Functional Representation of Query Sentences and Meaning Determination of Elliptical Sentences

Hsi-Jian Lee & Shin-Jiang Hwang

李錫堅　　黃詩建

National Chiao Tung University

國立交通大學

Proceedings of ROCLING I(1988)
R.O.C. Computational Linguistics Workshops I pp 179-210

中華民國第一屆計算語言學研討會論文集 179-210 頁

Functional Representation of Query Sentences and Meaning Determination of Elliptical Sentences

Abstract: A dialogue model is provided to describe the contents of a dialogue process between a user and a database management system. This model can be used as an intermediate representation between query sentences in a natural language and the underlying database query language. It is also capable of keeping the dialogue information, including user query sentences and the associated responses, for later processing of elliptical sentences.

A query sentence is decomposed into two components, a query phrase and a data description phrase. The functional representations of both components are analyzed in detail. Five types of representations for elliptical sentences, including subsetting, ordinal, remaining, projection, and substitution, are presented. We also detailedly discuss the determination of full meanings of elliptical sentences based on both the dialogue convention and functional representations.

Keywords: dialogue model, query sentence, functional representation, elliptical sentence

1. Introduction

To provide a friendly user interface in a database management system for information retrieval is an important task. Database query languages[1] are commonly adopted for information retrieval. They do provide rigid notations for a user to state his query sentences precisely without worrying about the physical structure of the database. But in order to express query sentences skillfully, the user must have some knowledge about query languages and the structure of the underlying database systems. It is sometimes very inconvenient for a novice. Therefore, it is necessary to provide a presentation more intuitive than database query languages. According to the experiments of Hendrix *et al.*[2], natural languages as user interfaces for information retrieval can satisfy this need. They can also shorten learning time and thus encourages the using of database systems. In addition to providing more intuitive expressing method, natural languages can save the user from the trouble of dealing with the physical and even logical structure of the database.

When we use natural languages, it is necessary to provide a model to describe the meaning of a query sentences in a natural language and to organize the dialogue information so that query sentences and system responses can be kept. In this paper, we present a dialogue model to describe the information involved in a dialogue process in a Chinese Intelligent Database Assistant (CIDA) for retrieving library information. This model contains a list of items composed of user query sentences and system responses. The purpose of keeping information of the dialogue process is to enable the user to state query sentences referring either to previous responses or to *elliptical sentences*, whose meaning can be determined from a previous query sentence. For example, after the user issues the following query,

(1) qǐng xiǎnshì A.I. fāngmiàn de qīkān.

(Please display the title of journals related to Artificial Intelligence.)

he can state the following elliptical query sentence,

(2) DBMS?

instead of

(3) qǐng xiǎnshì DBMS fāngmiàn de qíkān.

The problem about representation of sentences, usually very complicated, is a typical issue in natural language processing. So long as the universe of discourse is limited in a special domain, it is possible to devise a compact, precise and perspicuous representation. In the model proposed in this paper, the representation of a query sentence is called a *functional form,* which takes into account mainly the function of a query sentence. The syntactic structure is also reserved for resolving the full meaning of an elliptical query, which can be exemplified as follows.

(4) qǐng xiǎnshì ACM chūbǎn de qíkān zhōng yǔ A.I. yǒu guān de.
(Please display the title of journals published by ACM and related with A.I.)

(5) ACM chūbǎn de qíkān zhōng yǒu nǎxiē yǔ A.I. yǒuguān ?

(6) yǔ A.I. yǒu guān de qíkān zhōng yǒu nǎxiē shì ACM chūbǎn de ?

The semantics of these three sentences are the same. For the representations in functional form, abbreviated as *functional representations* hereafter, sentences (4) and (5) should have the same representation since the processing about either the current sentence or subsequent sentences are all the same. However, they should have different representation from sentence (6). It can be seen from the situation when sentence (7) below follows them.

(7) nǎxiē shì IEEE chūbǎn de ?
(Which are published by IEEE ?)

If sentence (7) follows sentence (6), it means

(8) yǔ A.I. yǒuguān de qíkān zhōng yǒu nǎxiē shì IEEE chūbǎn de ?
(Among the journals related to A.I., which ones are published by IEEE ?)

But it is meaningless if sentence (7) follows sentences (4) and (5).

All sample sentences in CIDA, as shown in Fig. 1, are classified into two classes: *basic sentences* and *elliptical sentences*. In the class of basic sentences, those without "*e*" in the sentence number, their meanings can be interpreted from themselves. On the other hand, in the class of elliptical sentences, those with "*e*" attached to the sentence number, their meanings must be interpreted by referring to the previous context.

(1) lièchū suǒyǒu ACM chūbǎn de qǐkān.
(2) lièchū 3 zhǒng IEEE chūbǎn de zázhì zhōng yǔ réngōng zhìhuèi yǒuguān de.
(3) qǐng xiānshì zhìliàokùxìtǒng fāngmiàn de shūjí de jiàqián jí chūbǎnshāng.
(4*e*) lièchū nàxiē jǔ zhuānjiā xìtǒng yǒuguān de zázhì de chūbǎn niándaì.
(5*e*) lièchū qǐzhōng Knuth xiě de.
(6*e*) lièchū tāmende neìróng.
(7*e*) lièchū qǐzhōng dì 4 běn de neìróng.
(8*e*) qǐng lièchū qǐyúde.
(9*e*) lièchū qǐyú yǔ réngōng zhìhuèi yǒuguān de.
(10) yǒu nǎxiē Winston xiě de shū ?
(11) nǎxiē zázhì shì ACM chūbǎn de ?
(12) North Holland chūbǎn de zázhì zhōng yǒu nǎxiē yǔ réngōng zhìhuèi yǒuguān ?
(13*e*) qǐzhōng yǒu nǎxiē IEEE chūbǎn de zázhì ?
(14*e*) qǐzhōng yǒu nǎxiē zázhì shì IEEE chūbǎn de ?
(15*e*) háiyǒu nǎxiē shì Knuth xiě de ?
(16*e*) háiyǒu nǎxiē réngōng zhìhuèi fāngmiàn de zázhì shì IEEE chūbǎn de ?
(17) The Art of Computer Programming de zhuòzhě shì sheí ?
(18*e*) neìróng shì shéme ?
(19) The Art of Computer Programming shì sheí xiě de ?
(20) The Art of Computer Programming fàngzaì nǎlǐ ?
(21*e*) chūbǎn daù dìjǐqí ?

Fig. 1. Sample Sentences in CIDA

The intelligent information systems incorporating natural languages as a front end of database systems include GUS[3], TEAM[4], KID[5], LUNAR[6], LDC[7], FRED[8] and LADDER[2]. Among all the aspects of the above systems, we are interested in the meaning representations and their treatments of elliptical inputs. Only GUS, LUNAR and LADDER, among these systems, can accept elliptical inputs.

LADDER uses syntax tree based on the semantic grammar as its meaning representation. An elliptical input can be accepted only if its syntax tree is analogous with a partial tree of a previous sentence. This is the most common type of ellipses. GUS uses frames to represent query sentences and controls the processings. The system asks the user a planned sequence of questions in order to obtain the full specifications of user query sentences. Though it can understand a few mixed initiated utterances by key-word matching, the overall interaction is guided by the system. LUNAR uses extended notational variants of the ordinary predicate calculus as a meaning representation language; it determines the meanings of elliptical and anaphoric expressions according to both syntactic structure and the language, which is in logical form.

There are some researches, though not real systems, focusing on meaning representation for information retrieval and on appropriateness of the meaning representation for determination of ellipses. Nash-Webber[9] proposed a formal meaning representation, and argued that logical meaning representation is superior to semantic network, especially in determination of ellipses. Horrigan[10] tested his dialogue model for the real dialogue between passengers and a clerk at an information booth in a train station. Spiegler[11] proposed a notation to represent user query sentences but he did not address the problems of ellipses and anaphora.

For an elliptical sentence, we should determine its full meaning by applying the syntactic and semantic information of both previous and current sentences and the domain knowledge. The approaches for the determination problem can be classified into three kinds. The first approach, adopted in GUS[3], is to construct the complete sentence from a sentence fragment and then parse it. The second one, adopted in LADDER and INLAND[2], is to match the elliptical syntactic structure of an elliptical sentence with that of a previous meaning complete sentence. After an analogous pattern is found, it is used to replace the current syntactic structure to construct a com-

plete syntactic structure. The last one, adopted in this system as well as LUNAR[6], infers the meaning of an elliptical sentence from the functional representations of the context and the current input. In the determination process, source sentences and syntactic trees are not involved.

Sec. 2 describes the method to represent query sentences in the functional form. The analysis of utterances based on their functions is also presented. Sec. 3 shows the functional representations of ellipical sentences. Sec. 4 describes the determination of elliptical sentences according to their function types. The full meaning of an elliptical sentence is evaluated by taking into account the structures of the current and previous context, responses and the dialogue convention. Some conclusions are given in Sec. 5.

2. The Dialogue Model

The dialogue model is actually a representation of the dialogue process. A typical dialogue process contains a series of exchanges, each of which composed of a query sentence and its corresponding response. Following the principle in the design of functional form, two sentences should be mapped into the same meaning representation if there is no difference in the processings of subsequent query sentences; conversely, they should be mapped into different representations.

2.1 Overall Description

The full specification of the dialogue model is given in Appendix. The top level specifications are :

DialogProcess == [Exches]

Exches == Exch | Exch, Exches

Exch == [Query, Response]

That means a dialogue process is composed of a series of exchanges and each exchange contains a query part and a response part. The response part, containing responsed information of the query sentence, is used to reduce the need of repetitive

access to the database and to resolve elliptical expressions with numeric determiners. This can be seen from the following two successive query sentences.

(9) qǐng xiǎnshì shuǒyǒu ACM chūbǎn de qīkān.

(Please display all titles of journals published by ACM.)

(10) qǐngwèn dì 3 běn fàng–zài nǎlǐ ?

(Where does the third one put?)

In this paper, we focus mainly on the representation of the query part. It is specified as

Query == basic(Complete) | elliptical (Ellipsis, Complete).

It means that a query sentence may be either a basic sentence or an elliptical sentence. There are two constituents in the functional representation of an elliptical sentence, "*Ellipsis*" and "*Complete*". "*Ellipsis*" denotes the original meaning-incomplete representation of an elliptical sentence and "*Complete*" represents its corresponding meaning-complete representation, derived from "*Ellipsis*" and the context.

Generally speaking, a query sentence is composed of two components: *a query phrase* and *a data description phrase*, where the latter phrase determines entities from which some information should be retrieved, and the former phrase determines the query type of the sentence; that is, it determines the information which should be retrieved from entities described by the data description phrase. For example, in the following sentences :

(11) Artificial Intelligence cóng nǎ–yi–qī dìng–qǐ ?

(What is the issue number from which Journal of Artificial Intelligence is subscribed ?)

(12) On Conceptual Modelling fàng–zài nǎlǐ ?

(Where does 'On Conceptual Modelling' put ?)

(13) yǒu jǐ zhǒng réngōngzhìhuì fāngmiàn de zázhì ?

(How many kinds of journals are related to Artificial Intelligence ?)

the underlined phrases are query phrases and the others are data description phrases. In the design of functional form, it is very important to decompose a sentence into a query phrase and a data description phrase.

2.2 Representations of Query Phrases

As mentioned above, the functional form is mainly used for describing the function of a query sentence. In the following, we will describe three functional types of query phrases: presentation, aggregation and predicate.

2.2.1 Presentation Query Type

A presentation query phrase is specified to show some information on line. This is the most typical of query phrases for information retrieval. Some examples of the presentation query type are "qǐng xiǎnshì" (please display), "qǐng lièchū" (please list), "yǒu nǎxiē" (which ones), "yǒu nǎ jǐ zhǒng" (which kinds), "yǒu nǎ jǐ běn" (how many books), etc. This type of query sentences can be specified as

Complete == present(DataSetDescriptor, Attributes),

where "DataSetDescriptor" denotes the representation of the data description phrase and will be described in the next subsection, and "Attributes" denotes the information to be retrieved from items of the data constrained by "DataSetDescriptor". The following sentences are typical examples.

(14) qǐngwèn Natural Language Processing shì shéi xiě de ?

(Who is the author of Natural Language Processing ?)

(15) qǐng xiǎnshì Natural Language Processing de zhùzuò .

(Please display the author of Natural Language Processing ?)

(16) yǔ Computer Graphics yǒuguānde zázhì yǒu nǎxiē ?

(How many journals are related to Computer Graphics ?)

(17) yǒu nǎxiē guānyǔ Computer Graphics fāngmiàn de zázhì ?

(18) qǐng lièchū yǔ Computer Graphics yǒuguān de zázhì.

2.2.2 Aggregation Query Type

An aggregation query phrases is usually used to enquire the number of items in a data set. It is specified as

Complete == count(DataSetDescriptor, Unit),

where "Unit" may be an element of the set { zhǒng, běn, qǐ, lèi, . . . }. Query sentences of this type include, for example, "yǒu jǐ běn", "yǒu jǐ zhǒng" and "yǒu jǐ qǐ". The following sentence is a typical example:

(19) yǒu jǐzhǒng A.I. fāngmiàn de shū ?

(How many kinds of books are related to A.I. ?)

2.2.3 Predicate Query Type

Predicate query phrases usually appear in Yes/No questions, and are represented as predicates in functional form. In this paper, we provide only a few predicates because most Yes/No questions can be replaced pragmatically by WH questions[2]. This fact will be explained in Sec. 3 in more detail. A sentence containing "yǒuméiyǒu" is represented as "exist(DataSetDescriptor)", "shìbúshì Borrower jiè de" as "lentBy(DataSetDescriptor, Borrower)", and "yǒuméiyǒu bèi jièzǒu" or "shìbúshì bèi jiè le" as "lent(DataSetDescriptor)". The following is a typical example:

(20) yǒuméiyǒu A.I. fāngmiàn de shū ?

(Is there any book related to A.I. ?)

2.3 Representations of Data Description Phrases

There is a difference between the representation of a data description phrase and that of a query phrase: the mapping from a query phrase onto its representation in functional form ignores the syntactic structure information while the mapping from a

data description phrase preserves some syntactic structure information. The data description phrases are divided into five types as illustrated below.

2.3.1 Restriction Type

The restriction type of data description phrases is the most common type which can be specified as

DataSetDescriptor = restrict(DataSetDescriptor, Constraint) | cida | dataSet(N),

where "DataSetDescriptor" is defined recursively and may denote either the overall database, "cida", or a data set corresponding to a previous sentence specified by a number, "dataSet(N)"; "Constraint" denotes the conditions derived from the data description phrase. For a basic sentence, the typical "DataSetDescriptor" is "restrict(cida, Constraint)", which means the data in the underlying database satisfying the constraint "Constraint". If a "DataSetDescriptor" contains "dataSet(N)", it denotes an elliptical sentence; for example, "restrict(dataSet(5), Constraint)" denotes the data in the data set of the fifth query sentence satisfying "Constraint". The full specification of constraints is given in Appendix. Two phrases of this type and their corresponding representations are shown as follows.

(21) "suǒyǒu ACM chūbǎn de zázhì "
(All journals published by ACM)
restrict(cida, and([publisher(ACM), bookType(journal)]))

(22) "yǔ A.I. yǒuguān de shū zhōng 1980 nián hou`chūbǎn de "
(All books related to A.I. and published after 1980)
restrict(restrict(cida, and([field(ai), bookType(book)])),
gt(publishedYear, year(1980)))

Compare the following two phrases and their corresponding representations.

(23) yǔ A.I. yǒuguān de zázhì zhōng ACM suǒ chūbǎn de

(Among the journals related to A.I., which are published by ACM?)

restrict(restrict(cida, and([field(A.I.), bookType(journal)])),

publisher(ACM)),

(24) ACM suǒ chūbǎn de zázhì zhōng yǔ A.I. yǒuguānde

(Among the journals published by ACM, which are related to A.I. ?)

restrict(restrict(cida, and([publisher("ACM"), bookType(journal)])),

field(ai)).

From this comparison, we can see that the syntactic structure of data description phrases is preserved. The reason why the two phrases have been mapped onto different representations has already been explained in Sec. 1.

2.3.2 Indefinite Specification Type

In English, "some" is used to express indefinite specification. In this paper an indefinite specification phrase is defined syntactically as

<Indef-Spec> :== jǐ <unit> | <Number> <Unit>,

<Unit> :== běn | zhōng | cè | qǐ,

Phrases like "5 běn", "3 zhǒng" and "4 qǐ" belong to this catogory. In functional form, it is specified as

DataSetDescriptor == some(DataSetDescriptor, Unit, Number),

where "DataSetDescriptor" in the right hand side denotes the data set modified by the indefinite specification phrase. For example,

(25) qǐng xiǎnshì 5 běn Knuth suǒxiě de shū.

(Please show 5 Knuth's books.)

is represented as

DataSetDescriptor = some(restrict(cida, and([author(Knuth),

bookType(book)])), ben, 5).

2.3.3 Ordinal Specification Type

An ordinal specification phrase appears only in an elliptical sentence and is used to specify one or more definite items of the responsed part of a previous sentence. It is described as

DataSetDescriptor == ordinal(DataSetDescriptor, Unit, Orders)

where "Orders" is a list of numbers, each of which denotes an ordinal or an index of the referred sentence in context, and "DataSetDescriptor" is derived by considering the context and will be explained in Sec. 4. The phrase, for example,

(26) dì 3 běn yǔ A.I. yǒuguān de shū.

(The third book related to A.I.)

is represented as

ordinal(dataSet(N), ben, [3]),

and the phrase

(27) qízhōng dì 4 běn

(Among these, the forth one)

is represented as

ordinal(dataSet(N), ben, [4]).

2.3.4 Remaining Type

A remaining data description phrase also appears only in an elliptical sentence and is used to specify the data which would be determined from previous sentences. It is specified as

DataSetDescriptor == diff(BaseSet, Complements)

BaseSet == DataSetDescriptor

Complements == dataSet([Numbers]).

This specification means that the remaining data description phrase specifies the data derived by substracting the data in "Complements" from "BaseSet". Some phrases of

this type are shown as follows.

(28) hái yǒu nǎxiē

(How many remained ?)

(29) qíyú yǔ A.I. yǒuguān de

(Others related to A.I. ?)

(30) qíyú de

(Others ?)

Let us look at the following scenario.

(31) A.I. fāngmiàn de zázhì zhōng yǒu nǎxiē shì ACM chūbǎn de ?

(Among the journals related to A.I., which are published by ACM ?)

(32) nǎxiē shì IEEE chūbǎn de ?

(Which are published by IEEE ?)

(33) nǎxiē shì NorthHolland chūbǎn de ?

(Which are published by North Holland ?)

(34) qíyú de nē ?

(How about the others)

The remaining phrase of sentence (34) obviously means all the journals related with A.I. except those published by ACM, IEEE and North Holland. Thus it is represented as

diff(restrict(cida, and([field(A.I.), bookType(journal)])),

dataSet([31,32,33])).

3. Functional Form of Elliptical Sentences

The strategy for representing an elliptical sentence is to map it onto an elliptical functional form with partial meaning and then construct its complete functional form from the functional forms of the context. There are five classes of ellipses in CIDA, including subsetting, ordinal, remaining, projection, and substitution. They are

categorized according to their functions.

3.1 Subsetting Ellipses

A subsetting ellipsis specifies a subset of the responses of a previous sentence. The subset satisfies the additional constraints in the elliptical sentence. For instance, the phrase "qǐzhōng nǎxiē shì ACM chūbǎn de" (Which are published by ACM?) designates the books or journals published by ACM in the responses of a previous sentence.

The representations of subsetting ellipses are derived from the elliptical sentences containing such subpatterns as "qǐzhōng nǎxiē . . . yǔ . . . yǒuguān" (Among these, which are related to ... ?) or "qǐzhōng . . . suǒ xiě de shì nǎxiē" (Among these, which are written by ...?). The phrases are represented as

RefExp == subset(Constraint, Attributes),

where "Constraint" and "Attributes" mean the same as before.

Two sample phrases and their corresponding functional representations are shown as follows.

(35) qǐzhōng nǎxiē yǔ A.I. yǒuguān ?

(Among these, which are related to A.I. ?)

subset(field(A.I.), ?Attributes),

where "?Attributes" denotes omitted specification.

(36) qǐzhōng ACM suǒ chūbǎn de fàng–zài nǎlǐ ?

(Among these, where do the ones published by ACM put ?)

subset(publisher(ACM), location),

3.2 Ordinal Ellipses

Ordinal ellipses are derived from elliptical sentences containing such subpatterns as "dì N běn" (the nth one), "dì N zhǒng" (the nth kind) and "qǐzhōng dì N lèi" (the

nth type among these). Phrases of this type identify one or more items in the responses of a previous sentence, either basic or elliptical. They are represented as

RefExp == ordinal(Constraint, Orders, Attributes, Unit).

In the following, some sample phrases and their representations are shown.

(37) qǐzhōng dì 3 zhǒng fàng zài nǎlǐ ?

(Where is the third kind put on ?)

ordinal(?Constraint, [3], location, kind)

(38) dì 4 běn A.I. fāngmiàn de shū

(The 4th book related to A.I. ?)

ordinal(and([field(A.I.), bookType(book)]), [4], ?Attributes, ben)

(39) dì 3,4,6 běn

(The third, fourth and sixth books)

ordinal(?Constraint, [3, 4, 6], ?Attributes, ben)

2.4.3 Remaining Ellipses

Remaining ellipses are the elliptical sentences containing such subpatterns as "qǐyúde" (others), "qǐtā (others), "háiyǒu nǎxiē", (others ?),etc.. A remaining phrase specifies the differences of two sets. It is formally represented as

RefExp == complement(Constraint, Attributes).

A case in which the phrases of remaining type appear has been illustrated in Sec. 2.3.4. Other cases will be illustrated in Sec. 4. Here we show some sample phrases and their representations.

(40) qǐyúde zuòzhě

(Other authors ?)

complement(?Constraint, author)

(41) háiyǒu nǎxiē DBMS fāngmiàn de shū

(How many other books related to DBMS ?)

complement(and([field(DBMS), bookType(book)]), ?Attributes)

3.4 Projection Ellipses

A projection ellipsis projects an attribute of an item of a previous response. The function of this type of phrases is analogous to the projection in relational algebra, a data base model. The representation is specified as

RefExp == attriName(Attribute).

Some sample phrases and their representations are shown as follows.

(42) fàngzài nǎlǐ

(Where are they ?)

attriName(location)

(43) shì shéi xiě de

(Who write it ?)

attriName(author)

3.5 Substitution Ellipses

Any sentence of this type consists of just a noun phrase which denotes an instance of an attribute. This phenomenon is very common in most natural languages, such as English and Chinese. It is the type of ellipses which can be processed in LADDER[2]. To give an example, if an ellipsis, "ACM nē", follows the sentence "yǒu nǎxiē IEEE chūbǎn de zázhì," it means "yǒu nǎxiē ACM chūbǎn de zázhì." The representations are specified as

instanceOf(Attribute, Val).

Two typical sentences and their representations are shown as follows.

(44) Knuth

instanceOf(author, Knuth)

(45) 1986 nián 8 yuè

instanceOf(date, date(1986, 8))

4. Determination of Elliptical Sentences

We have divided the elliptical functional form into five classes. Here, we follow this classification to discuss the determination of the full meanings of elliptical sentences. Since a query sentence can be decomposed into a query phrase and a data description phrase, the determination of the full meaning also employs two processes to determine these two phrases. In this paper we mainly focus on the determination of the data description phrases, which are based on both the dialogue convention and functional representations.

4.1 Subsetting Type

Case 1.

Consider the following sequence of query sentences.

(46) Knuth suǒ xiě de shū zhōng yǒu nǎxi ē yǔ Algorithm yǒuguān ?

(Among the books written by Knuth, which are related to algrithms ?)

(47) nǎxiē yǔ DBMS yǒuguān ?

(Which are related to DBMS ?)

Sentence (46) is a basic sentence while (46) is an elliptical one. According to the dialogue convention, the elliptical sentence is interpreted by replacing the constraint phrase with that of the corresponding basic sentence because both sentences are analogous syntactically. As an example, the sentence (47) is interpreted as

(48) Knuth suǒ xiě de shū zhōng yǒu nǎxiē yǔ DBMS yǒuguā ?

(Among the books written by Knuth, which are related to compiler ?)

Formally, sentence (46) is represented as

present(restrict(restrict(cida, and([author(Knuth), bookType(book)])),

field(Algorithm), ?Attributes),

and sentence (47) is originally represented as

subset(field(DBMS), ?Attributes),

From the substructure of the representation of sentence (46), it can be seen that "field(Algorithm)" is the representation of the additional constraint phrase, "yǔ Algorithm yǒuguān". Thus we can replace it with "field(DBMS)" to construct the complete representation of sentence (46) because they are analogous. In summary, the constructed complete representation of sentence (46) is

present(restrict(restrict(cida, and([author(Knuth), bookType(book)])),
field(DBMS)), ?Attributes).

Case 2.

Consider the following sequence of query sentences.

(49) yǒuméiyǒu A.I. fāngmiàn de shūjǐ ?
(Are there books related to A.I.?)

(50) yǒu jǐ běn shì Rich xiě de ?
(How many books are written by Rich?)

In the above, sentence (49) is a basic sentence while (50) is an elliptical one. This case is different from case 1 in that no constraint in sentence (49) can be replaced with that of the elliptical sentence. Ideally, the complete data description may be constructed by concatenating the data description phrase of sentence (49) with the constraint phrase of sentence (50) such as

count(restrict(restrict(cida, and([field(A.I.), bookType(book)]))
author(Rich)), aBook)

In practice, however, the complete representation of sentence (50) is

count(restrict(dataSet(49), author(Rich)), ben).

It is because the former representation is not efficient during the period of retrieving

the data from the underlying database.

Case 3.

Now, we illustrate a more complicated case which will indicate the necessity of matching analogous patterns. Consider the following scenario:

(51) yǒu nǎxiē A.I. fāngmiàn de shūjí ?

(Which books are related to A.I.?)

(52) yǔ Expert System yǒuguān de yǒu nǎxiē ?

(Among these, which ones are related to expert system?)

(53) qízhōng nǎxiē shì 1980 nián hòu chūbǎn de ?

(Which ones are published after 1980?)

(54) nǎxiē shì 1980 nián yǐqián chūbǎn de ?

(Which ones are published before 1980?)

(55) yǔ Natural Language Processing yǒuguān de yǒu nǎxiē ?

(Which ones are related to natural language processing?)

According to the domain hierarchy, sentence (52) refers to sentence (51). By the dialogue convention, the sentence that sentences (53) and (54) refer to is (52) rather than (51). Formally speaking, the constraint of sentence (52) is represented as "field(Expert System)" while those sentences (53) and (54) are "gt(publishedDate, year(1980))" and "lt(publishedDate, year(1980))" respectively. Thus, the latter constraint is concatenated with the former one to form a new complete functional representation. As for sentence (55), it refers to sentence (51) according to the domain hierarchy. The above illustrates that the semantics and domain knowledge may affect the results of the determination.

4.2 Remaining Type

The intuitive meaning of an elliptical sentence of remaining type is analous to that of sentences containing the subpattern of "<primary data description phrase>

except <some others>". Formally speaking, an elliptical sentence of this type is determined as the representation containing a partial representation of "diff(BaseSet, Complements)", where "BaseSet" is the representation of "<primary data description phrase>" and "Complements" is that of "<some others>". Acordingly, the processing of this type of elliptical sentences is to determine the two data sets, "BaseSet" and "Complements".

Case 1.

Consider the following sequence of query sentences:

(46) Knuth suǒ xiě de shū zhōng yǒu nǎxiē yǔ Algorithm yǒuguān ?

(47) nǎxiē yǔ DBMS yǒuguān ?

(56) qíyúde nē ?

(Others)

This case is extension of Case 1 in Sec. 4.1; the structure we will discuss here is sentence (56). By the dialogue convention, sentence (56) asks in what other fields than Algorithm and DBMS the books written by Knuth are. Ideally, the data of this type include the data corresponding to the primary data description phrase, "Knuth suǒxiě de shū" except the data responsed for either the basic sentence (46) or the elliptical sentence (47). Formally, the context pattern can be recognized from the functional representations of these sentences. For example, the representations of sentences (46) and (47) are

present(restrict(restrict(cida, and([author(Knuth), bookType(book)])),
field(Algorithm)), ?Attributes)

present(restrict(restrict(cida, and([author(Knuth), bookType(book)])),
field(DBMS)), ?Attributes).

The author restricted in both the representations indicates this fact. So far, we can determine that the "DataSetDescriptor" of the functional representation of the current elliptical sentence is

DataSetDescriptor = diff(restrict(cida, and([author(Knuth), bookType(book)])),

dataSet([46, 47]).

where "restrict(cida, and([author(Knuth), bookType(book)]))" is "BaseSet" and "dataSet([46, 47])" is "Complements". As for the query phrase, we can easily decide that the queried data is the fields because there is a constraint - field(Field), where "Field" is either "Algorithm" or "DBMS", in the representation of each sentence. Thus the complete functional representation of sentence (56) is determined as

present(DataSetDescriptor, field),

where "DataSetDescriptor" is described as above.

Case 2.

Consider the query sentence following sentences (49) and (50):

(57) qǐyúde nē ?

(Others?)

The "dataSetDescriptor" of sentence (57) is determined as

DataSetDescriptor = diff(dataSet(49), dataSet(50).

where "dataSet(49)" is "BaseSet" and "dataSet(50)" is "Complements". For the query phrase, it is determined that the queried data is about the authors because the representation of sentence (50) contains the constraint phrase representation: "author(Rich)". Thus the complete representation of sentence (57) is determined as

present(DataSetDescriptor, author),

4.3 Substitution Type

An elliptical sentence of the substitution type usually contains only a short noun phrase or even just a noun. It is also a very common type in natural languages such as English or Chinese. Consider the following sequence of query sentences:

(58) yǒuǎxiē 1980 nián gòurù de shūjǐ

(Which books are bought in 1980?)

(59.a) 1981 nián nē ?

(How about 1981?)

Intuitively, the second sentence means

(59.b) yǒunǎxiē 1981 nián gòurù de shūjǐ

(Which books are bought in 1981?)

Formally, sentence (58) is represented as

present(restrict(cida, and([eq(boughtDate, year(1980)),
bookType(book)])), ?Attributes),

and (59.a) is originally represented as

instanceOf(year, 1981),

By matching "year" in the two representations, we can determine the second sentence as

present(restrict(cida, and([eq(boughtDate, year(1981)),
bookType(book)])), ?Attributes).

Next, compare the following two sequences of sentences.

(60) yǒunǎxiē Codd suǒ xiě de shū ?

(Which are the books written by Codd ?)

(61) Ullman ?

(60) yǒunǎxiē Codd suǒ xiě de shū ?

(62) qízhōng nǎxiē yǔ DBMS yǒuguān

(Among these, which are related to DBMS ?)

(63) nǎxiē yǔ Expert System yǒuguān

(Which are related to Expert System ?)

The elliptical sentence in the first sequence belongs to the subsetting type and that in the second case belongs to the subsetting type. It can be seen easily that the elliptical sentences in the first sequence can not be stated in the way as those in the second

sequence and vice versa.

4.4 Ordinal Specification Type

Ordinal ellipses are very useful as a user interface in a natural language. By using these ellipses, a user can briefly state the constraints about his interested data and then query about them closely. Consider the following sequence of query sentences:

(64) qǐng xiǎnshì yǒuguānyú Natural Language Processing fāngmiàn de shū.
(Please display the titles of books related to natural language processing.)

(65) qǐzhōng dì 5, 6 , 8 běn fáng–zài nǎlǐ ?
(Where are the fifth, sixth and eighth ones ?)

(66) qǐng xiǎnshì dì 3 běn de mùlù.
(Please show the table of contents of the third one.)

For the sentence (65), the ordinals specify the items in the data responsed for sentence (64). By the dialogue convention, sentence (66) specifies the third item in sentence (64).

Consider another sequence of query sentences shown as follows:

(67) qǐng xiǎnshì yǒuguānyú Expert System fāngmiàn de shū.
(Please display the titles of books related to Expert System.)

(68) qǐng xiǎnshì yǒuguānyú Natural Language Processing fāngmiàn de shū.
(Please display the titles of books related to Natural Language Processing.)

(69) qǐng xiǎnshì dì 3 bèn Expert System de shū.
(Please display the third one about Expert System.)

Obviously, sentence (69) specifies the third item in sentence (67) rather than sentence (68). Comparing this example with the preceding one, we find that if an ordinal specification is not followed by a constraint phrase, it is determined as referring to the item in the most recent sentence without the ordinal specification. By matching the

constraint of an elliptical sentence with that of the referred sentence, we can determine correctly the ordinal specification.

Another convention about the ordinal specification should be addressed also. Consider the following sequence of query sentences:

(70) yǒunǎxiē Natural Language Processing fāngmiàn de shū.

(How many books are related to Natural Language Processing?)

(71) dì 1 běn fàng–zài nǎlǐ ?

(Where is the first one?)

(72) qǐng xiǎnshì qí jiàqián.

(How about the price?)

(73) dì 5 běn nē ?

(How about the fifth one?)

(74) dì 7 běn nē ?

(How about the seventh one?)

The query phrases of sentences (73) and (74) are omitted. By the dialogue convention, the user must desire to know the locations and prices of the fifth and seventh books in sentence (70). That is, sentences (73) and (74) must inherit the queried information or attributes of the books mentioned in the previous sentence.

4.5 Projection Type

An elliptical sentence of the projection type may be just a predicate or a noun phrase which is the name of an attribute. It describes a new data set different from any previous data. It is used primarily to incrementally query about interested data. This type of sentences is always determined as referring to the most recently activated data set.

The following sentences show some examples and their corresponding functional representations.

(75) zuòzě shì shéi ?

(Who is the author?)

attriName(author)

(76) shì shéi xiě de ?

attriName(author)

(77) zuòzě ?

attriName(author)

Though the above sentences have different syntactic structures, they have a common function; that is, they all project an attribute of the data of a previous sentence.

It is easy to determine the full meaning of elliptical sentences of this type since they refer to the most recent data set. Consider the following sequence of query sentences:

(78) yǒunǎxiē A.I. fāngmiàn de shū ?

(What are the books related to A.I. ?)

(79) shì shéi chū bǎn de ?

(Who are the publishers ?)

(80) héshǐ chūbǎn de ?

(What are they published ?)

The constructed complete functional representation has the format of

present(dataSet(N), Attribute),

where "N" denotes the index of the referred sentence in the dialogue model and "Attribute" denotes the attribute name in the elliptical sentence.

5. Conclusions

In this paper, we have presented functional representations of query sentences. The representations describe the semantics of query sentences and reserve some messages of syntactic structures. The problem of determining the data description phrase

of an elliptical sentence based on the functional representation in Chinese dialogue has been exploited. During the process of discussion, it is demonstrated that determining the missing components in an elliptical sentence must consider both the semantic attributes and the syntactic structure. The determination of query phrases remains to be a topic for further research.

In summary, a friendly and high level user interface should conform to the following guidelines.

- It must be able to be accepted like a natural language.
- It should provide the mechanism to express rigid combination of logical connectives.
- It should provide the mechanism for stating elliptical sentences to incrementally query objects.
- The logical structure of database must be transparent and the knowledge about task domain should be built into the user interface.

For the design of a good dialogue model, some conclusions are listed below. First, the responses should be kept in the model so that the elliptical sentences can be resolved correctly and repetitive access to the underlying database system can be avoided. Second, the meaning representation must be able to describe the necessary syntactic and semantic messages. Third, the meaning representation must be powerful enough so that the dialogue convention can be easily implemented into the determination processes. Fourth, the meaning representation should be database independent and language dependent so that the implementation can be independent of the underlying database system. Finally, the dialogue model should provide a mechanism so that the determined elliptical sentences can be expressed.

References

1. C. J. Date, *An Introduction To Database Systems*, vol. I, 1985.
2. G. G. Hendrix, "Developing a natural language interface to complex data," *ACM Transactions on Database Systems*, vol. 3, pp. 105-147, 1978.
3. D. G. Bobrow, R. M. Kaplan , M. Kay , D. A. Norman , H. Thompson, and T. Winograd, "GUS, a frame-driven dialog system," *Artificial Intelligence*, vol. 8, pp. 155-173, 1977.
4. B. J. Grosz, "TEAM : an experiment in the design of transportable natural-language interfaces," *Artificial Intelligence*, vol. 32, pp. 173-243, 1987.
5. H. Ishikawa, Y. Izumida, T. Yoshino , T. Hoshiai , and A. Makinouchi, "KID designing a knowledge-based natural language interface," *IEEE EXPERT*, pp. 57-71, 1987.
6. W. A. Woods, "Semantics and quantification in natural language question answering," *Advances in Computers*, vol. 17, pp. 1-87, 1978.
7. B. W. Ballard, J. C. Lusth, and N. L. Tinkham, "LDC-1: a transportable, knowledge-based natural language processor for office environments," *ACM Transactions on Office Information Systems*, vol. 2, pp. 1-25, 1984.
8. G. Jakobson, C. Lafond, E. Nyberg, and G. Piatetsky-Shapiro, "An Intelligent Database Assistant," *IEEE Expert*, pp. 65-79, GTE Laboratory, Inc., Summer 1986.
9. B. Nash-Webber and R. Reiter, "Anaphora and logical form:on formal meaning representations for natural language," *IJCAI*, vol. 1, pp. 121-131, 1977.
10. Horrigan, "Modelling Simple Dialogs," *IJCAI*, vol. vol. 1, p. 88, 1977.
11. I. Spiegler, "Modelling man-machine interface in a data base environment," *International Journal of Man-Machine Studies*, vol. 18, pp. 55-70, 1983.

Appendix Logical Specification of the Dialogue Model

```
DialogProcess == [Exches]

Exches == Exch | Exch, Exches

Exch == [Query, Response]

Query == basic(Complete) | elliptical (Ellipsis, Complete)

Complete == present(DataSetDescriptor, Attributes) |
      count(DataSetDescriptor, Unit) | exist(DataSetDescriptor) |
      lentBy(DataSetDescriptor, Borrower) | lendable(DataSetDescriptor) |
      lent(DataSetDescriptor)

DataSetDescriptor == cida | DataSetPointer | RefExp
      some(DataSetDescriptor, Unit, Number) |
      restrict(DataSetDescriptor, Constraint) |
      diff(BaseSet, Complements) | ordinal(DataSetPointer, Orders)

DataSetPointer == dataSet(Number)

RefExp ==
        complement(Constraint, Attributes) |
        ordinal(Constraint, Orders, Attributes, Unit) |
        subset(Constraint, Attributes) |
        instanceOf(Attribute, Val) | attriName(Attribute)

Unit == ben | aJournal | kind | year

Constraint == '?Constraint' | not(Constraint) | and(Constraints) |
          or(Constraints) | RelOp(Attribute, Val)

Constraints == [Constraints0]

Constraints0 == Constraint | Constraint, Constraints0

RelOp == eq | ne | ge | gt | le | lt

Attributes == [Attributes0]

Attributes0 == '?Attributes' | Attribute | Attribute, Attributes0
```

```
Attribute == bookType | author | cost | contentTable | donator |
        field | publisher | publishedDate | location | id | borrower
        bookName | journalName | beginDate | endDate
        donatedDate | vol | journalNo | lentDate | date
Val == '?Val' | Number | String | Date | Vol | JournalNo | Year | Month
Date == date(Year, Month)
Year == Number
Month == Number
JournalNo == journalNo(Year, AjournalNo)
Val == Number
AjournalNo == Number
BaseSet == DataSetDescriptor
Subsets == dataSet([Numbers])
Numbers == Number | Number, Numbers
Complements == dataSet([Numbers])
Orders == Order | Order, Orders
Order == OneOrder | OrderRange
OneOrder == Number
OrderRange == (StartNumber, EndNumber)
StartNumber == Number
EndNumber == Number
Borrower == String
ElliQuery == Complete
Ellipses == Ellipsis | Ellipsis, Ellipses
Ellipsis == RefExp | ElliQuery
Attributes == Attribute | Attribute, Attributes
Response == [Informations]
```

Informations == Information | Information, Informations

Information == (DataDescriptor, DataList)

DataDescriptor == [Attributes]

DataList == [ValLists]

ValLists == ValList | ValList, ValLists

ValList == [Vals]

Vals == Val | Val, Vals

Acknowledgement

This research work was supported by ERSO, ITRI, Hsinchu, Taiwan, under contract, MIST-E76006(1987) and by Taiwan International Standard Electronics Ltd..