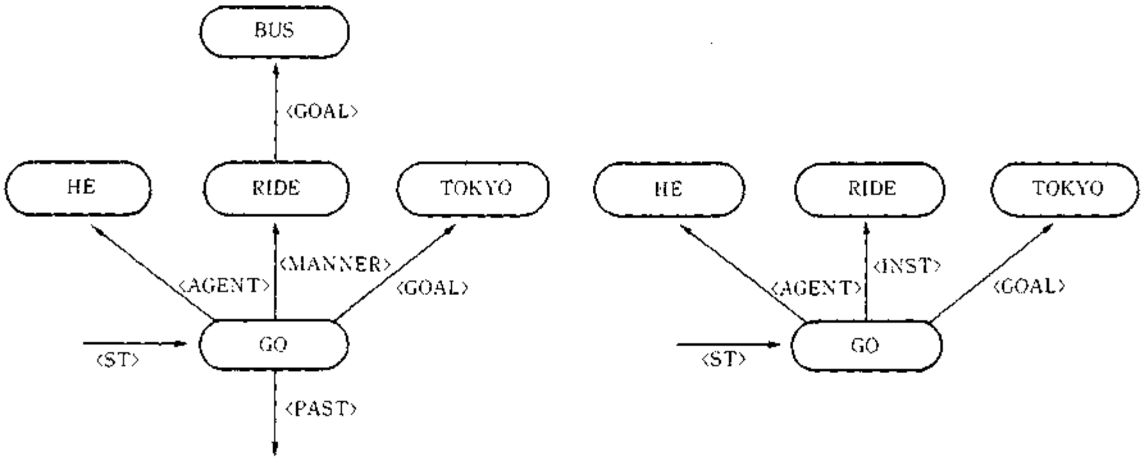

Fig. 8–Conceptual transfer.

```
((*0, RIDE, <GOAL>) (RIDE, *1, <MANNER>),   Subtree 1
 (*0, *1, <INST>),                           Subtree 2
 (VEHICLE, *0, < >) (MOVE, *1, < >))         Condition
```

Fig. 9–Conceptual transfer pule.

node is labeled with a corresponding semantic sybmol; each arc is to be placed between the nodes by extracting relations such as < GOAL >, < AGENT >, < MANNER > and additional information such as < ST >, < PAST >.

CSGEN verifies the conceptual structure by referring to the world model. ESPER can verify the semantic relation only between two nodes; CSGEN, however, can verify the entire conceptual structure. If it is incorrect, CSGEN can require ESPER to analyze again.

4.2 Transfer process

The transfer section is provided to fill the gap between the source language and the target language. Differences in languages stem from, among other things, the cultural background of the people speaking these languages. Superficially, it appears as a difference in words and grammar; internally, it appears as a difference in concepts and in the speaker's way of thinking.

ATLAS II compares these differences, not superficially, but internally; examining not the differences between words or grammar but the difference between concepts and thinking. The difference, therefore, is treated at the level of the intermediate representation, and the conceptual structure is transferred. However, the pivot approach, which does not require this transfer, is suitable for most cases.

We will illustrate some cases which would require such a transfer. For example, the sentence "Heya niwa mado ga futatsu aru." would be literally translated as "There are two windows in this room." But the natural translation would be "This room has two windows."

Another example involves the causative expression. The Japanese language expresses it using the auxiliary verb 'Saseru'; while English depends on an intransitive verb and the word order.

A general expression of the transfer rule is as follows: (Partial net 1, Partial net 2, relation, condition)

This rule indicates partial net 1 is replaced by partial net 2 if both the relation and condition are satisfied.

In the sample sentence the partial network for 'Basu ni notte Tokyo he itta". Figure 8 shows the conceptual structures before and after transfer. Figure 9 shows some of the transfer rules necessary for this transfer. In Fig. 9, *0 and *1 show variables of the corresponding symbols. Arc < GOAL > concatenates node RIDE and node *0; arc < MANNER > concatenates node RIDE and node *1. The concept of node *0 is included in the super concept VEHICLE, and the concept of node *1 is in-

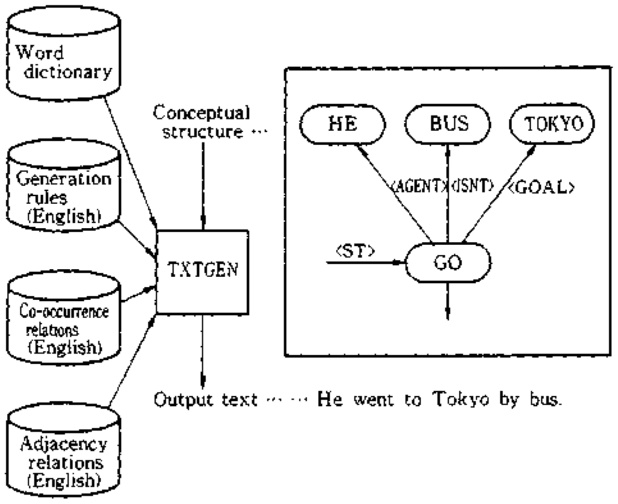
Word dictionary
Generation rules (English)
Co-occurrence relations (English)
Adjacency relations (English)
Conceptual structure ···
TXTGEN
HE
BUS
TOKYO
<AGENT>
<ISNT>
<GOAL>
<ST>
GO
Output text ······ He went to Tokyo by bus.

Fig. 10–Flow of sentence generation.

cluded in the super concept MOVE. When all three conditions are confirmed, a new conceptual structure, where are < INST > concatenates node *1 and node *0, is created. When *0 is replaced by BUS and *1 is replaced by GO, a new relation (BUS, GO, < GOAL >) is produced as shown in Fig. 8.

4.3 Generation process

Target language text is generated from the conceptual structure which is in the form of a semantic network.

This conceptual structure is converted to a linear word string. This direct conversion can eliminate the need for transformations, allowing not only the generation mechanism but also rules to be language-independent.

In this approach, generation rules can deal with both syntactic structuring and morphological synthesizing at the same time, thus simplifying the generation mechanism.

Figure 10 shows the flow of sentence generation.

The generation system consists of a generation window, output list and a rule interpreter. The rule interpreter traverses each node of the conceptual structure by moving the generation window and returns with the output list containing the translation results. Figure 11 shows the generation mechanism which consists of the generation rule, word dictionary, co-occurrence relation and adjacency relation.

The generation window is set at a node of the conceptual structure and is moved from node to node. This window is used to check the nodes and arcs. The output list stores each word in the order of generation. The contents of the output list indicate the surface-structure word order.

A node of the conceptual structure consists of a node name, a basket and a word list. The node name indicates semantic symbols such as 'GO', 'HE', 'BUS', 'TOKYO' as shown in Fig. 10. The basket stores messages sent from other nodes. The word list is a list of words which can represent the concept of the node.

An arc of the conceptual structure consists of an arc name and word list. The arc name indicates a relation between nodes such as < AGENT >, < INST >, < GOAL > as shown in Fig. 10. The word list is a list of words which can represent the concept between nodes. Both a node name and an arc name serve as a key to consult the word dictionary. Consulting the word dictionary produces word lists for both a node and an arc. The word dictionary contains generation symbols, word symbols and adjacency numbers.

The generation symbol serves as a key to access a set of generation rules. The word symbol is used to check the co-occurrence relation between words. The adjacency number is used to check the adjacency relation of the output list.

The rule interpreter interprets each geneation rule, traverses each node by moving the generation window, and selects words from nodes and arcs by checking the co-occurrence relation and adjacency relation. Each selected word is added to the output list. As the generation window is moved, the information in previously processed nodes and arcs is reserved.

Generation rules are classified by the parts of speech which they represent. The order to be applied is determined, and this order indicates the word order of the output sentence.

A general expression of the generation rule is as follows:

< CONDITION > < ARCNAME >

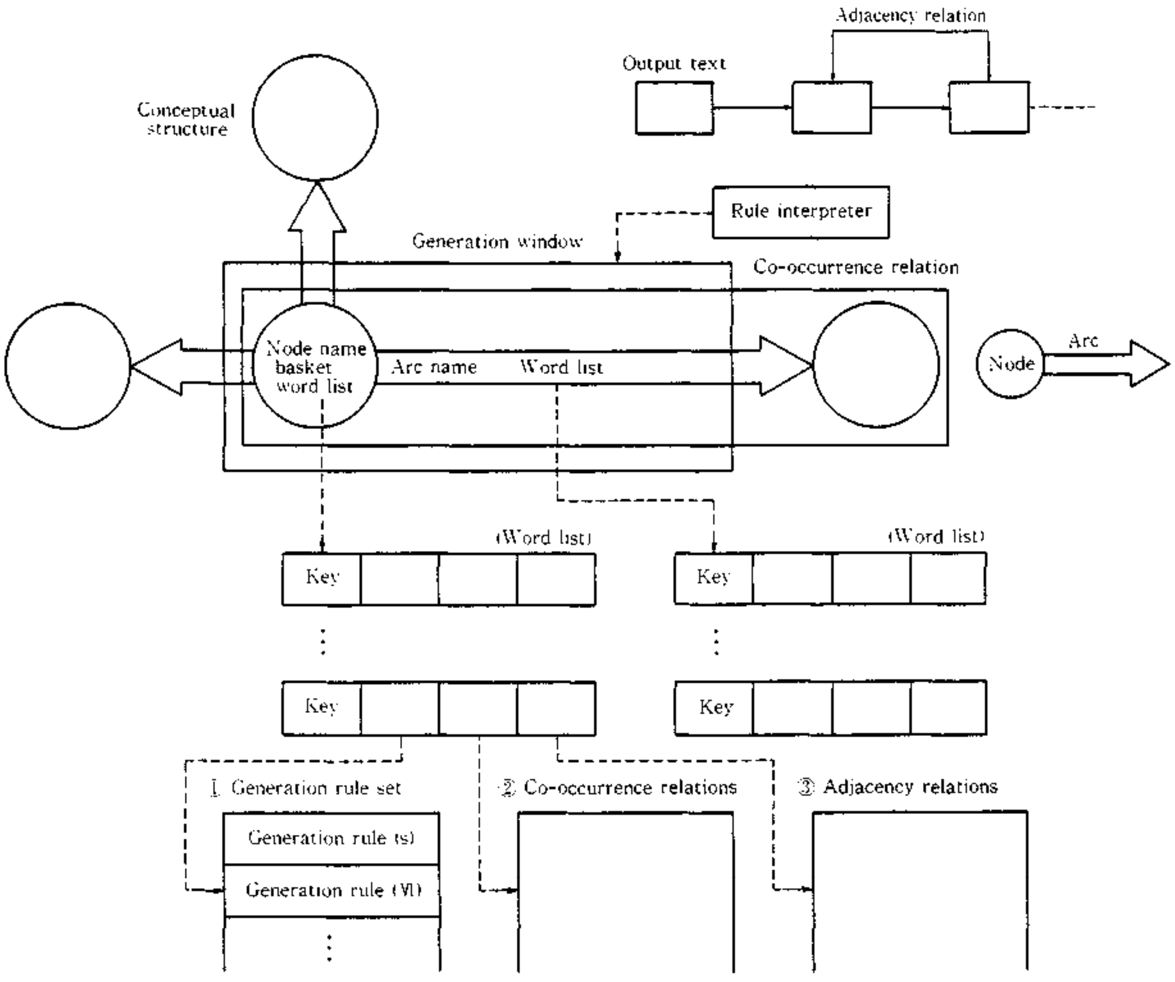

Fig. 11–Generation mechanism.

<ACTION> <MESSAGE>

<CONDITION> indicates the conditions under which this rule is applied.

<CONDITION> is checked with the messages in the BASKET. If they match, this rule is applied; if they do not match, the next rule is tried.

<ARCNAME> indicates an arc name to apply the rule.

<ACTION> specifies the type of the processing.

Major types are as follows:

1) Node generation rule for generating a word corresponding to the node
2) Out-arc generation rule for generating a sentence from a subnetwork starting at the specified out-arc
3) Inarc generation rule for generating a sentence from a subnetwork starting at the specified in-arc
4) Word generation rule for directly generating a word from the node

<MESSAGE> indicates messages to be sent to BASKET of the node.

The coocurrence relation between two words defines the true/false value of whether the two words can cooccur in the same sentence.

Generally a concept includes several words. For example, a concept indicating 'Sonzaisuru' in Japanese includes selection of a word from several candidates by checking the co-occurrence relation between the candidates.

Every word has two adjacency numbers. These numbers are checked with the adjacency matrix on the output list.

The generation system receives a conceptual

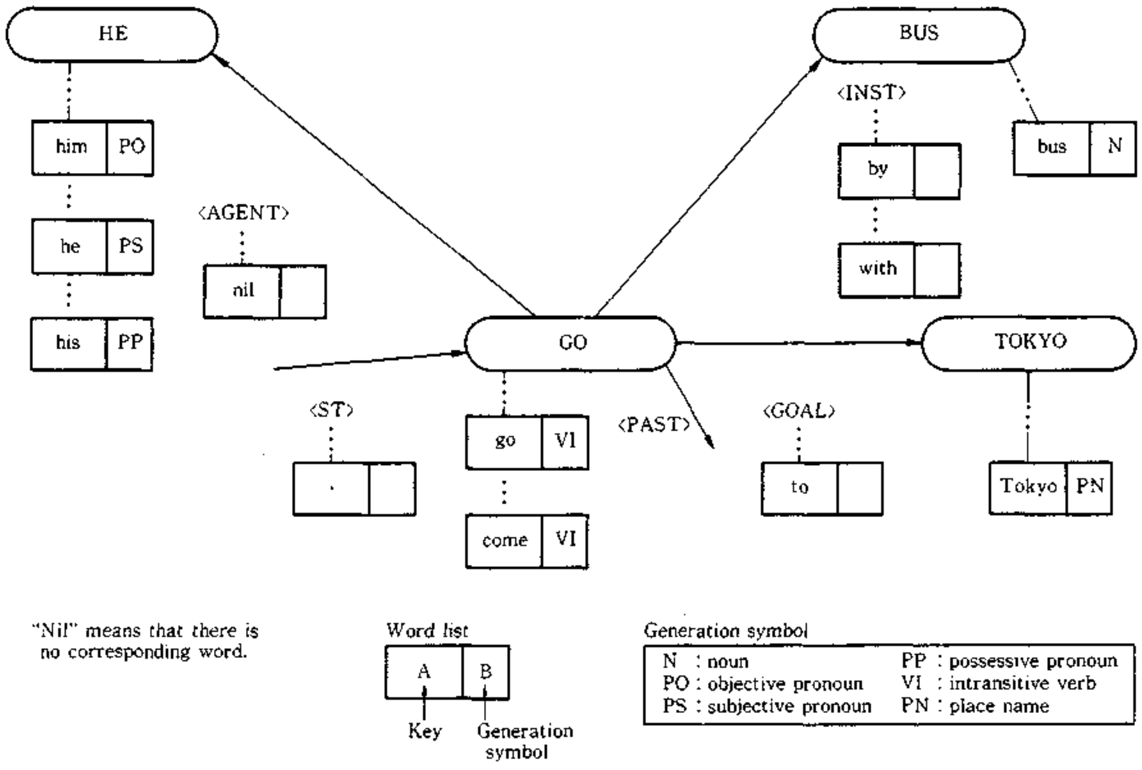

Fig. 12–Example of sentence generation.

Condition	Arc name	Action	Message
{VI}	〈AGENT〉/		
(1) *	/〈ODJ〉	/out arc	/SUBJ ;
(2) SUBJ is not put out	/ *	/out arc	/SUBJ ;
(3) auxiliary verb	/ *	/put out auxiliary verb	/ ;
(4) perfect	/ *	/put out perfect form	/ ;
(5) past	/ *	/put out past	/ ;
(6) present	/ *	/put out present	/ ;
(7) object & VT	/〈OBJ〉	/out arc	/OBJ ;
(8) takes IO	/〈GOAL〉	/out arc	/GOAL ;
(9) *	/〈INST〉	/out arc	/MOD ;
(10) not interrogative sentence	/ *	/put out " . "	/ ;
(11) interrogative sentence	/ *	/put out " ? "	/ ;
{PO}			
(1) SUBJ	/ *	/word change	/ ;
(2) POSS	/ *	/word change	/ ;
(3) plural, singular	/ *	/word change	/ ;
(4) *	/ *	/put out itself	/ ;
{PS}			
(1) OBJ	/ *	/word change	/ ;
(2) GOAL	/ *	/word change	/ ;
(3) POSS	/ *	/word change	/ ;
(4) plural, singular	/ *	/word change	/ ;
(5) *	/ *	/put out itself	/ ;
{PP}			
(1) SUBJ	/ *	/word change	/ ;
(2) GOAL	/ *	/word change	/ ;
(3) OBJ	/ *	/word change	/ ;
(4) plural, singular	/ *	/word change	/ ;
(5) *	/ *	/put out itself	/ ;

Fig. 13–Generation rules.

structure where each node and arc has a corresponding word list.

Sentence generation starts at a node with an in-arc < ST >.

Next, the process of generating an English sentence from the conceptual structure shown in Fig. 12 will be explained. Figure 13 shows some of the generation rules to be applied.

The rule interpreter sets the generation window at node 'GO'. The rule interpreter picks up a word from the top of the word list and calls for the generation rule using the generation symbol VI. Each rule is tried from the top of the generation rule VI, as shown in Fig. 13.

Rule(1), an out-arc generation rule, is applied. This rule specifies the subnetwork starting at outarc < AGENT > as the subject. The rule interpreter moves the generation window to node 'HE'. At the same time, message 'SUBJ' is sent to node 'He', and a flag meaning "Under processing" is set at node "GO' to reserve the present status.

Node 'HE' has a word list of 'he', 'his', 'him'. 'him' is first picked up from the top of the list. The rule interpreter finds the generation symbol PO and tries rule(1). This rule is applied and a word change occurs.

Next 'he' is picked up and rule(4) is applied.

The rule interpreter sends 'he' to the output list. Next rule(5) is applied. The rule interpreter finds the flag for node 'GO', and picks up a word from the top of the word list of arc <AGENT>. At this time the rule interpreter checks the co-occurrence relation among 'he', 'go' and <AGENT>. There are no rules left to be applied. The rule interpreter returns to node 'GO'.

The rule interpreter resumes processing starting at rule(2). Rules(2) to (4) are not applied. Rule(5) is applied and 'went' is output. Rules(6) to (7) are not applied. Rule(7) is applied and 'to Tokyo' is output. Rule(8) is applied and 'by bus' is output.

5. Conclusion

The biggest problem with any machine translation system is the quality of the translation. Unfortunately, current technology cannot achieve perfect results. We have to provide assistance functions such as pre-editing, post-editing and dictionary compilation.

The quality of translation depends on the accuracy of both rules and dictionaries, as well as the amount of information contained in the dictionary. But this presents another problem: the greater the amount of information, the longer the processing time. It is also difficult to guarantee the accuracy of a large amount of information.

The machine translation system operates around rules and dictionaries. The average number of words in a Japanese sentence ranges from 40 to 50. There are more than 1 000 rules. A single item of erroneous information will produce a mistranslation. The accuracy, therefore, must be better than 99 percent.

The number of words in normal use is said to range from 30 000 to 50 000. But actually, there are a vast number of technical terms (5 million to 10 million).

These problems cannot be solved by one company alone. We must ask for assistance from users, especially in the compilation of dictionaries.

We believe, however, that machine translation will eventually prove superior to manual translation in terms of speed and consistency, and will play an important role in international communication.

Hiroshi Uchida
Software Laboratory
FUJITSU LABORATOIES,
KAWASAKI
Bachelor of Physics
Osaka University 1970
Specializing in Natural Language Processing

Tatsuya Hayashi
Software Laboratory
FUJITSU LABORATORIES,
KAWASAKI
Bachelor of Applied Physics
Waseda University 1960
Specializing in Software Engineering, Data Base, Natural Language Processing, Artificial Intelligence

Hiroshi Kushima
Scientific Systems Development Dept.
Systems Engineering Group
FUJITSU LIMITED
Bachelor of Applied Physics
Tohoku University 1975
Specializing in Natural Language Processing