A Unification-Based Approach for Chinese Inquiry Sentences Processing*

Shu-hang Lee and Hsi-Jian Lee†

Department of Computer Science and Information Engineering

National Chiao Tung University

Hsinchu, Taiwan 30050

ABSTRACT

In this paper, we present a unification-based approach for processing natural inquiry sentences on the relational database. We also propose a method to process the proper nouns in the queries. The semantics of some words in the inquiry sentences may have close relationship with the relational schema; they therefore can be expressed in the relation algebra. We design four types of feature structures, *value specification type, attribute specification type, relation specification type* and *function specification type*, as semantic representations to describe the corresponding algebraic content of the signs of the inquiry sentence. Unification is used as the primary information-combining operation to construct the syntactic and semantic content. In this system, we perform the syntactic parsing and the semantic interpretation in an integrated way. As a result, when the parser unifies two constituents, their algebraic contents are formed. Lastly, according to the algebraic content of the inquiry sentence, we design transformation rules to translate the feature structure to database retrieval commands and then get the desired data from the database.

* This research is partially supported by Taiwan International Standard Electronics Ltd.

† To whom correspondence should be addressed.

I. INTRODUCTION

For retrieving information from the database, many formal query languages are designed to simplify the problem of obtaining the correct data. To express the queries skillfully, users must have knowledge about these artificial languages and the schema of the underlying database system. It is not convenient for a novice user. Using natural languages for database retrieval provides users a more intuitive method.

In general, a natural inquiry sentence processing system can be divided into three major modules [13]: the parser, the formal query generator, and the database access routines. In the first step, the system parses the input query to a uniform intermediate representation, such as D&Q, logic form [1], case frame [5], etc. In the second step, the system translates the intermediate representation to a target query. In the third step, the system interprets the target query and retrieves the data.

In this paper, we propose a unification-based approach for Chinese inquiry sentences processing. For the past decades, unification has become one of the primary operations in linguistic theories and natural language processing [16]. Many researchers have focused on the design of unification-based grammatical formalisms due to the common design features, including surface-based, informational, inductive, declarative, and complex feature based [15].

Unification-based grammatical formalisms use feature structures as an information-bearing object that provides phonological, syntactic and semantic information by specifying values for various attributes in a set of feature-value pairs, and take unification to be a primary information-combining operation. Some principles we adopt in CIDA (a *Chinese Intelligent Database Assistant*), a user interface for helping the users to retrieve library information, are inherited from HPSG (*Head-driven Phrase Structure Grammar*) [14].

It is an important application to incorporate natural language as a user interface into a database system. The intelligent information systems incorporating natural language as a front end include GUS [5], KID [8], TEAM [1], LUNAR [17] and LADDER [6].

Grammatical formalisms are languages intended to describe the set of sentences that language encompasses, the structure properties and the meanings of such sentences [15]. HPSG [14] is a successor to Generalized Phrase Structure Grammar (GPSG) [7]. A collection of syntactic features and their values describe the syntactic category of a sign. The HFP (Head Feature Principle) declares that the head feature of a phrasal sign is shared with its head daughter. For example, the head feature of a noun phrase is determined by that of its head daughter. The SP (subcategorization Principle) states that in any phrasal sign, each complement daughter must unify with a member of the head daughter's *subcat* (an abbreviation of subcategorization) list, and that the subcat list of the mother consists of those elements that remain to be satisfied on the head daughter's subcat list. The AP (Adjunct Principle) states that all adjuncts to be attached are according to the arguments of the adjuncts list in the head daughter.

The remainder of this paper consists of five sections. In Section II, we introduce the architecture of the proposed system. Then in Section III we describe feature structures that we design as intermediate representation to bear contents of the inquiry sentences. In Section IV, the parsing technique is introduced. We also present a method to process the proper nouns. In Section V, we illustrate the implemented system and some examples. Conclusion and future works are drawn in the last section.

II. SYSTEM ORGANIZATION

The system architecture of CIDA is illustrated in Figure 1. The input query is first segemented into a list of words according to the lexical signs in the lexicon. The parser then analyzes the list and generates complex feature structures, which we use as the intermediate representation. When the results are passed to the contextual interpreter and asserted in the dialogue base, the contextual interpreter applies rules in the rule base and the information in the dialogue base to transform the feature structure to its corresponding relational algebra commands. It executes finally a database access routine to retrieve information. In the current stage, the mechanisms not yet implemented are marked by '*' in Figure 1.

The data model imposes structural restrictions on natural language understanding. The relational data model is generally a standard for developing natural language query systems [13]. In detail, the model is a data structure of the contents of the database, which is relatively independent of the actual storage structure of the database.

Four relation schemes of the underlying database are shown as follows:

BOOK(OBJNAME, OBJ_ID, FIELD, AUTHOR, PUBLISHER, LOCATION PUB_DATE, PRICE)

JOURNAL (OBJNAME, OBJ_ID, FIELD, PUBLISHER, LOCATION, PRICE)

BORROW (OBJ_ID, USER_ID, BORROWTIME)

USER (USER_ID, USERNAME)

III. INTERMEDIATE MEANING REPRESENTATION

There is a universal agreement within the AI community that natural language understanding systems must provide some underlying meaning representation into which surface strings are mapped [2]. The semantic interpretation is to map the input natural inquiry sentence to its intermediate meaning representation. A variety of meaning representations were proposed in natural language interface systems in the literature. These formalisms not only bear the meaning of the inquiry sentence, but also provide a computer – interpretable charactoristics of natural language.

In our system, the complex feature structures are taken as the intermediate meaning representation. A feature structure α is defined as a set of feature values $\{(f_i, v_i), i = 1, 2, n\}$, where feature names f_i are atoms, feature values v_i are atoms or feature structures. Feature

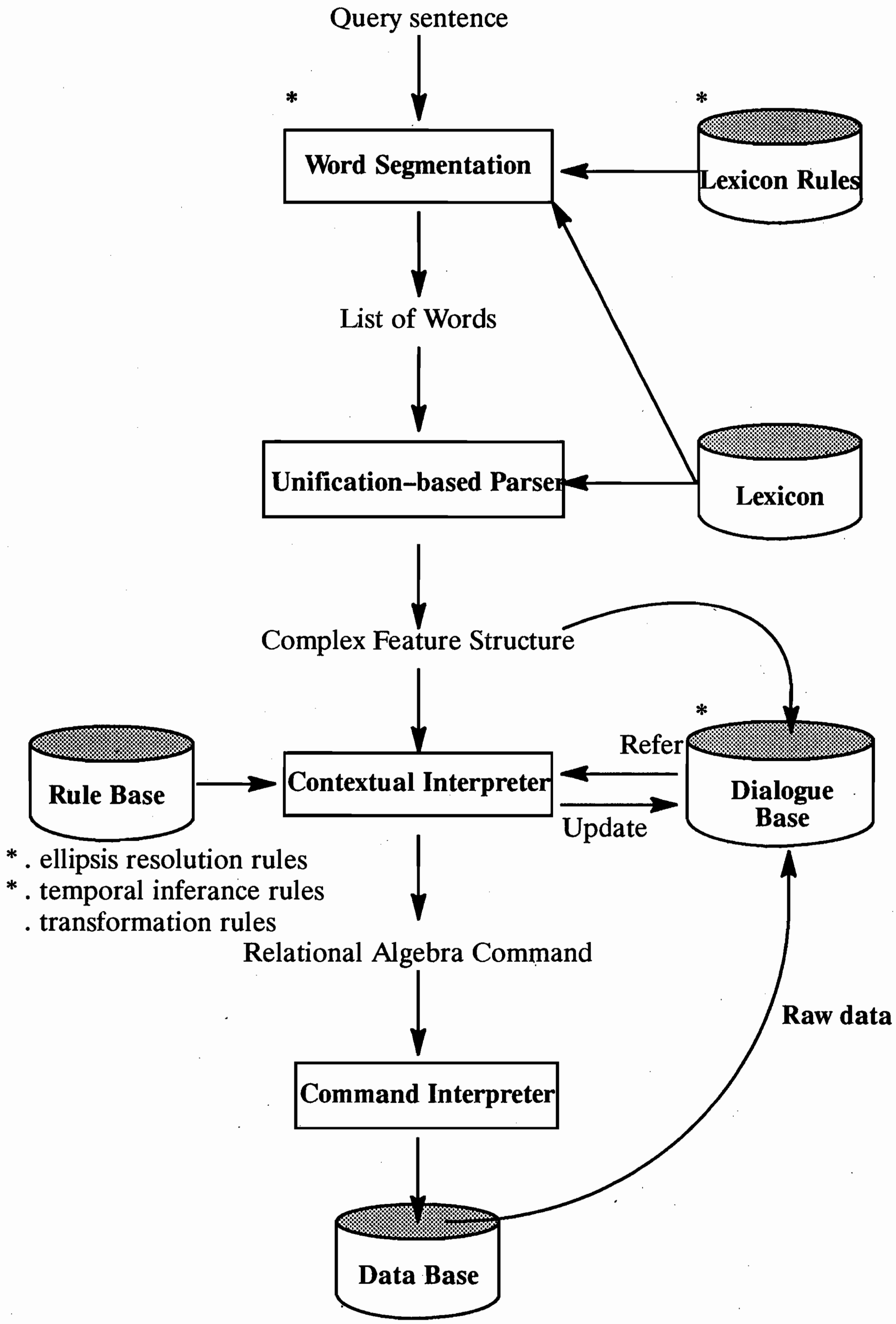

Figure 1. System architecture of CIDA.

structures resemble the first order predicate calculus, but several restrictions have been lifted [12]. For instance, substructures are labeled symbolically, not inferred by argument positions. Fixed arity is not required. Feature structures have been used to represent partial information, and unification operation is used to combine partial structures into larger structures.

The primary goal of semantic interpretation in CIDA is to extract the algebraic information from the natural queries into the intermediate representation. The overall structure of a sign in HPSG is shown below.

```
[phon α
 syn [head   [maj     β1,
              adjunct β2,
      subcat  β3]],
 sem γ]
```

where α, β_1, β_2, β_3, γ denote feature values. The *phon* attribute stores the phonological information of a word or a phrase. The *syn* attribute contains a set of syntactic features to represent the syntactic information. The feature *head* describes syntactic properties that a sign shares with its projection. The information of categories is kept in the feature *maj* and the possible adjuncts in the feature *adjunct.* The *subcat* list gives us the required arguments to form a bigger constituent. The *dbs* feature is to bear the relational algebraic information. The features that can be the values of *dbs* are listed below.

Feature	Explanation
rel	It contains a feature structure to specify a relation.
attrs	It contains a feature structure in which some attributes are restricted; it is similar to a *restriction predicate* in [3].
value	It contains an atom that specifies the instance of some attribute, or a fea ture structure that specifies the operation associated with the attribute.
quant	It contains a feature structure that describes the quantification of the relation or the attribute.

In CIDA, the hierachical information is keep in two features: *d_hier*, and *type*. The feature *d_hier* specifies the domain hierarchies of nouns and adverbs. The *type* feature specifies the syntactic type hierarchy of nouns. Type hierarchy is used to express knowledge about the structure of the things that it describes. The knowledge is useful in describing the relationships between different things, e.g., what kind of objects can play a certain role in some relations, or what kind of relations that a particular relation can modify it [19]. In CIDA, the domain hierarchies of nouns and adverbs are classified in accordance with the underlying domains of database.

There are four types of feature structures we propose for describing the algebraic content of a sign. They are value specification type, attribute specification type, relation specification type and function specification type. The features in these types we adopt are based on the syntax of the relational algebra and the functional forms in [16]. In the following sections, we will discuss these types of feature structures in detail.

3.1 Value Specification type

In Chinese inquiry sentences, some words represent instances of the attributes or operations on the attribute values. They are discussed as follows.

- Proper nouns that mean instances of certain attributes. For example, in the query *li3si4 jie4zou3 tian1long2ba1bu4 zhe4 ben3 shu1 ma1 ? (Does li3si4 borrow the book, tian1long2ba1bu4?)*, *li3si4* is a user name. The feature structure for the proper noun *li3si4* is:

```
[phon li3si4,
 syn  [head  [maj    n,
              type   proper,
              d_hier actor]],
 dbs  [value li3si4]].
```

- Verbs that associate scalar comparison operations with the attribute values, such as *da4yu2* (*is greater than*) and *deng3yu2* (*equal to*). For example, the query *lei4chu1 jia4qian2 da4yu2 yi4 bai3 yuan2 de5 shu1* (*Display the books of which the price is greater than 100 dollars.*). The lexical sign for verb *da4yu2* is:

```
[phon da4yu2,
 syn  [head [maj   v, ...
             subcat [[syn [head   [maj n,    d_hier Y]]],
                     [syn  [head   [maj n,    d_hier Y]],
                     dbs   X]]]],
 dbs  [value [gt X]]].
```

  where *gt* is a comparison operation that will check if the value of some attribute described by the subject of the verb *da4yu2* is greater than X described by the object.

- Some question words in Mandarin such as *shei2* (*who*), *na3li3* (*where*), denote that some attribute value will be projected. For example, in the query *tian1long2ba1bu4 zhe4 ben3 shu1 fang4 zai4 na3li3?* (*Where is the book located, tian1long2ba1bu4?*), *na3li3* represents a projection operation on the attribute LOCATION. The lexical sign for the interrogative pronoun *na3li3* is:

```
[phon na3li3, ...
 dbs  [value   [fun project]]].
```

  where the feature *fun* contains an atom that is a procedure name *project,* which means a projection operation that will put the projected attribute in a projection list.

- Adverbs that associate with some procedures to infer the attribute values of some domains. For example, *zou2tian1* (*yesterday*) represents operations to infer the values of domain 'time' (domain PUB_DATE or BORROWTIME). Its lexical sign is illustrated as follows:

```
[phon zou2tian1, ...
```

dbs [value [fun yesterday]]].

- A measure word that is the type of measure, such as *yuan2* (*dollar*) in the inquiry *lei4chu1 jia4qian2 da4yu2 yi4 bai3 yuan2 de5 shu1* (*Display the books whose prices are greater than 100 dollars.*), which quantifies the attribute value of domain PRICE. The lexical sign of measure word *yuan2* is shown as follows.

  [phon yuan2, ...
  dbs [value [quant [unit yuan2]]]].

3.2 Attribute Specification type

Some words in the inquiry sentences in CIDA implicitly indicate the attributes of some relation that will be specified in the future. The type is of the form:

dbs [attrs [ATTRIBUTE [VALUE-SPEC],
ATTRIBUTE [VALUE-SPEC], ...]

where ATTRIBUTE represents some attribute of some relation, and VALUE-SPEC is a feature structure of value specification type to restrict the attribute.

In CIDA, some nouns explicitly map to the attributes, such as *zou4zhe3* (*author*) to the attribute AUTHOR and *chu1ban3she4* (*publisher*) to the attribute PUBLISHER. Thus, the noun *zou4zhe3* has the following lexical sign:

[phon zou4zhe3
syn [head [maj n, ...
d_hier author],
dbs [attrs [author [value X]]]].

3.3 Relation Specification Type

Some words in the inquiry sentences specify some operations that are made on relations. The feature structures of relation specification type are taken as their algebraic content. These words may be as follows.

- The nouns that indicate the relations from which we select some tuples, such as *shu1* (*book*) and *qi2kan1* (*journal*), which denote the relations BOOK and JOURNAL, respectively.The signs for *shu1* are shown as follows:

```
[phon shu1
syn   [head   [maj n,   d_hier object,    type common,
               adjunct [[syn [head   [maj n, d_hier objname]]],
                         [syn  [head   [maj n, type classifier]]],
                         [syn  [head   [maj det]]]]]],
dbs   [rel   [select [rel  [relname  book],
                       attrs [objname [value X]]]]].
```

where the feature *select* represents a selection operation; it has two subfeatures *rel* and *attrs*. The former is a feature structure of relation specification type to specify the relation selected, and the latter is a feature structure of attribute specification type to specify some attributes restricted. The feature *relname* contains the name of relation.

- The proper nouns that are attribute values of some domain can evidently indicate which relations will be selected; that is, they are unique identifiers of some relations. For instance, the proper noun *tian1long2ba1bu4* in the query *tian1long2ba1 bu4 fang4 zai4 na3li3* (*Where is tian1long2ba1bu4 located?*) identifies the BOOK (or JOURNAL) relation in CIDA. Since *tian1long2ba1bu4* is a book name, the lexical sign for it is:

```
[phon tian1long2ba1bu4, ...
 dbs  [rel [select [rel   [relname book],
                  attrs [objname [value tian1long2ba1bu4]]]]]
```

- Some verbs that denote the selection of one relation on some attributes that may be restricted in the future. For example, *chu1ban3* (*publish*) in the query *ACM chu1ban3 na3xie1 za2zhi4?* (*What journals does ACM publish?*), means the selection of the relation JOURNAL on the attributes, PUBLISHER or PUB_DATE, which may be restricted.

- Verbs that implicitly indicates that we will join some relations. For example, *jie4zou3* (borrow) in the query *li3si4 jie4zou3 tian1long2ba1bu4 zhe4 ben3 shu1 ma1? (Does li3si4 borrow the book, tian1long2ba1bu4?)* associates the join operation with three relations, USER, BORROW, BOOK (or JOURNAL). The lexical sign for verb *jie4zou3* is:

```
[phon jie4zou3
syn  [head   [maj     v,
              adjunct [[syn [head    [maj adv, d_hier time]],
                        dbs  Y]]],
      subcat  [[syn [head    [maj n, d_hier object]],
              dbs  [rel     X2]],
              [syn  [head    [maj n, d_hier actor]],
              dbs  X1]]],
dbs  [rel [join  [[rel [select  [rel  [relname   user],
                                 attrs [username X1]]]],
                  [rel  [select [rel  [relname   borrow],
                                 attrs [borrowtime Y]]]],
                  [rel  X2]]]]].
```

where the feature *join* means the natural join operation, its value is a list of feature structures of relation specification type. For the verb *jie4zou3*, the list contains three elements. The first element is a feature structure that denotes the selection of the relation USER on the attribute NAME, described by the subject of the verb *jie4zou3*. The second specifies that the relation BORROW is selected on the attribute BORROWTIME restricted by the temporal adverb. The last is specifed by the object of *jie4zou3*.

3.4 Function Specification Type

The feature structures of this type are use to describe a few predicates that are operations made on the relations or on the attribute values. The words of this type are as follows:.

- A determinative–measure compound to reflect quantity. Determinatives may be classified into [9]:

1. Specifier, such as determinative pronoun, *zhe4* (*this*), or *na4* (*that*), or *qi2yu2de5* (the remaining), ordinalizing prefix such as *di4*. The following lexical sign illustrates the determiner *qi2yu2de5*.

   [phon qi2yu2de5, ...
   dbs [quant [spec remaining]]].

2. Number, such as definite qunatity, *san1* (*3*) or *wu3* (*5*), indefinite quantity, *duo1 shao3* or *ji3* (*how many*).

3. A measure word that is the type of classifiers, such as *ben3* (*volume*) in the query *lie4chu1 san1 ben3 jin1yuan1 xie3 de5 shu1* (*Display three volumes of books whose author is jin1yuan1.*), which is used to quantify the relation that satisfied the description. The lexical sign of measure word *ben3* is shown as follows.

   [phon ben3, ...
   dbs [quant [unit ben3]]]

- Verbs or particles that represent query types [16]. For example, the verb *you3mei2you3* (*exist*) in a Yes/No question is to check whether the relation is empty or not. The lexical sign for *you3mei2you3* is:

```
[phon you3mei2you3
 syn [head   [maj v]],
     subcat  [[syn    [head   [maj n,
                               d_hier object]]]]]],
 dbs [q_type yesno]].
```

where *q_type* means the query type, and *yesno* represents a Yes/No question.

IV. NATURAL QUERY PROCESSING

4.1 The Parsing Technique

A parser plays a significant role in natural language understanding systems. Its primary goal is to examine how the syntactic structure of a sentence can be computed and to generate an intermediate representation that can be processed furthermore. There are three types of parsers been developed in the recent two decades: top–down parsing, bottom–up parsing, and mixed mode parsing [10]. The parser we adopt is the bottom–up chart parser for it can list all ambiguous solutions and it is natural for the unification process.

The basic parsing strategy is to determine if two constituents, *C1* and *C2*, can be combined, which depends on whether *C1* is subcategorized for or adjunct of constituent *C2* or vice versa. For the former case, this relationship is checked by unifying the partial information of the *subcat* and *adjunct* features of constituent *C2* with the complete information of *C1*. When the unification is successful, the whole information of the constituent *C1* will be copied into the field of the *comp_dtrs* (complement daughters) feature or *adj_dtrs* (adjunct daughters) feature. The detailed control mechanism of the parser is stated in [18].

4.2 Interpretation Scheme

Semantic interpretation to natural language processing in earlier approaches is performed after syntactic analysis and before pragmatic processing. The separation of syntactic analysis and semantic interpretation modularizes the natural language processing system; that is, syntactic parsers can be developed without considering the problems of semantic interpretation. But it may produce a lot of syntactic structures that will be judged to be semantical anomalous. This results in the inefficiency of the system.

In our system, semantic interpretation is to get the algebraic representation of inquiry sentences. We will construct the algebraic representation of a sign that bears the relation algebraic content according to both syntactic information, such as its complements and (or) adjuncts, and relational algebra information. The basic control mechanism is very simple. The strategy is that while the parser unifies two constituents, their relational algebraic contents are also combined, which can be divided into two actions:

(1). Combining heads with adjuncts

The adjuncts of nouns may be adjectives, classifier, associative phrases, and relative clauses [Li and Thompson, 1981]; the adjuncts of verbs can be prepositional phrases, adverbial phrases or verbal phrase. For example, the relational algebraic content of the question word *duo1shao3* (*how many*) is:

dbs [quant [num howmany]].

and the content of classifier *ben3* is:

dbs [quant [unit ben3]]

By unification, *duo1shao3 ben3* has the content:

```
dbs [qunat [numhowmany,
            unit  ben3]]
```

(2). Combining heads with complements

In general, the lexical heads of phrases, such as verbs, adjectives, and prepositions, characterize a situation and the complements of them provide information about the fillers of the roles in the relations described by the lexical heads. For example, the verbal phrase *jie4zou3 dou1 shao3 ben3 shu1* (*How many books does ... borrow*) has the following content:

```
dbs [quant  [num    howmany,
             unit   ben3]],
     rel    [join [[rel [select  [rel   [relname user],
                                  attrs [name X1]]]],
                   [rel [select  [rel   [relname borrow],
                                  attrs [borrowtime Y]]]],
```

[rel [select [rel [relname book],
attrs X]]]]]]

Below we give an example to illustrate how to construct the relational algebraic content of the query *li3si4 jie4zou3 jin1yuan1 xie3 de5 na3xie1 shu1* (*Which books written by jin1yuan does li3si4 borrow?*). It will first combine the head *jie4zou3* with the complement *li3si4*. The result has the following algebraic content:

```
dbs [rel [join  [[rel [select  [rel  [relname user],
                                attrs [name [value li3si4]]]]],
                 [rel  [select [rel  [relname borrow],
                                attrs [borrowtime Y]]]],
                 [rel  X2]]]
```

Then the parser processes the noun phrase *jin1yuan1 xie3 de5 na3xie1 shu1* and produces the following result:

```
dbs [rel [select [rel [select  [rel  [relname book],
                                attrs [objname [value X]]]],
                  attrs [author [value jinyuan],
                         pubdate Y]]]].
```

Since the noun phrase is subcategorized for the verb *jie4zou3*. By structure sharing, the entire query has the following relational algebraic content:

```
dbs [rel [join [[rel [select  [rel  [relname   user],
                               attrs [name      [value li3si4]]]]],
                [rel [select  [rel  [relname   borrow],
                               attrs [borrowtime Y]]]],
                [rel [select  [rel  [select [rel  [relname book],
                                                  attrs [objname [value X]]]],
                               attrs [author [value  jinyuan],
```

pubdate Y1]]]]]]].

4.3 The Processing of Noun Phrases

While parsing, some parsing rules are fired to construct the syntactic structures of noun phrases that contain the particle "de5" [18]. In CIDA, three types of noun phrases occur frequently in queries. They are:

- Clause + de5 + N: The example is *ACM chu1ban3 de5 shu1* (*the book published by ACM*).
- Clause + de5: The example is *ACM chu1ban3 de5*.
- N1 + de5 + N2: that is N1 owns N2, and N2 represents an attribute in CIDA. The example is *li3si4 jie4zou3 de5 shu1 de5 zou4zhe3* (*the authors of the books borrowed by li3si4*).

While a noun phrase is of the pattern "Clause + de5 + N," we would first process the complements of the head verb of the clause, then check the word after *de5*. If its word category is a noun and can be subsumed by the verb, then we take the clause to be an adjnuct daughter of the head daughter and its relational algebraic content to be shared by that of the whole sign.

If a noun phrase is of the pattern "Clause + de5," that is, the word category after *de5* is not the noun, the control strategy is the same as above. We will retrieve the syntactic and semantic information of the empty NP slot in the SUBCAT list, and assign this information in the slot *head_dtr*. For example, in the phrase *ACM chu1ban3 de5,* the head verb *chu1ban3* subcategorized for two complements. First, the proper noun *ACM* is subsumed, and the corresponding entry in the subcat list of *chu1ban3* is removed. Then we check whether the word after the particle *de5* is unified with the information of the *subcat*. Since the word does not exist, we extract the information in the subcat list to the slot head_dtr.

The noun phrases of pattern "N1 + de5 + N2" denote a relation that we can select the relation described by N1 on the restricted attribute described by N2. The parsing strategy is to put N1 in the adjunct daughter of N2. Then we create the feature structure of relation specification type as shown below:

[rel [select [rel $N1_{algebra}$,
attrs $N2_{algebra}$]]]

where $N1_{algebra}$ represents the value of the feature *rel* copied from the algebraic content of N1, and $N2_{algebra}$ represents the value of the feature *attrs* copied from N2. For example, the algebraic content of the phrase *li3si4 jie4zou3 de5 shu1 de5 zou4zhe3* is shown below.

```
dbs [rel [select [rel [join [[rel [select [rel [relname user],
                                                attrs [name [value li3si4]]]]],
                              [rel [select [rel [relname borrow],
                                                attrs [borrowtime [value Y]]]]],
                              [rel [select [rel [select [rel [relname book],
                                                              attrs X]]]]]]],
         attrs [author [value X]]]]]
```

4.4 The Processing of Proper Nouns

It is not reasonable to store lexical signs for all proper nouns in the lexicon, since it will waste much memory for storing the redundant information, which the database already has, and the number of proper nouns is very large. To avoid this problem, we quote every proper noun in the inquiry sentence such as *'li3shi4' jie4zou3 'Jin1yuan1' xie3 de5 she2 mo5 shu1?" (What books written by 'Jin1yuan1' are borrowed by 'li3shi4'?)*. A proper noun may be a user name, a publisher name, a field name to which books (or journals) are classified, a book name, or a journal name. It may be classified as 'actor', 'field' or 'objname' in the domain hierarchy. Therefore, when the proper noun 'α' is passed to the parser, the parser automatically generates four lexical signs for it, instead of looking up the lexicon.

Then the parser will check the domain hierarchy to eliminate impossible lexical signs. In the query *shei2 xie3 'tian1long2ba1bu4'* the lexical signs for the proper nouns will be restricted in the domain hierarchy 'object'. There may still have several possible. In this situation, different algebraic contents are generated. Finally, when we retrieve data from the database, some algebraic contents of the query do not get the data and can naturally be eliminated. For example, if *tian1long2ba1bu4* is a book name in stead of a journal name, then the sign that represents a journal name will get no answer from the database.

4.5 Transformation to Retrieval Commands

According to the feature structures generated from semantic interpretation, some transformation rules will be fired to translate them to the corresponding retrieval commands. The form of the feature structures of attribute specification type is: "attrs [ATTRIBUTE [value INSTANCE]]" or "attrs [ATTRIBUTE [value FUN-SPEC]]." We use the following three auxiliary lists to record the restricted attributes and their associate values.

1. *instance–lst*: It records the restricted attributes and their instances.
2. *project–lst*: It records the projected attributes.
3. *operation–list*: It records the restricted attributes and the operation associated with them.

The transformation rules are stated briefly as follows.

- If the feature structure is:

  attrs [ATTRIBUTE [value INSTANCE]]

  where the value of the feature *value* is an atom, then we put [ATTRIBUTE, INSTANCE] into the list *instance–lst* as [[ATTRIBUTE, INSTANCE], ...].

- If the feature structure is:

  attrs [ATTRIBUTE [value FUN–SPEC]]

  and FUN–SPEC means a projection on the attribute ATTRIBUTE, then we record the projected ATTRIBUTE in the list *project–lst* as [ATTRIBUTE, ...]. Otherwise, we put [ATTRIBUTE, FUN–SPEC] in the list *operation–lst*.

We give an example, *li3si4 jie4zou3 jin1yuan xie3 de5 shu1 ma1? (Does li3si4 borrow the books written by jin1yuan?)* to illustrate the transformation of the feature structures of relation specification type. Its feature structure is listed below.

```
dbs [rel  [join [
                  [rel  [select  [rel  [relname   user],
                                  attrs [name       [value   li3si4]]]]],
                  [rel  [select  [rel  [relname   borrow],
                                  attrs [borrowtime Y]]],
                  [rel  [select  [rel  [select     [rel  [relname book],
                                                    attrs [objname [value X]]]],
                                  attrs [author     [value jin1yuan1],
                                         pubdate    Y]]]]]]]].
```

First, we fire the rule to interpret the three feature structures of the join relation. We will get the derived relation of the first structure, which is the relation USERS. It can be expressed in the relational algebra as "$\delta_{\Phi_{username='li3si4'}}(USER)$." The derived relation of the second is the relation BORROW, rather than a derived relation BORROWS, since the auxiliary lists, *instance–lst* and *operation–lst*, are all empty. The derived relation of the last feature structure named as BOOKS is the selection of the relation BOOK on the attribute AUTHOR whose value is *jin1yuan1*, its algebraic expression is "$\delta_{\Phi_{author='jin1yuan1'}}(BOOK)$." Next, we join three relations, USERS, BORROW, and BOOKS. The resulting relation of the whole structure is obtained and can be expressed as follows:

$$(\delta_{\Phi_{name='li3si4'}}(USER))\ \Omega\ BORROW\ \Omega\ \delta_{\Phi_{author='jin1yuan1'}}(BOOK)),$$

where Ω denotes natural join.

V. EXPERIMENTAL RESULTS AND DISCUSSION

CIDA is implemented by Quintus Prolog running on the SUN 3/60 system. The language is adopted because of its unification properties, facilities in creating the relational database and in natural language processing. We represent the relational database in a set of facts in Prolog. The following is a snapshot of the underlying databases.

book('Tian1long2ba1bu4', nov001, 'Xiao3shuo1', 'Jin1yuan1', 'Yuan3liu2', 1984, ec622)

book('An Introduction to Database Systems', dbs001, 'Zi1liao4ku4', 'C. J. Date', 'Addison–Wesley', 1986, ec622, 125)

borrow(dbs002, 7717514, 07/24/1989).

borrow(ai001, 7717543, 04/25/1990).

user(li3shu2hang2, 7717543).

user(zhang1yuan2sheng, 7717514).

Some of the sample queries are listed as follows:

1. *'Tian1long2ba1bu4' shi4 shei2 jie4zou3 de5? ('Tian1long2ba1bu4' is borrowed by whom?)*
2. *na3xie1 shu1 bei4 'Li3shu2hang2' jie4zou3? (What books are borrowed by 'Li3shu2hang2')*
3. *'Li3shu2hang2' jie4zou3 de5 shu1 (zhong1) na3xie1 shi4 'Yuan3liu2' chu1ban3 de5? (Among the books borrowed by 'Li3shu2hang2', which are published by 'Yuan3liu2'?)*
4. *bei4 'Li3shu2hang2' jie4zou3 de5 na4ben3 'Jin1yuan1' xie3 de5 shu1 shi4 shei2 chu1ban3 de5? (Is the book that is borrowed by 'Li3shu2hang2' and is written by 'Jin1yuan1' published by whom?)*
5. *'Li3shu2hang2' jie4zou3 de5 shu1 de5 zou4zhe3 shi4 shei2? (Who are the authors of the books borrowed by 'Li3shu2hang2'?)*

6. *'Li3shu2hang2' jie4zou3 na3xie1 yu3 'Zi1liao4ku4' you3guan1 de5 shu1? (What books related to 'Zi1liao4ku4' are borrowed by 'Li3shu2hang2'?)*

7. *'Tian1long2ba1bu4' zhe4 ben3 shu1 bei4 jie4zou3 duo1jiu3 le5? (How long has the book, 'Tian1long2ba1bu4', been borrowed?)*

8. *lie4chu1 san1 ben3 'Yuanliu2' chu1ban3 de5 shu1? (Display three volumes of books published by 'Yuan3liu2'.)*

9. *na3xie1 yu3 'Zi1liao4ku4' you3guan1 de5 shu1 bei4 jie4zou3? (What books related to 'Zi1liao4ku4' are borrowed?)*

10. *'Yuan3liu2' chu1ban3 de5 'Xiao3shuo1' fang1mian4 de5 shu1 you3 na3xie1? (Which books related to the field 'Xiao3shuo1' are published by 'Yuan3liu2'?)*

Below we take query *'Tian1long2ba1bu4' zhe4 ben3 shu1 shi4 shei2 chu1ban3 de5* as an example. For clarity, the algebraic content is shown in bold face.

-------------------- solution 1 --------------------

```
[phon    tian1long2ba1bu4 zhe4 ben3 shu1 shi4 shei2 chu1ban3 de5,
syn      [head   [maj v]],
dbs      [quant [unit  ben3,
                 spec  this],
         rel     [select  [rel  [select  [rel  [relname  book],
                                        attrs [objname [value tian1long2ba1bu4]]],
                                attrs [publisher [value  [fun  project]],
                                       pub_date  _3858]]]],
head_dtr
         [phon   shi4, ...
         dbs     []]
cmp_dtrs
         [phon   tian1long2ba1bu4 zhe4 ben3 shu1,
```

```
dbs     [quant [unit    ben3,
               spec     this],
        rel    [select [rel [relname  book],
                        attrs [objname [value  tian1long2ba1bu4]]]]],
head_dtr
      [phon   shu1, ...
adj_dtrs
      [phon   tian1long2ba1bu4, ...
      [phon   zhe4, ...
      [phon   ben3, ...
[phon   shei2 chu1ban3 de5, ...
dbs  [rel  [select  [rel  select  [rel  [relname  book],
                                  [attrs [objname  [value  tian1long2ba1bu4]]]],
                    attrs [publisher [value  [fun project]],
                           pub_date  _3858]]]],
      head_dtr
            [phon [],
            dbs  [rel    [select  [rel  [relname  book],
                                   attrs  [objname  [value  tian1long2ba1bu4]]]]]],
      adj_dtrs
            [phon   shei2 chu1ban3, ...
            head_dtr
                  [phon   chu1ban3, ...
            cmp_dtrs
                  [phon   shei2, ...
      de rule3
------------ There is total 1 solution. ------------
```

The following demonstration shows the actual dialogues between the user and CIDA, where the input of user is a list of words.

Quintus Prolog Release 2.2 (Sun–3, Unix 3.5)

1310 Villa Street, Mountain View, California (415) 965–7700

| ?– cida.

Welcome To
Natural Language and Image Analysis Laboratory
Departent of Computer Science and Information Engineering
National Chiao Tung University

CIDA: I am CIDA. I can help you to retrieve the libary information in AI lab. Please talk to me in Chinese.

USER: ['Li3shu2hang2',jie4zou3,na3xie1,yu3,'Zi1liao4ku4',you3guan1,de5,shu].

CIDA: Processing....Please wait for a while....

CIDA: I got it, bingo!!

An Introduction to Database Systems

USER: ['Li3shu2hang2',jie4zou3,na3xie1,shu1].

CIDA: Processing....Please wait for a while....

CIDA: I got it, bingo!!

An Introduction to Database Systems
Natural Language Understanding

USER: ['Li3shu2hang2',jie4zou3,de5,shu1,de5,zou4zhe3,shi4,shei2].

CIDA: Processing....Please wait for a while....

CIDA: I got it, bingo!!

C. J. Date
James Allen

USER: [lei4chu1,'Zi1liao4ku4',fang1mian4,de5,shu1].

CIDA: Processing....Please wait for a while....

CIDA: I got it, bingo!!

An Introduction to Database Systems
Principles of Database Systems

USER : [end].

CIDA : BYE! BYE!
yes
| ?-

VI. CONCLUSION AND FUTURE WORKS

In this paper, we have presented a new unification–based approach for processing natural inquiry sentences on the relational database. Since the semantics of some words in the inquiry sentences may have close relationship with the relational schema; they therefore can be expressed in the relational algebra. We have designed four types of feature structures, *value specification type, attribute specification type, relation specification type* and *function specification type*, as semantic representations to describe the corresponding algebraic content of the signs of the inquiry sentence. Unification is used as the primary information–combining operation to construct the syntactic and semantic content. We have performed the syntactic parsing and the semantic interpretation in an integrated way. As a result, when the parser unifies two constituents, their algebraic contents are formed. Lastly, according to the algebraic content of the inquiry sentence, we have designed transformation rules to translate the feature structure to retrieval commands and to get the desired data from the database.

The approach we propose has the following advantages: It uses feature structures as the intermediate representation. Only relevant information items are kept so that much memory space can be saved. It checks syntax of the query in a compositional way than keyword matching does. It can process more queries than template matching. The grammatical formalism is lexicon–based, so we can easily extend the system to process more inquiry sentences by adding more lexical entries and rules. It can easily be applied to the relational databases of other domains by creating a new lexicon and modifying the transformation rules according to the underlying relational schemes.

The success of implementation of the prototype of CIDA verifies that the unification approach proposed here is feasible. Some future research directions are listed below:

- To design a complex feature structure that can describe the sophisticated temporal information of the query, such as phrases, *1980 nian2 yi3hou4/yi3qian2 (after/before 1980), cong 1980 nian2 dao4 1984 nia2 (from 1980 to 1984), shang4 ge5 yue4 (last month),* etc. Also temporal inference rules should be constructed to infer the dates or time intervals described by the feature structures.

- To construct a dialogue base that contains the information of the previous queries, including the syntactic structures, the algebraic contents, and the response relations. We should construct some ellipsis resolution rules in the rule base that utilize the information of the dialogue base to resolve the elliptical queries.

- To enhance the parsing capability of the prototype of CIDA, so that the system can accept more types of queries, such as the queries that contain conjunction.

REFERENCES

1. B. J. Grosz, "TEAM: an experiment in the design of transportable natural language interface," *Artificial Intelligence*, vol. 32, pp. 173–243, 1987.

2. B. Nash–Webber and R. Reiter, "Anaphora and logical form: on formal meaning representations for natural language," *IJCAI*, vol. 1, pp. 121–131, 1977.

3. C. J. Date, *An Introduction To Database System*, vol. I. Addison–Wesley Publishing, 1986.

4. C. N. Li and S. A. Thompson, *Mandarin Chinese: A Functional Reference Grammar.* Berkeley and L.A.: University of California Press, 1981.

5. 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.

6. G. G. Hendrix, "Developing a natural language interface to complex data," *ACM Transactions on Database Systems*, vol. 3, pp. 105–147, 1978.

7. G. Gazdar, E. Klein, G. K. Pullum and I. A. Sag, *Generalized Phrase Structure Grammar*. Cambridge. Mass.: Harvard University Press, 1985.

8. 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.

9. H. T. Lin, *Essential Grammar for Modern Chinese*. Cheng & Tsui Company, Inc., Boston, Mass., 1981.

10. J. Allen, *Natural Language Understanding*. Menlo Park, California: Benjamin/Cummings, 1987.

11. J. D. Ullman, *Principles of Database Systems*. Pitman Publishing Limited, 1984.

12. K. Knight, "Unification: A Multidisciplinary Survey," *ACM Computing Surveys*, Vol. 21, NO. 1, pp. 93–123, 1989.

13. M. Wallace, *Communicating with Databases in Natural Language*. Ellis Horwood Limited, 1984.

14. C. Pollard and I. A. Sag, *Information-based Syntax and Semantics: Volume I, Fundamentals*. Stanford: Center for the Study of Language and Information, 1987.

15. S. M. Shieber, *Introduction to Unification-based Approaches to Grammar*. Stanford: Center for the Study of Language and Information, 1986.

16. S. J. Hwang, *A Dialogue Model for Information Retrieval*. Master Thesis, National Chiao Tung University, Republic of China, 1987.

17. W. A. Woods, "Semantics and quantification in natural language question answering," *Advances in Computers*, vol. 17, pp. 1–87, 1978.

18. Y. S. Chang, *Parsing Chinese Nominalizations and Coordination Constructions*, Master Thesis, National Chiao Tung University, Taiwan, 1990.

19. S. H. Lee, "A unifiction-based Approach for Chinese Inquiry Sentences Processing," M.S. thesis, National Chiao Tung University, Taiwan, 1990.