STUDIA PHONOLOGICA XXIV (1990)

A Speech Interface to an Information Retrieval System

Yasuhisa NIIMI and Yutaka KOBAYASHI

SUMMARY

This paper describes a speech interface to an information retrieval system. It consists of three main components; speech recognition system, command generator and response generator. The speech recognition system accepts a spoken command of Japanese sentence and passes the recognized sentence to the command generator, which translates it into a formal query command to operate the information retrieval system. The response generator receives retrieved data and produces a response to the user in a written sentence. We proposed a basic strategy in construction of the speech recognition system. It is that the top-down linguistic hypothesis is made at the lexical level while the verification of top-down hypotheses is made by using the unit independent of the word, the phonetic string bounded by robust phones (phones which can reliably be detected) in order to reduce the misrecognition of short function words which are easy to be phonetically affected by neighboring words. The interface was tested by using the speech corpus of 53 sentences spoken by each of three male speakers. The average rate of sentence understanding was 83.0%. Since few errors were found in short words, it could be concluded that the proposed strategy has been successful.

1. INTRODUCTION

This paper describes a speech interface we are developing as the main component of a speech dialogue system. It consists of a speech recognition system and a natural language interface to an information retrieval system. The natural language interface is composed of a command generator and a response generator. The speech recognition system accepts a spoken command of Japanese sentence and passes the recognized sentence to the command generator, which translates it into a formal query command to operate the information retrieval system. The response generator receives retrieved data and produces a response to the user in a written sentence.

Yasuhia NIIMI (新美康永): Professor, Department of Electronics and Information Science, Kyoto Institute of Technology.

Yutaka KOBAYASHI (小林 豊): Assistant· Department of Electronics and Information Science, Kyoto Institute of Technology.

The authors have done the research under the direction of Dr. Shuji Doshita, Professor of Information Science, Kyoto University.

Japanese has several short words like postpositions and auxiliary verbs. The misrecognition of these types of words might lead to the error which is not relievable by the latter semantic processing. In order to avoid this kind of errors, we proposed a basic strategy in construction of the speech recognition system. It is that the top-down linguistic hypothesis is made at the lexical level while the verification of top-down hypotheses is made by using the unit independent of the word, the phonetic string bounded by robust phones. The robust phone is a phone, such as an unvoiced fricative or a short pause before unvoiced plosives, which can reliably be detected. Owing to this strategy, it makes possible to cope with inter-word coarticulation as well as intra-word coarticulation.

The interface was tested by using the speech corpus of 53 sentences spoken by each of three male speakers. The average rate of sentence understanding was 83.0%. Most of errors occurred at long compound words, but few errors were found in short words. This shows that the proposed strategy is as effective as we have expected.

2. Speech Interface and Task Specification

2.1 *Speech interface and database*

We suppose the user of the speech interface reported here could utter questions about the contents of a relational database, and get responses as written sentences. Fig. 1 illustrates the configuration of the speech interface. It is composed of three main modules: speech recognition system, command generator and response generator. The speech recognition system recognizes an utterance of the user and passes to the command generator the result of recognition as a string of words. The command generator converts this sentence into a query command to the database sys-

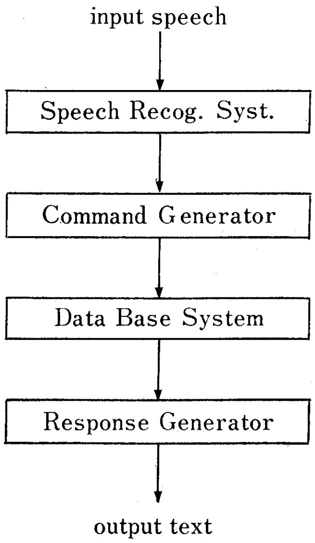


Fig. 1. A configuration of the speech interface.

tem. The response generator accepts the result of retrieval and produces a response to the user, which is shown on a computer terminal.

The database contains information on temples, shrines, museums, universities and hotels in Kyoto city. A relational database consists of tables like the one shown in Fig. 2. The name of each table is considered as a relation. A column contains values of relation components, called an attribute. A row corresponds to a record, an instance of the relation, which is a list of the values of the attributes. Fig. 3 shows an example of a relational table for temples and shrines.

Name of the table

attribute name 1	*attribute name 2*	:
attribute value 11	attribute value 21	:
attribute value 12	attribute value 22	:
:	:	:
:	:	:

Fig. 2. A relational table.

Table of temples.

name	*location*	*fee*	*open*	*close*	
kinkakuji	kinugasa	100	9:00	17:00	
ginkakuji	higasiyama	150	8:30	17:00	
rokuonin	uzumasa	300	9:00	17:00	

Fig. 3. An example of a relational table for temples and shrines.

Since the current database relevant to the task is a small portion containing less than one hundred records out of the original database of two thousand records, we declared each instance of the relations as a Prolog clause instead of using a commercial database system.

2.2 *Task syntax and semantics*

Fig. 4 shows a few Japanese sentences of sightseeing queries. Fig. 5 illustrates syntactic rules for query sentences by the definite clause grammar (DCG)—the context free grammar augmented by the semantic constraint. The terminal symbols of this grammar are underlined and the nonterminal ones are not. The sym-

Kousendai no denwabangou o oshiete kudasai.
(May I have the telephone number of Kousendai?)
Kinkakuji no haikanryo wa hyakuen desuka?
(Is the entrance fee of Kinkakuji temple 100 yen?)
Nyuujouryou ga nihyakuen ika de juuji yori hayaku hiraku
bijutukan o oshiete kudasai.
(Are there any museums which open before 10 o'clock and
whose entrance fee is less than or equal to 200 yen?)

Fig. 4. Examples of acceptable sentences.

$$
\begin{array}{rcl}
s & \rightarrow & cp(C), \underline{fn(wa)}, ppc(C,q). \\
 & | & cp(C), \underline{osh}. \\
 & | & tp(T), \underline{fn(wa)}, ppt(T,q). \\
 & | & tp(T), \underline{osh}. \\
tp(T) & \rightarrow & \underline{tcst(T)}. \\
 & | & \{cat(T,C)\}, cp(C), \underline{fn(hap)}, \underline{tn(C,T)}. \\
 & | & \{cat(T,C)\}, cp(C), \{verbf(C,F,T,V)\}, \\
 & & \underline{fn(F)}, \underline{verb(V,n)}. \\
cp(C) & \rightarrow & \underline{cst(C)}. \\
 & | & \underline{cn(C)}, \underline{uch}, ppc(C,rt), \underline{mono}. \\
 & | & ppc(C,rt), \underline{cn(C)}. \\
ppc(C,S) & \rightarrow & \{cat(T,C)\}, sp(T,C), ppt(T,S). \\
 & | & \{cat(T,C)\}, cfp(T,F), \underline{adj(T,F,ry)}, \\
 & & \{verbf(C,T,\_,V)\}, \underline{verb(V,S)}. \\
 & | & \{cat(T,C)\}, dp(T), \{verbf(T,F,C,V)\}, \\
 & & \underline{fn(F)}, \underline{verb(V,S)}. \\
ppc(C,q) & \rightarrow & \{cat(T,C)\}, \underline{qn(T)}, \\
 & & \{verbf(T,F,C,V)\}, \underline{fn(F)}, \underline{verb(V,q)}. \\
 & | & \{cat(T,C)\}, qav(T), \{verbf(T,F,C,V)\}, \underline{verb(V,q)}. \\
ppt(T,S) & \rightarrow & cfp(T,F), \underline{adj(T,F,S)}. \\
 & | & dp(T), \underline{aux(T,S)}. \\
 & | & \underline{qn(T)}, \underline{aux(T,S)}. \\
sp(T,C) & \rightarrow & \underline{tn(T,C)}, \underline{fn(subj)}. \\
 & | & \{verbf(C,F,T,V)\}, \underline{verb(V,n)}, \underline{fn(subj)}. \\
cfp(T,F) & \rightarrow & tp(T), \{compf(T,F)\}, \underline{fn(F)}. \\
 & | & \{cat(T,C)\}, cp(C), \{compf(T,F)\}, \underline{fn(F)}. \\
dp(T) & \rightarrow & tp(T). \\
 & | & cfp(T,F), \underline{an(T,F)}.
\end{array}
$$

Fig. 5. Syntax and semantics of the sight-seeing task.

bols in the parentheses are semantic markers. The terms enclosed by the brackets express the predicates for the semantic constraint. The vocabulary contains 248 words. The test set perplexity is 8.3 for the current task.

2.2.1 *Syntactic structures and categorization of words*

The nonterminal symbols describing the task grammar reflect the specific features of the database queries. Besides the starting symbol s, we introduced seven nonterminals.

cp: noun phrases which specify the search key in the relational tables, e.g., "Kinkakuji (Kinkakuji temple)", "400 en de hairu koto no dekiru hakubutsukan (the museums whose entrance fee is cheaper than 400 yen)".

tp: noun phrases which specify the attributes in the relational tables, e.g., "dobutsuen no nyujoryo (the entrance fee of the zoo)", "Kinkakuji ga taterareta toshi (the year when Kinkakuji temple was constructed)".

ppc, ppt: predicative phrases containing a verb, which terminate a sentence or modify a noun phrase. ppc and ppt modify cp and tp, respectively.

sp: noun phrases which specify the main topic in a ppc phrase.

cfp: phrases which represent the object of comparison, e.g., "300 en yori (than 300 yen)", "Kokedera to (as Kokedera temple)".

dp: noun phrases which specify case fillers in ppc or ppt phrases, e.g., "1000 nen mae (ni taterareta) ((built) before 1000 A.D.)".

We categorized the words in the vocabulary into fourteen task specific categories. Especially the nouns were subcategorized according to the concepts related to the database. The words other than the noun were classified according to the standard school grammar. The categorization of the noun is described below.

cn: names of relational tables in the database, e.g., "shaji (shrine-temple)", "daigaku (university)".

tn: names of attributes in a relational table in the database, e.g., "haikanryo (entrance fee)", "denwabango (phone number)".

cst: proper nouns which appear in the column of the name attribute, e.g., "Ginkakuji", "Miyako hoteru (Miyako hotel)".

tcst: nouns which appear in the columns other than the name attribute, e.g., "hyakuen (one hundred yen)".

qn: interrogative nouns, e.g., "nani (what)", "dare (who)", "doko (where)".

2.2.2 *Semantic markers and case structure*

In order to judge the possibility that a particular word pair may appear in a particular syntactic structure, we introduced parameters playing roles of semantic markers to the syntactic rules. The semantic constraint between noun phrases can be calssified into two categories in the present task.

(1) "A no B (B of A)" imposes a constraint that B is an attribute name of the record specified by the noun phrase A. A must be a noun phrase generated from cst or cp, and B from tn or tp. Moreover, both specified items must exist in a relational table.

(2) "C wa D desuka (Is C D ?)" implies that C and D refer to an attribute name and its value respectively. Therefore, C must be a noun phrase generated from tn or tp, and D from tcst or qn. In addition, both C and D must be relevant to the same attribute of the database, for example, "time" and "8:30".

We use semantic markers C and T for describing these constraints. The syntactic categories cn and cst are given the marker C as a parameter which holds the type of the relevant relational tables. The syntactic categories tn and qn are given the marker T as a parameter which holds a higher concept of the attribute name. The predicate {cat(T,C)} in Fig. 5 expresses the constraint that the value of T be an attribute of the relational table specified by the marker C.

We describe the semantic (or co-occurrence) relation between an adjective or a verb and other phrases in a sentence by the semantic marker and the case grammar. In the present task the case frame of both an adjective and a verb has two case slots. Syntactically one of them acts as the subject of a sentence.

Since the predicative pharase including an adjective is usually interpreted as a comparison operator, two case slots must be filled by pharases with the same semantic marker. Thus the adjective can be characterized by a semantic marker. The slot filler other than the subject of a sentence forms the predicative pharase with a postposition and an adjective itself. This co-occurrence relation between the adjective and the postposition can be tabulated. The predicate {comp(T,F)} in Fig. 5 expresses such a relation, where the parameter F denotes the class of postpositions.

In the case of verbs the two slot fillers are described by different semantic markers; one by T and the other by C. The predicate {verbf(T,F,C,V)} in Fig. 5 expresses a relation among these two semantic markers T and C, the class of verbs (denoted by V), and the postoposition which can be attached to the slot filler other than the subject.

3. The Speech Interface

3.1 *The configuration of the speech interface*

The speech interface is composed of five components: acoustic processor, matcher, phonological and linguistic processors, and controller. They are hierarchically organized as shown in Fig. 6.

The acoustic processor samples an input speech at 10 kHz and digitizes it with 12 bit precision. Then it performs signal processing based on the linear predictive

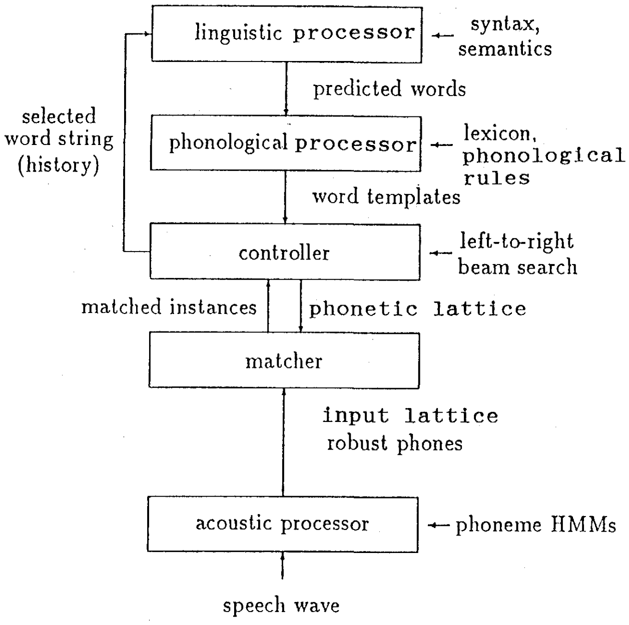


Fig. 6. A configuration of the speech recognition system.

coding (LPC). For each 10 ms of the speech the processor calculates the LPC-cepstral coefficients (LPC-CEPs), their time derivatives (DELTA-CEPs), and a pair of short-term energy and its time derivative (POWs). The time derivatives are calculated as differences of the LPC-CEPs and the short-term energy between the preceding and following frames. These three sets of parameters, LPC-CEPs, DELTA-CEPs, and POWs are vector-quantized separately with 256 code words each. Thus the input speech is transformed into a sequence of trigrams of different code words. Finally, the processor recognizes broad phonetic classes by using the discrete phoneme-based hidden Markov models(HMMs) with three states, and produces a phonetic lattice, which is called an input lattice below. Table 1 shows the 17 broad phonetic classes which are currently identified. Of these broad phonetic classes, the unvoiced fricative and the silence are used as the robust phone since they can reliably be detected.

The linguistic processor makes a top-down hypothesis which consists of a set of words capable of following a partial sentence selected by the controller. A partial sentence is a string of already recognized words covering the beginning portion of an input utterance. The hypothesization is based on the syntactic and semnatic analyses of the given partial sentence. As stated in the section 2.2, the syntactic rules for input sentences are described by the context free grammar, and semantic constraints are given by a set of semantic markers and the case grammar. Both are integrated into a definite clause grammar. The algorithm to make a top-down hypothesis is a version of BUP which is modified for the efficient top-down word prediction. We have developed a procedure to derive mechanically a program written in Prolog working as a word predictor from a definite clause grammar[1].

The phonoligical processor computes phonetic variants and the standard duration of a word. Given the phonetic representation of a partial sentence and one of

Table 1. 17 broad phonetic classes.

vowels		consonants	
A	a	N	m n ng
I	i	Z	z dz
U	u	R	r
E	e	D	b d
O	o	G	g
semivowels		S	s sh ts ch
Y	j	H	h
W	w	K	k
other		P	p t
Q	slience		

the words following it, this processor applies phonological rules to the word with the last portion of the partial sentence as a phonetic context, resulting in a lattice describing phonetic variants of the given word. We call this lattice a phonetic lattice below . During the application of rules, it also computes the standard duration of the word by summing up the length of each phoneme included in the word, and locates the robust phones in the phonetic lattice. A word is described by the orthographic transcription while the acoustic processor can identify only 17 broad phonetic classes. The processor translates the orthographic transcription of a word into the system-dependent phonetic transcription.

Given a phonetic lattice bounded by two robust phones and the current segment of the input lattice, the matcher first looks for candidate robust phones in the input lattice within twice the standard duration of the phonetic lattice. After deciding a few candidate intervals, the matcher calculates the distance between both lattices using the lattice vs. lattice DP matching algorithm[2]. The distance is returned to the controller with the end segment of the matched interval.

The controller invokes other components while extending likely partial sentence hypotheses. A partial sentence is a string of words which have been identified in the beginning portion of an input utterance. The controller preserves a set of partial sentences in the form of a tree. The fundamental strategy of the speech interface is that the linguistic top-down hypothesis is made at the word level, while the verification is made by using a unit independent of the word, that is, a phonetic string (lattice) bounded by two robust phones. Thus the neighborhood of leaves of the search tree is not usually verified. The controller selects one of these parts and asks other components to verify it. The order of the selection is based on the strategy which will be stated in the section 3.2.

3.2 *Search strategy*

3.2.1 *The structure of the search space*

A top-down hypothesis to be matched against a part of the input acoustic data is a sub-word lattice bounded by two robust phones of which the locations are independent to word boundaries, while a linguistic top-down hypothesis is given as a string of words. In order to facilitate this complicated matching scheme, the search space of the system is constructed with three levels of trees: category, word, and phoneme trees. The relation among these trees is illustrated in Fig. 7.

The category tree preserves the syntactic description of the partial sentences. A node of this tree, called category node, corresponds to a linguistic category of a word. Each of the category nodes keeps a partial parse tree, called parse history, resulting from the linguistic analysis of a sequence of linguistic categories corresponding to the path from the root to that node.

The word tree preserves the description of the partial sentences by the word. A node of the word tree, called word node, correspnds to a word and has a pointer to

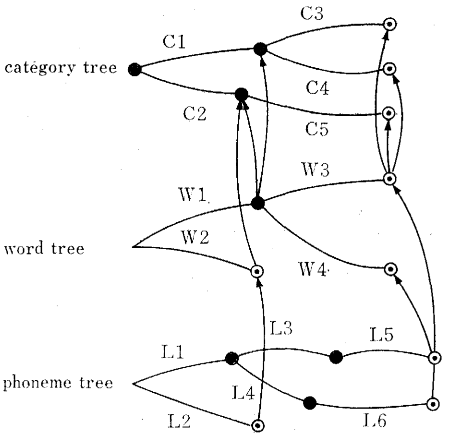


Fig. 7. The structure of the search tree.

a category node, that is, the category to which the word belongs.

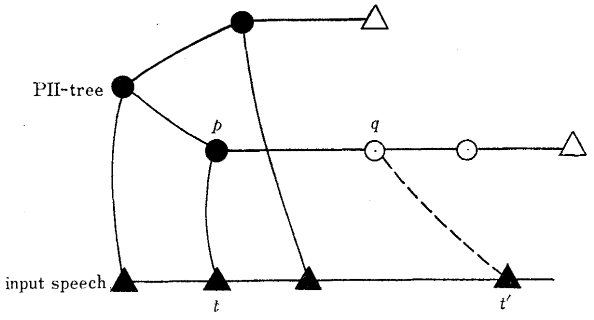
In the phoneme tree, only a leaf node, a node without a suceessor, has a pointer to a word node. A node other than a leaf node is independent of a word node, but represents a phonetic lattice terminated by a robust phone in a word or between words. The phoneme node is classified into three: verified, unverified, and leaf nodes. A node is called a verified node when the phonetic lattice corresponding to the path from the root to that node has been matched against a part of the input lattice, and an unverified node when the lattice has not verified and if it is not a leaf node.

A pointer is spanned from a location of the robust phone in the input lattice to each of different verified nodes. This means that the input lattice from the beginning to the robust phone has been matched with the phonetic lattice from the root to each of the pointed nodes. The confidence scores resulting from these matching operations are sorted and preserved in a score array associated with the location of the robust phone.

3.2.2 *The search strategy*

Fig. 8 illustrates a relation between a part of the phoneme tree and a part of the input lattice. Black circles indicate verified nodes, white circles unverified nodes, and white triangles leaf nodes. Black triangles indicate locations of the robust phones of the input lattice. Of the unverified nodes, the ones of which the parent node is a verified node can be matched against the input lattice. These nodes are called verifialbe nodes. In the figure they are identified as white circles which are successors of black circles. The black triangle pointing such a black circle gives the beginning instance of acoustic data to be matched with these lattices. We call it an expanadble robust phone.

After these definitions we can state the search strategy, that is, the strategy to decide which to select for the verification among the verifiable nodes. Our strategy



● node already matched (verified node),

○ node ready for matching (unverified node),

△ ·leaf node,

▲ robust phones.

Fig. 8. A relation between the phoneme tree and the input lattice.

is based on the time synchronous beam search method, and can be stated as follows.

(a) The robust phone whose instance is earliest among the expandable robust phones is selected, and designated by t.

(b) If it points verified nodes, one of them (denoted by p) is selected and the following steps (b–1) and (b–2) are repeated after removing the pointer from t to p. When there is no verified node pointed by the robust phone t, the control goes to the step (c).

(b–1) If the selected node p has an unverified node as a seuccessor, the unverified node (denoted by q) is verifiable. The phonetic lattice corresponding to the path from p to q is passed to the lexical processor for the verification, which returns a confidence score as the result of the verification.

(b–2) If p has no successor but a leaf node, it is expanded. Following from the structure of the search space mentioned in the section 3.1.1, a link exists from the leaf node to some node of the category tree. The request to make top-down hypotheses is issued to the linguistic processor by giving the parse history of that category node. Then the phonological processor is requested to exand the leaf node by applying phonological rules to the resulting linguistic hypotheses. This process continues until an unverified node appears, and then the control goes to the step (b–1).

(c) Confidence scores obtained in the step (b–1) are registered. A confidence score represents goodness in matching the phonetic lattice corresponding to the path from p to q against a part of input lattice between the instance t and the instance (denoted by t′) of some robust phone following t, which is decided by the matcher. If this score is greater than the predetermined value, it is stored in the score array at the instance t′ and a pointer is linked from t′ to q, which is then marked as verified. The values in the score array are sorted and the ones outside of the beam are discarded.

(d) The above cycle (a) through (c) continues until no expandable robust phone exists.

4. Natural Language Interface

4.1 *The outline of the natural language interface*

As shown in Fig. 1, outputs of the speech interface are passed to the command generator, interpreted as query commands therein and supplied to the database system. The retrieved information from the database is passed to the response generator, and translated into Japanese sentences to present to the speaker. The speech interface gives to the command generator a string of words as its output. The command generator analyzes the string syntactically and semantically, and transforms it into an internal representation. Although the speech interface makes the syntactic and semantic analyses of an utterance, their purposes are to make top-down hypotheses in order to reduce the search space in recognizing the utterance.

The internal representation of an utterance should give its semantic interpretation as well as have a form which can easily be converted into a command to the database system. Although we do not assume that the speech interface be connected to a specific database system, we have designed the internal representation so that it could easily be translated into a command of an existing query language, for example, SEQUEL-2[3].

Queries described in a natural language are generally classified into the two forms as shown in Fig. 9. The sentence (a) in Fig. 9 requires a value of an item of a record in some relational table, while the sentence (b) requires 'yes' or 'no' as an answer. Given these answers from the database system, the response generator produces a response to the speaker, referring to the internal representation of the input query sentence. Currently responses are presented as written sentences on the computer terminal.

```
(a) Kinkakuji no haikanryo wa ikura desuka ?
    (How much is the entrance fee of Kinkakuji temple ?)

(b) Kinkakuji no haikanryo wa 500 en desuka ?
    Is the entrance fee of Kinkakuji temple 500 yen ?)
```

Fig. 9. Two types of query sentences.

4.2 *The internal representation of an utterance*

The syntax for the internal representation is shown in Fig. 10. An internal form of an utterance is represented by a function and one or two quadrigrams, that is, lists of four terms. The function indicates an action of the information retrieval

```
semantic representation of a sentence
    ---> {function, quadrigram}
         |{function, quadrigram, quadrigram}
function
    ---> find | comparison operator
comparison operator
    ---> = | == | < | > | =< | >= | /==
quadrigram
    ---> [C, N, T, V]
C   ---> SM1
N   ---> SM1 | quadrigram
T   ---> SM2
V   ---> SM2 | quadrigram | comparison expression
comparison expression
    ---> (comparison operator variable1 variable2)
SM1 ---> value of the semantic marker C | null
SM2 ---> value of the semantic marker T | null
```

Fig. 10. Syntax for the semantic representation of a sentence.

system, corresponding to either of the two forms of the query sentences shown in Fig. 9. The query (a) includes an interrogative and requires to retrieve a value from a relational table, and the query (b) requires a decision on whether a given value, for example, 500 yen, is correct. The functions corresponding to these two are 'find' for the query (a) and '==' (equal to) for the query (b). Other six comparison operators, '/==' (not equal to), '>', '<', '>=', '<=', and '=' (substitution) are used for the latter.

The quadrigram is a list of the following four terms.

(a) C: a name of a relational table of the database.

(b) N: an indicator of a record of the relational table.

(c) T: an attribute of the record.

(d) V: a value of the attribute.

Three of these four terms are extracted from an input query to construct a query command to retrieve the rest one. The terms N and T can be defined by a quadrigram as shown in Fig. 10. This means that a query sentence can include embedded sentences.

4.3 *Semantic analysis*

The semantic analysis of an input sentence parallels to the syntactic one. The semantic definition of a word is given by an incomplete quadrigram, a function, or a combination of both. An incomplete quadrigram is the one in which some of four terms are not specified. Examples of the semantic definitions of words and terminal symbols are shown in Fig. 11.

Each nonterminal of the grammar illustrated in Fig. 5 has a semantic form

```
cn  ---> [C,_,_,_]  ("shaji"     ---> [temple,_,_,_])
tn  ---> [_,_,T,_]  ("haikanryo" ---> [_,_,fee,_])
cst ---> [C,N,_,_]  ("Kinkakuji"  ---> [temple,Kinkakuji,_,_])
adj ---> {comparison operator,T}
                    ("yasui"     ---> { >,fee})
```

Fig. 11. Semantic definitions of terminal symbols and words.

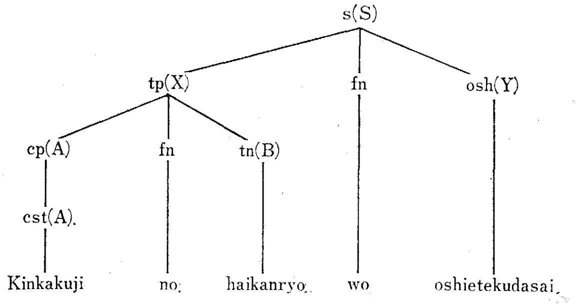


Fig. 12. The analysis of sentence "Kinkakuji no haikanryo wo oshietekudasai".
A =[temple, Kinkakuj, —, —]
B =[temple, —, fee, —]
X =[temple, Kinkakuji, fee, —]
Y ={find}
S ={find, [templel Kinkakuji, fee, —]}

similar to the semantic definition of a word. There is a semantic rule corresponding to a syntactic rule. It gives the way to construct the semantic form of the nonterminal at the left hand side of the syntactic rule from those of the syntactic categories at the right hand side. Both a semantic rule and a syntactic rule are applied at the same time during the linguistic processing. Fig. 12 shows the process of the syntactic and semantic analyses of a sentence "Kinkakuji no haikanryo wo oshietekudasai. (Please tell me the entrance fee of Kinkakuji.)" The two words "Kinkakuji" and "haikanryo (the entrance fee)" have the semantic forms [temple, Kinkakuji,—,—] and [temple,—, fee,—] respectively. These forms are combined to build the semantic form of the phrase "Kinkakuji no haikanryo (the entrance fee of Kinkakuji)", resulting in the form [temple, Kinkakuji, fee,—].

4.4 *Generation of responses*

The information retrieval system returns a complete quadrigram when the function of the query command is "find", and "yes" or "no" when it is a comparison operator. The semantic representation of a query sentence and this returned value are passed to the response generator. The response generator has a set of patterns of the response, each corresponding to the pattern of the semantic representation of a query sentence. Fig. 13 shows examples of the patterns of the response. The response generator generates a response using these patterns and passed information.

```
(1) {find, [C,N,T,V]}  (V is unknown.)
        response: "N no tn(T,C) wa V desu"
(2) {find, [C,N,T,V]}  (N is unknown.)
        response: "tn(T,C) ga V dearu C wa N desu"
(3) {op, [C1,N1,T1,V1], [C2,N2,T2,V2]}
        response: "N1 no tn(T1,C1) wa {N2 no tn(T2;C2)¦V2}
                   yori adj(op,T) {desu¦dewaarimasen}"
```

Fig. 13. Examples of patterns of the response.
op: comparison operator.

Consider the query sentence, "Kinkakuji no haikanryo wo oshiete kudasai". The semantic representation of it is {find, [temple, Kinkaji, fee,—]} as shown in Fig. 12. The response pattern for it is "N no tn (T,C) wa V desu. (the tn(T,C) of N is V.)", the first pattern in Fig. 13. The execution of the query command {find, [temple, Kinkakuji, fee,—]} returns a complete quadrigram which gives V (for example, 500 yen) as the value of its fourth term, while N and tn(T,C) are given in the input sentence. Thus the response for this sentence is "Kinkakuji no haikanryo wa 500 yen desu. (The entrance fee of Kinkakuji is 500 yen.)"

As the second example, consider the query sentence "Kinkakuji no haikanryo wa 400 en yori takai desuka. (Is the entrance fee of Kinkakuji more expensive than 400 yen?)" Since the semantic representation of this sentence is {>, [temple, Kinkakuji, fee,—], [—,—,—,400en]}, the third pattern in Fig. 13 is selected as the pattern of the response. If the answer from the information retrieval system is "no", then the response is "Kinkakuji no haikanryo wa 400 en yori takaku wa arimasen. (The entrance fee of Kinkakuji is not more expensive than 400 yen.)"

5. Experiment and Discussion

This section describes an experiment carried out to test the speech interface explained in the previous sections. The text composed of 53 sentences was designed according to the syntax stated in the section 2. The speech corpus used in the experiment consists of these sentences read by each of three male speakers, namely, KB, NY and HA. The corpus contains about 9.4 minutes of speech in total.

For each speakers, eleven sentences from the first half was used in order to design three separate vector-quantization codebooks for the LPC-CEPs, the DELTA-CEPs and POWs. For the sentence recognition, the speech corpus of each speaker was divided into two halves. Except the codebooks, the parameters obtained from the initial half were used to recognize the latter half, and vice versa.

Table 2 shows the sentence understanding rates obtained for the experiment. The columns, first and -fifth, stand for the rates of the correct sentences out of the 53 sentences in the first position and within the top five, respectively. The average rates were 83.0% and 93.1%. The command generator not only translated all the correctly recognized sentences into intended query commands, but also converted correctly mis-recognized sentences which were semantically understood. The figures

Table 2. The rates of sentence understanding.

speaker	correct (%)	
	first	–fifth
KB	81.1	90.6
NY	84.9	98.1
HA	83.0	90.6

in Table 2 contains such sentences.

Several errors of the recognition occurred at very long compound words, such as /umekojijokikikanshakan (/umekoji/, /joki/, /kikansha/, /kan/), but few errors at short functional words. Moreover, most of the latter errors were relieved by the semantic interpretation in the command generator. The current phonological processor cannot decompose a long compound word into several component words to insert optional pauses between them, although this often happens in actual utterances.

The average process time was about 15 minutes per sentence on an Apollo DN/4000 (4MIPS, 8MB memory). This figure is far from the practical use. We are very optimistic in this point, because the progress of the hardware technology is incredibly fast.

6. Conclusion

This paper has reported the speech interface we are developing as the main component of a speech dialogue system, and an experiment carried out to test it. It consists of three main components; speech recognition system, command generator and response generator. In construction of the speech recognition system, we proposed a basic strategy that the top-down linguistic hypothesis is made at lexical level while the verification of hypotheses is made by using the unit independent of the word, the phonetic string bounded by robust phones.

For 53 sentences read by each of three male speakers, the average rates of sentence understanding were 83.0% for the first choice and 93.1% within the top five, respectively, Although these figures are not sufficient for the practical use, most errors occurred at long compound words, but few errors at short functional words. We think mis-recognition of long words is more tractable than that of short words. This shows that the proposed strategy have been successful and is promising.

References

[1] Y. Niimi and Y. Kobayashi, "The procedure to constract a word predictor in a speech understanding system from a task-specific grammar defined in a CFG or DCG," Proc. of Int. Conf. on Computational Linguistcs, pp. 605–607 (1986).

[2] Y. Kobayashi and Y. Niimi, "Matching algorithm between a phonetic lattice and two types of templats—lattice and graph," J. Acoust. Soc. Jpn., Vol. E5, No. 4, pp. 267–270 (1984).

[3] D.D. Chamberlin, et al., "SEQUEL 2—A unified approach to data definition, manipulation and control," IBM J. Res. Develop., Vol. 20, No. 6, pp. 560–575 (1976).