PARSING CHINESE NOMINALIZATIONS BASED ON HPSG*

Yuan-Sheng Chang and Hsi-Jian Lee†

Department of Computer Science and Information Engineering

National Chiao Tung University

Hsinchu, Taiwan 30050

ABSTRACT

A Chinese sentence parsing system based on Head-driven Phrase Structure Grammar (HPSG) is proposed in this paper. It is designed as a module of CEMAT, a Chinese-to-English MAchine Translation system. Basically, it is a bottom-up data-driven chart parser with unification as its primary operation. We augment the unification process with structure sharing. The phrases or clauses included in this paper are nominalizations. A new feature **func** is proposed to specify declaratively the properties of the sentence constructions. Also a function-rule-firing mechanism is proposed to process constructions involving functional words, such as *de5*. Incorporating with Subcategorization Principle, Adjunct Principle, and Head Feature Principle, our parser is able to parse a lot of sentences.

* This research is partially supported by Ricoh Co., Ltd.

† To whom correspondence should be addressed.

1. INTRODUCTION

The development of Machine Translation (MT) systems begins from 1950s. People want to translate automatically between different natural languages by the help of computers. MT systems are now developed in the realization that MT can be very useful though imperfect.

Natural languages, unlike programming languages, usually lack of sufficient syntactic and semantic information in the surface strings, which is helpful in determining their internal structures. This inadequacy increases the parsing ambiguities, and significantly reduces the parsing efficiency. In comparison with other natural languages, Mandarin Chinese is even more difficult to be parsed. For example, it has no inflection in the morphologies. There are also no tense and aspect markers. Such kinds of facts make the parsing of Chinese sentences more difficult than the parsing of any other natural languages.

The purpose of this paper is to design a parsing system for Chinese nominalizations, the grammatical processes by which a verb, a verb phrase, or a portion of a sentence including a verb can function as a noun phrase. In Mandarin Chinese, nominalization involves placing the particle *de5* after a verb, a verb phrase, or a portion of a sentence including the verb [1].

Many researchers have developed Chinese language processing systems for the past few years. However, most of the examples reported in their works are sentences with simple nouns, such as *Zhang1san1* and *ni3* (you) or simple NPs (noun phrases), such as *xiao3hai2zi5* (little children). Very few examples of NPs involve nominalization, such as *jiao1shu1 de5* (teachers) and *jiao1 wo3men5 ying1wen2 de5 lao3shi1* (the teacher who teaches us English). Lum and Pun [2] dealt with this issue by using Case grammar. They implemented the processor as a rule–based system, which uses especially an individual preprocessor for detecting complex NPs.

Our Chinese–to–English Machine Translation System (CEMAT) includes six modules: word identifier, syntax parser, semantics interpreter, tense and aspect determiner, lexi-

cal selector and language generator. After the sequence of input character strings is segmented into words by the word identifier module, the parser then creates all possible charts to represent the possible sentence structures. Next, a semantics interpreter will analyze the properties and meanings of the sentences based on Situation Semantics. The results are also represented by feature structures. After semantics interpretation, there is a procedure to determine the tense and and aspect information. This procedure is required because the representations of tense and aspect information in Chinese are much different with those in English. Next, we enter the transfer stage to select the corresponding words, phrases in the target language (English). Finally, the generation stage produce the translated result -- the corresponding English sentence.

This paper is dedicated to the syntactic parsing module. To reduce the parsing ambiguities, some semantic information is included for constraint checking. In fact, the processing of syntax and semantics can not be definitely separated into two modules. We adopt a unification–based grammar, HPSG, as our linguistic theory, and use an integrated syntactic–semantic approach for our parsing strategy.

Hsu [3] developed a bottom–up chart parser to process simple Chinese declarative sentences. In this paper, we go further to include the nominalization phenomena into our problem domain. We also extend the parser's power by adding a rule–based mechanism for handling some anomalous cases and augmenting the unification process to involve structure sharing.

In Section 2, we review some contemporary linguistic grammars for processing natural languages. We also introduce the grammar and parsing strategy we adopt, and explain why we choose them and their properties. In Section 3, we introduce the nominalization phenomena in Mandarin Chinese. We will give a detailed analysis of this sentence construction, and show how we can parse it with our parser. In Section 4, we explain the details of implementa-

tion. Finally, in Section5, we give a short conclusion and present some future research directions. Some examples will be giving in the appendix.

2. THE GRAMMAR AND PARSER

To design a good natural–language–processing system, a grammatical formalism plays an important role. Some grammatical formalisms are developed by linguists to describe the string set, the syntax, and the semantics of a language [4]. Examples include transformational grammar [5], definite–clause grammar [4], lexical–functional grammar [6], generalized phrase structure grammar [6, 7], head–driven phrase structure grammar [8, 9], and so on. Some grammars are originally developed by computer scientists for parsing programming languages, such as context–free grammar [10], attribute grammar, augmented transition network [11], and so on.

GPSG developed out of work by G. Gazdar at the end of the 1970s [6, 7]. It has just one level of syntactic representation, and it can solve several long–standing problems. GPSG relies on being able to pass information around trees, in which information is encoded by means of syntactic features.

Head–driven Phrase Structure Grammar (HPSG) is an immediate successor of GPSG. It makes use of many ideas from GPSG in handling syntactic categories and features [8]. However, HPSG drops out most of GPSG's grammar rules by enriching the lexicon. Nevertheless, as cited in Computational Linguistic [12], HPSG is incomplete in both its universal and language–specific components. Pollard also did not consider the computational properties. The reason why we choose HPSG as our linguistic theory is due to its clarity and declarative properties.

The overall structure of a sign in HPSG is shown below.

[phon α

syn [head [maj β_1,

adjunct β_2,

subcat β_3]],

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 feature *maj* keeps the information of categories and the feature *adjunct* the optional modifiers. The *subcat* list gives us the required arguments to form a bigger constituent. Semantic information is specified as values for the feature *sem*. Take *hit* as an example.

[phon [hit],

syn [head [maj v,

adjuncts [ADV(frequency)]]

subcat [NP(singular,human), NP(object)]]].

When this sign is combined with some NP by a unification process according to both Subcategorization principle and Head feature principle, the resulting feature structure will have the same head features with its head daughter *hit*. In addition, the agreement constraint, saying, the subject of the verb must be a singular NP (*[NUM singular]*), is directly specified in the *subcat* list. The notation *adjuncts [ADV(frequency)]* says that the verb *hit* could be modified by a adverb of frequency type. The corresponding unification principle for processing this feature is called Adjunct Principle. HPSG is in fact a theory of describing the relations between various feature structures.

Subcategorization Principle and Adjunct principle are not powerful enough to substitute all the ID rules in GPSG. In fact, the purpose of this paper is to propose some features and corresponding unification principles, inheriting from HPSG's spirit, to recognize some kinds of Chinese sentence constructions.

Since HPSG puts most information into the lexicon and drops out most of the grammar rules, it prefers a unification-based implementation rather than a rule-based parsing strategy. Unification, in general, is a process which combines two feature structures to form a more informative feature structure. In HPSG, the unification process functions more like constraint checking. For example, when parsing the sentence *She cries*, the Subcategorization Principle will unify the NP *she* with one of the *subcat* element of the verb *cries*. This unification succeeds, because the feature structure of *she* suits the constraint *NP(singular,animal)* specified in the subcat list of *cries*. Note that the notation *NP(singular,human)* is the shorthand of the feature structure *[syn [maj n], sem [num singular, d_hier human]]*. We call this process as **conditional unification**. Obviously, the conditional unification is not a commutative process. That is, A unify $B \neq B$ unify A.

An input string that can be accepted by a computer should pass the verification of a parser. After verification, we say that this input string fits the grammar defined for the language. For a legal input sentence, the parser would generate an intermediate, internal representation that can be processed furthermore. The choice of parsing strategy is generally dependent on the kind of languages and grammars which you adopt.

Augmented Transition Network is not only a grammatical formalism but also a parsing strategy. Each network is essentially a context-free grammar rule, with a set of registers to represent the intermediate structure. Each node is a state, connecting with arcs to specify the transition requirements. With each arc, there are many tests to rule out illegal input strings and action routines to construct the intermediate syntactic structure for further processing. LR parsers are originally developed by computer scientists for parsing programming languages. With a precomputed shift-reduced table, an LR parser is able to parse most programming languages deterministically. Unfortunately, most natural languages contain lots of ambiguous structures. Simple LR parsers can not handle these cases, so many augmentation methods have been developed to improve their powers [10].

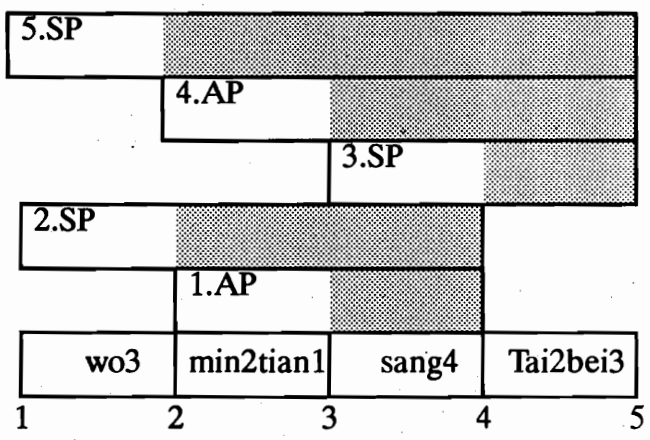
Most chart parsers nowadays are designed for rule–based systems. The basic data structures of these chart parsers are composed of three objects: grammar rules, charts, and input string queue. Cooperated with unification–based grammars, such as HPSG, the data structure of chart parsers can be simplified to only one object type: the charts. The control mechanism is now based on the unification principles, as shown by the following process.

```
%   N is the current position of the position pointer.
%   This predicate parse(N) looks for all charts adjacent the position pointer,
%   and tries to combine them according to various unification principles,
%   such as Subcategorization Principle, Adjunct Principle, and so on. The
%   predicate chart(P1,P2,F) looks for a feature structure F beginning from
%   position P1 to position P2.
parse(N) :- chart(P0,N,A), chart(N,P1,B), combine(P0,P1,A,B).
parse(N) :- M is N+1, M<max_length, parse(M).
parse(_) :- print_out_all_results.

%   This predicate combine(P0,P1,A,B) combines feature structures A
%   and B by each unification Principle. If any of them succeeds, add
%   a chart Result from position P0 to position P1, and find a new
%   chart C ending at position P0. Then, recursively call this predicate
%   to combine C and Result.
%   Subcategorization Principle
combine(P0,P1,A,B) :- subcategorization_principle(A,B,Result),
        add_a_chart(P0,P1,Result),chart(P,P0,C),
        combine(P,P1,C,Result),fail.
%   Adjunct Principle
combine(P0,P1,A,B) :- adjunct_principle(A,B,Result),
        add_a_chart(P0,P1,Result),chart(P,P0,C),
        combine(P,P1,C,Result),fail.
```

```
%   Function-rule-firing Mechanism
combine(P0,P1,A,B) :- function_rule_firing(A,B,Result),
        add_a_chart(P0,P1,Result),chart(P,P0,C),
        combine(P,P1,C,Result),fail.
%   Conjunction Principle
combine(P0,P1,A,B) :- conjunction_principle(A,B,Result),
        add_a_chart(P0,P1,Result),chart(P,P0,C),
        combine(P,P1,C,Result),fail.
```

The whole parsing process for the sentence *wo3 min2tian1 sang4 tai2bei3* is drawn in Figure 1 We put a mark on the left side of each chart to indicate the principle to be used. The numbers given on the left-upper corner of each chart indicate the generated sequence order of each chart.



Note: 1.SP stands for Subcategorization Principle, and AP stands for Adjunct Principle.
2.The shaded area separates a chart into two subparts which compose this chart.

Figure 1. The parsing process for sentence *wo3 min2tian1 sang4 Tai2bei3*.

As we can see from Figure 1, this parser works bottom-up, breadth-first, and adopts an exhaustive search. This implies a poor efficiency. However, it can generate all possible

parsing results. Thus, it is good for debugging during the developing process and is suitable for our needs.

3. NOMINALIZATION

Analysis of complex noun phrases (NPs) is a difficult issue in Chinese language processing. Many researchers have developed Chinese language processing systems for several years. Most of the examples reported in their works are sentences with simple nouns such as *zhang1_san1* and *ni3* (you) or simple NPs such as *xiao3hai2zi5* (little children). Very few examples of NPs involve nominalization or relative clauses using *de5* such as *jiao1_shu1 de5* (teachers) and *jiao1 wo3men5 ying1wen2 de5 lao3shi1* (the teacher who teaches us English). Lum and Pun dealt with this issue in their work [2]. Their method is based on Case grammar and is implemented as a rule-based system, which uses especially an individual preprocessing module for detecting complex NPs.

A nominalization may be used as a noun phrase, a relative clause construction, or may serve as the complement to an abstract head noun [1]. Zhu [13] regarded that a nominalization is that a nominalizer "de5" follows a verb or a verbal phrase and the whole construction serves as a nomal component. To illustrate these phenomena, consider the following two examples:

1. *zhei4 zhong3 zhi2wu4 ke3yi3 dang1zuo4 chi1 de5.*

   (This type of plants can be taken as food.)

2. *zhong4 shui2quo3 de5 nong2ren2*

   (the farmer who grows fruit)

3. *xie3 xin4 de5 mao2bi3*

   (the brush to write letters)

4. *xing2zheng4yuan4zhang3 ci2zhi2 de5 xin1wen2*

   (the news that the premier of the Executive Yuan resigns)

The nominalization *chi1 de5* in the first sentence functions as an NP. It means something eatable. While the nominalization *zhong4 shui2quo3 de5* (growing fruit *–de5*) in the second example functions as a relative clause modifying the head noun *nong2ren2* (farmer), where a relative clause is a clause that restricts the reference of the head noun. In other words, there is a noun phrase just following *de5*, and this noun phrase actually refers to one of the unspecified participants in the situation named by the nominalization. In this example, the noun phrase *nong2ren2* refers to the unspecified subject role of the situation named by the nominalization *zhong4 shui2quo3 de5.* And the nominalization *zhong4 shui2quo3 de5* serves as a relative clause modifying *nong2ren2.* Example 3 is another case where the nominalization *xie3 xin4 de5* serves as a relative clause modifying the head noun *mao2bi3*. However, there is one difference between examples 2 and 3. The head noun *nong2ren2* in example 2 actually refers to one of the unspecified but necessary participants of the verb *zhong4* of the nominalization *zhong4 shui2quo3 de5*, while the head noun *mao2bi3* in example 3 refers to an optional participant of the verb *xie3* of the nominalization *xie3 xin4 de5*. From the HPSG viewpoint, the head noun *nong2ren2* fits one of the unspecified arguments in the *subcat* list of the verb *zhong4*, while the head noun *mao2bi3* fits one of the unspecified arguments in the *adjunct* list of the verb *xie3*.

In the last example, the nominalization *xing2zheng4yuan4zhang3 ci2zhi2 de5* (the premier of the Executive Yuan resigns *–de5*) modifies the head noun *xin1wen2* (the news). It is very similar with example 3 in having a saturated verb phrase in the nominalization. But, different with examples 2 and 3, the head noun *xin1wen2* is neither a necessary nor an optional argument of the clause *xing2zheng4yuan4zhang3 ci2zhi2*. Instead, the nominalization modifies the head noun by specifying an event as its content. In other words, the difference is on the semantic relation between the nominalizations and the head nouns. In summary, we list a classification of the usages of nominalizations in Table 1.

Table 1. Classification of the Usages of Nominalizations

Function of the nominalization	Constituent following the nominalization
1. being a noun phrase itself	Nil
2. serving as a relative clause	a head noun, which can fit the specification of the
i) referring to an obligatory argument	1. obligatory argument
ii) referring to an optional argument	2. optional argument
3. serving as a modifier	an abstract head noun

In the following sections, we will discuss each of the above cases, and propose a special feature **func** and a corresponding unification principle to recognize all of these constructions. They have been implemented as a subset of our chart parser.

3.2 A Nominalization without a Head Noun

If the word or phrase following *de5* can not form a noun phrase, the nominalization is itself a head noun, called *nominalized noun phrase* later. We shall assign its syntactic and semantic features according to some conventions. Since the nominalized noun phrase is generally a relative clause, there is at least one unspecified participant of the verb. We need only check which one it is. Then we can determine the role of the nominalized phrase. For the phrase *jiao1 wo3men5 ying1wen2 de5* (teach us English –*de5*), the only unspecified syntactic role is the subject. Thus, we can first assign the syntactic and semantic information of the nomalization by copying the subject information from the *subcat* list of the head verb of the clause. The resulting structure is shown as follow.

[phon [jiao1, wo3men5, ying1wen2, de5],
syn [head [maj n]],
sem [d_hier human],
adj_dtr [phon [jiao1, wo3men5, ying1wen2],
syn [head [...],
subcat [

[subj, [syn [head [maj n]],
sem [d_hier human]]]],
...]].

The element **[subj, left, [syn [head [maj n]], sem [d_hier human]]]** in the **subcat** list of the relative clause *jiao1 wo3men5 ying1wen2* says that in regular case there must be an NP, which belongs to the **human** type in the domain hierarchy, on the left side of this cluase, and the NP functions as a subject (subj) of this clause. Similarly, the nominalization *ta3 mai4 wo3men5 de5* in *ta3 mai4 wo3men5 de5 dou1 shi4 ci4ji2pin3* (What he sold to us are all goods of inferior quality.) refers to the direct object (what is sold) of the verb *mai4*. However, if more than one participant are not specified, how can we determine the syntactic and semantic properties of the nominalization ? How do we know that the nominalization *wo3 mai4 de5* in *wo3 mai4 de5 shi4 zhong1guo2 huo4* (What I sell are Chinese merchandise) refers to the direct object, what is sold, rather than the indirect object, buyer? Even more, all the three participants in the nominalization *mai4 de5* in *mai3 de5 bu4 ru2 chu1_zu1 de5 hao3* (What is for sale is not as good as what is for rent) are not specified. Yet, we know that *mai4 de5* refers to the direct object (what is sold) rather than the others. To explain these phenomena, Li and Thompson proposed the following four rule:

1. To be used alone as a noun phrase, a nominalization must contain a verb with at least one of its participants unspecified.

2. If there is only one participant unspecified, the referent of the nominalization is the same as that of the missing participant.

3. If both the subject and direct object participants are unspecified in a nominalization, then the nominalization will generally be understood to have the same referent as the unspecified direct object participant of that verb.

4. A nominalization used alone as a noun phrase never refers to the indirect object participant.

Take the verb *song4* (give) as an example. Since the verb *song4* is a ditransitive verb, it requires three participants, either overtly specified or understood: a subject denoting the giver, an indirect object the receiver, and a direct object the given entity. In Chinese discourse, the receiver (indirect object) whom both the talker and the listener know is sometimes omitted. Yet we understand that the verb *mai4* actually requires three participants. According to the above four rules, the nominalizations, which are underlined, in the following four sentences can get suitable interpretations.

5. *song4 de5 bu4 ru2 zi4ji3 mai3 de5 hao3.*(direct object – goods)
   (What is given free is not as good as what is for sell.)

6. *wo3 song4 de5 shi4 yi2 ben3 shu1.*(direct object)
   (What I give is a book.)

7. *song4gei3 Li3si4 de4 shi4 zui4 qui4 de5.*(direct object)
   (What is given to Li3si4 is the most expensive.)

8. *song4 huo4 de5 da4ban4 dou1 shi4 nan2ren2.*(subject)
   (Goods delivers are mostly men.)

To implement these four rules in HPSG form, the feature structure of the grammatical particle *de5* is defined as the following form:

[phon [de5],
func de5].

Here we introduce a new feature **func** for firing grammar rules. During the bottom–up parsing process, if we find out any feature structure containing a **func** feature, then the value of this feature is adopted as a predicate name and is fired. We will call this firing operation as function–rule–firing mechanism:

If a feature structure A contains a feature **func**, then
the value of this feature is taken as a predicate name and is fired.

A word with a **func** feature is called a **functional word**, for the way to combine the word with others is different with the regular unification processes. Since we can hardly specify its behaviors by some declarative unification principles, we use a rule-based mechanism to describe the way which a functional word combines with other feature structures. For the nomalization, the value of **func** feature is *de5*, so the grammar rules named *de5* will be fired.

```
%     N is the current position of the position pointer, L is the total
%     length of the input string.
%     get_feature_value(A,list_of_features,Value) gets the value on the
%     path list_of_features from feature structure A.
parse(N,L):- chart(M,N,A), get_feature_value(A,[func],A_func),
                P=.. [A_func,M,N,L], call(P).

%     (M,N) are position of the functional word de5
de5(M,N,L):- chart(P0,M,A),
                get_feature_value(A,[syn,head,maj],A_maj),
                ( (A_maj= =a) -> de5_rule1(P0,N,A)
                | (A_maj= =n) -> de5_rule2(P0,N,A)
                | (A_maj= =v) -> de5_rule3(P0,N,A)
                | otherwise -> fail ).
```

The grammar rule for *de5* contains three sub-rules. What we care about is the case when there is a VP clause proceeding *de5*, that is, the nominalization construction. The rule *de5_rule3* is proposed as follows.

```
%     P0 is the starting position of clause A,
%     N is the ending position of de5
de5_rule3(P0,N,A):-
        get_feature_value(A,[syn,subcat],[H|T]),
        H=[Role,_,F], Role\= =indirect_obj,
```

```
Result=[phon      [A_phon,de5],
        syn       [head  F_head,
        sem       F_sem,
        head_dtr  F,
        adj_dtrs  [A] ],
add_a_chart(P0,N,Result),
chart(P,P0,C), combine(P,N,C,Result).
```

The above rule will always bind the variable **H** with the first element in the **subcat** list of **A** . In order to assign the direct object the highest priority, we need just to put it at the first position in the **subcat** list of the verb. As a result, the **subcat** list of a ditransitive verb will be:

[direct_obj, subject, indirect_obj]

and that of a transitive verb is:

[object, subject].

Note that above arrangement is different from the one proposed by Pollard and Sag, who claim that the order of the elements in the **subcat** list is determined by the so called obliqueness. Figure 2 shows the parsing result of the nominalized noun phrase *song4gei3 Li3si4 de5*. Note that the unsaturated direct object **NP(concrete)** of the clause *song4gei3 Li3si4* is copied to be a head daughter of the resulting nominalization. Thus the category of the nominalization is an NP ([maj n]) rather than a VP([maj v]).

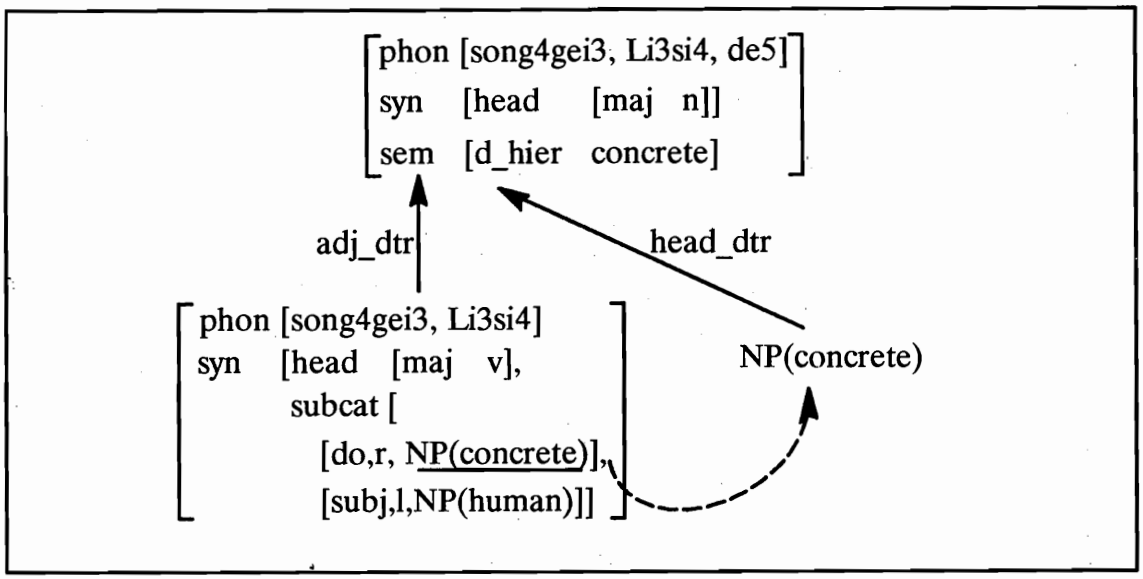


Figure 2. The nominalized noun phrase *song4gei3 Li3si4 de5.*

Unfortunately, there are some cases which do not obey the above analyses. For example, the nominalization *jiao1 wo3men5 de5* in *jiao1 wo3men5 de5 shi4 yi2 ge5 hao3 lao3shi1* (The person who teaches us something is a good teacher) obviously refers to the subject role (the teacher) of the verb *jiao1*, though both the subject and direct object participants are not specified. For this special case, we need only rearrange the subcategorization order of the verb *jiao1* as follows:

[subject, direct_obj, indirect_obj]

The nominalization *jiao1 wo3men5 de5* will now refer to the teacher, rather than the course.

3.3 A Nominalization with a Head Noun as a Complement

A nominalization can also serve as a relative clause of a head noun. In sentence *jiao1 wo3men5 ying1wen2 de2 liao3shi1 min2tan1 chu1kuo2* (The teacher who teaches us English will go foreign tomorrow), the noun phrase following *de5*, that is, *lao3shi1*, obviously refers to the unspecified subject role (teacher) in the *subcat* list of the verb *jiao1*. The nominalization functions like a relative clause modifying the noun phrase. The resulting feature structure is shown below.

```
[phon     [jiao1, wo3men5, ying1wen2, de5, lao3shi1],
syn       [head     X],
head_dtr  [phon     [lao3shi1],
           syn      [head     X] ]
adj_dtrs  [[phon    [jiao1, wo3men5, ying1wen2],
           syn      [...
                     subcat [Y] ]] ].
% NOTE: X is uniable with Y.
```

To recognize this kind of constructions, another rule for *de5* is designed below:

```
%     P0 is the starting position of clause A,
%     N is the ending position of de5
de5_rule3(P0,N,A):- chart(N,P1,B),
        get_feature_value(A,[syn,subcat],A_subcat),
        member(A_subcat,I),
        unify(I,B,IB)
        Result=[phon      [A_phon,de5,B_phon],
                syn       [head       IB_head],
                sem       [IB_sem],
                head_dtr  IB,
                adj_dtrs  [A] ],
        add_a_chart(P0,P1,Result),
        chart(P,P0,C),combine(P,P1,C,Result).
```

Taking sentence *ying2 de5 ren2 yiao4 qing3ke4* (The winner must be a host.) as an example, the *subcat* list of the ditransitive verb *ying2* (win) is:

```
subcat [    [dir_obj, right, NP(object)],
            [subj, left, NP(human)],
```

[indir_obj, right, NP(human)]].

When the parser finds the functional word *de5* at position (2,3), the rule packet of *de5* is checked. Since the *maj* of the feature structure immediate leading *de5* is v (verb), the rule *de5_rule3* is fired. It first checks if the chart immediately following *de5* can be unified with any element of the *subcat* list of the verb *ying2* (win). The checking obeys the order of the *subcat* list. Since the first *subcat* element NP(object) fails to unify with *ren2* which is an NP(human), the next element NP(human) is tried. This time the unification succeeds, a resulting feature structure is built and is asserted into the database.

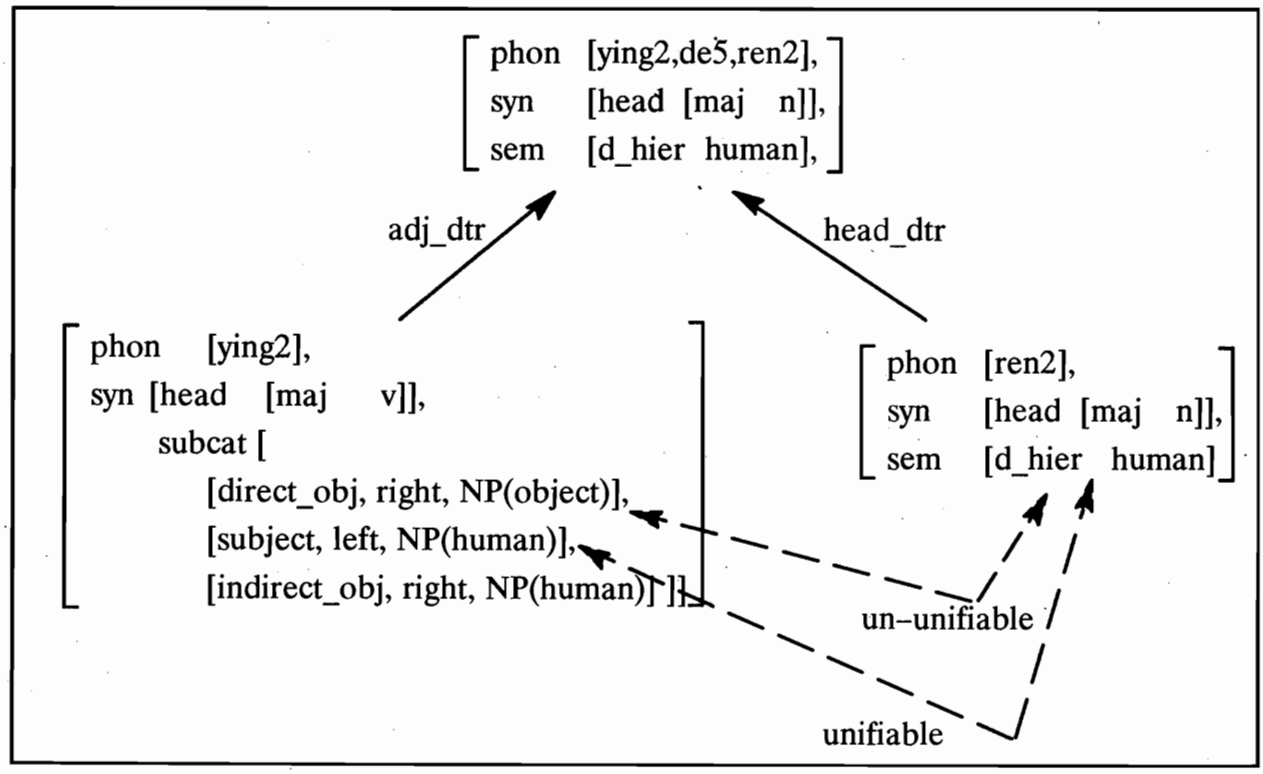


Figure 3. The parsing tree of *ying2 de5 ren2* (the winner).

Similarly, in the following sentences, the head nouns refers to the unspecified direct object, unspecified subject, and unspecified subject role of the verb in the nominalization, respectively.

9. *mai4 gei3 Li3si4 de4 yi1fu2 shi4 zui4 qui4 de5.*

(The clothes sold to Li3si4 is the most expensive.)

10. *jiao1 shu1 de5 ren2 da4_ban4 dou1 zhi4 hao3 ren2.*

(Teachers are mostly good people.)

11. *jiao1 wo3men5 de5 Wang2xian1sheng1 ming2tian1 chu1kuo2.*

(Mr. Wang who is our teacher will go abroad tomorrow.)

3.4 A Nominalization with a Head Noun as an Adjunct

There is another case where the head noun following *de5* does not refer to the obligatory participant in the **subcat** list of the verb of the nominalization. Instead, the head noun refers to some other participant involved in the situation named by the relative clause, such as an instrument used, the location or time at which the event happens, or even the reason for which or the method by which it occurs. Below are some examples.

12. *xiu1li3 shui3guan3 de5 ju4zi5* (instrument)

(the saw to repair the water pipe)

13. *zhang1san1 hua4 huar4 de5 fang2jian1* (location)

(the room where Zhang1san1 does his painting)

14. *lian4 zu2qiu2 de5 ji4jie2* (time)

(the season when one practices soccer)

15. *wo3 lai2 zher4 de5 yuan2gu4* (reason)

(the reason why I came here)

16. *pa2shou3 tou1 dong1_xi1 de5 fang1fa3* (method)

(the method by which pickpockets steal things)

When the head noun following *de5* fails to unify with the unspecified participants of the **subcat** list of the verb of the nominalization, we then check if the content of this head noun belongs to the type of instrument, location, time, reason, method and so on in the semantic domain hierarchy. If it does, a feature structure is constructed in which the head noun fits in the head daughter feature, and the nominalization fits into the adjunct daughters list.

For example, the feature structure for the noun phrase *pa2shou3 tou1 dong1xi1 de5 fang1fa3* (the way that pickpockets steal) is:

```
[phon     [pa2shou3, tou1, dong1xi1, de5, fang1fa3 ],
syn       [head     X],
head_dtr  [phon     [fang1fa3],
           syn      [head     X] ]
adj_dtrs  [[phon    [pa2shou3, tou1, dong1xi1],
           syn      [...]] ]
```

The rule for processing this case is shown below.

```
%     P0 is the starting position of clause A,
%     N is the ending position of de5.
de5_rule3(P0,N,A):- chart(N,P1,B),
        get_feature_value(B,[syn,head,maj],n),
        get_feature_value(B,[sem,d_hier],B_domain),
        is_a(B_domain,[instrument,location,time,reason,method]),
        Result=[phon      [A_phon,de5,B_phon],
                syn       [head      B_head],
                sem       B_sem,
                head_dtr  B,
                adj_dtrs  [A] ],
        add_a_chart(P0,P1,Result),
        chart(P,P0,C),combine(P,P1,C,Result).
```

3.5 A Nominalization with an Appositive Head Noun

The most important characteristic of this noun complement construction is that the head noun is always abstract and does not refer to any entity, specified or unspecified, in the

modifying clause. In other words, the nominalization functions as an appositive clause of the head noun. Below are some examples with the head noun underlined:

17. *wo3men5 he2zuo4 de5 wen4ti2 hen3 jian3dan1.*

(the problem concerning our cooperation is very simple.)

18. *wo3men5 zu1 fang2zi5 de5 shi4*

(the matter concerning our renting a house)

19. *xing2zheng4yuan4zhang3 ci2zhi2 de5 xin1wen2*

(the news that the premier of the Executive Yuan resigns)

Despite of the linguistic viewpoint, the process to recognize this noun phrase construction is almost the same with the previous one, except that the content of the head noun no longer belongs to the instrument, location, time, reason, or method type but rather belongs to the **event** type in the NP domain hierarchy. Thus we just need to add a new condition into the previous *de5_rule3* to check further if the domain hierarchy of the head noun is an event type. The third subgoal of the previous *de5_rule3* is now modified as follows.

is_a(B_domain,[instrument,location,time,reason,method,**event**]),

4. IMPLEMENTATION

Our parser has been implemented with Quintus Prolog on a Sun 3/60 workstation. The reason why we choose Prolog is due to its good facilities for unification, recursion, and data representation. In addition, Quintus Prolog has supposed an excellent environment involving necessary tools for developing systems

Since our chart parser adopts exhaustive search, all possible substructures will be built during the parsing process. The order to apply the above unification principles will not affect the parsing results. That is, for any two adjacent charts, all of the unification principles (SP, AP, CP) and the function–rule–firing mechanism will be applied to check if the two charts can be combined.

Grammar rules are used to handle anomalous cases which can not be declaratively specified in the feature structures. In this paper, we recognize the nominalization constructions by grammar rules. In order to reduce to rules to be tried, a feature **func** is used for firing the required rules. The value the **func** feature is taken to be the name of the rules. For example, a feature–value pair **[func de5]** is specified in the feature structure of the functional word *de5*. When the parser detects the **func** feature, it takes the value *de5* as a predicate name and fires it.

The reason why we use grammar rules to handle the nominalization constructions instead of some unification principles is that these constructions can not be easily specified. Sometimes a nominalization functions as an NP, sometimes it functions as an relative clause construction modifying the following head noun. In the former case, the parser need to reference two feature structures at a time (the clause and *de5*), while in the latter case, the parser needs to refer to three feature structures at a time (the clause, *de5*, and the following head noun). For either case, one needs to check the *subcat* list of the leading clause.

Below we give an example to illustrate the parsing result of the input sentence, expressed as a list of words, [wo3, xi3_huan1, xi3_huan1, zhong1_guo2, de5, nu3_hai2].

```
---------- Solution 1 ----------
phon      wo3,xi3_huan1,xi3_huan1,zhong1_guo2,de5,nu3_hai2
syn       head      maj       v
                    adjuncts  .....
          subcat    .....
sem
head_dtr
          phon      xi3_huan1
          syn       head      maj       v
                              adjuncts  .....
                    subcat    .....
          sem
cmp_dtrs
          phon      wo3
```

```
syn      head     maj      n
sem      var      per      1
                  num      sg
                  d_hier   human
         rest     reln     referring
                  referent speaker
phon  xi3_huan1,zhong1_guo2,de5,nu3_hai2
syn      head     maj      n
sem      var      per      3
                  d_hier   human
head_dtr
         phon     nu3_hai2
         syn      head     maj      n
         sem      var      per      3
                  d_hier   human
adj_dtrs
         phon     xi3_huan1,zhong1_guo2
         syn      head     maj      v
                           adjuncts .....
                  subcat   .....
         sem
         head_dtr
                  phon     xi3_huan1
                  syn      head     maj      v
                                    adjuncts .....
                           subcat   .....
                  sem
         cmp_dtrs
                  phon     zhong1_guo2
                  syn      head     maj      n
                  sem      var      d_hier   space
---------- There is total 1 solution.----------
```

5. CONCLUSIONS

In this paper, we have analyzed the Chinese nominalizations and have designed a special feature **func**. A function–rule–firing mechanism is introduced to recognize constructions involving functional words, such as *de5*. All of the work described in this chapter has been implemented, without a specific NP preprocessing module.

It is still a long way to implement a practical Chinese–to–English machine translation system. Among those modules, the parser is the most difficult module to implement because Chinese sentences contain many anomalous structures which are remained to be exploited.

In the future, we are going to analyze more Chinese sentence constructions, such as Serial Verb Construction (SVC), Topicalization, *ba3* and *bei4* constructions, etc. And we will augment the power of the parser, trying to integrate the syntax and semantics analyses in a single process. In addition, the efficiency of the parser needs some improvement. There are many redundant constructions of constituents being generated during the parsing process. We should apply some other mechanisms, such as top–down predictions, and other constraints to make the parser more efficient.

The lexicon is another big problem in MT systems; especially those based on unification–based grammars which keep most information in the lexicon. Building a complete lexicon is a huge work. In the future, we are going to discuss the structure of the lexicon, giving a more efficient data retrieval mechanism and a flexible and user–friendly updating method.

REFERENCES

1. C. N. Li and S. A. Thompson. *Mandarin Chinese: a Functional Reference Grammar*, Berkerley, CA: University of California Press, 1981.

2. B. Lum and K. H. Pum, "On parsing complex noun phrases in a Chinese sentence", in *Proc. Int Conf. on Comp. Processing of Chinese and Oriental Lang.* Toronto, Canada, 1988, pp. 470–474.

3. P.-R. Hsu, "Parsing Chinese Sentences in Head-driven Phrase Structure Grammar," Master thesis, National Chiao Tung University, Hsinchu, Taiwan, R.O.C., 1989.

4. S. Shieber, "An Introduction to Unification-Based Approaches to Grammars," CSLI Lecture Notes, No. 4, Standford: Center for the Study of Language and Information, 1986.

5. T. C. Tang, *Studies in Tansformational Grammar of Chinese, Vol. I: Movement Transformations,* Taipei, Taiwan: Student Book Co., 1986.

6. P. Sells, "Lectures on Contemporary Syntactic Theories: An Introduction to Government-Binding Theory, Generalized Phrase Structure Grammar, and Lexical Functional Grammar," CSLI Lecture Notes, No. 3, 1985

7. G. Gazdar, E. Klein, G. K. Pullum and I. A. Sag, *Generalized Phrase Structure Grammar.* Oxford: Basil Blackwell, 1985.

8. C. Pollard and I. A. Sag. "Information-based Syntax and Semantics: Volume I. Fundamentals," CSLI Lecture Notes, No. 13, 1987.

9. D. Proudian and C. Pollard, "Parsing head-driven phrase structure grammars," in *Proc. 23rd Annual Meeting of the Assoc. for Comput. Linguist.*,July 1985, pp. 167-171.

10. J. E. Hopcroft and J. D. Ullman, *Introduction to Automata Theory, Languages, and Computation.* Addison-Wesley Publishing Company, 1979.

11. J. Allen, *Natural Language Understanding.* Menlo Park, CA: Benjamin/Cummings, 1987.

12. E. P. Stabler Jr., Book Reviews, *Computational Linguistics*, vol. 15, No. 3, 1989, pp. 198-200.

13. D. X. Zhu, *Han4yu3 zhi1shi4 cong2shu1-Yu3fa3 da2wen4 (Inqueries and Answers of Grammars)*, (in Chinese), Peking, 1985.