International Conference on Information and
Knowledge, Nov. 1987, at Yokohama

Computer Environment for Meaning Structure Representation and Manipulation in Machine Translation System

Kiyoshi Kogure
ATR Interpreting Telephony Research Laboratories
TWIN-21, MID Tower, Higashi-ku, Osaka 540, Japan

Hirosato Nomura
NTT Basic Research Laboratories
Musashino-shi, Tokyo 180, Japan

1. Introduction

A machine translation system needs to apply various kinds of knowledge in translation if it is expected to understand the meaning of source language text deeply, and it then produces target language text. It is widely understood that deep understanding of the meaning of the text is needed to produce high-quality translation.[5),14)] The required knowledge is classified into linguistic and extra-linguistic knowledge. Linguistic knowledge concerns syntax, semantics and context, while extra-linguistic knowledge concerns domain expertise and common sense. The quantity and quality of the knowledge a machine translation system utilizes determine the translation ability and quality.

The entire knowledge required by such a machine translation system is too extensive to make preliminary arrangements. Therefore, it is necessary to start with the small amount of knowledge stored in the Knowledge-Base in the machine translation system, and then accumulate further fragments of knowledge into the Knowledge-Base through examination and experience during development and practical use. This means that the Knowledge-Base grows gradually with translation experience.

One of the most crucial problems arising in the development of a machine translation system by this strategy is how to accumulate the knowledge or how to extend the Knowledge-Base.[2),5),9),15)] This function is not accomplished by a fully automatic process but is accomplished by long-term interactive process between experimenters or users and the machine translation system step by step. Thus, it is very important to develop an interactive computer environment which provides facilities to support such a knowledge accumulation process.

The knowledge accumulation process at the early stage of machine translation system development consists of two tasks: 1) collecting linguistic and extra-linguistic data by carefully examining the translation of various kinds of sentences as a whole or in part, and then 2) assimilating the new fragments of knowledge learned from the experiences into the Knowledge-Base so that the machine translation system can use them in succeeding translation experiments. To conduct these tasks efficiently, it is impractical to provide dedicated individual support software for each purpose. Rather, a total environment which can support those various tasks in one computer system is needed.[4)]

This paper presents an interactive computer environment, the Reciprocal Environment for the Study of the Language Understander, Translater & Editor (RESOLUTE), which supports the knowledge accumulation process described above by examining the translation process of an experimental machine translation system called the Language Understander, Translater & Editor (LUTE).[5)~13)] The strategy adopted by the RESOLUTE might exemplify the development style of a knowledge-based machine translation system which could be developed as a high-ability and high-quality machine translation system for the future. Section 2 outlines the LUTE and the RESOLUTE. Section 3 describes a linguistic model and the knowledge used by the LUTE. Section 4 points out several problems of machine translation experiments which must be resolved by providing a proper computer environment. Finally, Section 5 thus presents the functions and facilities the RESOLUTE provides.

2. Outline of the LUTE and the RESOLUTE

This section outlines the LUTE and the RESOLUTE for preparing the following sections.

2.1 Overview of the LUTE

An experimental machine translation system, the LUTE[5)~13)], has been developed to examine our theoretical work on computational linguistics. Thus, the LUTE is not intended to be a practical system. It can only translate a limited number of Japanese sentences into English sentences and vice versa to demonstrate how the translation process is executed and

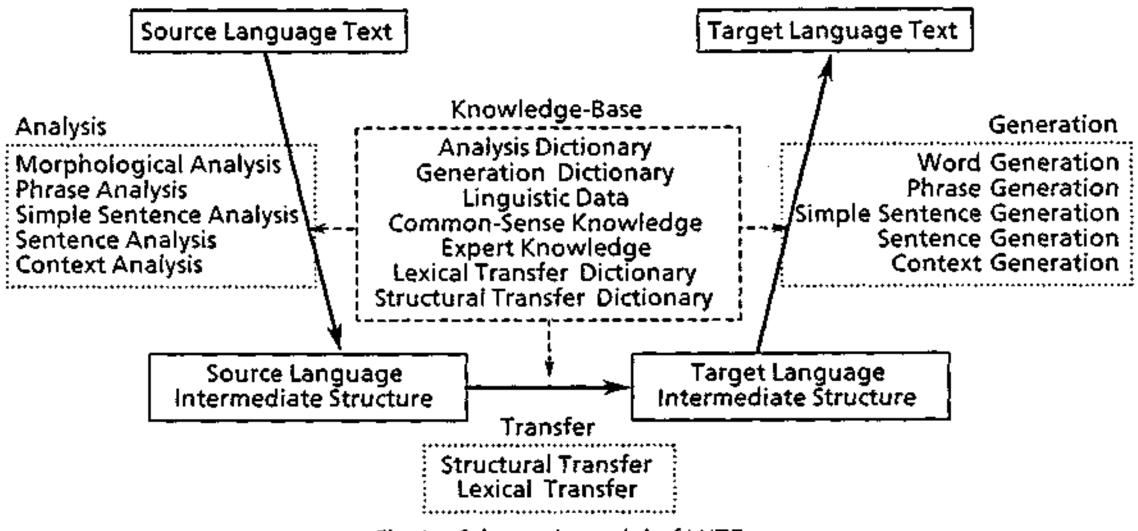


Fig.1 Schematic model of LUTE

what kind of information the translation uses.

The LUTE adopts the so-called transfer method: it divides the whole translation process into analysis, transfer, and generation. The analysis process deeply analyses both syntactic and semantic information in a source sentence and then produces a "deep" meaning structure as an intermediate representation of the information in the source sentence. The transfer process interconverts the meaning structure of the source sentence into a meaning structure for the target language. The generation process then produces a translated sentence from the interconverted meaning structure. In these processes, the system uses both linguistic and extra-linguistic knowledge. Figure 1 shows the LUTE processing flow and the data used by the LUTE.

The analysis or understanding process applies both types of knowledge concerning the utterance in the source language. The transfer process applies such knowledge with respect to the correspondence between two languages. The generation process applies it with respect to the linguistic expressions in the target language. Thus the LUTE translation process can be used to determine what kind of linguistic and extra-linguistic knowledge is needed for each progcess in translation. This makes it possible to know what fragments of knowledge are incorrect or insufficient thus what should be modified, and what fragments of knowledge have not been incorporated into the Knowledge-Base yet thus what should be added to the Knowledge-Base.

The LUTE design is based on a computational linguistic model called the Extended Case Structure model[3)~8),11)], which formalizes both the linguistic and extra-linguistic information in a sentence and provides a way to construct a meaning structure from those fragments of information. Both the meaning structure and the knowledge are represented in terms of a knowledge representation scheme called the Frame-Network[1),11)], which provides a uniform framework for incorporating them.

2.2 Overview of the RESOLUTE

An interactive computer environment, the RESOLUTE, has been developed to support the development and the examinations of the LUTE. It has been designed mainly for manipulating the complicatedly structured meaning information represented in the Frame-Network. Thus, the RESOLUTE provides facilities to check and modify fragments of information in the Frame-Network interactively on the display.

Additionally, it has extra facilities for examining the LUTE translation process as a whole or in part. As a result, the effect of the application of the various fragments of knowledge in the translation process can be tested either separately or simultaneously and then the Knowledge-Base can be edited from the results of the examination.

3. Knowledge in the LUTE

The LUTE has been developed to evaluate the Extended Case Structure model as mentioned above. The LUTE analyzes both the syntactic and the semantic information in a sentence in terms of the Extended Case Structure model, and then it produces a meaning structure from which the system generates a translated sentence after rearranging it as a target meaning structure also in terms of the Extended Case Structure model. In these processes, the Extended Case Structure model provides various kinds of knowledge for syntactic and semantic processing. In this section, the Extended Case Structure model and the type of knowledge used by the LUTE will be described.

3.1 Extended Case Structure Model

The Extended Case Structure model[11] is proposed as a linguistic model for representing both text information and text structure. The text information relates to both the linguistic and extra-linguistic knowledge on the words and phrases in a text. The text structure relates to how to combine them using the several kinds of relationships presented below so that they construct a meaning structure of the text. Thus, the meaning structure retains both syntactic and semantic sentence information, and the information in the meaning structure concerns both linguistic and extra-linguistic knowledge. As such, the Extended Case Structure model accounts for the syntactic and semantic well-formedness of the text using both types of knowledge.

The knowledge concerning words and phrases is defined by a feature bundle[12] consisting of feature-label and feature-value pairs. Each feature-value is defined by a set of both syntactic and semantic categories which are supposed as primitives to represent either grammatical restrictions in language expressions or selectional restrictions in identifying the meaning. The grammatical categories concern the parts of speech and the grammatical role while the semantic categories concern the semantic classification of each object or word such as animate, building and so on.

The relationships in the Extended Case Structure model is classified as follows[3),7),11),13)]:

1) Grammatical relation: grammatical relationships between a modifier and a modificant in a constituent (restrictions on adjoining words, restrictions on phrase structure, etc.)

2) Noun relation: relationships between nouns (whole-part, upper-lower, possession, material, etc.)

3) Case relation: relationships between a case element and a predicate (object, agent, instrument, place, etc.)

4) Modality relation: relationships between a modal element and a predicate (passive, causative, etc.)

5) Embedding relation: relationships between a modifying clause and a modified noun in a noun phrase (case relations, etc.)

6) Conjunctive relation: relationships between clauses in a sentence (cause-result, time-advance, assumption, etc.)

The meaning structure constructed using these feature bundles and relationships is hierarchical with respect to the way constituents are connected, iterative with respect to conjunctions, and recursive with respect to embedding. Using this formalism, the syntactic and semantic structure of a sentence can be represented uniformly in terms of the Extended Case Structure model. Also, the linguistic and extra-linguistic information is represented uniformly in terms of the Extended Case Structure model, as described in the rest of this section. An example of a meaning structure in terms of the Extended Case Structure model is shown in Figure 2.

3.2 Linguistic Knowledge

For incorporating the feature bundles and relationships into a meaning structure, the Extended Case Structure model provides an extra data structure called a frame[1),6),17),18)], which is a component of the representative framework for a meaning structure. The frame is also one kind of meaning structure. However, it is not necessarily a meaning structure for a sentence; it might be for a smaller constituent of a sentence such as a word or a phrase or a clause. Therefore, the hierarchical structure mentioned above is constructed using frames as components. There are the following kinds of frames, depending on the grammatical characteristics of the constituents[3),11),13)].

1) Noun frame: A semantic structure for a compound noun, a noun phrase consisting of more than one noun, possibly with particles such as "no" or "to" between nouns in Japanese and that consisting of a noun and some adjuncts such as adjectives or prepositional phrases in English, and a noun phrase with an embedded clause, which represents both syntactic and semantic constraints and relationships holding between a head noun and other constituents. The head noun in a noun phrase retains the noun frame.

2) Case frame: A semantic structure for a unit sentence, which represents both syntactic and semantic constraints and relationships holding between a head predicate and other case or modal elements. The predicate retains the case frame.

3) Event frame: A semantic structure for a complex sentence, which represents both syntactic and semantic constraints and relationships holding between clauses. The conjunctive element retains the event frame.

4) Heuristics: This is used for resolving ambiguity among categories and relationships by linguistic information such as restrictions on categories to be combined by a relationship, and preferences over several relationships.

3.3 Extra-linguistic Knowledge

Fragments of both expertise and common sense knowledge are represented again using basic elements such as feature bundles and relationships. Thus, extra-linguistic knowledge fragments are not essentially different from the linguistic ones, and they are again represented in the Frame-Network in the same way as linguistic knowledge[5),9),11)]. Therefore, there is no clear boundary between linguistic and extra-linguistic knowledge. However, they are different in accounting for relationships between constituents; extra-linguistic knowledge mainly concerns real world knowledge and situations as well as expertise but does not concerns linguistic ones. The following types of extra-linguistic knowledge are defined in terms of semantic relationships[9),11)]:

1) Concept relation: relationships such as synonym, antonym, whole-part, and possession. (the "whole-part" relationships between a train and a train window).

2) Event-state relation: relationships between two events or between an event and a state. (smell when grilling a fish).

3) Meta-Knowledge: this is used for reasoning, such as traversing internal constituents in a semantic structure and then checking semantic consistency among them according to concept and event-state relationships.

4. Problems on Machine Translation Experiments

The knowledge a machine translation system applies in its translation process is to be accumulated through examinations and experiments as mentioned earlier, while basic fragments of it might be given in advance. It is supposed that such given fragments occupy a very small part in the whole Knowledge-Base needed for a high-ability and high-quality machine translation system. Thus, accumulating the rest of the Knowledge-Base through careful and detail examinations of both Knowledge-Base and programs is needed.

The tasks in the experiments can be summarized as tasks related to Knowledge-Base editing and knowledge application to the translation process. From these, it is possible to get new information to be incorporated into the Knowledge-Base for further experiments. Thus, machine translation experiments can be seen as consisting of iterative application of these two tasks. In this section, problems related to conducting efficient experiments with them will be listed.

4.1 Problems Related to Knowledge-Base Editing

A Knowledge-Base can be seen as consisting of grammatical rules and dictionaries which are extended to represent both linguistic and extra-linguistic knowledge. Dictionaries are divided into those for source language, for target language, and for bi-lingual situations. A Knowledge-Base combines them as mentioned above.

Each lexical item in the dictionaries accompanies frames which bear such knowledge on one hand and extra information on controlling both during processing and editing on the other hand.

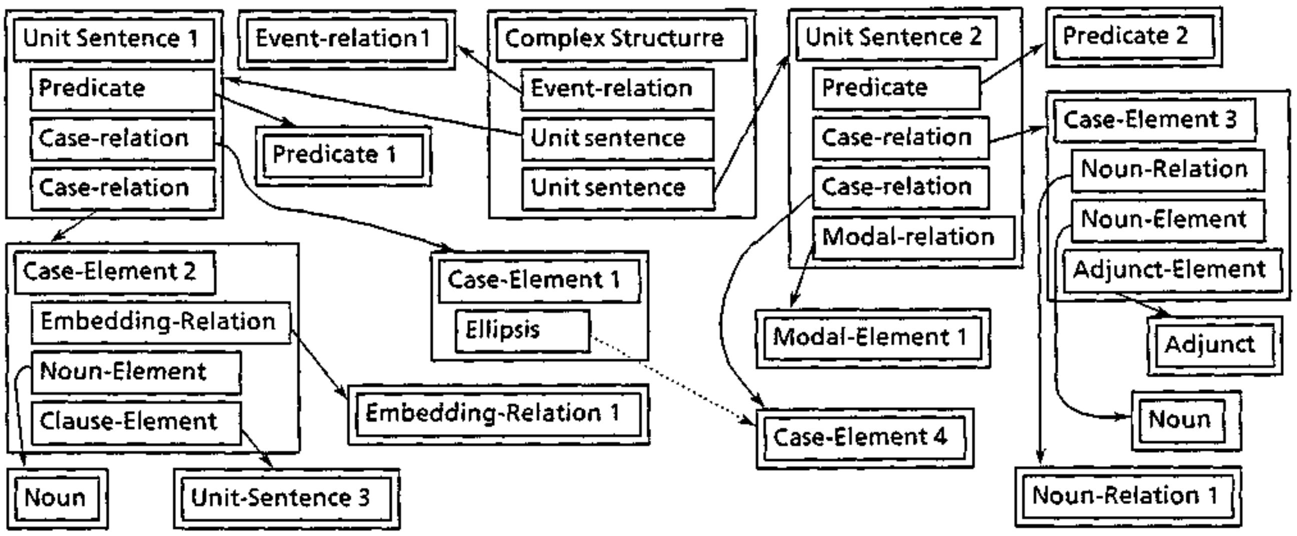


Fig.2 Frame-Network representation of a sentence in terms of Extended Case Structure

Since dictionaries to be used by the knowledge-based machine translation systems exemplified above have a large amount of description consisting of different kinds of information, there are a reasonable number of hard problems in editing and examining them.

Even if a machine translation system does not employ such a knowledge-based approach, most of the recent computational linguistics theories adopt the so-called lexical approach, and they describe almost of linguistic information in lexical items. This results again in plenty of linguistic descriptions put into each lexical item.

Thus, the task for editing dictionaries is one of the most important problems whichever approach is employed.

When developing such large dictionaries, many people used to work together. Accordingly, one person has the responsibility for only some particular part. This causes serious problems in keeping consistency among the lexical item descriptions, which emphasizes that a dictionary editing environment is crucial and has to have facilities to support and maintain consistency.

The problems related to dictionary editing are classified into three types; i) intra-lexical items, ii) inter-lexical items, and iii) inter-dictionary items. These will be explained in detail in the following subsections.

4.1.1 Intra-Lexical Item Problems

In general, a lexical item is described by using a dictionary representation language which must be powerful enough to represent the various kinds of knowledge mentioned above, and flexible enough to accept any modification of it and addition of extra data.

Moreover, in order to accomplish a correct and consistent description of each lexical item, the following conditions must be satisfied:

1)Each part of the description must be correct;

2)The contents of the each description must be sufficient for processing the deep meaning mentioned earlier;

3)Each part of the description must be consistent in relation to the whole description of the dictionaries.

Therefore, the dictionary description language has to provide special facilities for indicating whether those conditions are satisfied. Moreover, the function realized by these facilities must be able to check the conditions automatically whenever the dictionary is modified.

The LUTE uses the Frame-Network previously mentioned as a dictionary representation language. This provides a very powerful and flexible representative framework for that purpose. In the Frame-Network, each lexical item is represented by a frame with a three-level hierarchy consisting of embedded substructures; slots, facets and data[17),18)], as shown in Figure 3.

As mentioned earlier, each information in a frame is represented as a pair comprising feature-label and its feature-value. A frame name is an instance of a feature-label and its body is an instance of its feature-value. However, the feature-value can be a set of values. Thus, the representative scheme of the frame follows from that for FRL.[17),18)]

By using such facets, a frame can represent not only ordinary knowledge but also meta-knowledge. Such meta-knowledge is used to keep conditions which must be held with regard to other data. The conditions include the relationships to hold over the data and the daemon procedures to check them. They are represented in the same way as ordinary knowledge but can also possibly be described as programs which are executable directly. Thus, the conditions are written in a procedural manner as well as a declarative manner. As a result, this provides a powerful and flexible representative framework as a dictionary representation language. This type of frame construction is depicted in Figure 3.

There are two kinds of experimenters and editors, users and managers. A user or an average dictionary editor does not need access to such meta-knowledge. Only the dictionary manager needs this. Consequently, it would be convenient for editors if an environment were provided for them in which they need not access knowledge fragments not related to their current tasks. Moreover, such a facility would become to provide or assure using limited user interfaces efficiently and avoiding mis-operations on unrelated fragments. Thus, the environment must provide facilities to support editor's access to the data only related to his on-going tasks.

```
(frame   (slot  (facet   (datum)  ···  (datum))
                   ⋮
                (facet   (datum)  ···  (datum))  )
           ⋮
         (slot  (facet   (datum)  ···  (datum))
                   ⋮
                (facet   (datum)  ···  (datum))  ))
```

Fig.3 Frame construction by slots, facets and data

4.1.2 Inter-Lexical Item Problems

When more than a few people edit one dictionary cooperatively over a relatively long period, it becomes important to keep homogeneity and consistency among descriptions over the different lexical items in the dictionaries. Especially, maintaining semantic accuracy becomes problematic in areas such as semantic constraints between a head and its complements in a constituent structure. Since there are often no clear criteria on them, the following facilities to compare and check them need to be provided: facilities to advise editors by giving typical examples, and facilities to use hierarchical descriptions for reducing outsize descriptions.

The LUTE uses a multiple-inheritance mechanism with hierarchical descriptions realized on the Frame-Network. The inheritance mechanism is used to reduce redundant descriptions both for proper knowledge for language processing and for meta-knowledge in control processing. For example, the description on English verbs representing someone's desire such as "want" or "hope" does not take an deep PURPOSE case. This can possibly be given in the following way: 1) create a DESIRE-VERB class frame; 2) put into the frame a description on those verbs involved in this verb class that cannot take a deep PURPOSE case; and then 3) make these verbs instances of the class frame.

4.1.3 Inter-Dictionary Item Problems

Machine translation systems based on the transfer method use at least three kinds of dictionaries: 1) a source language dictionary, 2) a transfer dictionary which combines two different languages lexically and structurally, and 3) a target language dictionary. In editing the transfer dictionary, for example, an editor needs to know linguistic knowledge on both languages which is described separately in each language's dictionary, and then needs to know how to combine source meaning information with target meaning information. This puts a requirement on the editing environment that enables the editor to retrieve information from these dictionaries simultaneously.

Moreover, if the editor learns that the analysis can not produce sufficient or accurate data to choose an equivalent term in translation, he must research the analysis dictionary and then correct it so that it can produce enough and correct data for the purpose. Also, if the editor learns that the generation can not produce a correct translated sentence, he must resort to the generation dictionary. Consequently, a multiple dictionary editing facility is also required.

4.2 Problems Related to Knowledge-Base Examining

The process of knowledge application in the examination process relates to three tasks as listed below:

1) Collecting suitable data for examination: Data related to linguistic phenomena to be examined must be carefully selected in order to accomplish the examination effectively. Consequently, the environment must provide a facility for an experimenter to retrieve the related data both inside and outside the Knowledge-Base. In the second case, it includes a search for a text data-base which is not a part of the Knowledge-Base to be used for the translation, from which the experimenter can learn some examples. Such a facility might look like the so-called KWIC.

2) Invoking the machine translation process as a whole or in part: a machine translation system adopting a transfer method like the LUTE usually consists of three sub-processes: analysis, transfer, and generation. When debugging a transfer rule used by the transfer sub-process, for example, it is efficient to be able to invoke only the transfer sub-process which applies the transfer rule to the analysis result or a part of it.

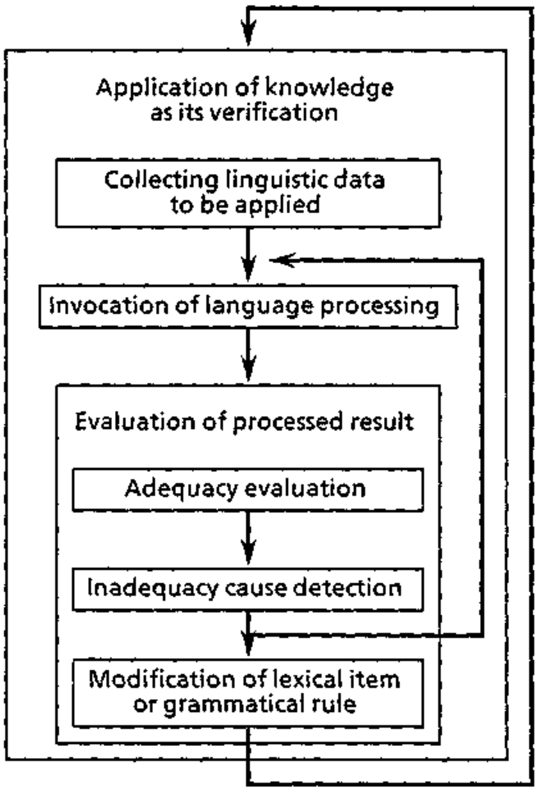
3) Evaluation of the analysis and transfer results: When checking the result of the application of the Knowledge-Base to the translation process as a whole or in part, if any inconsistency is found in the results, either the analysis result or the transfer result or both, the following processes are required.

   a) Finding the reason for the inconsistency: An experimenter must find the process that caused the incorrect result, so the experimenter must check partial results such as analysis results and transfer results. However, the analysis and transfer results are represented as complex semantic structures, and sometimes they become too complex for an experimenter to understand the details at a glance. Thus, a facility for displaying these structures with some abstraction is required. The RESOLUTE provides a graphic representation facility which has a data abstraction function for this purpose. The abstraction is effected in accordance with the experimenter's on-going tasks.

   b) Modifying the knowledge that caused the problem: When the experimenter can identify the incorrect part of the Knowledge-Base, it must be able to be modified it without being bothered by its irrelevant part. This means that the environment must provide facilities so that the experimenter can modify only the relevant part of it by using the result obtained from the endeavor conducted in the task in finding the reason. The side effect, if any, might be eventually induced from the modification. In the case, it should be managed automatically so that the relevant portion of the knowledge will be adjusted without an explicit instruction from the user.

c) Examining the modified knowledge: After modifying the data, it will be checked for whether the modification of the knowledge works out. This examination should be done only on the related part of the process.

The foregoing tasks provide the entire machine translation experiment process. From them, it can be seen that the machine translation experiment is simply iterative applications of these processes. Furthermore, if the experimenter can not identify the reason causing the problem in detail, the process enters into the recursive application of these processes: in the beginning the experimenter will nominate candidates for the reason, and then he will identify the real reasons from them after checking them precisely in a limited area where more detail and specific conditions can be applied. Therefore, an environment for knowledge examination is required to enable an experimenter to do these tasks both iteratively and recursively. Figure 4 shows the flow of the knowledge examination.


Application of knowledge as its verification
Collecting linguistic data to be applied
Invocation of language processing
Evaluation of processed result
Adequacy evaluation
Inadequacy cause detection
Modification of lexical item or grammatical rule
Fig. 4 Flow of knowledge examination and editing

4.3 User Friendliness

Every task for editing dictionaries and examining them should be accomplished without imposing any psychological burden on the experimenter. This concerns both friendliness of the physical man-machine interface realized on the screen and accessibility to the object the experimenter intuitively desires to be checked.

The physical man-machine interface is limited by the hardware technology available; however, the level will soon be reached where eye-pointing and voice-instructing will become easier and more accurate, while they are not the matters to be discussed here.

Direct accessibility to the object which the experimenter is doubting as a criminal should then be possible according to the user's feeling. This must be done without interpreting or translating his intuition concerning what object should be checked by the computer instructions. For that purpose, an environment must provide facilities which help him to identify the object freely and directly and then execute any sub-program for examining it. In the case, it also has to provide a facility which passes any kinds of data to be checked to it even if the sub-program is embedded deeply into a program. So, the issue on the accessibility relates tightly to the friendliness in identifying and checking a object.

5. The RESOLUTE Interactive Environment

An interactive environment called the RESOLUTE has been developed to solve the problems described so far. This environment provides an experimenter with a feeling of directly manipulating an object related to a fragment described in dictionaries or frames with facilities supporting both a multiple-window system as a graphic display device and a mouse as a pointing device. The RESOLUTE has been implemented on Symbolics Lisp machines.[20]

The RESOLUTE consists of two sub-systems: 1) a machine translation sub-system, and 2) a dictionary editing sub-system. They are closely related to each other so that an experimenter can conduct both Knowledge-Base examining and editing simultaneously in a single environment.

5.1 Machine Translation Sub-system

This sub-system is the top-level of the total RESOLUTE environment. Thus, it is invoked whenever an experimenter enters into the RESOLUTE environment. In this sub-system, the experimenter is allowed to invoke the dictionary editing sub-system which will be described in the next sub-section.

The machine translation sub-system consists of two editors responsible for handling two types of objects the LUTE uses; a ZMACS text editor and a FRED frame editor.

ZMACS is a program provided as a resident editor for its customers by Symbolics. It is incorporated here as a module to be used in the RESOLUTE environment. ZMACS handles text strings in a manner that strings to be processed by some LUTE module are supplied from ZMACS's buffer and the result strings the LUTE module produces are returned to the buffer. Thus, the buffer is an interface between LUTE modules and ZMACS for passing data to and from each other.

FRED is a multiple-window editor designed for handling frames in the Frame-Network, especially for handling complex structures constructed by combining frames. To handle these frames, FRED provides a facility to control each frame and each part of a frame in a visible style on the display. This facility enables an experimenter access to only those knowledge fragments that he requires in his task and that he can manipulate within his knowledge. This accelerates knowledge checking and avoids mis-operation on unrelated fragments. Moreover, this facility enables an experimenter to grasp a global view of the relationships among frames by means of its complexity reduction capability with visible objects.

In FRED, each frame or each sub-structure of a frame can be handled in the so-called object-oriented way. The frame pointed to by using the mouse displays a list of commands it can accept in the situation and then it invokes the specified command in its own way. FRED uses meta-knowledge, which tells how to display and edit it, for this purpose. This knowledge is stored in user-defined class frames and it is retrieved for use by using the multiple-inheritance mechanism mentioned previously.

FRED provides three types of windows: 1) a top-level window to manage the whole frame system; 2) frame-network windows to handle inter-frame relationships; and 3) frame table windows to handle the contents of each frame. Each of them has its own process specified as meta-knowledge which shows how to display its contents and how to handle its acceptable commands or its contents' acceptable commands.

This sub-system has been designed for supporting tasks which will be shown in the following sub-sections.

5.1.1 Invoking Language Processing Programs

In the RESOLUTE, a language processing program of the LUTE is invoked via a ZMACS buffer or a FRED window as either a whole or a part. For example, the command "Translate a Japanese sentence into English" can be executed only by pressing Hyper-K (pressing the Hyper-key and K simultaneously) after specifying a text to be translated. The translated text

Command	Contents
Invoke LUTE program modules	Execute a program module in LUTE.
Display Next	Display the next one from the set of the results produced by a process.
Analyze English	Apply English analysis to a text and display the result on a frame network window.
Analyze Japanese	Apply Japanese analysis to a text and display the result on a frame network window.
Translate English to Japanese	Translate an English sentence to Japanese.
Translate Japanese to English	Translate a Japanese sentence to English.
Retrieve English Dictionary Items	Retrieve English dictionary items whose entries match with the specified string.
Retrieve Japanese Dictionary Items	Retrieve Japanese dictionary items whose entries match with the specified string.
Edit Dictionary Item	Enter into the edit mode for the specified dictionary item.
Retrieve Japanese Electric Dictionary Item	Retrieve Iwanami Japanese dictionary[15] item (text format dictionary for human use).
Retrieve English Electric Dictionary Item	Retrieve English dictionary item (text format dictionary for human use).

Fig.5 Commands for language processing

will appear on the next line following the source text on the screen.

As such, most of the commands which invoke language processing modules are assigned to a single key so that they can be invoked by just pushing their assigned keys. A list of commands provided for invoking language processing modules is shown in Figure 5.

This easiness of language processing module invocation makes it easy to compare the results of various input strings and to find the reasons for inadequacy in the results. When a translated sentence appears to be incorrect, examining slightly different sentences can give information concerning the reason for the problem.

Modifying sentences and examining the modified sentences can be managed and accomplished on the same screen by simple operations. For example, suppose that an experimenter noticed an inadequacy in the translation result for a sentence having an embedded clause, and then he tries to identify which rule used in the process, either a rule related to the matrix sentence or another rule related to the embedded clause, is responsible for the inadequacy. In such a case, the fact will be revealed immediately after translating the sentence obtained by removing the embedded clause. Such an examination can be conducted easily and quickly in the RESOLUTE environment.

5.1.2 Looking at Semantic Structures as Graphic Objects

An intermediate representation produced in the translation process such as an analysis result or a transfer result has a complex structure as mentioned earlier and it is often too complex for an experimenter to handle. This is especially true when the processed sentence has a complex syntactic structure including such thing as embedded clauses or certain types of coordination.

The number of frames which make up a complete semantic structure sometimes extends to more than one hundred. Such a structure is almost impossible to display in detail even on a high-resolution bit-map display. Even if it were possible, its graphical structure becomes so complex that it is quite difficult to understand its details. Therefore, some abstraction facility is required that can help to reduce the complexity of the displayed objects and thus capture the whole structure at a glance.

In the RESOLUTE environment, a semantic structure is displayed in a Frame-Network window. For example, after the whole translation program is invoked, two intermediate structures, an analysis result and a transfer result, are displayed in each Frame-Network window. A Frame-Network window is controlled by an abstraction facility that is used to limit the complexity of objects on the display. The facility controls it so that only the necessary part of a semantic structure appears. This enables an experimenter to grasp the significant information at a glance. The decision as to what part of the frame should be displayed depends on the meta-knowledge the frame retains.

For example, suppose that an experimenter is debugging transfer rules related to a deep case and wants to pay his attention only to the deep case relationships under the matrix sentence. In this case, what the experimenter wants to look at is only information related to the deep case assignments of both the analysis and the corresponding transfer results. The Frame-Network windows can be controlled to display only such information when the system is in the CASE-ANALYSIS mode, which the user can choose or specify. In this mode, a Frame-Network window displays only a partial semantic structure related to the case analysis.

A Frame-Network window has other facilities for supporting visible controls. The objects in a Frame-Network window can accept commands not only for modifying structures but also for controlling their visibility. When an experimenter wants to look at more information than that currently displayed, he can display hidden sub-structures by specifying or pointing at a current node in the Frame-Network window displayed with abstraction by using the mouse. Moreover, he can display a frame-table window that displays information on the node in detail.

The LUTE assigns a score to each meaning structure when it produces more than one. The score represents the grade on its accuracy or correctness as a meaning structure for the source sentence analyzed. Such scores are also displayed on the frame-table window so that the experimenter can know with which he should start the examination.

By using these facilities, an experimenter can judge the adequacy of a semantic structure quickly and accurately.

5.1.3 Editing and Processing Semantic Structures

In the RESOLUTE environment, a complete semantic structure displayed on a window or any part of it can be edited independently and then applied to some processes simultaneously. For example, when a source language analysis program is recognized as assigning incorrect deep cases, it is possible to edit them and get the correct deep cases independently from other parts

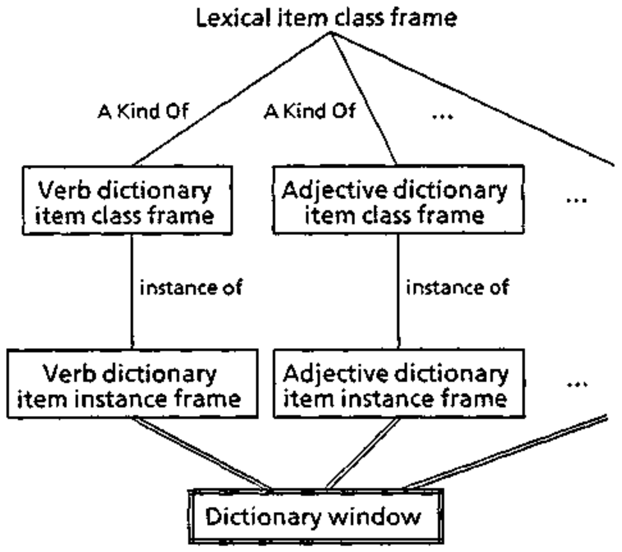

Lexical item class frame
A Kind Of
A Kind Of
...
Verb dictionary item class frame
Adjective dictionary item class frame
...
instance of
instance of
Verb dictionary item instance frame
Adjective dictionary item instance frame
...
Dictionary window

Fig.6 Dictionary editing sub-system

and then supply the edited semantic structure to the transfer sub-process for an immediate check of the transfer program by postponing the fixing of the analysis process. By using this facility, the transfer rules can be applied to the semantic structures just modified temporarily or intentionally. This provides a way for debugging each module separately.

This editing process is done in an object-oriented way such as: 1) choosing a node to be processed by pointing with the mouse; and 2) selecting a command from the menu of the commands that the node can accept. For example, the "transfer" command appears only on the menu table for transferable nodes but not on the menu table for non-transferable nodes. These facilities are used in the following cases:

1) Discovering causes of inadequacy in the transfer or generation processes: Suppose that an experimenter reaches at a judgment that an inadequacy occurred in the transfer program but he can not identify which part of the program caused it. In such a case, the cause of the inadequacy will be found by transferring some parts of the semantic structure separately and then checking their results individually.

2) Processing after editing lexical items or rules: In the similar situation mentioned above, when the experimenter recognized that the inadequacy arose from the incorrect or insufficient descriptions of the lexical items or rules, he can modify them and then check the effect of the editing on the same semantic structure immediately.

3) Processing after editing semantic structures: Suppose that an analysis result is incorrect. After modifying the resulted semantic structure, the evaluation of the modification can be executed immediately by applying the transfer process to the modified structure.

4) Generating an analysis result in the reverse way: One of the easiest ways of assuring the adequacy of an analysis result is by generating a sentence from the semantic structure produced by the analysis. This means that the experimenter applies a Japanese generation program to a Japanese analysis result. In Japanese, for example, ellipsis of predictable constituents (including zero-pronouns) happens very frequently. Thus, it is important for the analysis process to fill them up since it is necessary to produce an equivalent English sentence which does not allow the ellipsis any more. Japanese analysis by the LUTE fills them up by reasoning using knowledge. To assure a filled-up result, the Japanese analyzed structure is applied to the Japanese generation program to produce a sentence without ellipses. Thus, we can check it without examining the complicatedly represented intermediate structure.

Since the LUTE can translate both directions from and to between Japanese and English, such tests mentioned can be executed on both directions simultaneously.

A window made by the RESOLUTE will survive until an experimenter deletes it by his own intention. Consequently, comparing a window's currently processed contents with previously processed material makes clear the difference between the results.

5.2 Dictionary Editing Sub-system

This sub-system is invoked from the machine translation sub-system as mentioned earlier. The sub-system consists of dictionary windows and knowledge on how to edit and check dictionary items. The dictionary window is a kind of frame-table window with special commands for editing dictionaries. A dictionary window is created under the condition described in each dictionary item description frame.

Knowledge fragments on how to edit are described in dictionary item class frames and they are inherited by their instance frames, that is, dictionary item description frames. Such kinds of knowledge concerns the conditions as follows:

1) Restrictions on the set of feature bundles required to be a complete description of the lexical item.

2) Conditions imposed on feature-values to be assigned.

3) Attached procedures invoked whenever a feature-value is given or modified or removed. These are to keep consistency throughout the whole description.

4) Attached procedures such as how to input a feature-value and how to display the feature-value.

When an experimenter selects the "Input Feature-Value" command to add a feature-value into a dictionary item frame, he can input data via a kind of menu window or another type of window according to knowledge 4 listed above. After he assigns or selects a value, the value will be examined for whether it satisfies the requirements related to 2 mentioned above. If it satisfies the requirements, the value is written into the lexical item, and then the attached procedures 3 shown above are invoked to check and keep consistency over other descriptions. Dictionary editing by using this knowledge reduces a dictionary editor's effort reasonably. The construction of the data in the dictionary editing sub-system is shown in Figure 6.

6. Conclusion

An interactive environment, the RESOLUTE, in terms of the machine translation experiments is presented. The environment has the following characteristics:

1) The environment visualizes the process of the machine translation experiments.

2) The environment provides a facility for an experimenter to manipulate the knowledge he requires in ways according to his task and his level of knowledge on the machine translation system he uses.

3) The environment gives a feeling of direct manipulation of objects by displaying knowledge fragments graphically, and it enables a user to manipulate these graphic objects interactively.

Because of these features, the experimenter can conduct his machine translation experiments efficiently and effectively in the environment. For example, we could reduce the checking time of a semantic structure from one hour to less than ten minutes.

The RESOLUTE environment has been implemented on the Symbolics Lisp machine as mentioned earlier, and used for examining the LUTE system also implemented on the same machine, and then accumulating data into the Knowledge-Base. Figure 7 shows a snapshot of the experience on the RESOLUTE.

References

[1] Nomura H.: "A Model for Knowledge Acquisition Systems with Problem Solver", Proceedings of the Fourth International Congress of Cybernetics and Systems, 1978

[2] Nomura, H.: "Terminology Banks and Dictionaries in Japan and their Computer Processing", Term Banks for Tomorrow's World, Proceedings of ASLIB Conference '82, Translating and the Computer 4, 1982

[3] Shimazu, A., S. Naito, H. Nomura: "Japanese Language Semantic Analyzer based on an Extended Case Frame Model", Proceedings of the 8th International Joint Conference on Artificial Intelligence, 1983

[4] Saito, Y., H. Nomura: "JMACS : Screen Editor for Japanese and English and Programs — A Kernel for Unified Research Environment —", Proceedings of the International Conference of Text Processing with a Large Character Set, 1983

[5] Nomura, H.: "Towards the High Ability Machine Translation", Joint Japanese-European Work Shop on Machine Translation, 1983

[6] Iida, H., K. Ogura, H. Nomura: "A Case Analysis Method Cooperating with ATNG and Its Application to Machine Translation", Proceedings of the 10th International Conference on Computational Linguistics, 1984

[7] Naito, S., A. Shimazu, H. Nomura: "Classification of Modality Function and its Application to Japanese Language Analysis", Proceedings of the 23th Annual Meeting of the Association for Computational Linguistics, 1985

[8] Nomura, H.: "Experimental Machine Translation Systems: LUTE", Second Joint European-Japanese Workshop on Machine Translation, 1985

[9] Nomura, H.: " Modeling and Representative Framework for Linguistic and Non-linguistic Knowledge in Natural Language Understanding", Germany-Japan Science Seminar, 1986

[10] Nomura, H., H. Iida: "English-Japanese Interactive Translation System : LUTE-AID", Proceedings of the International Conference on the State of the Art in Machine Translation, 1986

[11] Nomura, H., S. Naito, Y. Katagiri, A. Shimazu: "Translation by Understanding: A Machine Translation System LUTE", Proceedings of the 11th International Conference on Computational Linguistics, 1986

[12] Starosta, S., H. Nomura: "Lexicase Parsing: A Lexicon-Driven Approach to Syntactic Analysis", Proceedings of the 25th Annual Meeting of the Association for Computational Linguistics, 1987

[13] Shimazu, A., S. Naito, H. Nomura: "Semantic Structure Analysis of Japanese Noun Phrases with Adnominal Particles", Proceedings of the 25th Annual Meeting of the Association for Computational Linguistics, 1987

[14] Nomura, H.: "Technical Prospects of Machine Translation", Proceedings of the MT SUMMIT, 1987

[15] Magnetic Tape of Iwanami Kokugo Jiten (A Japanese National Language Dictionary edited by M. Nishio, E. Iwafuchi, S.Mizutani), Iwanami, 1979

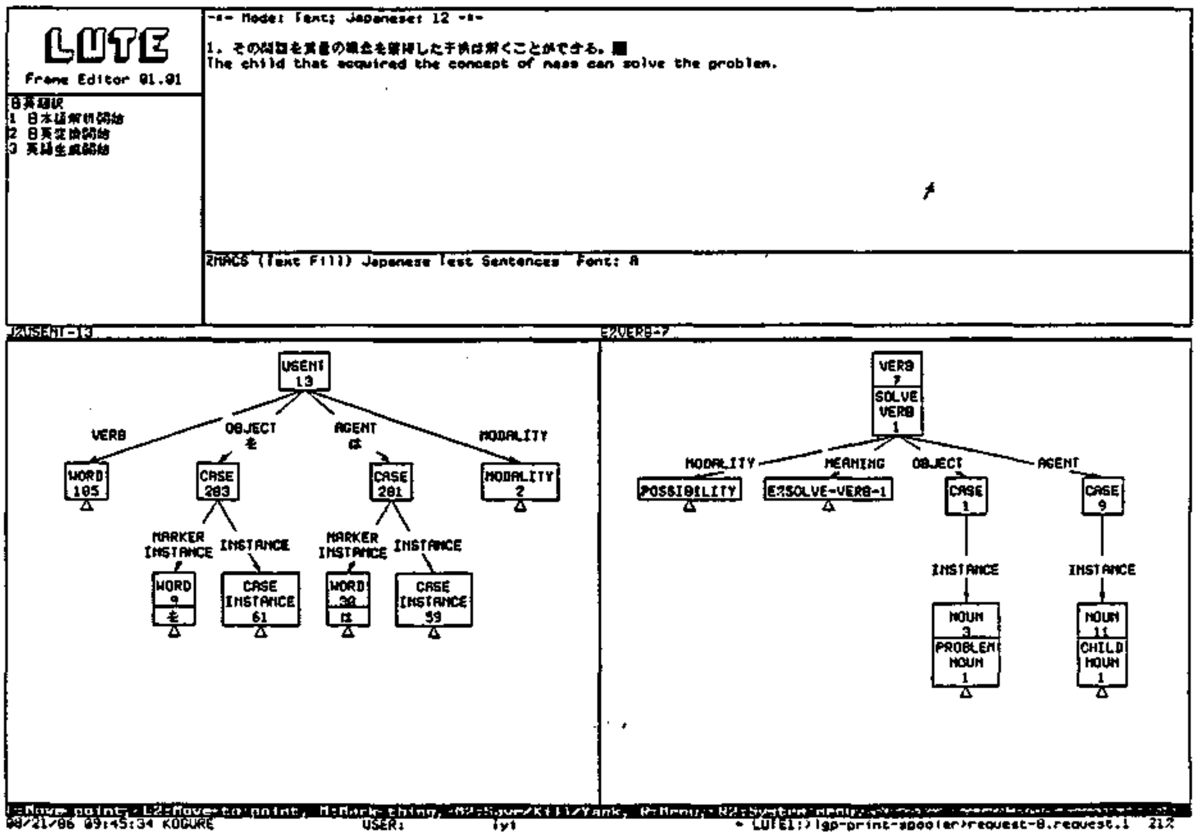
[16] Minsky, M.: "A Framework for Representing Knowledge", in Winston, P. H. (ed), The Psychology of Computer Vision, McGraw-Hill, 1975

[17] Roberts, R. B., P. I. Goldstein: "The FRL Primer", MIT memo 408, 1977

[18] Roberts, R. B., P. I. Goldstein: "The FRL Mannual", MIT memo 409, 1977

[19] Stallman, R. M.: "EMACS The Extensible, Customizable Self-Documenting Display Editor", SIGPLAN Notice, 16, 6, June 1981

[20] Symbolics Inc.: "Lisp Machine Mannual", 1985


```
-*- Mode: Text; Japanese: 12 -*-

1. その問題を質量の概念を獲得した子供は解くことができる。
The child that acquired the concept of mass can solve the problem.

ZMACS (Text Fill) Japanese Test Sentences  Font: A

08/21/86 09:45:34 KOGURE          USER:        Tyi          * LUTE1:>lgp-print-spooler>request-8.request.1  21%
```

Fig. 6 A snapshot of the experience by RESOLUTE

After a Japanese-English translation command was applied to the sentence 1 displayed in the ZMACS buffer, both meaning structures as an analysis result and a transfer result are displayed on each frame network window respectively. This makes it easy for an experimenter to compare or check Japanese and English meaning structures. A translated sentence is displayed on the line following the source sentence.