A COGNITIVE TREATMENT OF ASPECT IN JAPANESE TO CHINESE MACHINE TRANSLATION

Miao-Ling Hsieh and June-Jei Kuo
Matsushita Electric Institute of Technology
(Taipei) Co., LTD.

ABSTRACT

Sentential aspect is the integrated function of lexical main verbs, aspect markers, adverbials, subjects and objects, and other syntactic constituents. The present approach represents sentential aspect by situation types and further distinction of aspectual meaning. Situation types are the basic categorization of situations that human make on the basis of their perceptual and cognitive faculties. Seven situation types are distinguished in this article, which include events (accomplishments, processes, achievements and activities), states, habituals and generics. Accomplishments and achievements express perfectivity, whereas others imperfectivity. Perfectivity and imperfectivity can be further distinguished into subdivions such as 'telic' and 'perfective', and 'habitual', 'delimitative' and 'continuous' respectively. Based on the situation type and the further distinction of aspectual meaning of a source language, the generation of verbs, aspect markers and adverbials for a target language can be properly made. In this paper, the translation of aspect from Japanese to Chinese is described in detail. The problem that several aspect markers occur together in a sentence is also discussed.

I. INTRODUCTION

It has been observed that the aspectual properties of sentences are not determined simply by their lexical main verbs.[1] Rather, a large variety of syntactic constituents play a role in this determination, which include aspect markers, adverbials, subjects and objects, and other syntactic constituents such as prepositional phrases. According to different aspectual properties of a sentence, situation types, which are the basic categorization of situations that human make on the basis of their perceptual and cognitive faculties, can be distinguished. In this article, we follow Smith (1986) and distinguish seven types of situations as follows:[1]

Figure 1: Situation types and perfectivity

Except events and states, we also distinguish habituals and generics. Habitual situations are characterized by an extended period of time, whereas generic one refers to situations that denote facts, common sense, knowledge and so on. Events can be distinguished into happenings and actions. If the subject of a sentence is agentive, it denotes an action situation. Otherwise, a sentence denotes a happening situation. Both happenings and actions can be further distinguished into achievements and processes, and accomplishments and activities respectively. Achievements and accomplishments are percepted as a whole and thus are perfective, whereas processes and activities are regarded to continue for a period of time and are imperfective.

The perfective and imperfective oppositions can be mainly represented as follows: [2][2]

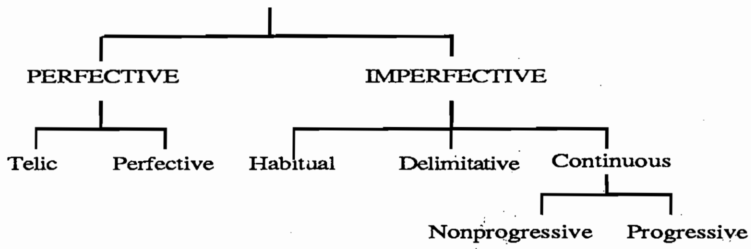


Figure 2: Aspectual oppositions

In the subdivisions of 'conitnuous', a distinction is made between 'progressive' and 'nonprogressive', the former being the combination of continuousness with nonstativity.

While many languages do have a single category to express perfectivity and another to express imperfectivity, there are other languages where perfectivity and imperfectivity are subdivided into a number of distinct categories, and yet others where there is some category that corresponds to part only of the meaning of perfectivity or imperfectivity.[2] The aspectual system of a particular language has to be investigated. Our approach is to represent sentential aspect by situation types and further distinctions of aspectual meaning and generate verbs, aspect markers and adverbials for a target language based on situation types and further distinctions of aspectual meaning of a source language. The present treatment of aspect in machine translation should be like the following figure:

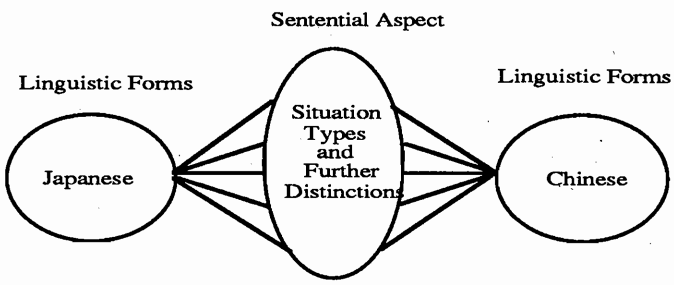


Figure 3: The treatment of aspect in machine translation

In the existing algorithms for processing Japanese, sentential aspect is identified to be such as 'start', 'continue', 'completion', and 'result.'[3][4] Our approach is different from those algorithms in that both the notion of situation types and further distinctions of aspectual meaning are suggested to be put into the treatment of aspect in machine translation mainly for distinguishing conceputal differences. For example:[3]

(1) 私が入った時、彼は死んだ。(Achievement, perfective)
當我進門的時候，他死了
'When I came into the door, he died.'

(2) 終戦の年には、もう祖父死んでいます。(State, perfective)
大戰結束那年，祖父已經死了
'At the year that the war ended, my grandfather was dead.'

The main clause of (1) is categorized into an achievement situation and (2) a state. Best translation should depend on both of the situation type and the further distinction of aspect that are intended by the speaker. In (2), the English adjective 'dead' is more proper than the verb 'die' to describe a state. For Chinese, the state of 'deadness' is better expressed by the verb '死' 'die' plus the perfective marker '了' and the adverb '已經' 'already' since the Chinese lexicon does not distinguish the action from the state for deadness.

On the other hand, sentential aspect can be marked by aspect markers or not depending on the aspectual systems of particular languages. For example, for some Japanese accomplishment and achievement situations they are not marked by any aspect markers and simply occur in past tense (as in (3) and (4)). Besides, certain syntactic constructions (as in (5)) also contribute to the sentential aspect of the sentence. For example:

(3) 私は食べました。(Accomplishment, perfective)
我吃了
'I have eaten.'

(4) 彼は死んだ。(Achievement, perfective)
他死了
'He died.'

(5) 私は日本へ行くことがありません。(State, experiential)
我沒去過日本
'I have not been to Japan before.'

The use of situation types and further distinctions are easy for the treatment of the sentential aspect of those sentences that are not marked by any aspect markers.

In section II and III we will first introduce the aspectual systems of Chinese and Japanese respectively. Section IV will show how such an approach can be implemented in a Japanese to Chinese machine translation, followed by section V that the combination of aspectual meaning will be discussed. Finally, the concluding remarks of this paper will be included in section VI.

II. ASPECTUAL SYSTEM OF CHINESE

In this section and the following one, we will first introduce the aspectual systems of Chinese and then Japanese.

The aspects of Chinese are perfective and continuous (or durative), typically indicated by the perfective marker '了', and the continuous markers '(正)在' and '著.' The former presents a situation from a perfective viewpoint and the latter an imperfective view. Besides, '過' is known as an experiential marker, and '起來' and '下去' are two continuous markers which derive their aspectual meaning from their lexical meaning 'up' and 'down' respectively.[4] Other markers such as '完' 'finish' and '好' 'good' are complements that mark telic and perfective meaning respectively.

In determining the situation type of a sentence, verbs can be classified according to their inherent meaning or to aspect markers that they can co-occur with. In Chinese, verbs can be divided into action, process, telic, and state verbs. The classification of Chinese verbs are exemplified as follows:

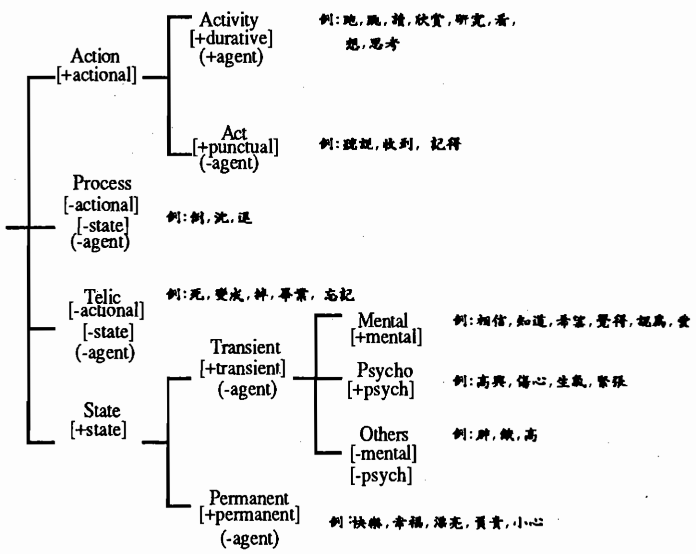


Figure 4: Verb classification of Chinese

Action verbs can be further distinguished into activity and act verbs. The former are durative and the latter are punctual. Two categories of states are distinguished: transient and permanent. The former are subdivided into mental state, psychological state and others. In the three types of state verbs, only psychological state verbs can occur with the continuous marker

(正)在.' This is very similar to English that lexically stative verbs can be used nonstatively and appear in the progressive.[2] On the other hand, only mental state verbs can be marked by the continuous marker '著.' Telic verbs cannot be marked by the constituous marker '(正)在' but adverbs such as '快要' 'almost,' or '漸漸' 'gradually' to refer to a process that necessarily comes to an end. Process verbs can be modified by '在' and means that a process is undergoing.

Action, process and state form a continuum. In Chinese, it is very often that action, process and state are polysemies. For example, the verb '開' 'open' and '停' 'stop' can be an action, a process or a state and occur in an accomplishment, an achievement and a state situation respectively as follows:

```
e.g. Accomplishment  Achievement          State
     *---------------------*------------------------->
     開了                  開了                開著

     Accomplishment  Achievement          State
     *---------------------*------------------------->
     停了                  停了                停著
```

Action and process verbs are often marked by the perfective marker '了' to mean an accomplishment and an achievement respectively. In some other cases, the vocabulary might distinguish. For example:

```
Accomplishment  Achievement          State
*--------------------*------------------------------>
走了                 走了                不見了

                   Achievement          State
                   *------------------------------>
                   死了                 已經死了
```

In the case of '死' 'deadness,' there is not any state verb for '死' 'deadness.' However, a state situation can be expressed by the verb '死' 'die,' the adverb '已經' 'already' and the perfective marker '了.' For the treatment of aspect, action, process and state verbs that form a continuum are suggested to be recorded in the dictionary as a unit.

In some languages, it is possible to derive verbs referring to specifically telic situations from verbs that do not necessarily refer to telic situations, usually as part of the derivational morphology.[2] Chinese is such a language that a resultative compound can be formed by an action verb plus a complement to express the resultant state of the action including such as '完' 'finish' and '好' 'good' and many other complements that the cooccurrence of which is mainly dependent on different verbs.

The situation types and the interpretations of Chinese aspect markers are listed as follows:

Table 1: The situation types and the interpretations of Chinese aspect markers

ST / AM	Accomplishment	Activity	Achievement	Process	State	Habitual	Generic
了	'perfective' 'telic'	*	'perfective' 'telic'	'inceptive'	'perfective'	*	*
過	'experiential'	*	'experiential'	*	'experiential'	*	*
(正)在	*	'progressive'	*	'progressive' 'repetitive'	*	'habitual'	*
著	*	'progressive'	*	*	'nonprogressive'	*	*
起來	*	'inceptive'	*	'inceptive'	*	*	*
下去	*	'continuous'	*	'continuous'	*	'continuous'	*
V看 or V一V	*	'delimitative'	*	*	*	*	*

ST: Situation types, AM: Aspect markers

The perfective marker '了' can both occur in an accomplishment and an achievement situation. It can mean that the action is bounded or the terminal point of the action is achieved. The two continuous markers '(正)在' and '著' behave differently. Both '(正)在' and '著' can occur in an activity situation and means 'progresssive.' '在' but not '著' is used in a habitual situation to denote habituals.

Having analyzed the twelve textbooks of Mandarin Chinese for Elementary Schools, we concluded that two categories of verbs often occur with the continuous marker '著' in a subordinate clause, which function as a background for the figure indicated by its main clause.

Spacial postural verbs:坐著 'sitting,' 躺著 'lying'

State verbs that are used causatively:仰著臉 'facing up,' 彎著腰 'curve waiste,'

低著頭 'facing down,' 壯著膽 'with guts,' 光著腳 'barefoot,' 紅著臉 'blushingly,'

歪著頭 'with oblique head,' 狠著心腸 'hardhearted'

Also, some verbs, though subcategorize for verb phrases or sentential complements, must contain aspect markers or the sentences will be unacceptable:

急著大喊 'cry anxiously,' 忙著除草 'busy in mowing,' 搶著提水 'rush for lifting water,' 爭著長大 'fight for growing up,' 忘了告訴他 'forget to tell him,' 忘記了他會去 'forget that he will come'

However, in the following types of sentences, we found that aspect markers never occur.

1.Verbs that subcategorize for verbal phrases occur. For example:

(6)* 我希望了去他那裡.
I hoped to go to his place.

2. Verbs that subcategorize for sentential complements occur. For example:

(7)* 我知道了他要去。
I knew that he would come.

3. Verbs that subcategorize for pivotal sentences occur. For example:

(8)* 他批評了這部電影不好。
He criticized that this movie is not good.

4. Verbs that subcategorize for a goal or locative phrase occur. For example:

(9) * 我看見他躺著在床上看書。
I saw him lie on the bed and read.

5. Sentences that contain resultant or extent complements occur. For example:

(10)* 這間房子佈置了得很漂亮.
This house was beautifully decorated.

6. Cleft sentences or causative sentences that occur. For example:

(11)* 他使了大家很開心。
He made everybody happy.

Also, in Chinese the first event in a sequence of successive events are often marked by the perfective marker. For exmaple:

(12) 他買了票進去.
He bought a ticket and came in.

III. ASPECTUAL SYSTEM OF JAPANESE

The notion of aspect is related to seven auxiliary verbs in Japanese. They are 'teiru', 'teshimau', 'teoku', 'tekuru', 'teiku', 'temiru' and 'tearu.'[5] In general, 'teshimau' marks perfectivity whereas 'teiru,' 'tekuru,' 'teiku,' 'temiru,' and 'tearu' mark imperfectivity. While 'teoku' can mark perfectivity and imperfectivity. Verbs in Japanese can be divided into state , action, and change verbs. Change verbs can be further distinguished into three types: process, telic and resultant verbs. All verbs except state verbs can be marked by the aspect marker 'teiru.' The three types of verbs are exemplified as follows:[6]

i. State verbs: ある、いる、できる

ii. Action verbs: 読む、食べる、話す、歩く

iii. Change verbs

a. Process verbs: 開く、沈む

b. Telic verbs: 知る、死ぬ

c. Resultant verbs: 愛する、似る、優る

The situation types and the interpretations of Japanese aspect markers are listed below:

Table 2: The situation types and the interpretations of Japanese aspect markers

ST	Accomplishment	Activity	Achievement	Process	State	Habitual	Generic
Teiru	*	'progressive'	*	'progressive' 'repetitive'	'experiential' 'perfective' 'nonprogressive'	'habitual'	*
Teshimau	'telic'	*	'telic'	*	*	*	*
Teoku	*	*	*	*	'perfective' 'nonprogressive'	*	*
Tekuru	*	'continuous'	*	'inceptive' 'continuous'	*	*	*
Teiku	*	'continuous'	*	'continuous'	*	'continuous'	*
Temiru	*	'delimitative'	*	*	*	*	*
Tearu	*	*	*	*	'nonprogressive'	*	*

ST: Situation types

AM: Aspect markers

The interpretations of the seven aspect markers and the Chinese translation are discussed as follows:

1. Teiru

In general, a verb in the '-te' form plus 'iru' refers either (1) to an action that goes on over a period of time, or (2) to the state resulting from an action.

a. teiru--> (Activity, progressive)

When 'teiru' occurs with an action verb, it can mean 'progressive' of an activity. Adverbials that denote a period of time such as '今' 'now' often occur in an activity situation. For example:

(13)今、英語の会話を習っています。
現在正在学英語會話

I am learning English conversation now.

(14)みんなが外で遊んでいます。
大家正在外面玩
All are playing outside.

b. teiru--> (Process, progressive)

'Teiru' can mean 'progressive' of a process when it occurs with a process verb. In such a case, the translation '在' is better than '正在.' For example:

(15)雨が降って川の水がどんどん増えている。
因爲下雨所以河川的水不斷的在增加
The water in the river is continuously increased due to the rain.

c. teiru-->(Process, repetitive)

'Teiru' can mean 'repetitive' of a process when it occurs with a telic verb. The subject of a sentence that describes such a case is usually plural. For example:

(16)多くのガラスは破れている。
很多玻璃在破
A lot of glass is breaking.

d. teiru--> (State, experiential)

'Teiru' can also mean 'experiential' when it occurs with an action verb. Frequency adverbs such as '二三度' 'two or three times' often occur in the states that expresses experience. In Chinese, 'experiential' is often expressed by an action verb plus '過', which is recognized as an accomplishment situation.[5]

(17)あの人はたくさんの小説を書いている。
那個人寫過很多小説
That man has written many novels.

(18)私は若い時に二三度この小説を読んでいた。
我年輕時看過這本小説二三次
I had read this novel two or three times when I was young.

e. teiru-->(State, perfective)

When 'teiru' occurs with a telic, a resultant, or an action verb (as in (22) and (23)), it means the resultant state. To express a state, '了' should be used if a nonstative verb is chosen.

(19)隣は空地になっている。

隔壁變成了空地
The neighborhood has became an open area.

(20)ところどころに、大木がたおれている。
到處都有樹木倒了
Trees have fallen down everywhere.

(21)昨日から桜の花が咲いている。
櫻花從昨天開了
Cherry blossoms have been in blossom since yesterday.

(22)彼はすでにその本を読んでいる。
他已經讀了那本書
He has read that book.

(23)彼はあの先生に三年教わっている。
他跟那位老師學了三年
He has learned from the teacher for three years.

In the case of action verbs, in order to distinguish the sentential aspect '(state, perfective)' from '(activity, progressive),' the definiteness of the subject and the object, and adverbials must be taken into consideration.

f. teiru-->(State, nonprogressive)

When 'teiru' occurs with a resultant verb (as in (24) and (25)) or an action verb (as in (26) and (27)), it can mean the 'nonprogressive' of a state. Most of Chinese state verbs are not marked for 'nonprogressive.' However, verbs such as '穿' 'wear' and '戴' 'wear,' which can denote both an action or a state, are marked by the continuous marker '著' in a state situation.

(24)私はあなたを愛しています。
我愛你
I love you.

(25)この道は曲っている。
這條道路彎彎曲曲
The road curves.

(26)陳さんは眼鏡を掛けている。
陳先生戴著眼鏡
Mr. Chen has glasses on.

(27)王さんは背広を着ている。
王先生穿著西裝
Mr. Wang is in a suit.

In the case of (25) and (26), both interpretations, '(state, nonprogressive) and '(activity, progressive),' are possible.

g. teiru-->(Habitual, habitual)

'Teiru' can also occur in a habitual situation, which means that the situation lasts for a period of time. Adverbs such as '毎朝' 'every morning' and '最近' 'recently' usually occur in habitual situations.

(28)彼は毎朝バイブルを読んでいる。
他每天早上讀聖經
He reads Bible every morning.

(29)この頃は栄養失調で人がどんどん死んでいる。
最近因爲營養失調人不斷的在死
Recently, people are dying due to malnutrition.

2. Teshimau

A verb in the '-te' form plus 'shimau' indicates either (1) that an action has been completed, or (2) that the speaker has regrets about the completion of the action.

a. teshimau-->(Accomplishment, telic)

When occurring with an action verb, 'teshimau' means that the action reaches the endpoint. Therefore, we categorize it as 'telic.' Its best Chinese translation is '完' 'finish.' For example:

(30)牛は昨日刈ってきた草をみんな食べてしまいました。
牛把昨天割來的草全部吃完了
The cow has eaten up all the grass that were mown yesterday.

(31)あの小説は面白いから一晩で読んでしまった。
那本小說很有趣所以一個晚上就看完了
That novel is very interesting so I have read it in a night.

b. teshimau-->(Achievement, telic)

'Teshimau' also means 'telic' when it occurs with a process verb and its Chinese translation should be '了.'[6]

(32)景気が悪いのでたくさんの会社が倒れてしまった。
因爲不景氣許多公司倒閉了
Due to the economic depression, many companies went bankrupt.

(33)あの娘は行ってしまいました。
那個女孩走了
That girl has gone.

3. Teoku

A verb in the '-te' form plus 'oku' refers to doing something for future use or something being left in a certain state.

a. teoku-->(Accomplishment, perfective)

When 'teoku' occurs with an action verb, it can mean that an action is done earlier than the time indicated or the speech act time. It is categorized into an accomplishment situation and can be translated by the complement '好' 'good.' Since Chinese does not have a single grammatical category to mark tense, the adverb '預先' 'in advance,' which indicates the time earlier, can be used. Thus the Chinese translation of this type of sentential aspect is '預先..好' 'in advance...down' if the verb is transitive. Otherwise, '先' 'first' will be given. For example:

(34)試験の準備をしておきます。
預先做好考試的準備
Have the preparation for the examination done in advance.

(35)友達が来るだろうと思ってビールを買っておきました。
我想朋友大概會來所以預先買好啤酒
I guessed my friends might probably come so I had bought beer in advance.

(36)報告の要点を記録しておくことが必要だ。
有必要把報告的要點預先記錄好
It is necessary to have recorded the main points of the report in advance.

Adverbials such as '前もって' 'in advance,' '先に' 'first,' 'ために' 'for the reason that' and 'に' 'for' often occur in such a situation.

b. teoku-->(State, nonprogressive)

When 'teoku' occurs with an action verb, it also means 'nonprogressive.' In such a situation, it emphasizes the state that is caused by the action remains. It is better translated into '著' if Chinese verbs such as '開' 'open' nd '關' 'close,' which can both denote the action and the state, occur. For example:

(37)電灯は消さないで、朝まで付けておこう。
不要關燈一直開到早上吧
Do not turn off the light till the morning.

(38)窓を締めておきなさい。

讓窗户關著吧
Let the door be closed.

(39)相手を羨ましがらせて老いた。
讓對方羨慕
Let your opponent be jealous.

4. Tekuru

A verb in the '-te' form plus 'kuru' indicates either (1) that a change has been taking place up to certain time, or (2) that an action is done in the direction of the speaker.

a. tekuru-->(Activity, progressive)

When 'tekuru' occurs with an action verb, it can mean the action is 'progressive' if a period of time is indicated as is exemplified below:

(40)私はこの工場で三十年間働いてきた。
我三十年間一直在這間工廠工作
I had been working for this factory during the thirty years.

Translation for 'progressive' should be '在' and the adverb '一直' 'always' can be used.

b.tekuru-->(Process, inceptive)

'Tekuru' can also mean inception when it occurs with a process verb or a resultant verb. In such a situation, '起來' or '了' can be translated. [7]

(41) 私は体重が増えてきた。
我的體重增加起來了
I was putting on weight.

(42)温度の上昇につれて砂糖がどんどん融けてきた。
隨著溫度的上昇砂糖不斷的溶化起來了
The sugar started to melt as the temperature went up.

(43)一時間も作業をつづけていると、疲れてきた。
連續工作一小時就疲倦起來了
Having working for one hour without a stop, I started to be tired.

c. tekuru-->(Process, continuous)

When 'tekuru' occurs with a telic or a process verb, it means 'continuous.' It should be translated to '了' if a Chinese telic verb is used, or '一直在' 'always..been' if a process verb is chosen. For example:

(44)その本性がだんだん現われてきた。
它的本性漸漸暴露了
Its nature was revealed.

(45)ちょうどその場にいるような実感さえ湧いてくる。
甚至產生了彷彿在現場的身歷其境感
The feeling to be personally in the situation comes up.

(46)私は上車する前から下車するまで小便を堪えてきた。
我從上車前到下車一直在忍小便
I had been bearing urine from the time that I got up till I got off.

In the case of (46), the process verb is interpreted as 'continuous' since a period of time is used.

d. tekuru-->(Habitual, habitual)

'Tekuru' can mean 'continuous' of a habitual situation when adverbs such as '毎日' 'every day' occur. For example:

(47)毎日、一時間ずつ時間を決めて本を読んできた。
每天規定平均一小時的時間念書下去
Read averagely one hour a day everyday.

5. Teiku

A verb in the '-te' form plus 'iku' refers to a change taking place for a period of time.

a. teiku-->(Activity, continuous)

When 'teiku' occurs with an action verb, it means 'continuous' and can be translated as '...下去.' For example:

(48)本箱の本を片っ端から読んでいく。
把書架的書從這一邊依次讀下去
Read the books on the shelf from this side.

(49)新しい道ができて家がどんどん建てていく。
新路一做好房子就不斷的蓋下去
As soon as the road was done, new houses are built one after another.

b. teiku-->(Process, continuous)

When 'teiku' occurs with a process or a resultant verb, it means 'continuous' of a process. The continuousness of a process is marked by the aspect marker '了' with adverbs such as '快要' 'almost' and '漸漸' 'gradually, if a Chinese telic verb is used. The aspect marker '下去'

can be used, if a Chinese state verb is used. For example:

(50)だんだん英語を忘れていく。
漸漸忘了英文
I gradually forget English.

(51)新しい先生にも慣れていく。
漸漸熟悉了新的老師
I am getting used to the new teacher.

(52)体が毎日に弱っていく。
身體一天一天的弱下去
The body is getting weaker.

c. teiku-->(Habitual, continuous)

'Teiku' can occur within a habitual situation and means 'continuous.'

(53)毎日、一時間ずつ時間を決めて本を読んでいく。
每天規定平均一小時的時間念書下去
Read averagely one hour a day everyday.

6. Temiru-->(Activity, delimitative)

In general, a verb in the '-te' form plus 'muru' means 'to try and see.' 'Temiru' can only occur with an action verb to mean 'delimitative' since volition is involved in 'delimitative.' The Chinese 'delimitative' is expressed by reduplicating a verb as 'VV' or using the pattern 'VV-看.'

(54)日本の食物を食べてみます。
吃吃看日本的食物
Eat some Japanese food.

(55)物差で長さを測ってみて下さい。
請用尺量量看長度
Please use a ruler to measure the length.

7. Tearu-->(State, nonprogressive)

A transitive verb in the '-te' form plus 'aru' refers to a state that has resulted from an action. We represent it as 'nonprogressive.' In Chinese, a state can be expressed by adding the continuous marker '著' or it can be translated into passive sentences, if a verb that can both denote actions and states are chosen. For example:

(56)地図が黒板に貼ってある。
地圖被人貼在黑板上
(黑板上貼著地圖)
The map is pasted on the blackboard.

(57)字が紙に書いてある。
字被人寫在紙上
(紙上寫著字)
Words are written on the paper.

In each first translation of the above examples, the continuous marker '著' is not used since a prepositional phrase indicating location is used.

In summary, the relation between verb classes and aspect markers is presented in the following table:

Table 3: Sentential aspect related to verb types and aspect markers

VC / AM	Action verbs	Change Verbs			State Verbs
		Process	Telic	Resultant	
Teiru	(Activity, pg) (State, exp) (State, nonpg) (State, pf) (Habitual, hb)	(Process, pg)	(State, pf) (Process, rept)	(State, nonpg)	*
Teshimau	(Accmp, tlc)	(Achv, tlc)	*	*	*
Teoku	(Accmp, pf) (State, nonpg)	*	*	*	*
Tekuru	(Activity, pg) (Habitual, cnt)	(Process, cnt) (Process, incpt)	(Process, cnt)	(Process, incpt)	*
Teiku	(Activity, cnt) (Habitual, cnt)	(Process, cnt)	*	(Process, cnt)	*
Temiru	(Activity, dlm)	*	*	*	*
Tearu	(State, nonpg)	*	*	*	*

VC: Verb classes, AM: Aspect markers
Accmp: accomplishment, Achv: achievement, pg: progressive, cnt: continuous
exp: experiential, tlc: telic, dlm: delimitative, incpt: inceptive, rept: repetitive
hb: habitual

Except for the above combinations of situation type and further distinction, other combinations are possible. For example, the sentential aspect of those sentences that contain no aspect markers would be (Accomplishment, experiential) as follows:[8]

e.g.私は二三度日本へ行きました。
我去過日本兩三次
I have gone to Japan two or three times.

IV. COMBINATIONS OF ASPECTUAL MEANING

For a sentence, it is either perfectivity or imperfectivity that is focused. However, there are situations that several aspect markers occur together. Logically, there are two possible combinations of perfectivity and imperfectivity. One is perfectivity in imperfectivity and the other is imperfectivity in perfectivity. In the first case, it is imperfective if the resultant state of a perfective situation is emphasized and it can be marked by an imperfective marker. For example,

(58)私はもうあの本読んでしまっていた。
我已經讀完那本書了
I had already read that book.

In the above example, the first aspect marker 'teshimau' indicates that the situation is an accomplishment and can be interpreted as 'telic.' The second aspect marker 'teiru' indicates that the resultant state of the accomplishment situation is in the focus. Thus, it can be further marked by the imperfective marker 'teiru.' Thus we categorize the sentential aspect as '(State, telic).'

On the other hand, the other case that imperfectivity in perfectivity is impossible because it is only perfectivity or imperfectivity that can be focused in a sentence. Though an activity or a process can start and end before the reference time, yet it can not be marked by a perfective marker. Nevertheless, it can be expressed by simple past, past perfect or present perfect instead of perfective.

V. MECHANISM FOR THE PROCESSING OF SENTENTIAL ASPECT AND THE GENERATION OF ASPECT MARKERS

The treatement of aspect is shown in the following figure:

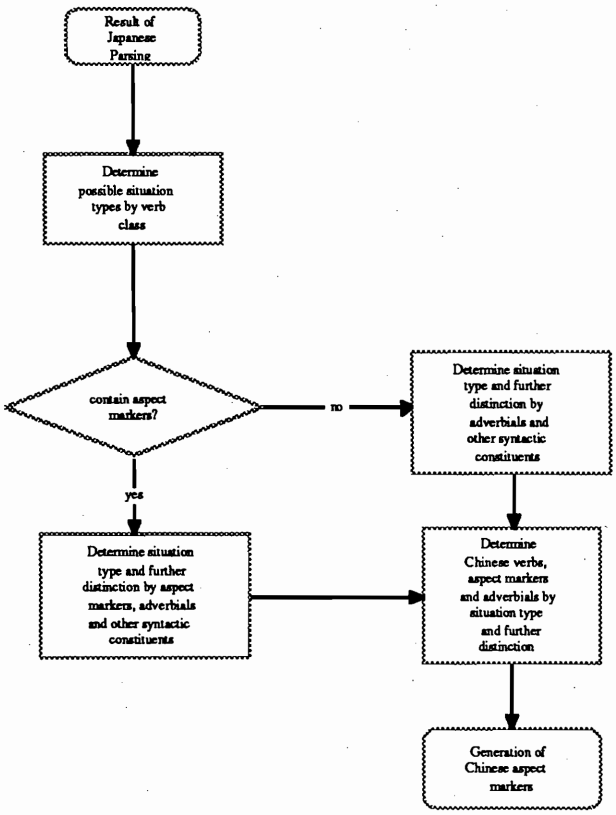


Figure 5: The flow of the treatment of aspect

Let us use sentence (22) as an example here. First, the class of a main verb is checked and possible situation types are determined:

Verb class=action
Possible situation types=
{(Accomplishment, x), (Activity, x), (State, x), (Habitual, x), (Generic, x)}

Following the check of verb class, aspect marker(s) are checked. When 'teiru' occurs with an action verb, the possible sentential aspects are as follows:

Aspect marker-->'teiru'
Possible sentential aspects:
{(Activity, progressive), (State, experiential), (state, nonprogressive), (state, perfective), (Habitual, habitual)}

Except aspect markers, the situation type can also be determined by the following factors:

i. Adverbials

Adverbials are important for determining situation types. For example, Japanese time adverbs such as '毎日' 'every day' denote habitual situations.

ii. Tense

Since accomplishments and achievements may only occur in simple past tense and may not be marked by any aspect markers, tense must be taken into consideration in the algorithm. Sentences in simple past tense may denote accomplishments and achievements. Also, a generic situation is described in present tense.

iii. Definite vs. indefinite subject and object

Definite subjects and objects are usually used in an accomplishment situation. Thus definiteness of subject and object is useful in distinguishing an accomplishment situation from an activity one.

iv. Number of the subject and the object

The number of the subject and object in a sentence is related to situation types. For example, generic sentences often occur with plural subjects and objects.

v. Quantified subject and object

An accomplishment situation accompanies with quantified objects, or quantified subjects that are more than one.

vi. Complements

Complements implies aspectual meaning, which must be considered in the overall algorithm.

In sentence (22), due to the adverb is ' すでに' 'already' and the object is definite, we can derive the sentential aspect as '(State, perfective).'

Adverbials: 'すでに' 'already'-->'perfective'
Object: definite-->'probably perfective'

In this case, the adverb 'already' is featured by 'perfective' and the definite object is characteristic of 'probably perfective.' Thus it can be interpreted as '(State, perfective),' which should be translated by adding the perfective marker '了' and it results in the translation '他已經讀了那本書' 'I have read that book already.' After all the constituents are checked, if there is still ambiguity in situation types or further distinctions, all of them will be included as the result of parsing. Nevertheless, only a default one will be translated into Chinese.

Finally, the Chinese translation is based on the situation type and the interpretation of the aspect marker. In this case, as long as the action verb '讀' 'read' is translated the perfective marker '了' can be given.

VI. CONCLUDING REMARKS

Chinese is different from Japanese and English in that it does not have a grammatical category to mark tense but has perfectivity and imperfectivity markers. In the machine translation systems that aim Chinese as a target language, the treatment of aspect must be carefully dealt with or the translation will be quite unacceptable. With this model, sentences that contain one aspect marker, several aspect markers or no aspect markers can be considered by the same approach. Moreover, it can also be applied to the translation of other languages with the only requirement that the aspectual systems are investigated language by language.

NOTES

1. In this article, we do not distinguish predicative adjectives from verbs. When we refer to verbs, we mean predicative adjectives and verbs both in Japanese and Chinese.
2. We add 'telic' and 'delimitative' to Comrie's subdivion of perfectivity and imperfectivity

respectively.

3. In the examples and the followings, two values indicated in the parentheses mean situation type and further distinction of aspect respectively.

4. Aspect markers 'tekuru' and 'teiku', which are derived from 'come' and 'go', are very similar to the Chinese aspect markers '起來' and '下去', which are derived from 'up' and 'down.'

5.'Experience' can be represented as an accomplishment or a state. In Japanese, an accomplishment that expresses experience is as follows:

e.g.私は二三度日本へ行きました。
我去過日本兩三次
I have gone to Japan two or three times.

A state that expresses experience is as follows:

e.g.私は二三度日本へ行っている。
我去過日本兩三次
I have been to Japan two or three times.

6. 'Telic' of achievement situations can be translated into Chinese by adding complements such as '掉' 'fall' and '下去' 'down,' which can be quite different depending on the meaing of verbs. The generation of these complements is possible if only if they are recorded in the dictionary.

7. In Chinese, there is a perfect marker 了', which occurs in the final position of sentence, indicates that the situation happens earlier than or at the reference time or the speech act time. In (41) to (43), the perfect marker '了' is used.

8. For the easy treatment of aspect, all the combinations of situation type and further distinction are stored in the computer.

REFERENCES

[1] C. Smith. "A Speaker-Based Approach to Aspect." In *Linguistics and Philosophy*. Vol.9-No.1-February, 1986.

[2] B. Comrie. *Aspect: An Introduction to the Study of Verbal Aspect and Related Problems.* Cambridge University Press, 1976.

[3]野垣内出、鈴木雅実、榊 博史 動詞意味素性の付加による日本語文アスペクトの解析および翻訳について 自然言語処理 48-4, 1985.

[4]賀来直子 日本語のアスペクト 情報処理学会第31回（昭和60年後期）全国大会, 1985.

[5]金田一春彦 日本語動詞のアスペクト むぎ書房, 1976.

[6] 中園 薫、古瀬 蔵、野村浩郷 アスペクト情報の処理について 自然言語処理 61-2, 1987.

AN ERS MODEL FOR TENSE AND ASPECT INFROMATION IN CHINESE SENTENCES

Hsi–Jian Lee* and Ren–Rong Hsu

Department of Computer Science and Information Engineering

National Chiao Tung University

Hsinchu, Taiwan, 30050

R. O. C.

ABSTRACT: In this paper, we address a ERS model of time and aspect in Chinese sentences. We first discuss the possible tense forms in simple sentences when translating from Chinese into English. Next, we also address the possible combinations of the tense forms of complex sentences joined by linking elements and some complex sentences containing one subclause. Finally, we provide some conventional rules in English to modify the tense forms before generating the output English sentences.

*To whom all correspondences should be sent.

1. INTRODUCTION

Both tense and aspect in sentences play important roles in transmitting temporal and situational information between human beings. For a machine translation system, the analysis of tense and aspect has great influence on the quality of the output sentence. Lacking the analysis of tense and aspect may result in syntactic or semantic errors. Take the following Chinese sentence as an example.

(1) ta1 zuo2-tian1 mai3 le5 yi4 ben3 shu1

The corresponding sentence in English should be

(1a) He bought a book yesterday.

The translation such as

(1b) He buys a book yesterday.

would be illegal. Without the analysis of tense and aspect, a human being can hardly translate sentences correctly, neither can the machine.

The tense can be viewed as determined by the relationship between the time of the action or the state denoted by the verb and the time of speech. Thus, the tense information involves at least three components: the event time, the speech time and the degree of understanding of the subject between the hearer and the speaker. On the other hand, aspect does not refer to the time relation between the event time and speech time, but to how the situation itself being viewed from its internal structure[1]. All the information of tense and aspect reflecting in the *verb sequence* and *tense form* will be defined in the following section.

In order to transfer tense and aspect information of Chinese sentences into those of English sentences, we have to find the relationships between these two languages, which has been done in [2] and will be summarized in Section 2. In this paper, we will address the problem about the possible tense forms in a sentence. Still, the sentences we will analyze are confined to simple declarative sentences, which contain only one predicate. We will also analyze some more complex sentences joined by linking elements such as "de5 shi2–hou4"(when), "yi3–hou4"(after), "yi3–qian2"(before) and some sentences containing one subclause. We also provide some conventional rules to modify the tense forms before generating English sentences.

2. FORMALISM OF TIME AND ASPECT

In an English sentence, the verb sequence is one of the most important constituents, which are generally affected by the tense and aspect. Winograd [3] defined the verb sequence as a sequence of verbs and auxiliaries following a fixed order, as illustrated below:

(Modal) (Have) (Be1) (Be2) Main Verb

where *Modal* has either the tense meaning or modal meaning; *Have* represents the auxiliary has (or have or had); *Be1* represents the verb *be*; *Be2* denotes the clause being passive if present and *Main verb* is self–explained. Each combination of *Modal*, *Have*, *Be1*, *Be2* and *Main verb* is also called a tense form. For example, the sentence

(2) He		has			finished	his job.
	Model	*Have*	*Be1*	*Be2*	*Main verb*	

has the option *Have*, but has no other options (*Modal, Be1, Be2*). Indeed, it has the tense form of "present–perfect".

Many linguists take only present and past as tense, future is considered as the modal. This kind of classification is not intuitive since human beings always partition the time axis into three parts: past time, present time and future time. Winograd [3] proposed that the first element of the tense sequence is the tense of the first element in the verb sequence unless it is a modal. The rest, correspond to the entire auxiliary sequence, excluding the main verb, are assigned *tense features* according to the following mapping:

element in a verb sequence	tense feature
Modal	Future if the word is shall or will

	Modal	otherwise
Have	Past	
Be1	Present	

Consider the following sentence,

(3) By the time he comes home, I will have finished my job.

The tense sequence for the clause "I will have finished my job." is (Future, Past) since "will" gives the feature Future and "have" gives the feature Past. Now, we summarize the tense sequences (excluding the feature Modal) of possible verb sequences in Table 1.

Table 1

Tense sequences for the sequences

tense sequence	examples
Pres	(4) I *read* every day.
Pres Pres	(5) I *am* reading now.
Pres Past	(6) I *have* lived here for six years.
Pres Past Pres	(7) It *has been* raining on and off since this morning.
Past	(8) I *wrote* a letter yesterday.
Past Pres	(9) I *was* reading when he came in.
Past Past	(10) When he came in, I *had* finished my job.
Past Past Pres	(11) I *had been* reading a novel till he came to see me.
Past Fut	(12) She *was going to* give me her address.
Past Fut Past	(13) He once said he *was going to have* fixed all radios in town by the time he was forty.
Fut	(14) I *will* go to Taipei tomorrow.
Fut Pres	(15) I *will be* reading when you come back.
Fut Past	(16) When you come back, I *will have* finished my job.
Fut Past Pres	(17) I *will have been* learning English for five years by next month.

A problem in the above formalism appears when the sentence involves the progressive or perfective aspect. For example, it gives sentences (8) and (9) the tense features (Past) and (Past, Pres) respectively to capture their differences by time. However, both events in these two sentences occurred during the past time. What makes them different is not the time but the aspect. Thus we would like to give both sentences (8) and (9) the same time denotation and use other feature to capture their difference.

As mentioned above, the tense sequence is got by tracing the elements in the verb sequence in order. Unfortunately we can not apply it in Chinese sentences since there is no tense marker in Chinese. Thus, we have to use other information in a sentence. Here, the information we use is called *temporal information*. To represent this information, we partition the time axis into three parts which are *past time*, *present time* and *future time*. We can also divide each partition recursively. Therefore, the temporal information can be used to denote the relative relationship in the time axis. The representation for this relative time is called "*time marker*".

There are two aspects in English - *perfective* and *progressive*. Since these two aspects greatly influence on the tense form of English sentences, we choose aspect as the other attribute to represent the information for the tense form. Thus perfective or progressive are the two aspect marker.

Based on the time and aspect markers mentioned above, we distinguish the tense form of English by the values *present*, *past*, *future*, *perfective* and *progressive*. The tense forms of English sentences and their attribute values are listed in Table 2. Taken the tense form "past-perfect" in item 7 as an example, its time marker past-past denotes the event specified by the verb sequence occurred in the past, with respect to the past reference time. Consider the following sentence

(18) ta1 hui2-jia1 de5 shi2hou4, Zhang1san1 yi3jing1 zuo4-wan2 ta1 de5 gong1zuo4 le5

(When he came home, Zhangsan had finished his job.)

The pair of time and aspect markers for the verbs "hui2–jia1" (go home) and "zuo4–wan2" (finish) should be {past – } and {past–past perfective } respectively. The method to determine these markers has been discussed in [2].

Table 2

Table of Tense Forms

tense form	time marker	aspect marker
1. present		present
2. present–progressive	present	progressive
3. present–perfect	past	perfective
4. present–perfect–progressive	past	perfective–progressive
5. past		past
6. past–progressive	past	progressive
7. past–perfect	past–past	perfective
8. past–perfect–progressive	past–past	perfective–progressive
9. past–future		past–future
10. past–future–perfect	past–future–past	perfective
11. future		future
12. future–progressive	future	progressive
13. future–perfect	future–past	perfective
14. future–perfect–progressive	future–past	perfective–progressive

3. POSSIBLE TENSE FORMS IN SENTENCES

We have provided a formalism of time and aspect to determine the tense form of the sentence. That is, if we can find the time and aspect markers of a verb, we can find the tense form through Table 2[2]. However, there is another approach to determine the tense form of an English sentence which will be translated from the Chinese sentence. This approach is based on the relations among speech time (S), reference time (R) and event time (E) proposed by Han Reichenbach in 1947 [4]. In this section, we will discuss the possible tense form in simple sentences and those in sentences containing two events.

3.1 Formalism of E, R, S and Aspect

To analyze the tense system of English, Han Reichenbach found that the analysis based on the absolute event time or the relative time relation between event and speech is not enough. Thus, he proposed the speech time, reference time and event time to account for the underlying structure of the English tense system. The speech time and event time are self-evident, while the reference time is the "temporal perspective" from which the described event is viewed[5]. There are six relations among E, R and S in simple sentences, which correspond to different tense forms, as Table 3 shows.

Table 3
The Mapping Between (E, R, S) and The Tense Form

relation of E,R,S	tense form
E = R = S	present
E < R = S	present-perfect
E = R < S	past
E < R < S	past-perfect
S< E = R	future
S< E < R	future-perfect

From this table, we find that the relationships among E, R, S are not enough to represent all various tense forms in English. Take the following sentences (19)-(20) as examples. The relations among E, R, S are all E = R = S.

(19) He usually plays the basketball.

(20) He is playing the basketball.

In fact, the role of the relations among E, R, S is the same to the time marker proposed in our time and aspect formalism. They all denote the time relation. The difference is that former is explicit, while the latter is implicit in the denotation of the reference time of an event. Thus, to distinguish the various tense forms in English, we still need the aspect marker to denote the progressive attribute. We summarize this result in Table 4.

Table 4

The Mapping Between (E, R, S), Aspect Marker and The Tense Form

E, R, S relation	aspect marker	tense form
E = R = S	none	present
E = R = S	progressive	present–progressive
E < R = S	none	present–perfect
E < R = S	progressive	present–perfect–progressive
E = R < S	none	past
E = R < S	progressive	past–progressive
E < R < S	none	past–perfect
E < R < S	progressive	past–perfect–progressive
S < E = R	none	future
S < E = R	progressive	future–progressive
S < E < R	none	future–perfect
S < E < R	progressive	future–perfect–progressive

Now, we can make use of Table 4 to determine the tense form of an English sentence if we can find the E, R, S relation and the aspect marker from the Chinese sentence. As an example, see sentence (21). In sentence (21), since the relation among E, R, S is E=R<S and the aspect marker is progressive, we have the "past–progressive" as the tense form of the translated English sentence.

(21) zuo2–tian1 ci3–shi3 ta1 zai4 kan4–shu1.

(He was reading this time yesterday.)

We have discussed the tense forms of English by E, R, S and aspect marker. However, not all tense form can be used in a simple sentence. In a sentence containing more than one event, the use of tense forms is further restricted. In the following sections, we will discuss the possible tense form in a simple sentence and those in a sentence containing two evens.

3.2 Possible Tense Forms in Simple Sentences

In Table 2, we have listed fourteen possible tense forms of English sentences. However, some of them can not be used in simple sentences. To discuss the possible tense forms in sentences, we first define some terms used below. The first one is the time category. Since time phrases contain the message of time, we partition the time phrases into four categories: Past, Present, Future, Nondeterministic[2], according to the time these phrases represented. The second term is the time precedence relations " < ", " = ", " > ", where " < " represents "happen before"; " > " represents "happen after" and " = " represents "happen simultaneously", respectively. In this section, we will discuss and give an example for each of them according to the E, R, S formalism mentioned in the last section.

The "present", "present-progressive" "present-perfect" and "present-perfect-progressive" tense forms are all current relevant (i.e, R = S), thus they do not need time phrases to specify their reference time. Sentences (22)–(25) show this case. The events in "past", "past-progressive", "past-perfect" and "past-perfect-progressive" tense forms are relevant to some past time, they need some phrases in Past time category to specify their reference time. Sentences (26)–(29) show such case. The word "qu4–nian2 ci3–shi2" (this time last year) in sentence (28) specifies the reference time for the event "zuo4–wan2 gong1–zuo4" (finish the work) and the words "zao3", "yi3" (already) specifies that the event happened before the reference time, thus the event "zuo4–wan2 gong1–zuo4" (finish the work) happened in the past–past time with respect to the speech time (i.e, E < R < S). The sentences of "future", "future-progressive", "future-perfect" and "future-perfect-progressive" tense forms usually have some phrases in Future time category to specify their reference time. The Sentences (30)–(33) denote this condition. The other two tense forms "past-future" and "past-future-perfect" are seldom used in a simple sentence. The reason is that these two tense forms are not current relevant and there does not exist any time phrase that can both specify the past meaning and the future meaning (relevant to the past time).

From above discussion, we show that we can use twelve tense forms in simple sentences, some of which need time phrases to modify the events in sentences. In the following section, we will discuss the tense forms in complex sentences.

(22) wo3 xi3-huan1 hua1 (I like flowers.)

(23) ta1 zai4 kan4 shu1 (He is reading now.)

(24) ta1 yi3-jian1 zuo4-wan2 gong1-zuo4 le5 (He has finished his job.)

(25) ta1 yi3-jing1 zai4 du2-shu1 le5 (He has been reading now.)

(26) *zuo2-tian1* ta1 qu4 le5 Tai2-bei3 (He went to Taipei yesterday.)

(27) *zuo2-tian1 ci3-shi2* ta1 zai4 kan4-shu1 (He was reading this time yesterday.)

(28) *qu4-nian2 ci3-shi2* wo3 zao3 yi3 zuo4-wan2 gong1-zuo4 le5 (This time last year, I had finished my job.)

(29) *qu4-nian2 ci3-shi2* wo3 zao3 yi3 zai4 kan4-shu1 (This time last year, I had been reading.)

(30) *ming2-tian1* ta1 jiang1 qu4 Tai2-bei3 (He will go to Taipei tomorrow.)

(31) *ming2-tian2 ci3 shi2* wo3 jiang1 zai4 kan4-shu1 (I will be reading this time tomorrow.)

(32) *ming2-nian2 ci3-shi3* ta1 jiang1 yi3 bi4-yeh4 (He will have graduated this time next year.)

(33) *ming2-nian2 ci3-shi2* ta1 jiang1 yi3-jing1 zai4 zhun3-bei4 kao3-shi4 le5 (He will have been preparing the examination this time next year.)

3.3 Possible Tense Forms in Sentences with Two Events

As we have analyzed in the last section, we can use one of the twelve tense forms to represent a simple sentence. When there are two events in a sentence, the tense forms of these events can not be any combination of the twelve tense forms, however. There may have some new tense forms such as "past-future" and "past-future-perfect" which can not be used in the simple sentence. There may also have some illegal combinations such as {past, present–progressive} as in the sentence "When I came in, he is eating his lunch.". The requirement of correct combination for tense forms in a sentence is called *tense agreement.*

Some people define the tense agreement as that the tense forms in a sentence should be the same, as the sentence (34) shows. The sentence (34) is correct because the tense form of the verbs "knew" and "read" are the same while (35) is illegal because the tense forms of the verbs "knew" and "reads" are different. Some might think that the tense form of the subordinate clause should follows that of the main clause (such as the sentence (34) shows), or that the tense form of the main clause should follows that of the subordinate clause (such as the sentence (36) shows). However, these still have deficiencies[6]. They can not explain the correct sentence shown as sentence (37).

(34) I knew that he read a book.

(35) *I knew that he reads a book.

(36) The boy I talked to had blue eyes.

(37) The boy I grew up with is the president of America.

Lin & Chen[7] made some notes about the tense agreement in English. They claimed that in a complex sentence, if the main clause has the tense form of " present", "future", or "present–perfect", the tense form for the subordinate clause can be any of the twelve tense forms mentioned above. When the tense form of the main clause is "past", change the tense forms of the subordinate clause to the past form. Thus, the tense forms of the two clauses in a sentence may be any one in Table 5, ignoring the progressive aspect in the main clause.

Table 5

Possible Tense Forms for Sentences Containing Two Clauses

tense form of main clause	tense of subordinate clause

present, present-perfect future, or future-perfect	present, present-progressive present-perfect present-perfect-progressive past, past-progressive, past-perfect, past-perfect-progressive future, future-progressive future-perfect future-perfect-progressive
past, past-perfect	past, past-progressive, past-perfect, past-perfect-progressive past-future, past-future-perfect

Now, we consider the situations in Chinese. Taking sentence (38) as an example, the main clause has "past" as its tense form while its subordinate clause has the "past-future" tense form.

(38) ta1 zuo2-tian1 da1-ying4 wo3 lai2 zhe4-li3.

(He promised me yesterday that he would come here.)

(He allowed me to come yesterday.)

To find out the possible combinations of the tense forms for the two clauses in a sentence, we divide our domain into six parts, i.e., (a) nondeterministic, (b) E2 > E1, (c) E2 < E1 in serial verb construction sentences and (d) E1 "de5 shi2-hou4" (when) E2", (e) E1 "yi3-qian2" (before) E2, and E1 "yi3-hou4" (after) E2[2].

(a). nondeterministic for verbs in serial verb constructions

The possible tense forms of the events in sentences of serial verb constructions will be the same to those in Table 5. The reason is that the two events in the sentence have no predetermined relation. Thus, any possible relation is permitted to exist between them. For example, the combinations {present, present-perfect}, {future, present} are all legal as sentences (39)-(40) show.

(39) wo3 zhi1-dao4 ta1 yi3-ying1 qu4 ri4-ben3 le5.

(I know that he has gone to Japan.)

(40) ming3-tian1 ta1 jiang1 zhi1-dao4 Zhang1san1 xi3-huan1 A1-Hua1.

(Tomorrow he will know that Zhangsan likes A-Hua.)

(b). E2 > E1

The possible tense forms of the clauses in sentences of such kind are more restricted since the relation between the two events is specified. We list the possible combinations of tense forms for them in Table 6 and use the sentence (41) to illustrate it. In sentence (41), the tense forms for the verbs "da1-ying4" (promise) and "lai2" (come) are "past-perfect" and "past-future" respectively.

(41) ta1 zao3 yi3 da1-ying4 wo3 lai2 zhe4-li3.

(He had promised me that he would come here.)

(He had allowed me to come here.)

Table 6

Possible Tense Forms for E1 and E2 in E2 > E1

tense form of E1	tense form of E2
present	future
present-progressive	future
present-perfect	future
present-perfect-progressive	future
past	past-future
past-progressive	past-future
past-perfect	past-future
past-perfect-progressive	past-future
future	future
future-progressive	future
future-perfect	future
future-perfect-progressive	future

(c) E2 < E1

The possible tense forms of the events in sentences of such kind are also restricted as those of E2 > E1. We list the possible combinations of tense forms for them in Table 7 and illustrate it by the sentence (42). In sentence (42), the tense forms for the predicates "pi1-ping2" (criticize), "lan3-duo4" (lazy) are "past" and "past-perfect" respectively.

(42) ta1 pi1-ping2 wo3 tai4 lan3-duo4.

(He criticized that I had been too lazy.)

Table 7

Possible Tense Forms for E1 and E2 in E2 < E1

tense form of E1	tense form of E2
present	past
present-progressive	past
present-perfect	past
present--perfect-progressive	past
past	past-perfect
past-progressive	past-perfect
past-perfect	past-perfect
past-perfect-progressive	past-perfect
future	past
future-progressive	past
future-perfect	past
future-perfect-progressive	past

The roles of events E1 and E2 in serial verb construction sentences are different from those of E1' and E2' in sentences linked by linking element. In fact, the clause containing the event E1 is the main clause and the clause containing the event E2 is the subordinate clause, while the clause containing the event E1' is the main clause and the clause containing the event E2' is the subordinate clause. As we have mentioned in Sec. 3.2, there are twelve possible tense forms for the main clause of the sentence and the tense form of the subordinate clause may be determined by the time relation between E1 and E2. However, any time phrase in the subordinate clause may change the reference time for E2. Thus, we can hardly use the event time of E1 as the reference time of E2. For example, the reference time of verb "lai2" (come) in sentence (43) is not the event time of "da3–suan4" (propose) but some time in tomorrow.

(43) ta1 da3–suan4 ming2–tian1 lai2.

(He proposes to come tomorrow.)

On the contrary, in sentences linked by "de5 shi2–hou4" (when), "yi3–hou4" (after), "yi3–qian2" (before), the possible tense form of the event E2' can be usually determined by the relations of E, R, S. This is because in sentences of such cases, the event time of E1' can be used as the reference time for the event E2'. In the following paragraph, we denote the event times of E1' and E2' as E_1 and E_2, respectively. We also denote the speech time as S and the reference time for the event E2 as R_2. We analyze them as follows:

(d).E1 de5 shi2–hou4 E2

(i) If the tense form of the event E1' is "past" or "past–progressive", the event time of E1 (i.e., E_1 or say, R_2) is past. Thus, we have $R_2 < S$. From Table 4, we have four pairs of { E, R, S relation, aspect} : { $E_2 = R_2 < S$, none }, { $E_2 = R_2 < S$, progressive }, { $E_2 < R_2 < S$, none }, { $E_2 = R_2 < S$, progressive } with the corresponding tense forms "past", "past–progressive", "past–perfect", and "past–perfect–progressive" for E2'.

(ii) If the tense form of the event E1' is "future" or "future–progressive", the event time of E1' is future. Thus, we have $S < R_2$. From Table 4, we have four pairs of {E, R, S relation, aspect marker} : { $S < E_2 = R_2$, none }, { $S < E_2 = R_2$, progressive }, { $S < E_2 < R_2$, none }, { $S < E_2 < R_2$, progressive }. From these pairs, we can get the possible tense forms : "future", "future–progressive", "future–perfect" and "future–perfect–progressive" for E2'.

(iii) If the tense form of the event E1' is "present" or "present-progressive", the event time is viewed as the same to S (i.e., $E_1 = S$). That is, $R_2 = S$. Thus, we have the pairs of {E, R, S relation, aspect}: { $E_2 = R_2 = S$, none }, { $E_1 = R_2 = S$, progressive} and E2' may possess the tense forms: "present", "present-progressive".

We summarize the results in Table 8 and give three sentences (44)–(46) with the tense forms as examples. One might find that some of the tense forms in the English sentences has been changed. For example, the tense form of the verb "lai2" (come) in sentence (45) is "present" while its original tense form is "future". This phenomena is due to the convention of English. We will discuss them in the next section.

(44) zuo2-tian1 ta1 lai2 de5 shi2-hou4, wo3 zai4 kan4-shu1.

(When he came yesterday, I was reading.) {E1':past, E2':past-progressive}

(45) ta1 lai2 de5 shi2-hou4, wo3 jiang1 kai1 men2.

(When he comes, I will open the door.) {E1':future, E2':future}

(46) mei3 ci4 wo3 zai4 kan4 shu1 de5 shi2-hou4, ta1 dou1 zai4 shui4-jiao4.

(Every time when I'm reading, he is sleeping.){E1':present-progressive, E2':present-progressive}

Table 8

Possible Tense Forms for E1' and E2' in E1' de5 shi2-hou4, E2'

tense form of E1	tense form of E2
past	past, past-progressive, past-perfect, past-perfect-progressive
past-progressive	past, past-progressive, past-perfect, past-perfect-progressive
future	future, future progressive, future-perfect, future-perfect-progressive
future-progressive	future, future progressive, future-perfect, future-perfect-progressive
present	present, present-progressive

present-progressive	present, present-progressive

(e). E1' yi3-qian2 (before) E2'

In this case, the relation between E1' and E2' is specified. This relation restricts the possible combinations of the tense forms for E1' and E2'.

(i)If the event time of E1' is past (i.e. $E_1 = R_2 < S$), we have the relation $E_2 < R_2 < S$. Thus, the tense forms for E2' may be "past-perfect" or "past-perfect-progressive", as the sentence (47) shows.

(47) wo3 qu4 Tai2-bei3 yi3-qian2 mai3 le5 yi4 ben3 shu1.

(I had bought a book before I went to Taipei.)

(ii) If the event time of E1' is future (i.e., $S < E_1 = R_2$), we have the relation $S < E_2 < R_2$. Thus, the tense forms for E2' may be "future-perfect" or "future-perfect-progressive". Sentence (48) illustrates such a example.

(48) wo3 qu4 Tai2-bei3 yi3-qian2 jiang1 yi3 zuo4-wan2 gong1-zuo4 le5.

(I will have finished my job before I come to Taipei.)

We also summarize the possible combinations of tense forms in Table 9.

Table 9
Possible Tense Forms for E1 and E2 in E1 yi3-qian2 E2

tense form of E1	tense form of E2
past	past-perfect
past	past-perfect-progressive
future	future-perfect
future	future-perfect-progressive

(f) E1 yi3-hou4 E2

In this condition the relation between E1' and E2' is $E_1 < E_2$. There are two cases:

(i) If the event time of E1' is past (i.e., $E_1 = R_2 <$ S), we have the relation $R_2 < E_2$ < S. However, there is no such case in Table 4. Therefore, we have to make some changes. Intuitively, the event E2' happened before the speech time, thus we may give the tense form of E2' "past" or "past–progressive". The event E1' happened before E2', i.e., it happened in the past–past with respect to the speech time. It possesses the "past–perfect" as its tense form. The sentence (49) shows a good example.

(49) ta1 lai2 zhe4–li3 yi3–hou4 mai3 le5 yi4 ben3 shu1.

(He bought a book after he had come here.)

(ii) If the event time of E1' is future (i.e., S $< E_1 = R_2$), we have the relation: S $< R_2 < E_2$. Still, there is no such case in Table 4. The obvious solution to this case is to set the tense forms for E1' and E2' as {future–perfect, future} or {future–perfect, future–progressive}. See sentence (50) as an example.

(50) ta1 lai2 zhe4–li3 yi3–hou4 jiang1 mai3 yi4 ben3 shu1.

(He will buy a book after he comes here.)

The result is summarized in Table 10.

Table 10

Possible Tense Forms for E1 and E2 in E1 yi3–hou4 E2

tense form of E1	tense form of E2
past–perfect	past
past–perfect	past–progressive
future–perfect	future
future–perfect	future–progressive

In this section, we have discussed the combinations of tense forms in sentences containing two events. However, there are some conventions in English that may change the tense forms of English sentences. We will discuss them in the next section.

4. Some Conventional Rules in English for Translation

When translating a Chinese sentence into English, the sentence pattern and the syntactic roles of corresponding words may be changed. In this paper, we assume a verb in Chinese sentence is translated into a verb in English. In most sentences, this assumption is correct.

Some tense forms in English need be modified according to other world knowledge. For example, the sentence

* (51a) Einstein has visited Princeton.

is incorrect because Einstein is dead, but sentence (51b) is correct.

(51b) Einstein visited Princeton.

In this paper, we propose a rule-checking method to solve this problem. After we generate the tense form, we test the tense forms to see whether they satisfy the following rules to avoid errors. We list some of them below[6,7].

(a). If the tense form we get is "present-perfect" and the person referred by the subject is dead, then change the tense form to "past". Sentence(51b) is an example.

(b). If the tense form is "present-perfect" and the time phrase is given specially, use "past" tense form instead. Take sentence(52) as an example.

(52) ta1 zou2-tian1 yi3-jing1 qu4 le5 Tai2-bei3 le5.

* (He has gone to Taipei yesterday.)

(He went to Taipei yesterday.)

(c). If two clauses are linked by the conjunction "before" or "after", the tense form "past-past" can be changed to "past" and "past-future-past" to "past-future". It is because the conjunction "after" or "before" is sufficient to indicate the relation between the two clauses it connects. As an example, see sentence (53).

(53) wo3 zuo4-wan2 gong1-zou4 yi3-hou4, cai2 chi1 wan3-fan4.

(I ate my dinner after I had finished my job.)

(I ate my dinner after I finished my job.)

(d). In subordinate clause, such as when-clause, if-clause, and so on, replace the "future" tense form by the "present" tense form. Sentence(54) shows such an example.

(54) dang1 chun1-tian1 lai2 de5 shi2-hou4, yan4-zi5 jiu4 hui4 hui2-lai2.

* (When the spring will come, the swallows will return.)

(When the spring comes, the swallows will return.)

(e). If the tense form of the main clause is "past" and the statement in the subordinate clause is a fact or a truth, the tense form of the subordinate clause is set to "present". The sentence (55) illustrates this convention. However, this rule is difficult to implement because the maintenance of the world knowledge is hard.

(55) wo3 gao4-su4 ta1 cheng2-shi3 wei2 shang4 ce4.

(I told him that honest was the best policy.)

(I told him that honest is the best policy.)

To further demonstrate the applicability of above rules, we analyze sentence (56) in detail.

(56) ta1 da1-ying4 wo3 zai4 wo3 li2-kai1 yi3-qin2, qian3 wo3 chi1 wan3-fan4.

The time markers for verbs "da1-ying4" (promise), "li2-kai1" (leave) and "chi1" (eat) in the sentence are past, past-future and past-future-past, corresponding to the tense forms "past," "past-future" and "past-future-perfect" respectively. By rule (c), the tense form of the verb "chi1 wan3-fan4" (eat dinner) is changed to "past-future" (that is, item 9 of Table

2) since there is a conjunction "before." By rule (d), the tense form "past–future" of "li2–kai1" (leave) is changed to "past." In summary, the tense forms of verbs "da1–ying4" (promise), "li2–kai1" (leave) and "chi1" (eat) are changes to "past," "past," "past–future" respectively. Because there are several ways to express the future tense of a verb, we still have to select the representation of its tense forms. Finally, the sentence generated would be "He promised, that he would give me a dinner before I left."

5. CONCLUSION

The paper has presented a preliminary analysis of possible tense forms in both simple sentences and complex sentences through the ERS model. We have found that different sentential structures may have different tense forms for the predicates in sentences. Thus, we may predict the possible tense forms from the structure or the linking element possessed by the sentence.

The problem in this paper is restricted by the domain anyway. The analysis and implementation of the tense forms for simple sentences have been done. However, we still have a long way to go for more complex sentences, discourse, and so on.

REFERENCES

1..C. N. Li and S. A. Thompson, *Mandarin Chinese: A Functional Reference Grammar*, Taipei: Wen–He Pub., 1983.

2..H. J. Lee and R. R. Hsu, "The analysis of tense and aspect in Chinese sentences for translating mechanically to English," *Proc. 1990 Intern. Conf. on Computer Proc. of Chinese and Oriental Languages*, Hunan, 1990, pp.21–26.

3..T. Winograd, *Language as a Cognitive Process,* Addison–Wesley Pub., 1983.

4..H. Reichenbach, *The Elements of Symbolic Logic*, New York: The Free Press, 1966.

5..B. L. Webber, "Forward," *Computational Linguistics*, Vol. 14, No. 2, June 1988.

6..X. F. Huang, *Mastering Modern English Grammar*, (in Chinese), Taipei: Wen–He Pub., 1982.

7..C. S. Lin and Y. Z. Chen, *A New Standard English Grammar*, (in Chinese), Tainan: Dai–Shan Pub., 1987.

A Logic-based Temporal Knowledge Representation in Mandarin Chinese

Hsiu-Hui Lee, I-Peng Lin, and Chien-Ping Wu

Graduate Institute of Computer Science and Information Engineering
National Taiwan University
Taipei, Taiwan 10764, R.O.C.
T8506027@TWNTUCC1

Correspondence :

Hsiu-Hui Lee
Graduate Institute of Computer Science
and Information Engineering
National Taiwan University
Taipei, Taiwan 10764, R.O.C.
(TEL) 886-2-3630231 EXT 3234
(FAX) 886-2-3628167
(E-MAIL) T8506027@TWNTUCC1

Abstract

This paper concerns the representation of temporal knowledge in Mandarin Chinese. We investigate a variety of temporal information and adopt a logic-based approach to analyze and specify interpretation procedures for each of the constituents. Then we represent temporal knowledge in a formal way. Finally a temporal semantics algorithm is proposed to construct the temporal meaning of sentences.

1. Introduction

The concept of time is crucially important because of its highly frequent use in utterances. However, the temporal information in natural languages is fairly complex. It may refer to some date in the calendar such as Nov. 18, 1989, or some deictic time adverbial such as yesterday on which the event occurred. Temporal reference can be an exact point of time such as 2 o'clock, vague period of time such as this year. The above information may refer to the beginning point, the end point, or the duration of event. And the temporal relationships between events can be explicitly or implicitly specified. In summary, most of temporal knowledge is introduced without reference to an explicit date. That is, relevant temporal information is distributed within and across distinct constituents and may be implicit and imprecise.

A fundamental issue in natural language processing is the determination of the semantic structure or the meaning of a sentence from its syntactic structure. We observed that natural language sentences contain several kinds of elements expressing their meaning. Among these are **propositions, tense, aspect**, and **time adverbials** at least. Propositions should be viewed as corresponding to the core verb phrases rather than simply to the verbs. Tense has some specific markers (e.g. *-s*, *-ed* in English) of past versus present to indicate relative position in time with respect to the moment of speaking. Aspect, which concerns the different viewpoints that can be utilized for describing a situation, has to do with such distinctions as perfective vs. imperfective, extension in time vs. instantaneity [5, 15]. And time adverbials are used to specify temporal relations between events, or give the time span occupied by the event [24]. For example,

John has finished his homework today.

The tenseless proposition *John finish his homework* can be interpreted as some kind of logical formulae. Since many formalisms have been proposed to represent the logical formulae successfully [4, 6, 16], we will not consider them here. However, the other constituents of tense, aspect and the time adverbial (*has*, *-ed*, *today*) which are all time-related notions should also be represented in a formal way. In recent years, studies on natural language analysis have mainly concerned with propositions, but algorithms for analyzing tense, aspect and time adverbials which are indispensable for natural language understanding and machine translation have not been sufficiently developed [11, 21, 22]. This paper concerns the temporal representation in natural languages especially in Mandarin Chinese. We will adopt a logic-based approach to analyze temporal knowledge, specify interpretation procedures for each of the temporal constituents and represent temporal knowledge in a formal way.

In English, many linguists and philosophers have investigated the temporal phenomena about verb tense, aspect, and temporal adverbials [8, 9, 19, 20]. However, till now only a few formalisms [7, 10, 18] have been proposed to represent such constituents, and how to use them is still not obvious. Kowalski and Sergot's calculus of events [12], and Lee *et al.*'s logic of time and events [13] concern only the explicit date associated with the event. Those are not general in natural language processing. Though the problem of time is repeatedly discussed in artificial intelligence, much of the work focuses on such domains, e.g. planning [1], database updates [12, 13], medical texts [17], etc. Those discussions are not suitable for reasoning in natural language sentences, either.

In terms of temporal concept, Mandarin Chinese is different from English at least in the following two respects. First, Chinese is one of the languages that have no specific tense markers. Many Chinese sentences without overt temporal reference and other context information can be translated into English with more than one kind of tense. For example,

張三在看這本書。

can be translated into past progressive tense *ChangSan was reading this book* or present progressive tense *ChangSan is reading this book.* Second, Chinese punctual verbs (e.g. '畢業', '死') can also co-occur with durational adverbials. For example,

張三畢業兩年了。

In English, punctual verbs cannot co-occur with FOR-adverbials. Thus, it is ungrammatical to say that

* ChangSan has graduated for two years.

Instead it should be translated into

ChangSan graduated two years ago.

This paper will analyze and extract all the time-related information in Chinese sentences, and put them into an appropriate semantic form. This form can show when the events occur, how long they carry on, and what their chronological orders are. Though the language investigated here is restricted to Mandarin Chinese, the concepts are quite general, and may be applied to other natural languages. Since we use the term **situation** as the combination of proposition, tense, and aspect, we first describe the classification of situations and their representation in Section 2. Section 3 gives an analysis of time adverbials and their interpretations, and Section 4 gets temporal semantics of the whole sentence based on this scheme. Section 5 is the concluding remarks.

2. Situations

2.1 Classification of Situations

In a sentence, the meaning of situation is composed of proposition, tense, and aspect. Since Mandarin Chinese is one of the languages that have no tense and use aspect and other constituents to denote their temporal meaning, we need not consider tense in situations. Traditionally, situations have been classified into four categories, i.e. **accomplishments**, **achievements**, **activities**, and **states** [23]. Achievements (e.g. dying, graduating, arriving at the station, finding an umbrella) are non-extended dynamic situations that occur momentarily in time. Accomplishments (e.g. walking to the store, singing a song) and activities (e.g. playing the piano, looking for an umbrella), both are extended dynamic situations that last or endure through time. In some literatures, accomplishments and activities both are referred as **processes.** Accomplishments are distinguished from activities in that the former are completed in time, have their natural terminal points, rather than merely going on and coming to an end in time. States (e.g. knowing the answer, liking somebody) are like processes in that they too last or endure through time, but they differ from processes in that they are homogeneous throughout the period of their existence. Because aspect serves to distinguish such things as whether the beginning, middle, or end of an event is being referred, and whether the event is completed or possibly left incomplete, it seems reasonable for us to further expand traditional classification of situations. The subcategories of situations are shown in Table 1.

Situation Types		Examples
achievement	simple progressive	張三死了。 張三快死了。
activity	simple progressive inchoative terminative	張三看書。 張三在看書。 張三開始看書。 張三看完書。
accomplishment	simple progressive inchoative terminative	張三蓋了一棟房子。 張三在蓋一棟房子。 張三開始蓋一棟房子。 張三蓋好一棟房子。
state	simple inchoative	張三喜歡李四。 張三開始喜歡李四。

Table 1. Situation classification and examples.

It is noted that the identification of situation types is not a trivial problem. Here, though we do not discuss how we identify situation types in sentences, we can see that many constituents (such as verbal aspect, locative prepositional phrase, objects, etc.) have something to do with it. For example,

(1)張三看書。(simple activity)

(2)張三看這本書。(simple accomplishment)

In sentence (1), the activity-denoting verb '看' with the generic object '書' denotes a simple activity situation, but the same verb '看' in sentence (2) with the referential object '這本書' denotes a simple accomplishment situation. This is because the event '看這本書' can reach a natural terminal point of '看完這本書' and the event '看書' just goes on and comes to an end. Let's take another example,

(3)張三找一把傘。(simple accomplishment)

(4)張三找到一把傘。(simple achievement)

The situation described in sentence (4) is a simple achievement, but this is not the case with sentence (3), which indicates a simple accomplishment situation. The reason is that the verb '找到' is a kind of resultant verb compound (RVC, [14]) that has the meaning of success in '找'. Besides, the event '找到一把傘' can occur momentarily in time and the event '找一把傘' may endure a lot of time.

2.2 Representation of Situations

In our system, each subcategory of situation is represented by a unique two-place situation predicate

pred_of_situation(P,T)

with two arguments: one for proposition P and the other for time T. This predicate not only indicates the situation type, but also associates the proposition with the time on which it occurs. All the situation predicates are listed in Table 2. For example, if the proposition 張三看書' is denoted by P, then the following predicates

simp_acty(P,T1)

prog_acty(P,T2)

inch_acty(P,T3)

term_acty(P,T4)

denote situations of simple activity, progressive activity, inchoative activity and terminative activity respectively.

Situation Types		Predicates
achievement	simple	simp_ahvt
	progressive	prog_ahvt
activity	simple	simp_acty
	progressive	prog_acty
	inchoative	inch_acty
	terminative	term_acty
accomplishment	simple	simp_apsh
	progressive	prog_apsh
	inchoative	inch_apsh
	terminative	term_apsh
state	simple	simp_state
	inchoative	inch_state

Table 2. Situation types and their corresponding situation predicates.

2.3 Aspectual Properties of Situations

It is obvious that there exists some relationship among those situations with the same propositions. Here, we use a kind of PROLOG form to reflect it. For example,

prog_acty(P,T1) :- simp_acty(P,T2), d(T1,T2).

That means if a simple activity situation of proposition P occurs at time T2, and T1 is during T2, then a progressive activity situation with the same proposition occurs at T1. Note that we adopt Allen's thirteen kinds of primitive relations among intervals of time [2]. They are <, >, eq, m, mi, o, oi, d, di, s, si, f, and fi. This rule can reflect the so called **sub-interval property** [3]. It refers to the fact that any segment of interval of certain kinds of events (i.e. activities and states) is an event of the same type (at least down to a certain grain size). Thus, we have the rules

simp_acty(P,T1) :- simp_acty(P,T2), in(T1,T2).

simp_state(P,T1) :- simp_state(P,T2), in(T1,T2).

According to Allen's notation, *in(T1,T2)* means the relation of T1 and T2 may be eq, s, d, or f.

As a consequence of the sub-interval property is that, for activities, if *John V-ed* then at some time *John was V-ing*, and if *John was V-ing* then at some time *John V-ed*. For example,

張三看書。(ChangSan read books.)

implies that at some time

張三在看書。(ChangSan was reading books.)

and vice versa. That is,

inch_acty(P,T1) :- simp_acty(P,T2), s(T1,T2).

prog_acty(P,T1) :- simp_acty(P,T2), d(T1,T2).

term_acty(P,T1) :- simp_acty(P,T2), f(T1,T2).

simp_acty(P,T1) :- inch_acty(P,T2), si(T1,T2).

simp_acty(P,T1) :- prog_acty(P,T2), di(T1,T2).

simp_acty(P,T1) :- term_acty(P,T2), fi(T1,T2).

In other words, the predicate *simp_acty* can imply the predicates of *inch_acty*, *prog_acty* and *term_acty* as well as the relation of time among them, and vice versa. States also have the sub-interval property, that is,

inch_state(P,T1) :- simp_state(P,T2), s(T1,T2).

simp_state(P,T1) :- inch_state(P,T2), si(T1,T2).

On the other hand, accomplishments and achievements have no the sub-interval property but have the problem of **imperfective paradox** [8]. The problem of imperfective paradox refers to the fact that the truth of the progressive form proposition does not imply the truth of the corresponding simple form proposition, while the truth of the simple form proposition does imply the truth of the corresponding progressive form proposition. In other words, if *John V-ed* then at some time *John was V-ing*, and if *John was V-ing* then it does not imply that at some time *John V-ed*. For example,

張三在蓋一棟房子。(ChangSan was building a house.)

merely says that ChangSan was engaged in some process, and says nothing about whether he completed it or not. But

張三蓋了一棟房子。(ChangSan built a house.)

means that ChangSan built a house and at some time ChangSan was building a house. In our system, we can reflect it by the following rules:

inch_apsh(P,T1) :- simp_apsh(P,T2), s(T1,T2).
prog_apsh(P,T1) :- simp_apsh(P,T2), d(T1,T2).
term_apsh(P,T1) :- simp_apsh(P,T2), f(T1,T2).

The predicate *simp_apsh* can imply the predicates of *inch_apsh*, *prog_apsh* and *term_apsh* as well as the relation of time among them, but the predicate *prog_apsh* cannot imply the predicate *simp_apsh*.

As for achievements, the predicate *prog_ahvt* cannot imply the predicate *simp_ahvt*, either. Because achievements are relatively point-like in nature and consequently do not tend to occur in the progressive. For some situations of progressive achievement, they refer to only a portion of the complete process. For example,

張三快死了。 (ChangSan was dying.)

does not imply that

張三死了。 (ChangSan died.)

because we can say that

張三本來快死了，但新藥救了他所以他並沒有死。
(ChangSan was dying but the new medicine cured him so he didn't die.)

3. Time Adverbials

It is common in sentences with time adverbials. The functions of time adverbials are to specify temporal relations between events, to place events in calendrical intervals, and to give the durations of events. In our system, the classification of time adverbials is shown in Figure 1.

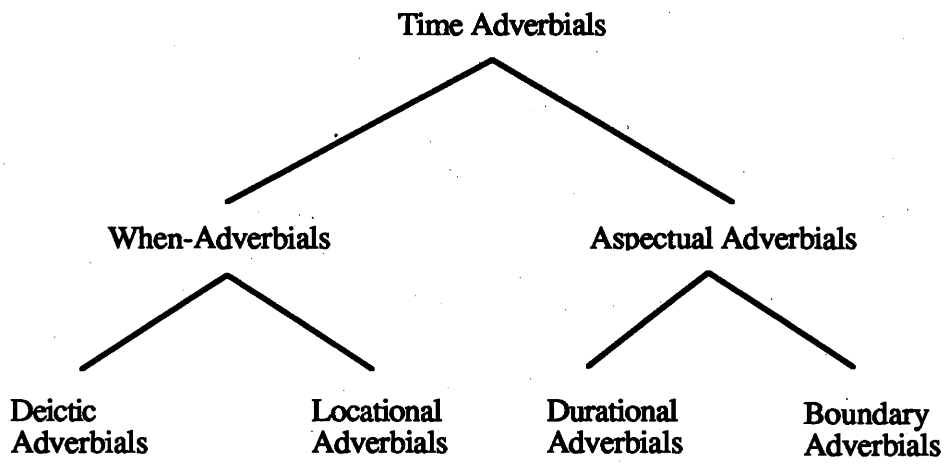


Figure 1. The classification of time adverbials.

Time adverbials are divided into two categories: one for when-adverbials, and the other for aspectual adverbials. When-adverbials that are used as answers to "when" questions are subclassified into deictic adverbials and locational adverbials. Deictic adverbials are adverbials that have explicit relations with the moment of speech, for example, '昨天', '去年', '下星期', '這個月', '兩天後', etc. Locational adverbials are adverbials that identify the location of time without any reference to the speech time, such as dates (like '一九八九年十一月十八日'), days (like '星期三') and times (like '兩點鐘'). Aspectual adverbials that are used as answers to "how long" questions are subclassified into durational adverbials and boundary adverbials. Durational adverbials specify the periods of time (like '兩個月', '三天'). And boundary adverbials (like '自…', '從…', '到…', '從…到…') give the beginning points and/or the end points of the intervals.

When-adverbials can be represented as predicates and have separate interpretation procedures to interpret their meanings. For example, '昨天' is represented by the predicate *yesterday(D)*, and its interpretation is "one day ago". Thus, we have the rule

yesterday(D) :- n_days_ago(1,D).

where the predicate *n_days_ago(N,D)* means that the day D is N days ago and it is further interpreted as

n_days_ago(N,D) :- today(DT), distance_day(D,DT,N).

The predicate *today(DT)* means that DT is the day of today, and the predicate *distance_day(D,DT,N)* means the distance between today DT and the day D is N. Of course, there are several more primitive predicates to interpret these predicates in our system.

Durational adverbials are represented as *length(I,Qty,Unit)*. For example, '三天' is represented by

length(I,3,day)

that means the period of time I is three days long.

As for boundary adverbials, they are represented by predicates *start* and *end*. For example, '從三天前到明天' is represented by

start(I,Ds), end(I,De), n_days_ago(3,Ds), tomorrow(De)

It means that there exists an interval I starting with Ds and ending with De, Ds is the day of three days before today, and De is tomorrow.

4. Representation of Sentences

In preceding sections, we has proposed a formal representation of situations and time adverbials. Now, it is the time to give the meaning of the whole sentences. For simplicity, we restrict ourself to discuss the representation of simple sentences.

4.1 Sentences without Time adverbials

For Mandarin Chinese is a language of no tense, it does not obligatorily relate the time of the situation being described to the time of utterance by any systematic variation in the structure of the sentence. It is also very common in Chinese sentences without time adverbials. Perhaps, we can make some defaults in semantics of aspect for Mandarin Chinese. For those nonstative verbs without aspect marker or with the aspect marker '了' (-le) or '過' (-guo), we assume that they have perfective meaning. And for those nonstative verbs with the aspect marker '在' (Zai) before the verb or the aspect marker '著' (-zhe) after the verb, we assume that they have imperfective meaning. Stative verbs take no marker, and have only imperfective meaning.

In fact, in the absence of any contextual indication of time reference (e.g. when-adverbial), the imperfective forms (i.e. '在' (Zai) and '著' (-zhe) forms of nonstative verbs or simple stative verbs) are interpreted as referring to the present. For instance,

張三在看這本書。 (ChangSan is reading this book.)

張三喜歡瑪琍。 (ChangSan likes Mary.)

While the perfective forms (i.e. 了' (-le) or 過' (-guo) forms of nonstative verbs) are interpreted as referring to the past. For instance,

張三看了這本書。 (ChangSan read this book.)

Only in the presence of an overt indication of time would the imperfective forms be interpreted as referring to the past. For example,

昨天張三在看這本書。 (ChangSan was reading this book yesterday.)

Thus, there is a close relationship between imperfective aspect and present time, and between perfective aspect and past time in Mandarin Chinese. For the sentence without when-adverbial, we, therefore, use predicate *present(R)* as its predicate of when-adverbial for imperfective aspect, and use the predicate *past(R)* for perfective aspect.

4.2 Sentences with Time Adverbials

For situations being represented as pred_of_situation(P,T), and when-adverbials as pred_of_when(R), we need another predicate rel(T,R) to relate the event time T and the time of when-adverbial R. The predicate rel(T,R) can be *d(T,R)* or *eq(T,R)*, depending upon the granularity of the system. The minimum scale of time is readjustable to any scale by a straightforward extension. In our system, we choose the granularity of time as "day". Then for those when-adverbials referring to "large" intervals of time (such as '下個星期') or "very large" intervals of time (such as '去年') during which the events of sentences can occur, we use the predicate *d(T,R)* to indicate the event time is during the time of when-adverbial. And for those when-adverbials (such as '昨天') referring to "small" interval of time, we use the predicate *eq(T,R)* to indicate the event time is equal to the time of when-adverbial. Of course, for predicates of *present(R)* and *past(R)* that represent "not small" intervals, we should use *d(T,R)* not *eq(T,R)*.

For those sentences with aspectual adverbials, we have predicates of *start(I,Ds)*, *end(I,De)*, or *length(I,Qty,Unit)*. We likewise should use another predicate to relate the event time T and the interval I denoted by the aspectual adverbial. For non-achievement situations, we use the predicate *has_duration(T,I)* to indicate that the event lasts for a period of time. Because achievement situations denoted by punctual verbs can occur momentarily in time, we use the predicate *has_distance(T,now,I)* to indicate the interval I being the distance between the event time T and now.

4.3 The Temporal Semantics Algorithm

For we have discussed each constituent of temporal knowledge and have given their meanings, respectively, the algorithm to get temporal semantics of the whole sentence can be summarized as follows:

<u>The Temporal Semantics Algorithm</u>

Step 1 : Determine the correct situation type of the input sentence to get the appropriate predicate

pred_of_situation(P,T).

Step 2 : Check for any occurrence of when-adverbials. If yes, we can get

pred_of_when(R), d(T,R),

or

pred_of_when(R), eq(T,R),

else we will get

past(R), d(T,R)

or

present(R), d(T,R).

Step 3 : If there exists some durational adverbial in non-achievement situations, then we can get

length(I,Qty,Unit), has_duration(T,I)

else we will get

length(I,Qty,Unit), has_distance(T,now,I)

Step 4 : If some boundary adverbial is occurred, then we can get

start(I,Ds), end(I,De), has_duration(T,I).

Let's take some examples to illustrate this mechanism.

<u>Example 1</u>: 張三在看這本書。

In Step 1, by '看這本書', it can be determined as an accomplishment. And by the progressive aspect marker '在', this sentence is further subclassified as a progressive accomplishment, and represented as *prog_apsh(P,T)*. Assume that P is the proposition '張三看這本書'. In Step 2, because of the imperfective form and absence of any time adverbial, we can get *present(R)*, *d(T,R)*. Therefore, the temporal meaning of the sentence is

prog_apsh(P,T), present(R), d(T,R).

Example 2: 昨天張三在看這本書。

This sentence resembles the sentence of Example 1 in that it has the same proposition '張三看這本書', but differs from Example 1 in that it has the when-adverbial '昨天'. '昨天' is represented as *yesterday(R)*. In addition, since the granularity of time in our system is "day", we should use the predicate *eq(T,R)* to relate the event time and the time adverbial. So, the temporal meaning of the sentence is

prog_apsh(P,T), yesterday(R), eq(T,R).

Example 3: 張三看這本書看了兩年。

The same proposition '張三看這本書' is denoted by P. The durational adverbial '兩年' appears in this sentence, and is represented by *length(I,2,year)*. Since it is a non-achievement situation, we should choose the predicate *has_duration(T,I)* to indicate that the event lasts for a period of time. Then we get the temporal meaning of this sentence as

simp_apsh(P,T), length(I,2,year), has_duration(T,I).

Example 4: 張三畢業兩年了。

The durational adverbial '兩年' appearing in this sentence is the same as Example 3 and it is also represented by *length(I,2,year)*. Because the verb '畢業' denotes an achievement situation, by Step 3, we choose the predicate *has_distance(T,now,I)* to indicate the distance between the event time and the present moment. Therefore, we get temporal representation as

simp_ahvt(Q,T), length(I,2,year), has_distance(T,now,I).

Example 5: 張三兩年前畢業了。

This sentence has the same meaning as Example 4, but differs from example 4 in that it uses when-adverbial '兩年前' (that is represented by the predicate *n_cal_elts_ago(R,2,year)*). Because our basic units of time are days, we should use *d(T,R)* to relate the event and the time adverbial. Thus, the temporal meaning of the sentence is

simp_ahvt(Q,T), n_cal_elts_ago(R,2,year), d(T,R).

5. Conclusion

This paper touches on the semantics of temporal information in Mandarin Chinese. We has proposed a formal representation for simple sentences that is composed of situations and time adverbials and also describe the interaction of each temporal constituent on sentence level. The representation of situations can easily reflect their aspectual properties such as the sub-interval property and imperfective paradox. And we have interpretations for a variety of time adverbials. So far, our approach has been limited to process one sentence at a time and independent of all other sentences. At last, the semantic form constructed by a temporal semantics algorithm can tell us when something happened, how long it lasts, and how events are related to one another in time. How to accurately capture temporal meaning of the sentences is central aim of this study. That is very important for further temporal reasoning.

References

[1] J. F. Allen and J. A. Koomen, "Planning Using a Temporal World Model," *Proceedings of 8th International Joint Conference on AI*, 1983, pp. 741-747.

[2] J. F. Allen, "Towards a General Theory of Action and Time," *Artificial Intelligence, 23*, pp. 123-154, 1984.

[3] M. J. Almeida, "Reasoning about the Temporal Structure of Narratives," Ph.D. Thesis, Department of Computer Science, State University of New York at Buffalo, 1987.

[4] A. Colmerauer, "An Interesting Subset of Natural Language," in K. L. Clark and S. -A. Tarnlund (editors), *Logic Programming*, 1982, pp. 45-66.

[5] B. Comrie, *Aspect : An Introduction to the Study of Verbal Aspect and Related Problems*, Cambridge University Press, 1976.

[6] V. Dahl, "Translating Spanish into Logic through Logic," *American Journal of Computational Linguistics*, Vol. 7, No. 3, 1981, pp. 149-164.

[7] M. Dalrymple, "The Interpretation of Tense and Aspect in English," *Proceedings of the Association for Computational Linguistics Conference*, 1988, pp. 68-74.

[8] D. R. Dowty, *Word Meaning and Montague Grammar*, Reidel, 1979.

[9] D. R. Dowty, "Tenses, Time Adverbs, and Compositional Semantic Theory," *Linguistics and Philosophy, 5*, 1982, pp. 23-55.

[10] M. P. Harper and E. Charniak, "Time and Tense in English," *Proceedings of the Association for Computational Linguistics Conference*, 1986, pp. 3-9.

[11] R. R. Hsu and H. J. Lee, "The Analysis of Tense and Aspect in Chinese Sentences for Translating Mechanically to English," *Proceedings of International Conference on Computer Processing of Chinese and Oriental Languages*, 1990, pp. 21-26.

[12] R. A. Kowalski and M. J. Sergot, "A Logic-based Calculus of Events," *New Generation Computing*, **4**, 1986, pp. 67-95.

[13] R. M. Lee, H. Coelho and J. C. Cotta, "Temporal Inferencing on Administrative Databases," *Information Systems, **10***, 1985, pp. 197-206.

[14] C. Li and S. A. Thompson, *Mandarin Chinese*, University of California Press, 1981.

[15] J. Lyons, *Semantics*, Vol. **2**, Cambridge University Press, 1977.

[16] M. C. McCord, "Focalizers, the Scoping Problem, and Semantic Interpretation Rules in Logic Grammars," in D. H. D. Warren and M. Canegham (editors), *Logic Programming and Its Applications*, 1986, pp. 223-239.

[17] K. K. Obermeier, "Temporal Inferences in Medical Texts," *Proceedings of the Association for Computational Linguistics Conference*, 1985, pp. 9-17.

[18] R. J. Passonneau, "A Computational Model of the Semantics of Tense and Aspect," *Computational Linguistics, **14(2)***, pp. 44-60, 1988.

[19] B. Richards, "Tense, Aspect and Time Adverbials," *Linguistics and Philosophy, **5***, 1982, pp. 59-107.

[20] C. Smith, "A Speaker-based Approach to Aspect," *Linguistics and Philosophy, **9***, 1986, pp. 97-115.

[21] J. Tai, "Verbs and Times in Chinese: Vendler's Four Categories," In: D. Testen, V. Mishra and J. Drogo (eds.), *Lexical Semantics,* Chicago: Chicago Linguistic Society, pp. 289-296, 1984.

[22] F. Van Eynde, "The Analysis of Tense and Aspect in EUROTRA," *Proceedings of 12th International Conference on Computational Linguistics*, pp. 699-704, 1988.

[23] Z. Vendler, "Verbs and Times," in *Linguistics in Philosophy*, Ithaca, NY: Cornell University Press, pp. 97-121, 1967.

[24] 張莉萍，"漢語的時間詞組和語言剖析"，中華民國第一屆計算語言學研討會論文集，pp. 77-86, 1988.

The Semantic Score Approach to the Disambiguation of PP Attachment Problem

*Chao-Lin Liu, *,**Jing-Shin Chang and *Keh-Yih Su

*Department of Electrical Engineering
National Tsing Hua University
Hsinchu, TAIWAN, R.O.C.

**BTC R&D Center
2F, 28, R&D Road II, Science-Based Industrial Park
Hsinchu, TAIWAN, R.O.C.

ABSTRACT

In a Natural Language Processing System which takes English as the source input language, the syntactic roles of the *prepositional phrases* in a sentence are difficult to identify. A large number of ambiguities may result from these phrases. Traditional rule-based approaches to this problem rely heavily on general linguistic knowledge, complicated knowledge bases and sophisticated control mechanism. When *uncertainty* about the attachment patterns is encountered, some *heuristics* and *ad hoc* procedures are adopted to assign attachment *preference* for disambiguation. Hence, although the literatures about this topic are abundant, there is no guarantee of the *objectiveness* and *optimality* of these approaches.

In this paper, a probabilistic semantic model is proposed to resolve the PP attachment problem without using complicated knowledge bases and control mechanism. This approach elegantly integrates the *linguistic* model for semantics interpretation and the *objective* characteristics of the probabilistic *Semantic Score* model. Hence, it will assign a much more *objective* preference measure to each ambiguous attachment pattern. It is found that approximately 90% of the PP attachment problem in computer manuals can be solved with this approach without resorting to any heuristics-based rules and complicated control mechanism. The mapping between the abstract Score Function paradigm and the real PP attachment problem will be addressed in this paper. Future expansion of the semantic score function for resolving general ambiguity problems is also suggested.

1. PP Attachment Problem

In a natural language processing system, there are many sources which may cause a given sentence to be multiple analyzed. One of the problems is the *uncertainty* on the placement of the modifiers. Such problem is known as the attachment problem. The most well-known attachment problem in English is the *PP attachment problem* (hereafter, PPAP), where a given prepositional phrase (PP) may modify either the main verb or the preceding noun phrase of the sentence. (Modification to other constituents, for example, the whole sentence, is also possible.) This uncertain characteristics on the placement of the prepositional phrase may lead to a large number of ambiguities.

The importance of resolving the PP attachment problem is twofold. First, the sentences with prepositional phrases are common in English. Secondly, the number of ambiguities resulted from PPs increases with the number of PPs. It is estimated that the number of ambiguities approximately follows a combinatoric series called the Catalan numbers (1, 1, 2, 5, 14, 42, 132, 469, 1430, 4862... and so on) [CHUR 82]. Therefore the importance of PP attachment disambiguation can not be overlooked.

In the past few years, there is an abundance of literature concerning the resolution of the PP attachment problem. For example, Frazier and Fodor [FRAZ 78] uses the well-known heuristic principles *Right Association* (RA) and *Minimal Attachment* (MA) to cope with the attachment problem in their *Sausage Machine* (SM). Marcus [MARC 80] uses a case-frame interpreter to decide the proper attachment pattern in his PARSIFAL parser, which relies heavily on selectional restriction. Ford, Bresnan and Kaplan [FORD 82] proposed their *Theory of Closure* to tackle this problem. Other solutions to the PPAP is numerous. Most of them are not explicitly aimed at the PPAP but aimed at more general semantic analyses. For example, Wilks applied *Preference Semantics* [WILK 75a, 75b, 83] in his intelligent analyzer and understander of English. More detailed information and comments on these approaches can be found in [HIRS 87].

2. Problems with Conventional Approaches

The rule-based approaches in the previous section do resolve certain attachment problem in some specific domain. However, these approaches share some common characteristics which make them difficult to adopt in a large practical system such as a commercialized machine translation system. The following problems are frequently encountered with such mechanisms :

[a] The rule-based systems are inappropriate for handling *uncertain* knowledge. When dealing with a large system with wide coverage, this problem becomes even worse.

[b] The heuristic measures used to assign *preference* to various ambiguities by rule-based systems are usually *ad hoc* or *heuristics-based.* There is no objective measure for evaluating the effectiveness of such rule systems, nor is there any formal way to predict whether the evolution of the rule system is toward the direction of the optimal solution.

[c] A large number of *rules* or *templates* are required which impose a heavy load on the linguists. Any variation in the rules may have unpredictable effect on the whole system. Hence, the large rule system also makes the maintenance of the system a hard task. In addition, complicated control mechanisms are usually required to handle such a rule system.

[d] In sum, these approaches are *non-systematic* in that there is no simple systematic approach for *extracting* the required linguistic knowledge, *verifying* the validity of these rules, maintaining the *consistency* of the rule system and *manging* the rules or templates in an effective way.

In the following sections, we will propose a probabilistic semantic model to resolve the PP attachment problem. Due to the inherent properties of *objectiveness*, *trainability* and *consistency* of a probabilistic model, such model will be more appropriate than rule–based systems when the above problems are taken into account.

The probabilistic semantic model is based on the *Score Function* paradigm suggested in [SU 88, 89, 90b, 90c], which combines both *semantics* and *statistics* to deal with general disambiguation problems. By taking *semantics* into account, we can avoid *blind* preference assignment as suggested in some heuristic approaches like RA and MA principles. By introducing *statistics*, the probability will provide a more *objective* preference measure than heuristically assigned scores in some other systems. Moreover, because a semantic score is given to each analysis, requirement for imposing rigid priority order on conflicting rules are eliminated in *uncertain* situations. Furthermore, the rules or templates will be revealed in the form of probability distribution if they do have some linguistic reality. Hence the linguists can be relieved of writing *exact* rules such as those used in selectional restriction or templates for lexical preference.

3. The Score Function and Semantic Score Approach

Due to the problems mentioned above on conventional disambiguation methods, we seek to find a systematic way to overcome these problems during the development of our English-Chinese Machine Translation System, **ArchTran** [SU 85, 87]. Since the ArchTran is meant to be an *operational* MTS rather than just a laboratory system, a *systematic* approach to semantic interpretation is very important. To achieve this goal, we have proposed a probability based Score Function as the overall evaluation function for *preference* measurement of a sentence [SU 88, 89, 90b, 90c].

To state formally, for a given parse tree (or more generally, a subtree) T which is annotated with semantic feature values on its nodes, the score associated with this particular interpretation of the sentence is given by the following Score Function :

$$Score(T) \equiv P(SEM, SYN, LEX|WRD) \quad ...(1)$$

$$= P(SEM|SYN, LEX, WRD) \times P(SYN|LEX, WRD) \times P(LEX|WRD)$$

where SEM, SYN and LEX are the specific set of *semantic annotation*, *syntactic structure* and *lexical features* attached to the nodes of the parse tree, and WRD is the set of terminal words of the sentence. For example, in the analysis :

$$\left[[saw]_{v\{+stat\}}\ [the\ girl]_{np\{+anim\}}\ \left[with_{\{+means\}}\ [a\ telescope]_{np\{+tool\}}\right]_{pp}\right]_{vp}$$

the verb phrase is given a specific set of semantic annotation SEM = { saw/+stat(ive), [the girl]/+anim(ate), ... }; the syntactic structure SYN is identified by the subtree of the parse, namely VP[v NP[det n] PP[p NP[art n]]]; and the lexical feature is represented by the lexical categories (and other lexical information) of the lexical items as LEX = { v(erb), det(erminer), n(oun), p(reposition), art(icle), n(oun) }. The score for this subtree is then defined as the conditional probability of SEM, SYN, LEX, given the input words WRD = { saw, the, girl, with, a, telescope}. In this sense, we can measure the degree of preference of a semantically annotated parse tree with the conditional probability of any specific set of SEM, SYN and LEX, given the known WRD.

As shown in equation (1), the *score function* can be further divided into three product terms, which are called *semantic score*, *syntactic score* and *lexical score*, respectively. Intuitively, the *semantic score* P(SEM | SYN, LEX, WRD) corresponds to the control mechanism of the semantic analysis phase in traditional stratified analyses. By dividing the score function into these components, it is much easier to apply them to different phases of the analyses or to incorporate them into the system incrementally.

The score function can be shown to be *optimal* as a decision rule *in Bayesian sense*; and the function (or its component functions) has been adopted for several applications [SU 88, 89, 90a, 90b]. For example, a simulation has been conducted to select the preferred parse among a set of ambiguous constructions [SU 88] based solely on the *syntactic score*. The syntactic score paradigm successfully models the parsing process in which arbitrary degree of context sensitivity can be handled. The result is quite promising. It shows that the correct syntactic structures of more than 85% of the test sentences are successfully ranked at the *first* place when a total of *three* local left and right context symbols are consulted. In addition, over 93% of the correct syntax trees are ranked at the *first* or *second* place based on the syntactic score and *two* context symbols.

With this promising result, we were encouraged to develop the semantic score model for ambiguity resolution. In particular, in this paper, we show how the semantic score (more exactly, the *partial* semantic score for a *verb phrase* in a sentence) can be used to solve the PP attachment problem. To state briefly, the semantic score approach to PP attachment problem adopts a simplified version of the *semantic score* as the *preference* measure for possible attachment patterns. The attachment pattern with the highest semantic score is regarded as the most probable attachment. In the next section, we will give a more detailed introduction to the mapping between the PP attachment problem and the semantic score function.

4. Semantic Score for PP Attachment

To simplify the discussion of the semantic score approach, we shall only consider a special case of the PPAP which is characterized by the four major components, [V N_1 P N_2], of a verb phrase . A typical verb phrase of this type is : "*saw the girl with a telescope.*" The symbols V, N_1, P and N_2 refer to the main verb, the head noun of the object, the preposition, and the head noun in the PP, respectively. These four components are selected to characterize the attachment problem because the resolution of the attachment problem depends heavily on their semantic features. Some examples will be shown later. If V-PP is used to mean that PP is attached to the main verb and N_1-PP to mean that PP is attached to the preceding noun, then the (partial) semantic score associated with these attachment patterns can be formulated as :

$$\begin{aligned}
&SC_{SEM}(X|V,N_1,P,N_2) \qquad \ldots(2)\\
&\equiv P(X|V,N_1,P,N_2)\\
&= \sum_{v,n_1,p,n_2} [P(X|v,n_1,p,n_2,V,N_1,P,N_2) \times P(v,n_1,p,n_2|V,N_1,P,N_2)]
\end{aligned}$$

where X can either be V-PP or N_1-PP, and the summation is taken over all *semantic features* v, n_1, p, n_2 of V, N_1, P, N_2, respectively. In other words, we try to assign the attachment preference by evaluating the probability of a particular attachment pattern conditioned on the [V, N_1, P, N_2] 4–tuple. If $SC_{SEM}(V-PP|V,N_1,P,N_2)$ is greater than $SC_{SEM}(N_1-PP|V,N_1,P,N_2)$, then the PP will be attached to the main verb, otherwise the attachment to N_1 is preferred.

An alternative formulation is to compute the joint conditional probability of the attachment pattern and the possible combination of the semantic features [v n_1 p n_2] based on the input strings. The score can then be formulated as :

$$\begin{aligned} & SC_{SEM}(X,v,n_1,p,n_2|V,N_1,P,N_2) \qquad \ldots(2') \\ & \equiv P(X,v,n_1,p,n_2|V,N_1,P,N_2) \\ & = P(X|v,n_1,p,n_2,V,N_1,P,N_2) \times P(v,n_1,p,n_2|V,N_1,P,N_2) \end{aligned}$$

which is exactly the individual terms in Eqn. (2).

If $SC_{MAX}(V-PP) \equiv \underset{v,n_1,p,n_2}{\text{MAX}} [SC_{SEM}(V-PP,v,n_1,p,n_2|V,N_1,P,N_2)]$ is greater than $SC_{MAX}(N_1-PP) \equiv \underset{v,n_1,p,n_2}{\text{MAX}} [SC_{SEM}(N_1-PP,v,n_1,p,n_2|V,N_1,P,N_2)]$, then the attachment pattern V-PP is preferred over N_1–PP, otherwise N_1–PP is the preferred attachment pattern. In other words, the preferred attachment pattern is determined by the most probable attachment pattern and the sequence of semantic classes given the [V N_1 P N_2] 4–tuple. Furthermore, the semantic feature [v n_1 p n_2] which corresponds to the maximal score is assigned to the input [V N_1 P N_2]. Hence, with this formulation, the *lexical ambiguity* on multiple word senses, can also be resolved at the same time when the most preferred attachment pattern is decided.

It is obvious that the second alternative requires less computation than the first one. However, due to the constraints on the amount of tagged corpus and the consideration of producing significant statistics, we use the first formulation as the basis in our simulation.

We can simplify Equation (2) further if we make the following assumptions [LIU 89] :

1. Once the semantic feature of a word is known, the word itself does not affect the score significantly. For example, if the semantic feature {v} of the verb is known, then the input {V} can be ignored from the conditional probability. If this is the case, we can assume that $P(X|v,n_1,p,n_2,V,N_1,P,N_2) \approx P(X|v,n_1,p,n_2)$ by ignoring {V, N_1, P, N_2}. In other words, the attachment pattern is more closely related to the semantic features of the words.

2. The semantic feature of a given word is not strongly influenced by other words or the semantic features of these words so that we can assume that

$$
\begin{aligned}
&P(v, n_1, p, n_2 | V, N_1, P, N_2) \\
&= P(v | n_1, p, n_2, V, N_1, P, N_2) \times P(n_1 | p, n_2, V, N_1, P, N_2) \\
&\times P(p | n_2, V, N_1, P, N_2) \times P(n_2 | V, N_1, P, N_2) \\
&\approx P(v | V) \times P(n_1 | N_1) \times P(p | P) \times P(n_2 | N_2)
\end{aligned}
$$

In other words, we assume that the dependency between the semantic feature of a given word and its context symbols is nearly context-free.

Under these assumptions, equation (2) can be simplified as :

$$
\begin{aligned}
&SC_{SEM}(X | V, N_1, P, N_2) \qquad \ldots(3) \\
&\approx \sum_{v, n_1, p, n_2} P(X | v, n_1, p, n_2) \times P(v | V) \times P(n_1 | N_1) \times P(p | P) \times P(n_2 | N_2)
\end{aligned}
$$

(See [LIU 89] for more details on the derivation of the simplified formula.)

Although it is not known whether the second assumption is true, we make the assumption so as to simplify the problem. The tests show that this assumption still leads to satisfactory results. To take contextual information into account, we can simply retain the items that are significant to the resolution of the PPAP, and extend the above formulation to an arbitrary degree of context sensitivity. In this paper, we will not discuss such topics.

Eqn. (3) is used in our tests to show the effects of the semantic score approach to PPAP when all four components in {V, N_1, P, N_2} are considered. We shall refer to such test scheme as [VNPN] in the following sections. To reduce the computational complexity and to show the individual effect of each component in {V, N_1, P, N_2}, we also conduct a series of tests with some of the terms in equation (3) ignored. The testing schemes and their simplified score functions are listed as follows :

[VxPN] (Ignore the contribution of the object)

$$
\begin{aligned}
&SC_{SEM}(X | V, N_1, P, N_2) \\
&\approx \sum_{v, p, n_2} [P(X | v, p, n_2) \times P(v | V) \times P(p | P) \times P(n_2 | N_2)] \qquad \ldots(4)
\end{aligned}
$$

[VxPx] (Ignore the contribution of the nouns)

$$
\begin{aligned}
&SC_{SEM}(X | V, N_1, P, N_2) \\
&\approx \sum_{v, p} [P(X | v, p) \times P(v | V) \times P(p | P)] \qquad \ldots(5)
\end{aligned}
$$

[xxPx] (Consider the contribution of P only)

$$SC_{SEM}(X|V, N_1, P, N_2) \approx \sum_{p} [P(X|p) \times P(p|P)] \quad \ldots (6)$$

(An "x" in the test scheme means to ignore the contribution of the corresponding component in [VNPN]; that is, "Don't Care".)

The [xxPx] scheme will cover the simplest cases in which the *preposition* strongly implies the attachment preference. For example, the preposition *of* usually leads to the N_1-PP attachment preference such as in :

- "change the format *of* the disk" (N_1-PP).

The [VxPx] scheme further includes the cases in which the *subcategorization* feature of the main *verb* or its *feature co-occurrence* characteristics with the *prepositional phrase* provides extra information for assigning attachment preference. This scheme will assign different preferences to the sentences such as :

- "*sent* the ticket *to* Taipei" (V-PP), and
- "*lost* the ticket *to* Taipei" (N_1-PP).

When the head *noun* (N_2) of the noun phrase in the prepositional phrase reveals strong evidence on the *case role* of the prepositional phrase, including N_2 will definitely be helpful for assigning attachment preference. The [VxPN] scheme formally encodes such preference. It can be useful in resolving such ambiguities as :

- "*eat* the apple *in* the *box*" (N_1-PP), and

  "*finish* the job *in* two *minutes*" (V-PP).

In the latter case, the noun *minutes* strongly implies a [+TIME] feature. Hence, the prepositional phrase "*in* two *minutes*" has the preference of being filled into the case slot of the verb with [+TIME] constraint. Hence, V-PP attachment is preferred.

Finally, when the case is so complicated that we must jointly take into account the *subcategorization features* of the main *verb*, the *predicate-argument structure* among the main *verb*, the object *noun* N_1 and the possible case filler N_2, and the *feature co-occurrence* constraints between the [VNPN] 4–tuple, then we might need a more complicated model such as the one suggested by the [VNPN] scheme.

Note that we have encoded the attachment preference with the *semantic attributes* of the [VNPN] 4–tuple only. Hence, we can easily determine the attachment preference without resorting to *complex knowledge bases* and *control mechanism* as traditional rule-based systems do. We can also benefit from such approach in that an *objective, trainable* and *consistent* system for assigning attachment preference can be easily acquired.

5. The Classification of Semantic Attributes

Before the computation of the required scores, the semantic features must be assigned to each of the four components V, N_1, P, N_2. Among the four components, the semantic features of the *verbs* are considered to be of most importance. To see how the semantic features of the verbs affect the resolution of PPAP, we have tried three different semantic feature sets/hierarchies which are suggested by Givón [GIVÓ 84], Tang [TANG 88] and Chodorow [CHOD 85], respectively.

According to Givón's classification, each *sense* of a lexical item is unique to the language. Hence, each word sense can be regarded as one class. Therefore, the verbs "contain", "have" and "hold", though all have the sense of "inclusion", will be regarded as three *different* verb classes which differ only slightly. With such criterion, the selected verbs in the test sentences are classified into 16 classes [LIU 89], corresponding to 16 word senses of the 14 most frequently used verbs in our test set and training set. The semantic classes thus defined is show in Figure 1 for the verbs used in our preliminary experiments.

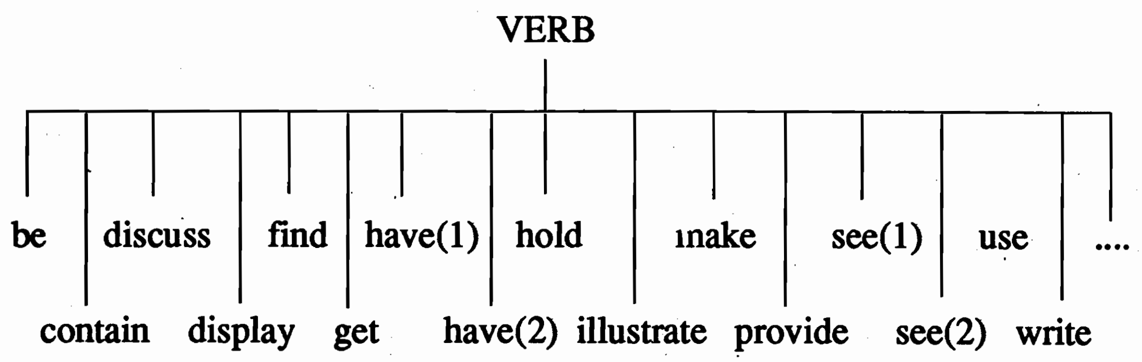


Figure 1 Verb Classification according to [GIVÓ 84].

In Tang's analyses, the verbs can be classified into *stative* and *non-stative* according to their semantic features. Non-stative verbs can be divided further into *non-dynamic* and *dynamic* verbs, which in turn consists of *accomplishment* verbs and *activity* verbs. Such classification forms a semantic hierarchy of the verbs which is characterized by the *syntactic functions* and *aspect features* of the verbs [TANG 88, LIU 89]. Figure 2 shows the hierarchy for the verbs used in our tests.

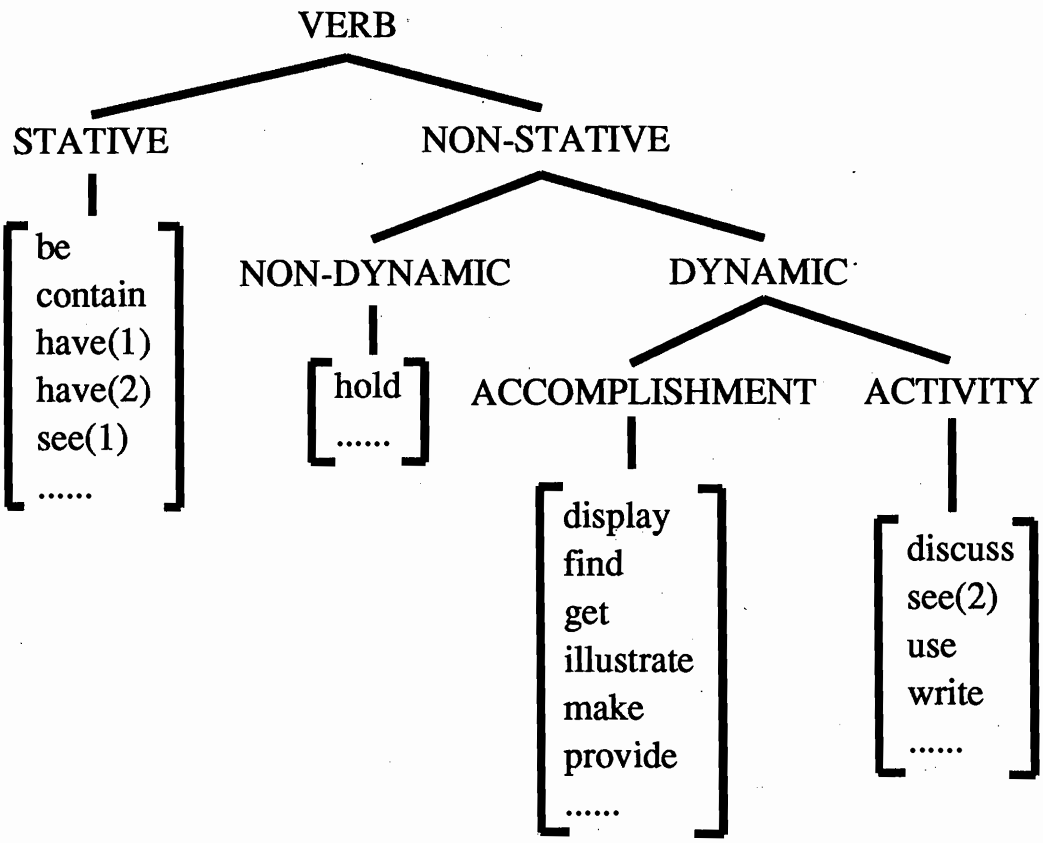


Figure 2 Verb Classification according to [TANG 88].

In the third classification system, Chodorow classifies the verbs which have "similar sense" into the same class. Hence, for example, the verbs "contain", "have" and "hold" will be classified into the *same* class as opposed to Givón's classification. The classification is meant for extracting *semantic hierarchies* from on-line dictionaries. In our testing, we adopted the definitions in Webster's Dictionary (1988 edition) to determine whether two verbs have "similar sense". According to such criterion, a list of 10 verb classes are selected for the test sentences. They are shown in Figure 3.

For prepositions and nouns, only one classification system for each category is adopted. The semantic features for prepositions came from Quirk [QUIR 85], where semantic features such as "duration", "manner" and "means" are used [LIU 89]. The semantic features for nouns, on the other hand, came from [CKIP 88]. They are classified into a hierarchical structure of "physical entity", "non-physical entity", "animate", "non-animate" ([LIU 89]), and so on.

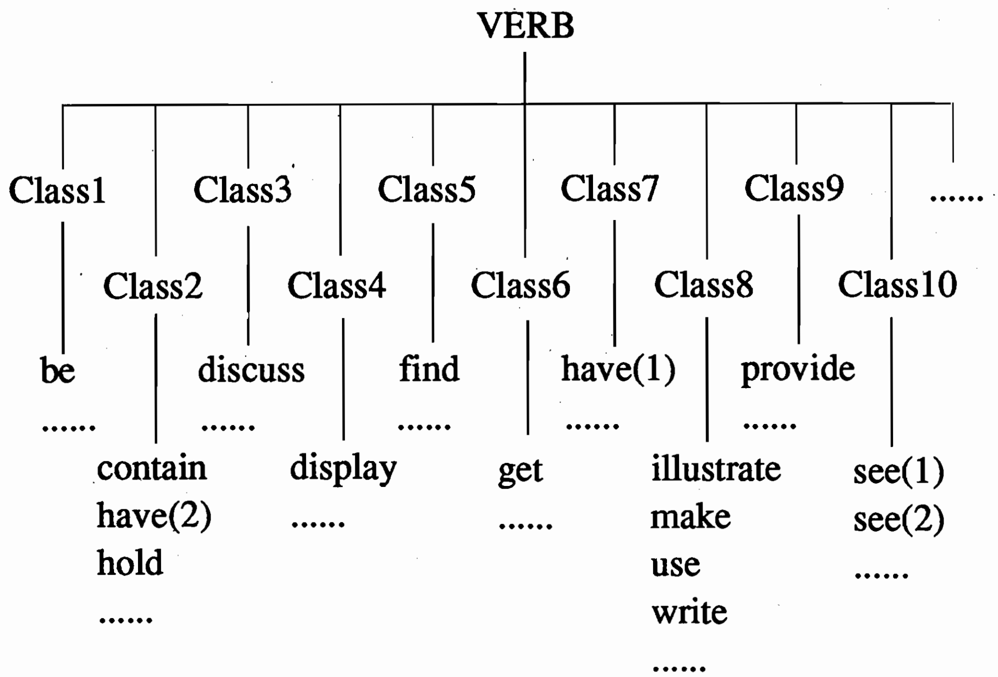


Figure 3 Verb Classification according to [CHOD 85].

With these semantic features, the test sentences in the corpus are either tagged by the ArchTran MTS or manually tagged for testing.

6. Simulation Results

To test the validity of the semantic score approach, we selected the most frequently used verbs, nouns and prepositions from *ten* books in the *computer* field and 1607 sentences parsed by the ArchTran MTS. The result is a list of 14 verbs, 10 prepositions, 6 nouns for N_1 and 18 nouns for N_2. Because not all the sentences have the required verbs, nouns and prepositions in the selected list, we divided the training data and test sentences into 6 databases for different tests. The four *training databases* are called 1607PC(1), 1607PC(2), 1607PC(3) and 1607PC(4), each containing 595, 370, 109 and 31 sentences, respectively. They are chosen from the 1607 sentences and *one* of the ten books. The 1607PC(1) database is a superset of 1607PC(2), which, in turn, is a superset of 1607PC(3). The 1607PC(4) database is the subset of all the other three. The 1607PC(1) database contains [VNPN] 4–tuples which include one of the prepositions in the list of common prepositions; it is used to train the probability for the [xxPx] scheme. Similarly, 1607PC(2), 1607PC(3) and 1607PC(4) contain the required patterns for training [VxPx], [VxPN] and [VNPN] schemes, respectively. There are two test sets, AL and B9, each containing 235 and 115 sentences, respectively. The AL database is selected from *two* of the ten books, while B9 is selected from *nine* out of the ten books.

The probability entries are estimated with relative frequency counts. If a null entry is found, it is replaced with the reciprocal of the number of semantic features of the corresponding lexical category. In other words, if we do not have any information about the semantic feature of a given word, we assume that it can be assigned with any of the possible semantic features with equal probability. Moreover, the probability of the form P(X | v n_1 p n_2) is assigned 1/2 if the [v n_1 p n_2] combination is not found. In other words, we assume that only attachment to V and N_1 is possible.

With the semantic features properly tagged, the following results are observed for the test schemes. Eqn. (3), (4), (5), (6) are used in the test schemes [VNPN], [VxPN], [VxPx] and [xxPx] respectively. Through the result of these schemes, we can estimate which of the four components in {V, N_1, P, N_2} is more significant, and evaluate the number of components required to solve the PPAP. In the following tables, we will show the effects of the score function on PPAP. The semantic features will be included gradually in the order of P, V, N_2 and N_1 in the tests.

TEST I

In this test, only the contribution of the *preposition* is considered in resolving the PPAP. The training data is the 1607PC(1) database, and it also serves as the testing data for *close* data set test. The *open* data set test, which takes sentences *not* in the training data, uses the AL database for testing. The success rates are shown unparenthesized in the table. To compare the semantic model with the RA-MA heuristics in Frazier and Fordor's work [FRAZ 78], the success rate for the RA-MA heuristics is shown *parenthesized* in the table. (For the following tests, these two types of success rates will be shown in the same manner.) It is evident that the semantic score approach outperforms the RA and MA heuristics drastically.

Test Data	[xxPx]
1607PC(1)	78.82 (33.45)
AL	81.28 (18.72)

Number of Sentences : 1607PC(1) = 595 & AL = 235

TEST II

In this test, the effects of both *verb* and *preposition* are considered. The effects of different semantic feature sets for verbs as proposed in [GIVÓ 84], [TANG 88] and [CHOD 85] are also shown in the table. It shows a success rate of about 90% in both open and close data set tests. The semantic feature set proposed by [TANG 88] is slightly better in the open data set test. However, the difference is not distinct.

Test Data	[VxPx]		
	GIVO	TANG	CHOD
1607PC(2)	88.92 (28.92)	86.22 (28.92)	87.84 (28.92)
AL	88.94 (18.72)	92.77 (18.72)	89.79 (18.72)

Number of Sentences : 1607PC(2) = 370 & AL = 235

TEST III

Another test scheme [VxPN] is conducted by taking account of the *noun* in a PP. Comparing the test schemes [xxPx], [VxPx] and [VxPN], the success rates increase as the head nouns of the PPs are taken into consideration. Over 90% of the attachment can be correctly decided in these cases.

Test Data	[xxPx]	[VxPx]			[VxPN]		
		GIVO	TANG	CHOD	GIVO	TANG	CHOD
1607PC(3)	79.82 (30.28)	95.41 (30.28)	86.24 (30.28)	94.50 (30.28)	96.33 (30.28)	90.83 (30.28)	96.33 (30.28)
B9	74.78 (6.96)	90.43 (6.96)	93.04 (6.96)	90.43 (6.96)	92.17 (6.96)	90.43 (6.96)	92.17 (6.96)

Number of Sentences : 1607PC(3) = 109 & B9 = 115

TEST IV

For the last test, the database 1607PC(4) is used. Because there are only 31 sentences in the training set, we do not conduct any open data set test. The results for different test schemes are shown in the table. The table shows a high success rate for all test schemes. Because the database size is small, this table is meant for reference only. However, it shows the trend of increasing success rate when more and more semantic features are involved.

	GIVO	TANG	CHOD
[xxPx]	90.32 (51.61)		
[VxPx]	100.00 (51.61)	90.32 (51.61)	96.77 (51.61)
[VxPN]	100.00 (51.61)	96.77 (51.61)	100.00 (51.61)

[VNPN]	100.00 (51.61)	100.00 (51.61)	100.00 (51.61)

DataBase = 1607PC(4) & Number of Sentences = 31

7. Perspectives and Conclusion

In this paper, we propose a *semantic score* approach to solve the Prepositional Phrase Attachment Problem (PPAP) without resorting to *complex knowledge bases* and *complicated control mechanism.* The probabilistic semantic model elegantly bridges the gap between linguistic knowledge and probability theory, and hence provides both *systematic* and *non-heuristic* approach to resolving the ambiguity problem. In the various simulations, about 90% of the attachment patterns can be correctly determined with this approach. In essence, the only semantic information used to acquire this performance is the *feature co-occurrence* distribution of the semantic classes of the [VNPN] 4-tuple. Hence, no complicated lexical entries and control mechanism are required in such paradigm. This is a very attractive property over conventional rule-based approaches.

We have also attempted to explore the disambiguation effects of different semantic *feature sets* (for verbs). Although the differences are not distinct in the preliminary tests, they do present another important issue in constructing a systematic mechanism for solving general ambiguity problem. Hence the selection of the most significant semantic features will be studied in greater detail and be incorporated into the ARCHTRAN MTS in the future.

In this paper, the semantic score approach is applied to the PP attachment problem only. For more general problems of disambiguation in which various sources of ambiguities can occur simultaneously, a *generalized* probabilistic semantic model, such as in [CHAN 90], will be required to deal with the semantic part of the ambiguity problems. In addition, the *integration* of *lexical* preference, *syntactic* preference and *semantic* preference will be very important for resolving more complicated ambiguity problems in various context. Some of the approaches of integration, such as [SU 90c], will be studied more extensively.

ACKNOWLEDGEMENTS

This paper is a revised version of the master thesis [LIU 89] of the first author. We would like to express our deepest gratitude to BTC R&D Center for providing the required text corpus and many helpfui supports from the ArchTran MTS group.

REFERENCES

[CHAN 90] Chang, J.-S., *GPSM : A Generalized Probabilistic Semantic Model for Ambiguity Resolution*, Master thesis, Institute of Computer Science and Information Engineering, National Chiao Tung University, Hsinchu, TAIWAN, R.O.C., 1990.

[CHOD 85] Chodorow, M.S., R.J. Byrd and G.E. Heidorn, "Extracting Semantic Hierarchies from a Large On-Line Dictionary," *ACL Proc. of 23rd Annual Meeting*, pp. 299-304, 8–12 July 1985.

[CHUR 82] Church, K. and R. Patil, "Coping with Syntactic Ambiguity or How to Put the Block in the Box on the Table," *AJCL*, vol. 8, no. 3-4, pp. 139-149, 1982.

[CKIP 88] CKIP group, *Syntactic Categories for Chinese (revised edition)*, Technical Report #2, Computing Center, Academia Sinica, Taipei, R.O.C., 1988.

[FORD 82] Ford, M., J.W. Bresnan and R.M. Kaplan, "A Competence-based Theory of Syntactic Closure," in Bresnan, J.W. (ed.), *The Mental Representation of Grammatical Relations*, MIT Press, Cambridge, MA, U.S.A., 1982.

[FRAZ 78] Frazier, L. and J.D. Fodor, "The Sausage Machine : A New Two-Stage Parsing Model," *Cognition* 6, pp. 291-325, 1978.

[GIVÓ 84] Givón, T., *SYNTAX : A Functional Typological Introduction*, John Benjamins Publishing Company, Amsterdan, Holland, 1984.

[HIRS 87] Hirst, G., *Semantic Interpretation and the Resolution of Ambiguity*, Cambridge University Press, Cambridge, U.K., 1987.

[LIU 89] Liu, C.-L., *On the Resolution of English PP Attachment Problem with a Probabilistic Semantic Model*, Master thesis, Department of Electrical Engineering, National Tsing Hua University, Hsinchu, TAIWAN, R.O.C., 1989.

[MARC 80] Marcus, M.P., *A Theory of Syntactic Recognition for Natural Language*, MIT Press, Cambridge, MA, U.S.A., 1980.

[QUIR 85] Quirk, R., et al., *A Comprehensive Grammar of English Language*, Longman Group Limited, London, U.K., 1985.

[SU 85] K.-Y. Su, H.-H. Hsu and M.-L. Hsiao, "Development of an English-To-Chinese Machine Translation System," *Proc. of Natl. Computer Symposium (NCS) 1985*, vol. 2, pp. 997-1001, Kaohsiung, R.O.C., Dec 20-22, 1985.

[SU 87] Su, K.-Y, J.-S. Chang and H.-H. Hsu, "A Powerful Language Processing System for

English-Chinese Machine Translation," *Proc. of 1987 Int. Conf. on Chinese and Oriental Language Computing*, pp. 260-264, Chicago, ILL, U.S.A., 15–17 June 1987.

[SU 88] Su, K.-Y and J.-S. Chang, "Semantic and Syntactic Aspects of Score Function," *Proc. of COLING-88*, vol. 2, pp. 642-644, 12th Int. Conf. on Computational Linguistics, Budapest, Hungary, 22-27 August 1988.

[SU 89] Su, K.-Y, J.-N. Wang, M.-H. Su and J.-S. Chang, "A Sequential Truncation Parsing Algorithm Based on the Score Function," *Proc. of Int. Workshop on Parsing Technologies*, pp. 95-104, CMU, Pittsburgh, PA, U.S.A., 28-31 August 1989.

[SU 90a] Su, K.-Y. and T.-H. Chiang, "A Framework for a Chinese Phonetic Typewriter System," *Proc. of 1990 Int. Conf. on Computer Processing of Chinese and Oriental Languages*, Changsha, Hunan, China, 25–28 April 1990.

[SU 90b] Su, K.-Y., J.-N. Wang, M.-H. Su and J.-S. Chang, "GLR Parsing with Scoring," to appear in Tomita, Masaru (ed.) : *Generalized LR Parsing*, Kluwer Academic Publisher, 1990.

[SU 90c] Su, K.-Y., J.-S. Chang and Y.-C. Lin, "A Unified Approach to Disambiguation Using A Uniform Formulation of Probabilistic Score Functions," in preparation, 1990.

[TANG 88] Tang, T.-C., "The Semantic Attributes and Syntactic Functions of English Verbs," *A Cognitive Grammar of English : Form, Meaning and Function*, vol 1, pp. 173–245, Student Book Co., Ltd., Taiwan, R.O.C., 1988.

[WILK 75a] Wilks, Y., "Preference Semantics," In Keenan, Edward Louis, III (ed.), *Formal Semantics of Natural Language*, Cambridge University Press, 1975.

[WILK 75b] Wilks, Y., "An Intelligent Analyzer and Understander of English," *Communications of the ACM*, vol. 18, no. 5, pp 264–274, May 1975.

[WILK 83] Wilks, Y., "Preference Semantics, Ill-formedness, and Metaphor," *AJCL*, vol. 9, no. 3–4, pp 178–187, July-Dec. 1983.

電話轉接對話模式與表達轉接要求句型的分析

[Analyses of Dialogue Patterns and Sentence Patterns of Telephone Switching Requests]

胡海旭 [Hai-Hsu Hu]　李臻儀 [Jen-i Li]　胡祖櫻 [Tzu-Ying Hu]　許仁榮 [Ren-Rong Hsu]

交通部電信總局電信研究所

[Telecommunication Laboratories, R.O.C.]

摘要

本文是根據電信研究所台北連絡處的電話總機值機員和客人間的對話語料，在「對話模式」和「客人表達轉接電話要求的句子句型」兩方面所做的分析報告。從分析中歸納出的對話模式類型和表達轉接電話要求的句子句型，可做為建構「國語文句對談式電話自動轉接系統」的部分知識基礎。並兼論選擇此系統的「知識代表法」和「系統架構」時的基本考量和可取法的範例。

ABSTRACT

Based on the corpus of dialogues between customers for switching telephone calls and the switchboard operator of the Taipei Liaison Office of the Telecommunication Laboratories, this article analyzes both these "dialogue patterns" and those "sentence patterns of sentences spoken by customers for requesting telephone switching". Both types of patterns induced from these analyses construct partial knowledge base for implementing an "Automatic Telephone Switching System capable of talking in text form Mandarin utterances with its custoumers". Considerations on selecting appropriate "knowledge representation" and "system architecture" for this implementation and their paradigms are also included.

1. 簡介

在通訊發達的現代生活中，電話是不可或缺的通訊工具，對一個具有多具電話分機之部門或公司，總機值機人員就成了該部門或公司的通話門房，擔任該部門或公司與外界通話接觸的第一線工作。

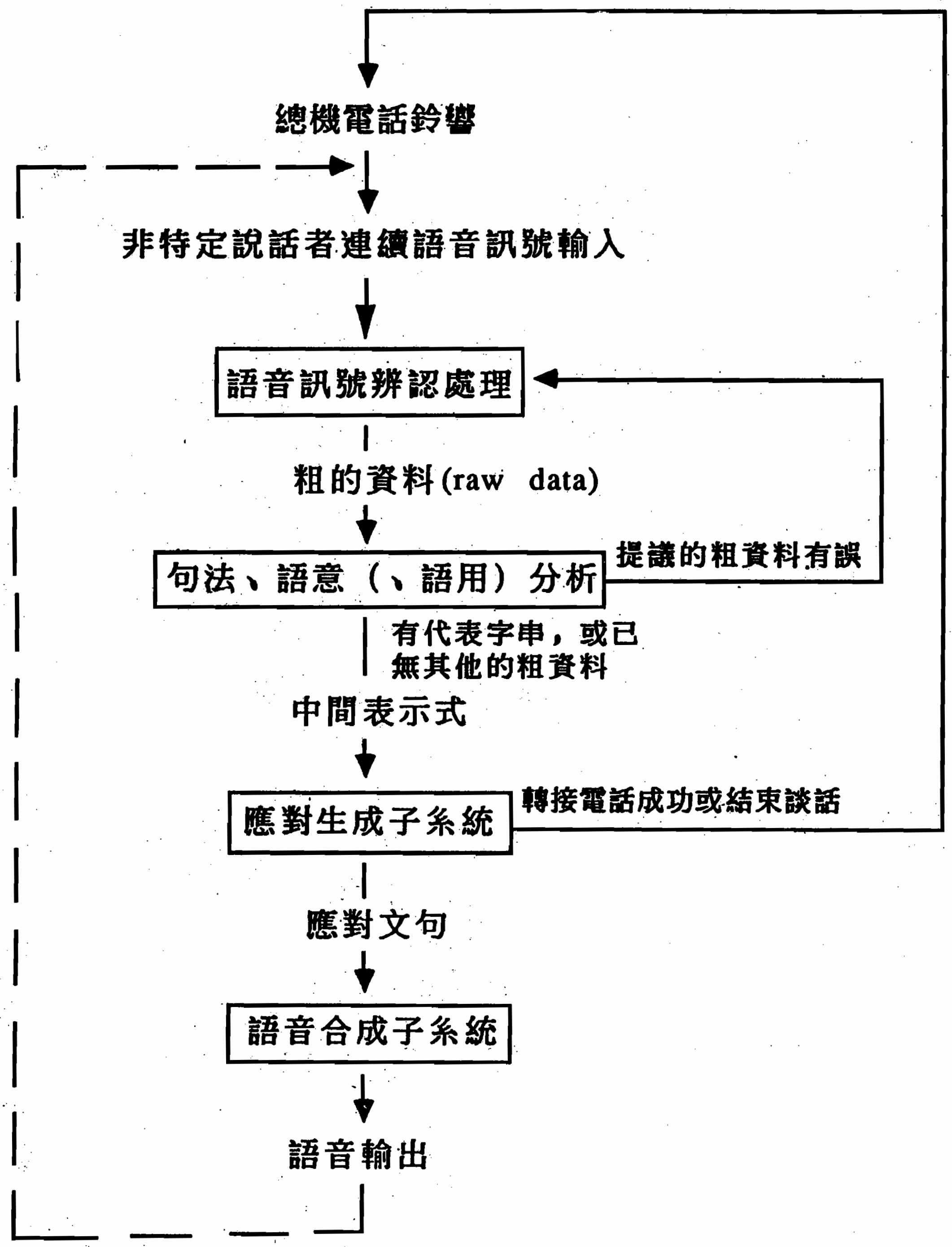
總機值機員的工作繁重，他要轉接外來的電話，對外界詢問與此單位有關的訊息時，也需能妥善的應對；另外，如幫同仁轉撥外線電話，留話的處理…等等，都是十分枯躁乏味的工作，不僅容易疲倦，而且可能使人的脾氣變得暴躁，所以在最近幾年內，總機值機員的工作有些已經逐漸由機器所取代。例如，交通大學的總機機器本身允許打電話

去的人直接撥分機號碼，而不用透過總機值機員轉接，只有該分機在忙線的狀況下才由總機值機員接聽轉接；但是這種半自動化的功能有其限制，因為它只能接受分機號碼，對於不知道分機號碼的人，轉接的工作仍然必須以人工的方式（由總機值機員接聽）來完成。至於詢問事項、留話、轉話等複雜的對話的處理，更是要由人才能做好。近幾年來，由於自然語言的技術又開始蓬勃發展，各式各樣的理論以及應用系統不斷的推出，再加上語音辨認與合成的技術已有些許進步，因此完全由電腦代聽代言的電話總機（簡稱「國語對談式電話自動轉接系統」）已不是夢想。

國語對談式電話自動轉接系統的一個架構如圖（一）所示。在此圖中，語音辨認技術已融入自然語言處理的知識。使用此系統的客人輸入一個語句的語音訊號後，系統利用存放語音訊息和語言知識的字典對此語句做訊號處理、辨認等初步的工作，找出由這語句所代表的最有可能的文字串（我們稱之為「粗的資料（raw data）」），然後再經相關的句法(Syntax)、語意(Semantics)、語用(Pragmatics)的分析來確認；如果在確認過程中發現有問題，就由語音訊號辨認處理子系統再產生其他的字串以供分析，直到找到輸入語句的適當代表文句或無其他可能的字串為止。在選出了輸入語句的代表文句、找出它的中間表示式後，經由應對生成子系統判斷是否執行轉接、結束談話或是產生應對文句。如果產生應對文句，經語音合成輸出後，系統就處理下一個語句的輸入；否則系統就等待下一次的電話轉接工作。

以世界上現有的技術而言，發展此系統主要的瓶頸在於如何將非特定客人所說的連續語音轉換成文字串，或直接轉換成該段連續語音所代表的中間表示式，但發展其餘各子系統時，情況卻很樂觀：只要將此種系統的對話能力限制在處理電話轉接對話和一些與電話轉接有關的對話範圍之內，以現有的「自然語言處理」技術和「語音合成」技術，「應答生成子系統」和「語音合成子系統」是可以在適當的研發之下完成的。

雖然電話轉接時，總機值機員和要求轉接的客人間的談話或「言談領域 (domain of discourse)」有所限制，但是一般人要求轉接時，常用那些句型、詞彙來表達要求轉接電話或其他有關的意願？轉接對話時，雙方交談的內容、方式、過程等是怎樣的？這些都必須就總機值機員和客人間各種不同類型的對話實例加以探討、分析、比較，才能找出各種有關的規則。



圖（一）國語對談式電話自動轉接系統架構

本文是[胡海旭等 '90a]的濃縮版，其所根據的語料由約 2,500 組的對話所組成，這些語料謄寫自10個（每個1小時）卡式錄音帶上所記錄的「交通部電信研究所台北連絡處」的總機值機員與客人的對話[胡海旭等'90b]。我們依對話的目標將這些語料分成兩大類：第一類是直接轉

接；第二類是非直接轉接；此二類對話分別約佔上述語料中對話組數的95%和5%。本文將對此二類對話做分析，找出對話時客人說出轉接要求句子的句型和對話的各種模式，做為建立「國語文句對談式電話自動轉接系統」時，所需要的資料庫、知識庫及可能句型的基礎。這裡所謂的「對話模式」是指：在轉接的過程中，對話的雙方為達成某種共同的目標（例如轉接電話至某分機）所採用的相互問答的方式、內容，總機值機員對應的動作等，所有與完成此目標有關的整個過程。

2. 直接轉接的對話分析

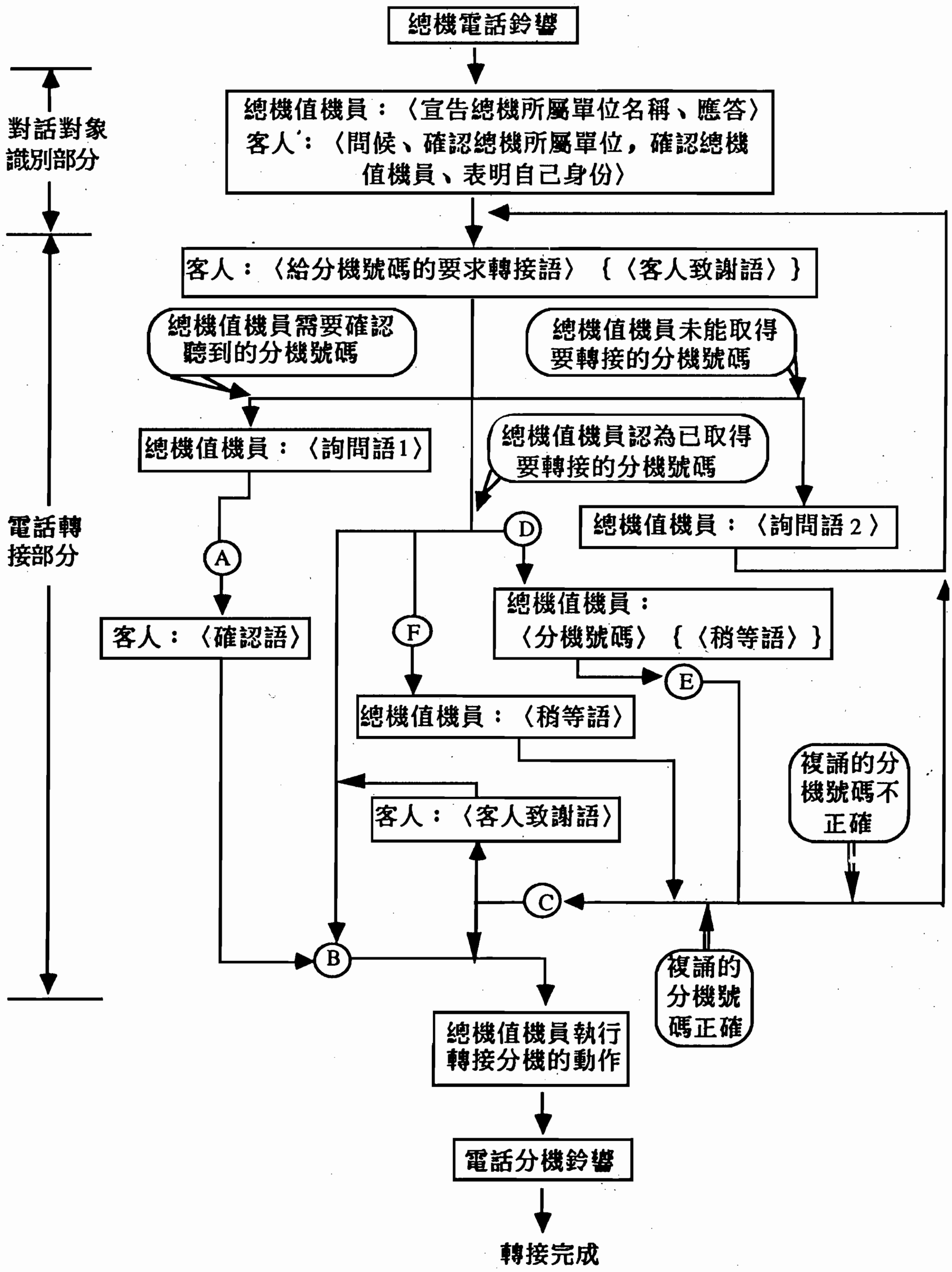
大部分打給總機的電話，都是直接要求總機值機員幫忙轉接某人（或某分機），我們稱這類型的對話為「直接轉接對話」；有少部分的電話則是先與總機值機員談論事情，然後要求轉接（通常這一類的客人都與總機值機員認識）或是並未要求轉接，只是詢問總機值機員所屬單位的一些訊息，或是要求留話，這三者都屬於「非直接轉接」型對話。本節先探討「直接轉接對話」，「非直接轉接對話」則留待第3節討論。

直接轉接對話可分為「簡單轉接對話」和「複雜轉接對話」兩類。「簡單轉接對話」指的是：總機人員在聽懂了客人的轉接要求後，第一次轉接就有人接聽的轉接狀況下，客人和總機人員間的對話。直接轉接對話中不屬於簡單轉接對話的，則歸類為複雜轉接對話。

2.1 簡單轉接對話

簡單轉接對話佔了轉接對話語料的80%左右，客人在這類對話中說出轉接的要求時，都使用簡明扼要的句子，以達成轉接的唯一目的。

根據客人說出轉接要求時所給資料的不同，簡單轉接對話可細分為三小類：第一類是「客人給了分機號碼」；第二類是「客人給了要轉接的人、地點、單位名稱等三者之一」；第三類是「客人既給了要轉接的人名，又給了分機號碼，且該分機確為此人所用」。不論上述那一類對話，都可以用客人提出轉接要求的句子，或總機值機員詢問客人打電話來的目的為何的句子為界線，將整個對話依其部分對話所要達到的目標分為兩段，可有可無的前段是「對話對象識別」部分，後段則是「電話轉接」部分。其中第一小類對話模式的流程圖繪於圖（二）中。



圖（二）客人給了分機號碼，且初次轉接就成功的「電話分機轉接對話模式」流程

「對話對象識別」部分的對話模式在每一類型的轉接都相同，它包括由總值機員聽到總機的電話鈴響(T-R: Telephone-Rings) 後，接通外線打入的電話起，到上述作為界線句子之前，所有總機值機員和客人間的對話。這部分對話的主要目的在互相識別說話的雙方是否正是彼此想要說話的對象。在整個對話中，這部分是可有可無的，下面就列舉一個有「對話對象識別」的例子，其中該部分是以「→」標示的：

《1A-17a》： T-R〔外線電話打入時，總機電話鈴響。〕

→ Op：研究所。

→ Cus：ㄟ，研究所是不是？

→ Op：是。

Cus：ㄟ，麻煩你跟我接257好不好？

Op：257啊？

Ding-T〔總機值機員撥分機號碼後，分機的鈴聲。〕

這裡的《1A-17a》所代表的是取自第一號錄音卡帶 A 面上的第17a 個對話。列在方括弧（即〔〕）內的是註釋或補充說明（以下各頁內的例子中，各同類符號的意義，請類推）。在對話中，總機值機員（Op）首先表明自己所在的單位是研究所，而客人（Cus）為確定自己撥對了號碼就問總機值機員：「ㄟ，研究所是不是？」，總機值機員應答「是」，然後才進行電話轉接或詢問的服務。

除了有「問候」、「宣告或確認總機所屬單位名稱」、「應答」等功能的句子外，「對話對象識別」部分也可以包括客人「確認總機值機員」和「表明自己的身份」的句子，如下例所示：

《5B-130》： T-R

Op：研究所。

→ Cus：吳小姐 ㄏㄚˋ。

Op：對。

→ Cus：我王金土 ㄏㄚˋ。

Op：ㄏㄟˋ，王副座 ㄏㄟˋ。

...

經過或略過對話對象識別後，整個轉接對話就進入了電話轉接的核心---「電話轉接」部分。「電話轉接」部分的對話是由客人提供充分的資訊，以便總機值機員完成電話的轉接。客人以‹要求轉接語›提出轉接要求，之後有些客人還會以「謝謝」…等‹客人致謝語›表示謝意。根

據客人提出轉接要求的句型在型態上的差異，我們將‹要求轉接語›依據「語意語法(Semantic Grammar)」模型[Allen '87:250-253,364; Rich '83:320-322] 分類，並且定出了16 個句型，其中5 個屬於‹給分機號碼的要求轉接語›，其餘的11個屬於‹給人名的要求轉接語›或‹給地點或單位名稱的要求轉接語›。除了這16種基本句型之外，也有少數句子屬於‹給人名也給分機號碼的要求轉接語›，這類型的句子是由分別屬於‹給人名的要求轉接語›和‹給分機號碼的要求轉接語›的句子所組成。下面以BNF符號系統來說明這些句子集合間的關係：

‹要求轉接語›::=‹給分機號碼的要求轉接語›|
‹給人名的要求轉接語›|
‹給地點或單位名稱的要求轉接語›|
‹給人名也給分機號碼的要求轉接語›
‹給分機號碼的要求轉接語›::= ‹P1›|‹P2›|‹P3›|‹P4›|
‹P5›|‹聽不清楚›
‹給人名的要求轉接語›::= ‹P6›|‹P7›|‹P8›|‹P9›|‹P10›|
‹P11›|‹P12›|‹P13›|‹P14›|
‹聽不清楚›
‹給地點或單位名稱的要求轉接語›::= ‹P15›|‹P16›
‹聽不清楚›::=〔總機值機員聽不清楚客人說了什麼。〕

採用語意語法來描述‹要求轉接語›的原因，將在本文第4節「結論和未來的工作」中說明。

2.1.1 客人給了分機號碼，且初次轉接就成功

要求轉接句子的句型分析

在‹要求轉接語›中，這一類給了分機號碼的句子佔的數量最大；也許是因為一般人如果知道要找的人的分機號碼，他會給分機號碼，而不給人名。‹給分機號碼的要求轉接語›有5種句型，都是以分機號碼為轉接的核心。其中‹P1›是5種中的基本句型，表示如下：

‹P1›::={‹P1H1›|‹P1H2›}‹P1T›
‹P1H1›::={我}{‹徵求同意語›}{‹P1H1T›}
‹P1H1T›::={‹請求你語›}(‹轉接語詞›)|
(‹請求語›){‹轉接語詞›}
‹P1H2›::={你}{‹徵求同意語›}{‹P1H2T›}
‹P1H2T›::=(‹轉接語詞›)|(‹請求語›){‹轉接語詞›}
‹P1T›::={‹定指語›}(‹分機號碼›){‹請求句尾›}|

{<請求句尾>}{<定指語>}(<分機號碼>)

<徵求同意語> ::= 能不能 | 可不可以

<請求你語> ::= 請你 | 拜託你 | 麻煩你 | 勞駕你 | 請麻煩你

<轉接語詞> ::= 接 | 接一下 | 轉 | 轉一下 | 找 | 要找 | 叫 | …

<請求語> ::= 請 | 拜託 | 麻煩 | 麻煩請 | 請麻煩 | 勞駕 | …

<定指語> ::= 那個

<分機號碼> ::= 292 | 分機292 | 內線292 | 212分機 | …

<請求句尾> ::= 好嗎 | 麻煩一下 | 拜託一下 | 好吧 | …

<P1>不僅涵蓋較複雜的句子，也允許簡單的、只有分機號碼的句子；下列對話中就有個此種簡單型的句子：

《1A-2》：　T - R

Op：研究所。

→　Cus：ㄟ，請接252。

Op：252是啊，稍等一下啊。

Ding-T

<P2>句型與<P1>句型不同的是<P2>句型中有<幫助語>（幫、跟、給）。當<幫助語>在<P2>句型中出現時，「我」和<轉接語詞>也要同時出現，下面的例子中可以看到這個特質。

《1A-41》：　T - R

Op：研究所。

→　Cus：您好，請幫我接261。

Op：261啊。稍等一下ㄏㄚˋ。

Ding-T

<P3>句型以<P1>或<P2>句型為主體，加上<請問語>（請問）。<請問語>可以說是多餘的，因為它並不傳遞要求轉接的訊息；由於說話者邊說邊改口，這個句型也許就是因此而產生的。請參閱下面的例子。

《7B-88》：　T - R

Op：研究所。

→　Cus：喂，請問，請幫我轉一下292。

<P4>與<P1>、<P2>、<P3>不同的是，客人提供了二門分機的號碼供總機值機員選擇轉接；而<P5>不同於前四個肯定敘述句的句型的是：客人以疑問句的方式要求轉接，如「ㄟ，你好，我可以轉246嗎？」。

電話轉接部分的對話模式分析

客人以‹給分機號碼的要求轉接語›說出轉接的要求之後，絕大部分的情形下，「總機值機員認為已取得要轉接的電話分機號碼」，這一類就是我們第一個要討論的對話模式。此時，總機值機員可以直接執行轉接分機的動作；有的時候總機值機員會以‹稍等語›（稍等、等一下、…）請客人等他進行轉接；但是大多數的對話中，總機值機員會復誦所聽到的分機號碼，由客人確認，避免因聽錯了號碼而轉錯了分機。在所收集的語料中，只有一個這樣的例子：

《2A-97》：　T - R
　　Op：研究所。
　　Cus：喂，請轉 236。
→　Op：246。〔Op 復誦。〕
→　Cus：236。〔Cus 以‹P1›型句子更正。〕
→　Op：236。〔Op 再復誦。〕
　　Cus：謝謝。
　　Ding-T

第二類是「總機值機員需要確認聽到的分機號碼」的對話模式。在這種對話模式裡，總機值機員會因客人說話時的音量不夠強，或其他原因，雖然聽到了客人說的分機號碼，但是不敢確定，因而以‹詢問語1›問客人，期待此號碼能被確認；而客人則以‹確認語›肯定回答。

‹詢問語1› ::= {‹大聲點›} ‹分機號碼› 是 ㄏㄤˋ
‹大聲點› ::= 聲音很小，麻煩你再講一下，好不好 | …
‹確認語› ::= （˙ㄝ）| …

《1A-29》正是一個這類對話模式的例子：

《1A-29》：　T - R
　　Cus：請接 254。〔總機值機員沒完全聽清楚。〕
→　Op：ㄟ，大聲一點，好不好，254 是 ㄏㄤˋ？
→　Cus：˙ㄝ，˙ㄝ。
　　Ding-T

第三類是「總機值機員未能取得要轉接的分機號碼」的對話模式。在此一模式中，客人的聲音實在太小了，總機值機員根本無從分辨，或只聽清楚分機號碼中的部分數字。在這種狀況下，總機值機員就必須用下述的‹詢問語2›問客人，從客人的回答中得到需要的訊息：

<詢問語 2> ::= 多少 | <大聲點> | 2多少〔總機值機員只聽清楚分機號碼的頭一個字是 2 。〕| …

下述的《1B-47》即是使用 <詢問語 2> 的例子，而且還用了兩次。

《1B-47》： T - R
Op： 研究所。
Cus：請轉239。
→ Op：ㄜ，聲音很小，麻煩你再講一下，好不好？
Cus：239。
→ Op：2 多少？〔有部分號碼總機值機員仍未聽清楚。〕
Cus：239 。
Op：239 啊！
Ding-T

總機值機員經 <詢問語 2> 取得要轉接的分機號碼後，可能的後續對話或動作就和「總機值機員認為已取得要轉接的電話分機號碼」狀況下的後續對話或動作一樣，因此不再贅述。

根據上面的討論，我們已經詳細說明了繪於圖（二）的，「客人給了分機號碼，且初次轉接就成功」而不再繼續對話的對話模式的流程。

2.1.2 客人只給了要轉接的人、地點、單位名稱等三者之一，且初次轉接就成功

要求轉接句子的句型分析

給了要轉接的人、地點、單位名稱等三者之一的簡單電話轉接對話和給予分機號碼的簡單轉接對話的前段都屬於「對話對象識別」部分，兩者這部分的對話模式相同，所不同的是「電話轉接部分」的對話模式。

「客人給了要轉接的人（的名稱），以要求轉接」的句型裡，客人如何稱呼要轉接的人，是這類句子分類的主要依據。稱呼要轉接的人的方法有多種，例如以下述的 <人名> 方式來稱呼就是一種。若以此 <人名> 分別取代 <給予分機號碼的要求轉接語> 的5個句型中之 <分機號碼>，就可以產生另外 5 個句型。不過在所收集的語料中，只找到類似 <P1> 或 <P2> 的「客人給了要轉接的人的名稱以要求轉接」的句型。這種現象的發生也許是由於 <人名> 和其他概念（或詞彙）間有關聯，而 <分機號碼> 則無的緣故。

<人名> ::= 郭素琴 | 郭小姐 | 施松男先生 | 所長 | 王副座 | …

屬於 <給人名的要求轉接語> 的句型有9個，從<P6>到<P14>。<P6>和<P7>分別包括了由<P1>和<P2>經由以<人名>取代其中的<分機號碼>所產生的句型。以<P6>為例：

<P6> ::= {<P1H1> | <P1H2>}<P6T>
<P6T> ::= {<定指語>} <要轉接的人> {<請求句尾>} |
{<請求句尾>} {<定指語>} <要轉接的人>
<要轉接的人> ::= {<地點> <所屬工作單位>的 | <地點> 的 |
<所屬工作單位> 的} {<定指語>} <人名>
<所屬工作單位> ::= 資訊技術研究室 | 資訊室 | 資訊 | …
<地點> ::= 三樓 | 三樓會議室 | 影印室 | …

<P6>句型中可以只有<人名>，但是<人名>只是 <要轉接的人> 這一組成成份中的一部分，和 <人名> 有關的其他概念（或詞彙，像 <所屬工作單位> 或 <地點> 等）常會出現在<人名>之前(如「網路規劃的曾貴三」等)，使得 <P6>更複雜。下面是個只有<人名>的例子：

《1A-72》：　T - R
　　　　Op：研究所。
→　　　Cus：郭素琴。
　　　　Ding-T

<P8> 到 <P14>裡有的是疑問句，在句尾部分加了<詢問句尾>（在不在、在嗎、來了沒有、…）；有的則提供兩個人名供總機值機員選擇，其中以<P10>和<P13>比較特別。<P10>句型以代表客人自己的「我」為主體，有別於其他以代表總機值機員的「你」為主體的句型，下列《1B-31》中就有個例子。前面所分析的各個句型之中的<人名>都出現在該句型中的動詞（像「接」、「轉」、「找」、「叫」等）之後，只有<P13>句型中的<人名>是在此句型中動詞（像「聽」或「接」等）之前，下列《9A-192》中就有個例子。

《1B-31》：　T - R
　　　　Op：研究所。
→　　　Cus：ㄟˋ，麻煩你，我找王樂基先生。
　　　　Op：王樂基。
　　　　Ding-T

《9A-192》：　T - R
　　　　Op：研究所。
→　　　Cus：麻煩請劉政先生聽電話。

Op：劉政啊。

Ding-T

「客人給了地點或單位名稱，以要求轉接」的句型有<P15>和<P16>兩個，客人使用這兩類句型的好處是，可以把電話轉到某人會出現的某一地點，因而找到此人。《9A-53》中有這樣的例子。

《9A-53》：　T-R

Op：研究所。

→　Cus：喂，你好，能不能接那個四樓會議室啊？

Ding-T

電話轉接部分的對話模式分析

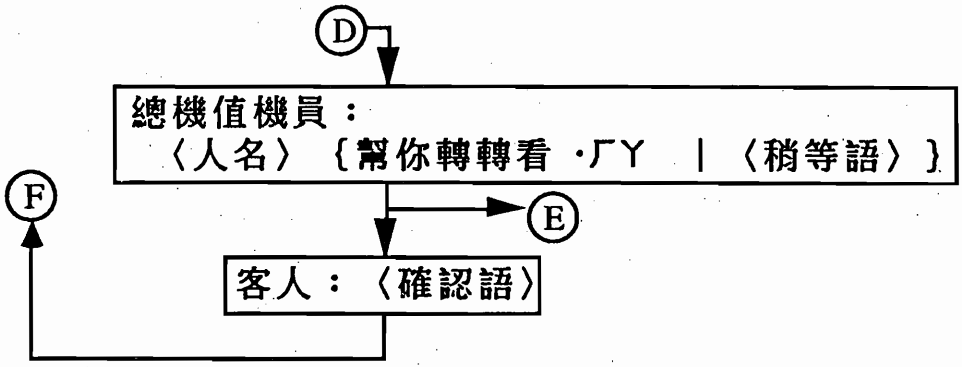
「客人給了要轉接的人名，初次轉接就成功」與「客人給了分機號碼，初次轉接就成功」兩類對話中的「電話轉接部分」的對話模式大同小異，只要經由下列的轉換過程，就可將繪於圖（二）中，後者「電話轉接部分」內的三種對話模式流程，轉換為前者相應部分的三種對話模式流程。

(1) 將圖（二）中出現「分機號碼」的地方用「人名」取代，就可將<給分機號碼的要求轉接語>改成<給人名的要求轉接語>，並修改<詢問語1>的定義；

(2) 更改<詢問語2>和增訂<確認語>的定義，可使原來的「總機值機員未能取得要轉接的分機號碼」對話模式變為「總機值機員未能取得要轉接的人名」對話模式；

(3) 將經上述步驟轉換後的圖（二）中，接至「B」節點的各箭頭移接至圖中的「C」節點；另外以「A」節點為起始點，畫一支線，並將此支線的箭頭指向並接至「B」節點，這樣原來的「總機值機員需要確認聽到的分機號碼」對話模式便成為「總機值機員需要確認聽到的人名」的對話模式；

(4) 要是再將總機值機員復誦分機號碼的流程刪除，以圖（三）取代，使總機值機員復誦的是人名，而不是號碼，就可使原來的「總機值機員認為已取得要轉接的分機號碼」的對話模式變成「總機值機員認為已取得要轉接的人名」的對話模式。



圖（三）與總機值機員復誦人名有關的部分流程

至於「客人給了『地點名稱』或『工作單位名稱』的簡單轉接對話」中的「電話轉接」部分的對話模式流程圖，只要以『地點名稱』或『工作單位名稱』取代「客人給了人名」的簡單轉接對話模式流程圖中的「人名」即可獲得。

2.1.3　客人既給了要轉接的人名，又給了分機號碼，且該分機確由此人所用，並於初次轉接就成功

有些客人，或許是「好心」想給總機值機員方便，在提出轉接要求時，既說出了「人名」，也說出了他所記得的此人使用的「分機號碼」。然而，總機值機員處理這種「過度充分的資訊」時，必須先判定客人所說的這個人是否確實使用客人所說的這門電話分機，反而延後了轉接的動作。這種判定「人機一致」的動作是必須的，因為有些客人記性不佳，給了錯誤的分機號碼。當「人名」和「分機號碼」不一致的情況出現時，總機值機員必須詢問客人，確定要轉給誰，這時的對話已屬於第2.2.1 節中討論的複雜轉接對話；而當「人名」和「分機號碼」一致，而且要轉接的電話在轉接後，有人接的情況下，整個轉接對話就是一種簡單轉接對話。在這種轉接對話中，由於客人所給轉接要求中的資訊充分，因此這種對話模式是所有「電話轉接部分」的對話模式中最簡單的一種，這一點可從語料中的下述對話實例看出來。

《4A-78》：　T - R
Op：研究所。
→　Cus：203 范鳳英。
Ding-T

2.2 複雜轉接對話

除了第 2.1 節的簡單轉接外，其他涉及直接轉接的對話類型，都屬於複雜直接轉接對話。然複雜直接轉接對話也有「對話對象識別」部分，但是這部分和簡單直接轉接對話中的同一部分並沒有什麼不同，所以不再贅述。本節分析討論的重點是各個複雜轉接對話中，除「對話對象識別」部分外的其餘部分的對話模式。我們依客人一開始所給的訊息、總機值機員的處理方式及轉接結果的不同，分成七個次類加以討論。

2.2.1 客人同時給了人名及分機號碼，但該分機並非此人所用

這一類型與2.1.3 節所討論的類似，不同的是，此類型中，客人所說的人並不使用所給的分機。總機值機員的處理方式是：根據人名來更正分機號碼，再予以轉接。

2.2.2 客人給了人名或分機號碼，但電話在講話中，轉接無法完成

客人給了人名或分機號碼，但是和第 2.1 節的簡單轉接對話不同的是，所要轉接的電話在講話中，轉接無法完成，而產生後續對話。後續對話模式可依客人有無提出要求，及要求的類別，細分成三個次類：

〈A〉 客人未提出其他轉接要求： 這種情況下，總機值機員的處理方式有三：(一)要求客人稍後再打，或客人自動表示稍後再打；(二)總機值機員知道有另一門電話，而自動幫客人轉接；(三)總機值機員先詢問客人進一步的訊息，再要求客人稍後再打，或自動幫他轉其他電話。當然，這一類型也包括總機值機員告訴客人電話在講話中，客人詢問（不包括轉接）所內或同仁的資料等服務性事項。以下為本類型的一個例子。

《7A-60》： T - R

Op：研究所。

Cus：請轉252。

Ding-T

B-T

Op：講話中，我給你轉旁邊的ㄏㄚ。

Cus：好好。

Ding-T

〈B〉 客人要求轉其他電話：此時，客人要求總機值機員轉的是另一門電話，好找同一個人或找另一個人。若是找同一個人，則總機值機員依是否另有一門電話可轉接來決定是否轉接或要求客人稍後再打；若是找另一個人，總機值機員則依客人所提供的人名或分機號碼予以轉接。只要是再轉接，就由此再開始另一個新的轉接電話的過程。以下是一個例子：

《7B- 114》： T - R

Op：研究所。

Cus： 請轉202。

...

Op：對不起在講話ㄋㄟ。

Cus：講話啊！

Op：˙ㄏㄟ。

Cus：噢，那就麻煩您幫我轉到隔壁好不好？

Op：好。

Cus：謝謝。

〈C〉 客人要求留話：此時，總機值機員則視自己的情形，讓客人留話（如電話號碼、姓名等）。但此種類型出現的比例很低，在所搜集的語料中，只出現兩次。

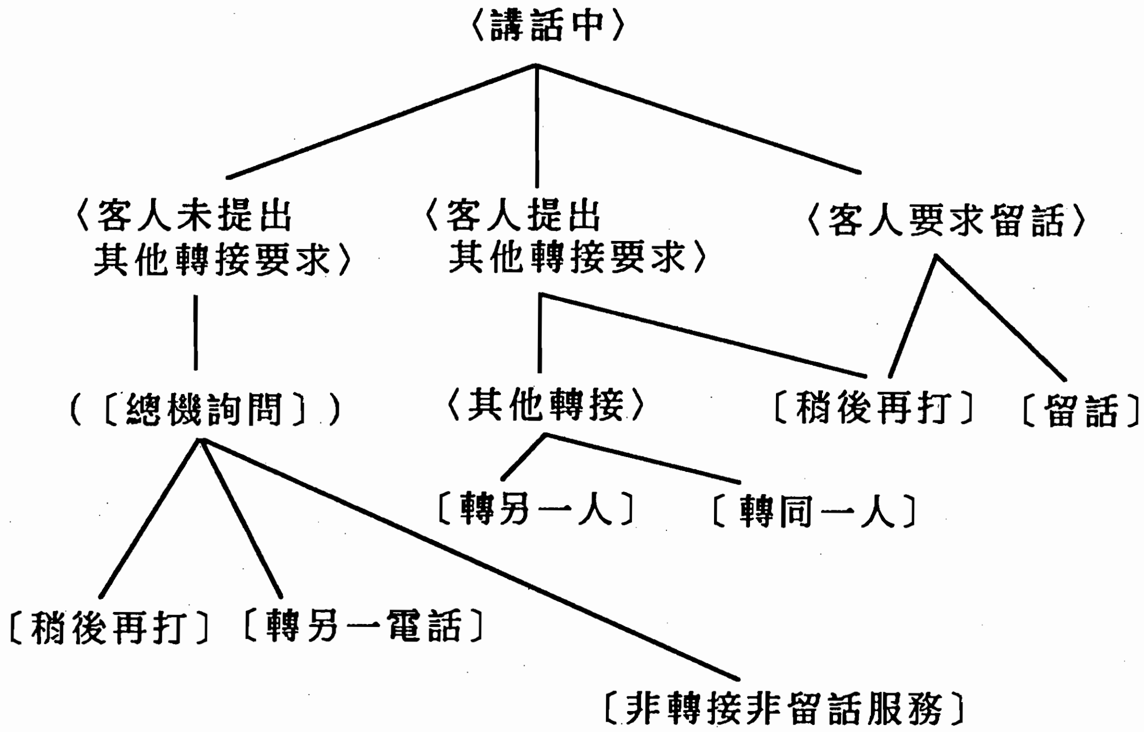
總括而言，這種要轉接的電話在講話中，轉接無法完成所造成的後續對話類型，可依客人的要求，總機值機員的處理方式等摘要如圖(四)。

2.2.3 客人給了人名或分機號碼，但電話沒人接，轉接無法完成

在這一類型的對話中，包含因電話沒人接所引起的後續對話；依客人有無提出要求，及所提出要求的類別，可將後續對話細分成：客人未提出其他轉接要求、客人要求轉接其他電話和客人要求留話三個次類，其分析方式與第2.2.2節類似。

2.2.4 客人給了人名或分機號碼，但總機值機員知道該人不在或分機沒有人接，故無實際轉接動作，轉接無法完成

這一類型的對話，客人給了總機值機員要轉接的人名或分機號碼要求轉接時，總機值機員已經知道客人要找的人不在或該分機沒人接，所以沒有實際的轉接動作；在後續的對話中，值機員先告訴客人無法轉接



圖（四）客人給了人名或分機號碼，但電話在講話中，轉接無法完成的後續對話模式中的分類。在此圖中，〈〉表條件(condition)，（）表可有可無(optional)，〔〕表動作(action)。

的上述原因；然後的對話，也和前兩節一樣，可依客人有無提出要求及所要求的類別，再加以細分成和第2.2.2 節一樣的 三個次類： 客人未提出其他轉接要求、客人要求轉其他電話、客人要求留話。這些對話的分析方式和第2.2.2 節類似，不再重敘。

2.2.5 客人再次打來，轉接訊息不完整，但總機值機員知道客人的轉接目標，直接予以轉接或告知相關訊息

這一類型的對話與前面四節最大的不同是，客人所給的轉接訊息不完整或不清楚，但是總機值機員卻知道客人要找誰或是要轉那個分機，而予以轉接或直接告知該人不在或分機沒人接。總機值機員所以會知道該轉那個分機，主要原因是客人與總機值機員認識或不久前客人已打過電話。所以雖然轉接的訊息不完整或不清楚，仍能轉接。以下是個例子。

《10B-23》： T - R
Op：研究所。
Cus：請問那個周先生回來沒有？

Op：周先生回來了，你等一下˙ㄏㄡ。

…

2.2.6 客人要求轉接的人名或分機號碼都不清楚的情況下的轉接處理

有時客人所給的轉接訊息不清楚，而且客人並不是再次打來，因此總機值機員處理此類轉接，有時是憑藉本身的知識，有時是與客人再溝通澄清後才轉接或告知相關的訊息。《7B-42》就是一個例子：客人給的分機號碼本所台北連絡處沒有；總機值機員告知無此號碼，再加以詢問。

2.2.7 午休時間的轉接處理

中午休息時間打來要求轉接的電話雖然不涉及實際的轉接動作，但客人所給的是轉接的訊息，所以也是與轉接相關的一種類型，總機值機員一般處理的方式是告訴客人當時是休息時間，要他下午上班以後再打。

3. 非直接轉接對話模式分析

佔語料中5%的「非直接轉接對話」包含三類：第一類是間接轉接類；第二類是客人要求非轉接性服務的電話對話；第三類是其他與電話轉接有關的對話。由於這類對話中的「對話對象識別」部分和第2.1節中所討論的簡單轉接對話中的並無不同，所以本節各對話類型的討論重點是除「對話對象識別」部分外的其餘部分的對話模式。

3.1 客人要求轉接，但無法直接轉接

客人要求轉接，但總機值機員卻無法直接轉接的原因有二：一是客人要求轉接的人不屬於本總機轉接範圍內（即交通部電信研究所台北連絡處），而是屬於中壢本所所本部。總機值機員發現屬於中壢本部，即給予適當電話號碼或地址，下面的《7A-9》就是一個例子；二是客人並沒有特定的目標，只是要辦理某些事情，總機值機員則依情況，給予可能人選的電話號碼、或幫他轉接、或婉拒他，如《8B-115》所示。

《7A-9》：〔客人提出要轉接的人名〕客人要求轉接張港基先生，而張先生屬於中壢本所所本部，此時總機值機員給予客人張先生的電話號碼。

《8B-115》：〔總機值機員給予電話號碼〕客人詢問IC電話卡是那個部門負責，總機值機員請他打電話到本所所本部企劃室去詢問，並給相關電話號碼。

3.2 客人要求非轉接性服務

客人要求非轉接性服務約可分成六類，分述於下：

〈A〉查詢本所所名、地址、電話號碼或某同仁的電話號碼、其所屬工作單位

例《2A-11》：〔查詢所名〕客人詢問本所名稱是否為電子技術研究所，總機值機員告知不是後，客人繼續詢問本所全名。

〈B〉要求留話

例《10A-81》：客人打電話來請總機值機員留話給某同仁，請該同仁回來後回話給他，並留下電話號碼及姓名。

〈C〉要求開傳真機接收傳真資料

例《7A-121》：客人在電話接通後即說：「我要傳真，麻煩一下。」

〈D〉詢問某電話通話狀況：客人詢問他剛剛打進來的電話是否沒人接，或者詢問電話為何打不進去。

例《2A-78》：〔沒人接〕客人詢問剛才打的270分機，是不是沒人接。

〈E〉詢問某同仁情形：第一種是客人的目的是要找人，但不確定要找的人是否在本所台北連絡處，所以先詢問此人情形，如《8B-55》所示；第二種是客人先詢問某件事情，再要求轉接。有時詢問的事與轉接之間無關連，《5B-126》就是這樣的例子，有時有關連。

《8B-55》：〔客人沒有其他要求〕客人在詢問要找的人在作什麼，總機值機員回答：「去開會。」，客人得知後並未提出其他要求，對話即結束。

《5B-126》：客人先詢問上班時間，再要求轉接。

〈F〉要求總機值機員給予服務台式的服務：這裡所指的服務台式的服務，類似每棟辦公大樓服務台小姐所給予的服務。

例《1A-83》：同仁自外線打進來，詢問與某同仁派公務車出勤有關的事情。

3.3 其他與電話轉接有關的對話

其他與電話轉接有關的對話主要包括兩大類：

〈A〉打錯或忘了打電話目的的電話：打錯電話的類型有很多種，其對話模式差異很大。有時在客人提出要求轉接的人名後，總機值機員發現要轉接的人非本所同仁，即告知打錯電話；有時是客人在詢問是否為某公司或單位，或詢問此為何處時，就發現打錯電話；也有客人在接通總機後，忘記要作什麼事，這種情形在所收集的語料中只有一例，即《8B-1》。

《8B-1》：

Op：ㄏㄟˋ！

Cus：我要轉299，哇！忘記了。

Op：ㄚˊ！

Cus：啊！沒關係，我待會兒再打來。

Op：好。

Cus：好，對不起。

‹B› 同仁由內線轉回來，因先前轉接時轉錯了人的電話：這種情形發生在總機值機員依客人所給的分機號碼轉接成功後，接通那方的同仁發現客人要找的人，其實並不屬於這門分機，因而又將之轉回給總機值機員，此時總機值機員再依據所要找的人的姓名，轉到正確的分機，這種例子在我們所收集的語料中只有一個。

4. 結論和未來的工作

從語料的分析中，我們將電話轉接對話分成兩大類，一為「直接轉接」，一為「非直接轉接」。前者在電話轉接對話中，佔了極大部份，是指客人清楚的指出要找的人的姓名或電話號碼。基於對話型態上的變化，此類又可分成「簡單型」及「複雜型」兩個次類。非直接轉接是指客人要求轉接但所提出的訊息不夠進行直接轉接，或不屬於此轉接範圍，以及一些對客人的服務性電話。

本文對直接轉接對話中的簡單型的分析，除了對話模式外，也包括客人所說的、用來表達轉接電話要求的句子的句型分析和分類。對直接轉接對話中的複雜型，以及非直接轉接對話的分析則只涉及其對話模式。很明顯的，與「電話總機值機員和客人對話」有關的分析應不只是在本報告中已討論的部分；其他的有關分析，如在此等對話中所使用的其他句子的句型分析，句子的語意、語用分析…等，都可以根據所收集的對話語料進行。要發展一個能像總機值機員一樣，能夠處理本報告前述各節所描述的各種對話狀況的「國語文句對談式電話自動轉接系統」，就要以這些分析所獲的成果為基礎。

我們發展此系統，採用發展人工智慧(Artificial Intelligence)系統時，常用的逐步增強系統功能的策略；我們的初期目標是使其具有處理「直接轉接電話對話中簡單型對話」的能力，下一目標則是使其能處理所有「直接轉接電話對話」，最後才是使其有處理「非直接轉接對話」的能力。就此三個目標，尤其是次二目標而言，本文並未能提供各對

話中所使用的各種句子在句型、語意和語用等方面的分析，以及更為詳細的「複雜型直接轉接對話」和「非直接轉接對話」的對話模式分析。由於此「國語文句對談式電話自動轉接系統」是一個以知識為基礎的系統(Knowledge-based System)，從上述這些分析所獲得的知識是發展建構此系統所必須的，因此進行這方面的分析就是我們下一個必須進行的工作。

除了尚需深入的分析語料外，要使「國語文句對談式電話自動轉接系統」能有我們所期望的處理各種對話的能力，替此系統選擇適當的「知識代表法」和「系統架構」，來分別代表和運用得自語料分析的知識，也是實際發展建構此系統時必須探討的課題；在選擇知識代表法和系統架構時，必須特別考量支持我們發展此系統的上級機關（交通部電信總局）的立場：希望發展製作出的系統在正式運作後，能受到使用者大眾的歡迎，而有助於電信事業的發展。因此，所做的選擇絕不能使發展出的系統，會令使用者在使用後有「此系統不夠和藹、熱心（即親和力不夠）」的不良印象；而產生「除非逼不得已，否則自己下次絕不再用；也不會推薦給親友」的意願。因為，對一般使用者而言，此系統只是個黑盒子(black box)；他們不瞭解，也不想知道系統內部的設計和運作方式，因此不會投其所好，選擇系統容易懂的句子來回應它；使用者所關心的是：當他們以合作的態度，用他們熟悉的句子表達出他們的意思，來回應此系統所說的文句後，系統也能像總機值機員一樣，以「合作的反應(cooperative response)」回應[Kaplan '83:167]。

基於此種使用者的心態，在設計系統架構時，就不能將「句法分析(syntactic analysis)」作為處理使用者所輸入句子的整個過程中的，第一個或主要的步驟；因為句子的「通不通」或「合不合法」，在活的自然語言中，尤其是國語，常是因人、地、時而不同的〔湯廷池， 民國68〕。更何況使用者在說話時，常是邊說邊想，所以語句中很自然的有口頭語、贅語，甚至說了一半又改口的句子。在此情況下，如果將句法分析作為處理句子的最先或主要的步驟，很可能，使用者自己認為足以表達其意思而且值機員也聽得懂的句子，反而會被此系統判定為不合法，並要求使用者以另一種方式重說；因此令使用者對此系統有了不良的印象。

影響此系統形象的另一個因素是處理對話的能力。從電話轉接語料中，可以觀察到下述五種對話時常有的現象；總機值機員能輕鬆而妥善的處理這些現象，並以「合作的反應」回應；除非選對了「知識代表法

」和「系統架構」來代表並運用與此現象有關的對話知識，一般系統通常無法表現得像人一樣好。

<1> 混合式主動(mixed initiative) 發言：對話雙方都可能主動發言，而由對方回應；

<2> 間接語言行為 (indirect speech acts)：客人被問時，會以間接方式回答值機員的問題；

<3> 據實回答暗含預設(implicit presuppositions) 的問題：客人向值機員提出的某些問題不是根據事實，而是客人自己的假設；對這類型問題，值機員會據實以告，而不是以「是」或「否」來回答；

<4> 前照應指涉(anaphoric references)：對話中常用代名詞代表已經提到的事物，值機員能很容易的決定某個代名詞代表那個事物；

<5> 以縮句(ellipses)回答問題：被問者常以問者，用來表達其所問問題的句子中的組成片段(sentence fragments)來回答問者。

為避免因過分依賴句法處理句子所造成的上述困擾，在此系統中將使用能同時包含單獨句子中的句法和語意資訊的「語意語法」[Allen '87: 250; Rich '83: 320-321] 來描述和處理使用者所輸入的句子，此種描述以句中詞的語意類別(semantic categories) 為本，不同於以詞類(part of speech)為本的句法描述。本文前面所說明的，客人所使用的‹要求轉接語›，就是藉語意語法來分析成句型的。使用語意語法的附帶好處是，使此系統在了解句子時，能處理句子中某種程度的意思模糊(fuzziness)或不確定(uncertainty)現象[Barr & Feigenbaum '81:250] ，因而也有助於使此系統有強健的(robust) 處理對話的能力。

至於處理上述五種對話現象，此系統應可採用「和藹慇勤的瞭解者系統(Genial Understander System)」(即 GUS, [Rich '83: 334-5; Bobrow et al. '77]) 中處理對話時，以框架推動對話(frame-driven dialog) 的知識代表法和系統架構，以使其能和GUS一樣，和使用者進行逼真的對話(realistic dialog)。雖然GUS和此系統一樣，所涉及的「言談領域」是很有限的---GUS 在對話中扮演旅行社的工作人員，經由和其使用者以自然語言形式的英文對話，替此人預定由美國加州某個城市回另一個城市的飛機票---GUS和其使用者的對話中也有上述五種現象，因而使我們相信取法於 GUS是此系統處理對話時很正確的選擇。

5. 致謝

本文係交通部電信總局電信研究所79年度計畫(計畫代號：79312)中自然語言處理部分所得成果之一。感謝本所呂學錦所長、王金土副所

長、電腦通訊與人工智慧計畫主持人盧清松博士、人工智慧分項計畫主持人蕭振木博士對有助於本文完成的各項先導工作的支持，另外要特別感謝本所台北連絡處的電話總機值機員吳碧環小姐熱心幫助語料的錄製，也要謝謝促成此錄製工作的本所莊淑慧小姐和張港基先生。自然語言處理組同仁唐憲章參與錄音語料謄寫成文字的工作，對本文的完成也有很大的貢獻，特此申謝。

6. 參考資料

胡海旭編輯，李臻儀、胡祖櫻、許仁榮著，1990a，*電話總機值機員和客人對話分析研究報告，第一卷：「對話模式分析」以及「客人表達轉接電話要求的句子句型分析」*，中華民國交通部電信總局電信研究所。

胡海旭編輯，李臻儀、胡祖櫻、胡海旭、唐憲章、許仁榮 語料謄寫及校對，1990b，*電話總機值機員和客人對話分析研究報告，第二卷：「對話原始語料」第一冊至第五冊*，中華民國交通部電信總局電信研究所。

湯廷池，民國68，〝從句子的「合法」與「不合法」說起〞，在湯廷池編 *國語語法研究論集*，台北：台灣學生書局。

Allen, James, 1987, *Natural Language Understanding*. Menlo Park, California: The Benjamin/Cummings Publishing Company, Inc. pp. 250-3, 364.

Barr, A. & Feigenbaum, E.A., Eds. 1981, *The Handbook of Intelligence, Vol. II*, Los Aitos, Calif., U.S.A.: Kaufman.

Bobrow, D.C., Kaplan,R.M., Kay, M., Norman, D.A., Thompson, H., & Winograd, T., 1977, "Gus, A Frame-Driven Dialog System," *Artificial Intelligence*, Vol.8, 1977, also in Grosz, B.T., Jones, K.S., & Webber, B.L., Eds. 1986, *Readings in Natural Language Processing*, Los Altos, California, U.S.A.: Morgan Kaufmann Pub. Inc., pp.595-604.

Kaplan, S.J., 1983, "Cooperative Responses From a Portable Natural Language Database Query System", in Brady, M., & Berwick, R. C. Eds., 1983, *Computational Models of Discourse* , Cambridge, Masscchusetts, U.S.A.: MIT Press.

Rich, E., 1983, *Artificial Intelligence*, New York, U.S.A.: McGraw-Hill, Inc. pp.320-322.

詞彙訊息的層次表達與管理

簡立峰　　國立台灣大學資訊系
陳克健　　中央研究院資訊所

摘要

在這篇文章裡我們說明近來在自然語言處理所採行的語法訊息表達方式已有逐漸減少使用詞類訊息而改以加重詞彙個別特徵的趨勢。但是這類表達方式在實際運用上會導致(1)詞彙訊息建構不易，(2)訊息的正確性與一致性維持不易,(3)儲存空間耗費太大等問題出現。因此本文嘗試就這些問題加以探討，並且提出一套層次表達與管理方式藉以改善上述問題。利用層次表達具有以下的優點，1.保持資料的一致性，2.重覆訊息得以壓縮，3.訊息更動容易，4.有良好的訊息聚焦性，5.具彈性化的層次分類及訊息取得，6.有助於找出同類間細分類的準則，本文並討論了系統完成的方法。

1. 緒論

近代自然語言處理所採行的語法訊息表達方式，已有逐漸減少使用詞類細分類而改以加重詞彙個別特徵的趨勢。因爲一來個別特徵能更精確描述每個詞的特性，再者配合聯併語法[黃'88]的運用,這類訊息表達方式比傳統大量依賴詞組結構規律的表達方式會有更好聚焦性（執行效率較好)。但是在實際運用上,這類表達方式也會導致(1)詞彙訊息建構不易，(2)訊息的正確性與一致性維持不易，(3)儲存空間耗費太大等問題出現,如果這些問題無法加以克服,將使得依賴這種訊息表達方式的語法模式在實際運用上滯礙難行，因此本文嘗試提出一套層次化的詞彙訊息表達以及管理方法，希望藉此可以改善上述問題。

傳統上詞彙的語法特性都是依靠詞類分類以及詞組結構律來表達，例如，'打'在分類上屬於及物動詞Vt，而及物動詞的語法特性就是當形成動詞片語時需要後一個名詞片語，即結構律VP→Vt NP所表示，因此'打'的語法特性即可以Vt的特性來代表。雖然這種表達方式頗爲精簡，但也常常面臨一個難題，即同類的詞彙常未能具有完全相同的語法特性，因此通常得將這些詞彙再加以細分。如同生物類上有界、門、綱、目、科、屬、種等層次，在詞類分類中也會有層次的關係，例如一般動詞可分及物，不及物。而及物動詞可再分爲單及物或雙及物，甚至進而再細分可還因賓語的不同而區分爲接受名詞、動詞片語或句子的及物動詞。由於性質完全相同的兩個詞很難存在，就如同你、我被歸類於人科人屬人種，但你、我之間還是會有一些細微區別。對同類的兩個詞而言，一樣會有語法特徵上的差異。

爲了避免傳統這種以分類加詞組律的描述方式會有如 (1)只能表示分類的共同訊息，個別差異無法修正，(2)一致性與共存限制等問題必須以相似規律重覆表示， (3)無法兼顧語意的訊息，(4)表示法不夠精簡，對於詞序比較自由的語言，表達時更爲繁瑣，(5)因爲分類過細而造成較差的聚焦性等缺陷[陳'89]。近代語法理論上，多只選擇最重要的語法特徵加以分類，而對於一些次要或不適合分類的特徵 (例如單、複數) 多改爲以附帶特徵的方式來表達而不再以共同訊息視之。因而在強化詞組語法架構下(Augmented Phrase Structure Rules)會有以詞類+特徵的方式來表達語法訊息[Gazdar'85]。在這種架構下，詞類共同特性仍是表現在相關的詞組律上，至於同類詞彙間的差異則以個別特徵視之。此外，還有一些語法表達架構，例如HPSG[Pollard'87]，Categorial Grammar[Uszkoreit'86]甚至有逐漸簡化詞類特徵，轉而注重個別特徵的傾向，其中HPSG將詞組律減至數條而 Categorial Grammar則從根本上以類別取代了詞組律的使用。近來，[陳'89]更提出一個訊息爲本的格位語法表達模式(ICG)，其假設每個詞彙所有的重要語法、語意訊息都完全表達在特徵結構上，藉此詞組律將可完全捨去，如圖一所示動詞"勸"的例子。

特徵：言談類

- 語意訊息
  - 論元
    - 主事者：
      - 特徵：人類
      - ：
      - ：
    - 對　象[1]：
      - 特徵：人類
    - 題　旨：
      - 特徵：動作
      - 論元：
        - 主事者：[1]
  - 附加成分：　{時間，地點，法相，副詞修飾語，語尾助詞，介詞片語}
- 語法訊息
  - 分類：$V_{NP \cdot VP}$
  - 線性律：
    - 主事者[NP] ＜ ＊ ＜ 受事者[NP] ＜命題[VP]
    - 時間 ＜介詞片詞 ＜副詞 ＜ ＊ ＜ 語尾助詞

圖一　詞彙'勸'的特徵結構

述訊息表達方式的優點是利用個別特徵可以更精確描述每個詞的特性，而詞組律的減少也使得聚焦性變得更好（畢竟在傳統一個有數百條詞組律的系統裡，與某一個詞類有關的規律一定十分有限，其聚焦性自然較差，這也是為什麼會有 LR Parsing Table— 一種在剖析時協助聚焦的方法出現[Tomita'85]）。圖二進一步表列了上述有關敘述。

相關語法模式	儲存在詞彙各別詞項之訊息	以詞組律表達詞類分類訊息的情形
PSG	詞類別	大量的詞組律
GPSG	詞類別+個別特徵	強化詞組律
HPSG	以複雜特徵結構表示 所屬類別訊息+個別特徵	少數詞組律
Categorial Grammar	"	無
ICG	"	無

圖二　相關語法模式的詞彙訊息表達方式

然而儘管上述這些注重詞彙個別特徵的語法表達架構有較多優點，但是在實際應用上這種表達方式也衍生不少問題。如因儲存詞彙個別特徵所大量增加的空間需求絕不是一般小型電腦所能負荷。還有個別詞彙所應儲存訊息多半需要異動。然而異動的影響常常導致相同性質的詞彙會有不一致的訊息，例如，我們希望在所有的及物動詞的賓語中增加一語意限制，就必須修改所有及物動詞的詞項。可是萬一有某個詞彙沒有修改到，則除非在日後藉由聯併某些句子而得知，否則這種不一致是很難被發現。這樣自然使得詞彙訊息的構建變得非常困難。因此在這種處理環境裡我們是不能直接把詞彙訊息完全儲存在詞典的各別詞項底下而不加管理，我們得尋求其它更理想的表達與管理方法。因此本文嘗試提出一套層次化的詞彙訊息表達及管理方法以改善上述問題。目前這套方法已被用在建構根據ICG所編定的中文詞檔維護系統。在以下幾節裡，第2節介紹以層次結構來表達詞類訊息。第3節則提出以編輯記錄的方式表示詞彙與其所屬詞類差異。第4 節則是系統製作上的其它考。最後，第5節是結論。

2. 層次化詞彙訊息表達

根據上節所述，在注重詞彙個別特徵的自然語言處理環境下，直接將詞彙所有的訊息完全儲存在詞典個別的詞項底下基本上是不可行的，必須有更

好表達方式來替代。由於即使詞彙所有的語法、語意訊息都以複雜特徵結構表示，但在同類的詞彙間還是會有相同的訊息。因此詞類訊息的區分在這種環境下依然是很重要的。我們可以以如下的關係表示：

完整的詞彙訊息=同類詞彙間的共同訊息(即所屬詞類訊息)+
同類詞彙間之差異訊息(即與所屬詞類訊息的相對差異)。

以這種方式表達詞彙訊息在系統建構以及儲存上將會方便且有效率多。因爲對於每個詞彙而言，建構訊息所需的動作只是擇定所屬類別以及列出與此類別所屬特徵結構的差異即可。例如中文詞'勸'在ICG裡屬$V_{NP,VP}$類，其特徵已如圖一所示，由於$V_{NP,VP}$的特徵結構 (如圖三所示) 與'勸'頗爲近似，因此在勸的詞項下只要表示出與$V_{NP,VP}$的差異即可，而不必儲存整個訊息(表示差異的方法將在下節裡說明)。

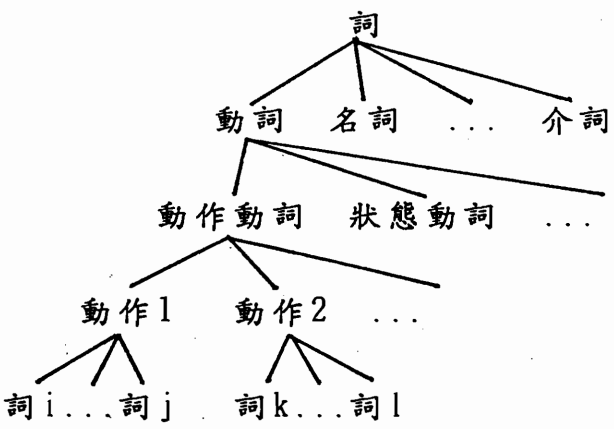
語意訊息
- 論元：
  - 主事者：
    - 意義： ?
    - 特徵： ?
  - 對　象：
    - 意義： ?
    - 特徵： ?
  - 題　旨：[ ]
- 附加成分：{時間，地點，法相，副詞修飾語，語尾助詞，介詞片詞}

語法訊息
- 分類：$V_{NP,VP}$
- 線性律：
  - 主事者[NP] < * < 受事者 [NP] < 命題 [VP]
  - 時間 < 介詞片詞 < 副詞 < * < 語尾助詞

圖三　$V_{NP,VP}$的特徵結構

此外由於詞類分類，如前所述不論何種語言都會有大分類、細分類、再細分類等。如在ICG 內、其大分類包括中文三大詞類介詞、名詞、動詞，而動

詞二類、…，等。因此這些類別很自然地形成如圖四所示的層次結構關係。



圖四　ICG的詞類分類舉例

事實上我們可以以這種層次結構來表示詞類訊息，而個別詞彙的訊息即可視爲是儲放在這個層次結構的最底層。由於在層次結構裡，上層節點與直屬下層節點有一種 is- a關係，這種關係具有特性承襲的性質(Inheritance Property)。所謂特性承襲事實上就是把高層次的訊息聯併給低層次的子孫。這種結構所具有的優點包括訊息分享（上層訊息對下層節點而言可以直接繼承而不必重建)和一致性維持(任一節點的訊息異動都可傳承給下層子孫而不會有不一致的現象）。我們可以利用這種層次結構來表示詞類訊息，讓每一個詞類在結構上都有一節點，而這詞類的訊息即是由聯併其所有祖先節點的訊息以及本身節點的訊息而成。

因爲詞類的訊息較易整理，利用這種層次結構對訊息分享（減少儲存空間，表示彼此的相同特性)以及一致性的維持(可以修改上層節點的訊息）都有助益。此外，因爲下層的特徵值有較高的優先性（priority),如果有例外的情形還可加以修正 ,例如企鵝是鳥類，企鵝承襲了鳥類的所有特徵，包括會飛的特徵。但是企鵝實際上不會飛，因此只要在企鵝的特徵裡註明會飛，則由於詞彙特徵是以下層爲主，就可以簡單的把例外情形解決。

綜合上述層次化訊息的表達方式具有下列的優點：

綜合上述層次化訊息的表達方式具有下列的優點：

(1) 資料的一致性(data consistancy)

由於訊息具有承襲的特性，同類詞彙的語法特性若欲修改，只要修改上層的類別訊息，不必一一更改同類悚的每一個詞彙資料，不致產生掛一漏萬的情形。資料維護容易，也可以保持正確性及一致性。

(2) 資料的壓縮(data redution)

由於同類的詞彙訊息,只在高層的類別訊息上表示一次,減少了許多不必要的重複訊息，達到了分類的主要目的。

(3) 訊息更動容易

修改或增加訊息可以適切的選擇正確的層次加以更動。其作用是全面性的更動了層次以下的所有詞彙訊息。

(4) 訊息的聚焦性

從層次結構上很容易獲得每個詞彙的完整訊息，摒除了其它無關的訊息。對剖析器而言，只須處理相關詞彙的語法語意訊息。可以減少許多不必要的檢測。

(5) 彈性化的分類及訊息取得

層次結構代表的是類別及細分類關係，對不同的應用而言，可能要求的訊息詳細程度有所不同。可以彈性的選取至不同的層次，此外分類也較爲容易。

(6) 有助於找出同類間細分類的準則

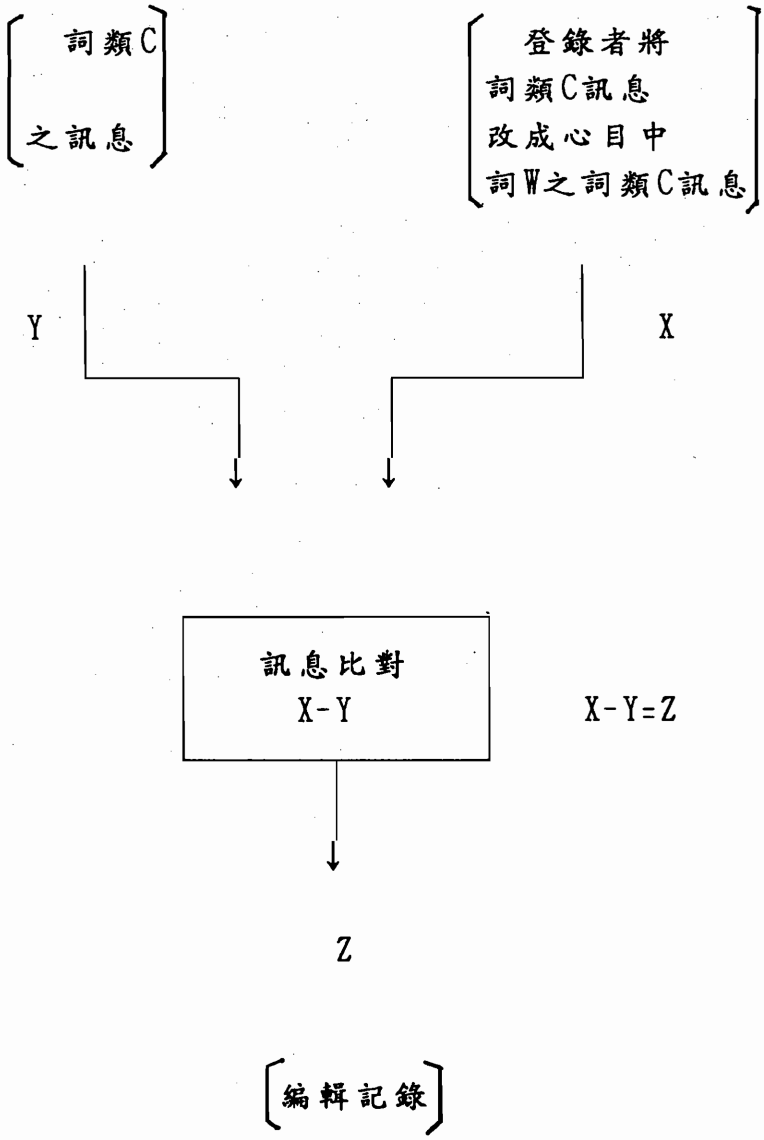
要找出細分類的標準通常並不容易，必須非常小心的觀察同類元素間的相同、相異點、找出足以作爲細分類的特徵。而這些細分類的特徵卻很容易的顯現在同類詞彙的個別差異訊息當中。

3. 編輯記錄與詞彙個別差異

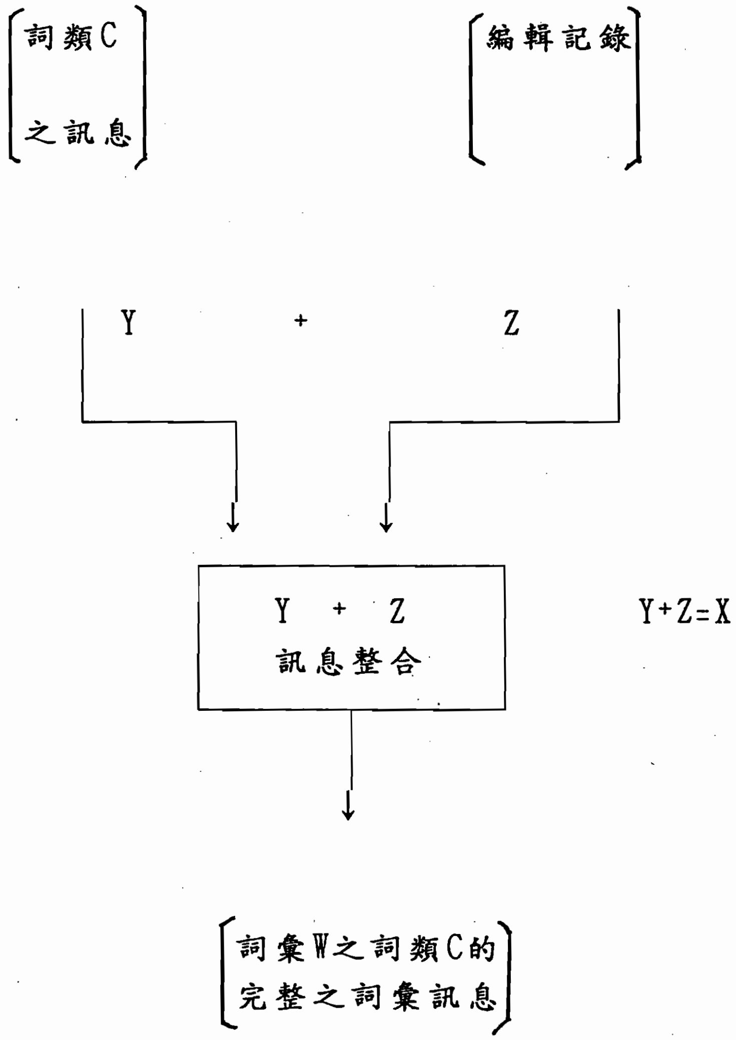
在上一節裡我們介紹以層次結構來表示詞類訊息，但是表達一完整的詞彙訊息還得包括相對於所屬詞類的差異訊息。然而這個差異訊息如何建立呢?事實上對訊息登錄人員而言要他們在詞項下描述與所屬詞類訊息的差異並不容易，例如填上類似" '勸'的主事者必須具有人類的特徵值，但類別 $V_{NP,VP}$

內則無此限制"這樣的敘述並不容易。但是如果將 $V_{NP.vp}$拷貝一份讓他們利用編輯器改成心目中'勸'的特徵結構，他們似乎很容易地就會在 $V_{NP.vp}$的主事者特徵後面加上了人類這個特徵值。於是我們進步提出編輯記錄(Editing Record)的表示方法來使得系統得以自動地表達出這種差異。這個處理方式可以如圖五所示：

(i) 詞彙訊息構建



(ii)詞彙　　　　訊息整合



圖五　編輯記錄和詞彙訊息構建與整合

圖五(i)所示是資料登錄者初次建構詞W之詞類C的訊息所採行的流程。圖五(ii)則是日後異動或審視時的流程。所謂編輯記錄Z即是登錄者心中認為的完整詞彙訊息X和其所屬詞類C之訊息Y的差異 (Z=X-Y)。不論訊息構建或整合,對於登錄人員而言會以為儲存在詞檔裡的訊息即是X，然而事實上卻是Z。這種由系統自動產生差異的方式會使得登錄人員感到非常方便而且系統效率也很好。然而編輯記錄是什麼形式的資料且系統又如何產生?事實上在ICG詞檔

維護系統裡不論詞類或詞彙訊息都被視爲一正規語言，有文法來描述以及剖析器來檢錯並將訊息轉換成串列的形式，因此圖五(i)中的 X和Y即是經由剖析產生的串列，至於X-Y即是X,Y二串列的比對，而最後的Z 就是有關X，Y二串列差異的描述，即所謂的編輯記錄。這類編輯記錄的每一筆格式如下：

路徑名/動作/字串
如果前述C代表如下特徵：

```
⎡sem: ⎡arg: [agent: ]
⎣     ⎣adjunct:
```

而登錄者將其修改爲如下W的特徵：

```
⎡sem: ⎡arg: [agent:[sem:human]
⎣     ⎣adjunct:
```

則Z就是如下之記錄：sem.arg.agent/Insert/sem human

也即在C 中Agent的特徵下增加一 "人類"語意特徵限制。而最後儲存放在詞典W詞項下的訊息將只有類別名稱C和編輯記錄Z(以代碼方式儲放)。

其實這種編輯記錄的最大好處還是訊息一致性的維持。例如若我們希望在所有 $V_{NP \cdot VP}$類的詞彙增加一個特徵。我們只需在 $V_{NP \cdot VP}$ 類別訊息加上此一特徵即可。因爲當初建構詞彙時這個特徵並不存在 $V_{NP \cdot VP}$ 中，所有屬於這個詞類的詞彙其編輯記錄內都不會有任何有關此特徵的動作(也即沒什麼不同)，因此這個訊息便可輕易地就加入所有這類詞彙中。若同樣地日後我們要除去此特徵,只要將其從 $V_{NP,\ VP}$中除去即一樣地等於所有有關的詞彙內也同時刪除。

4. 製作上的其它考量

綜合上述介紹，我們以層次結構來表達詞類訊息，因而在系統製作上有了詞類訊息管理子系統的需求。加上將個別差異訊息（其實是編輯記錄)儲放在詞彙個別詞項,因而有了詞彙訊息管理子系統。然而成爲一個完整的詞檔維護系統,在ICG的詞檔維護系統裡我們還做了如下的考量：

a.運用詞樹與直接

檔案取存技術快速存取訊息由於詞彙眾多且詞彙資料變動長，因此訊息存取必須有相當之技巧，我們利用變動長直接檔存取技術(direct file access with variable record size)和詞樹 (word tree)配合hash searching之應用使得訊息檢索之存取相當便捷。

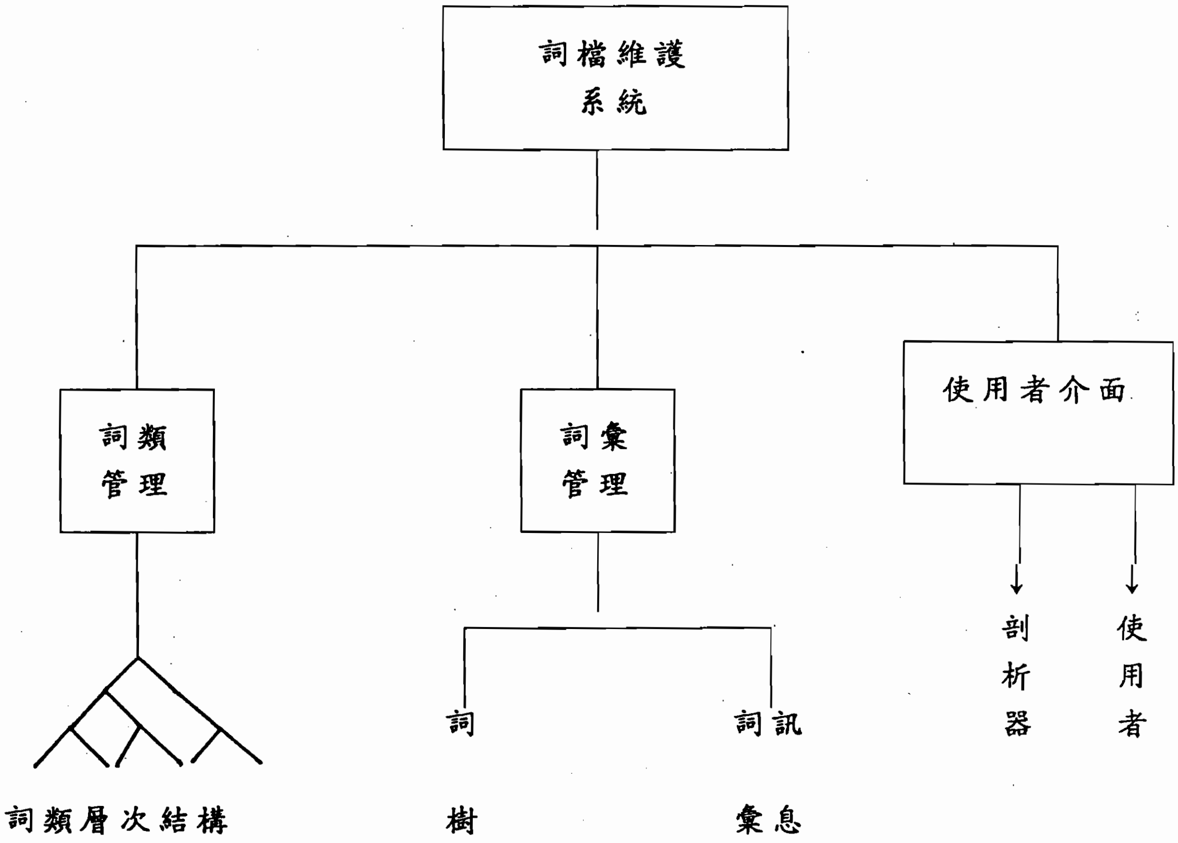
b.視訊息爲一種正規化語言

詞彙訊息必須能夠被剖析器解釋，而且詞彙訊息的正確性也必須加以維護。所以詞彙訊息絕對是一種有規律的知識，而非雜亂無章的資料。因此我們將訊息視爲如程式語言一樣的正規化語言，我們訂定一文法來描述，進而可以利用此文法來剖析資料之正確性。在詞類與詞彙兩個子系統裡我們都利用文法來協助登錄並檢錯。

c.目標導向式系統設計

在本系統內每個詞彙、每個詞類的每一個單項訊息都被視爲一目標(object)，每個目標有其個別的操作動作，如存取、編輯。因此系統設計上非常方便，擴充性也極強。

目前我們完成的管理系統其結構如下圖：



圖六 ICG詞檔維護系統架構

這個系統現在己開發完成並且開始建構詞彙訊息。

5. 結論

在本文裡我們首先提出近來在自然語言處理所採行的語法訊息表達方式已有逐漸減少使用詞類分類訊息而改以加重詞彙個別特徵的趨勢,然而這類表達方式在實際運用上會導致(1)詞彙訊息建構不易,(2)訊息的正確性與一致性維持不易,(3)儲存空間耗費太大等問題出現。因此我們提出利用層次結構來

表達詞類訊息，以及編輯記錄來表示詞彙個別差異。因爲利用層次結構管理詞類訊息這種複雜特徵結構，對儲存空間節省與詞彙分類都有很好的效果。加上以編輯記錄的方式來表示詞彙與其所屬詞類訊息差異的情形，可使得訊息構建自然且容易得多;另外正確性與一致性的維持也可以達成。有關這些處理方式我們已成功的運用在構成ICG詞檔的系統製作,此外我們還加上一些考量使整個系統非常有效率，我們相信這套詞彙訊息表達與管理方法對其它性質相近的聯併語法理論而言，都有進一步參考的價值。

6.参考資料：

1.[黃'88] 黃居仁，聯併：語法理論與剖析，ROCLING'88

2.[陳'89] 陳克健、黃居仁,訊息爲本的格位語法— 一個適用於表達中文的語法模式，ROCLING'89

3.[Gazdar'85]
Gazdar. G.,E. Klein, G.K. Pullum,and I. A. Sag 1985. Generalized Phrase Structure Grammar. Cambridge: Blackwell, and Cambridge, Mass.: Harvard University Press.

4.[Pollard'87]
Pollard,C.and Ivan A. Sag 1987. Information-based Syntax and Semantics Vol.1 Fundamentals, CSLI Lecture Notes Series No. 13. Stanford: CSLI.

5.[Tomita'85]
Tomita,M. 1985, Efficient Parsing for Natural Language, Kluwer Academic Publishers.

6.[Uszkoreit'86]
Uszkoreit, H. 1986. Categorial Unification Grammars. In Proceedings of Coling 1986. Bonn University of Bonn. Also appeared as report No. CSLI-86-66, Stanford: CSLI.

*本論文的研究得到國科會中文語句剖析的語法模式研究計畫(NSC78-0408-E001-01)及中央研究院與工研院電子所合作「中文剖析系統合作研究發展計畫」(X2-79007)之部分經費補助，特此申謝。此一層次維護系統的程式設計員有張孟元、蔡正茂、莊清華及溫佩樺，謝謝他們對本篇論文的幕後貢獻。論文寫作的部分例子取材於中文詞庫小組的研究成果，並感謝詞彙小組對此一層次維護系統所做的實際測試。

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, lexical 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 implementation. 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].

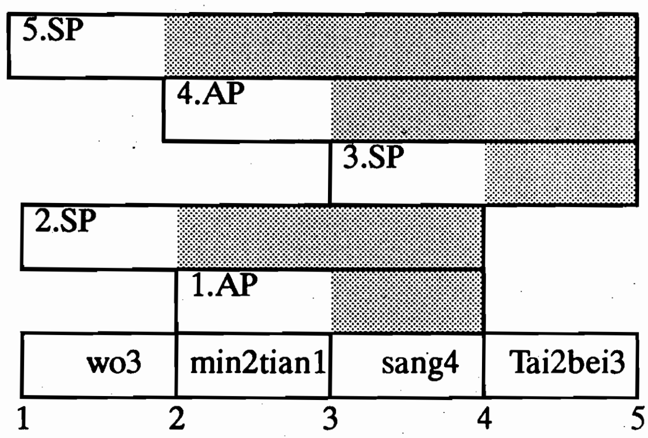
Most chart parsers nowadays are designed for rule-based systems. The basic data structures of these chart parsers are composed of three objects: grammar rules, charts, and input string queue. Cooperated with unification-based grammars, such as HPSG, the data structure of chart parsers can be simplified 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. <u>*song4 de5*</u> *bu4 ru2 zi4ji3 mai3 de5 hao3.*(direct object – goods)

   (What is given free is not as good as what is for sell.)

6. <u>*wo3 song4 de5*</u> *shi4 yi2 ben3 shu1.*(direct object)

   (What I give is a book.)

7. <u>*song4gei3 Li3si4 de4*</u> *shi4 zui4 qui4 de5.*(direct object)

   (What is given to Li3si4 is the most expensive.)

8. <u>*song4 huo4 de5*</u> *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 <u>function–rule–firing mechanism:</u>

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]).

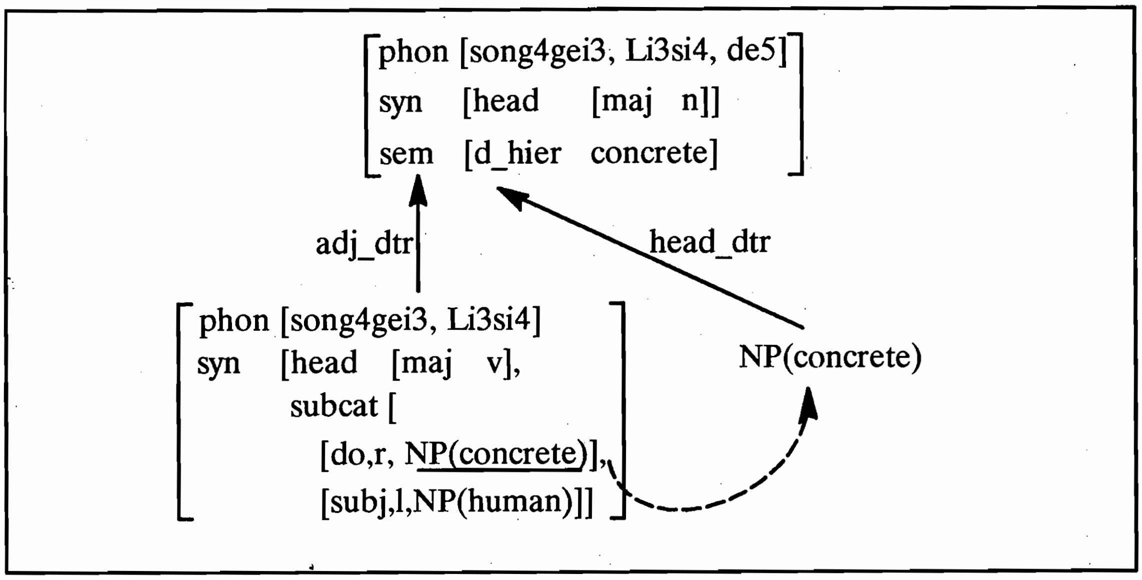


Figure 2. The nominalized noun phrase *song4gei3 Li3si4 de5.*

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.

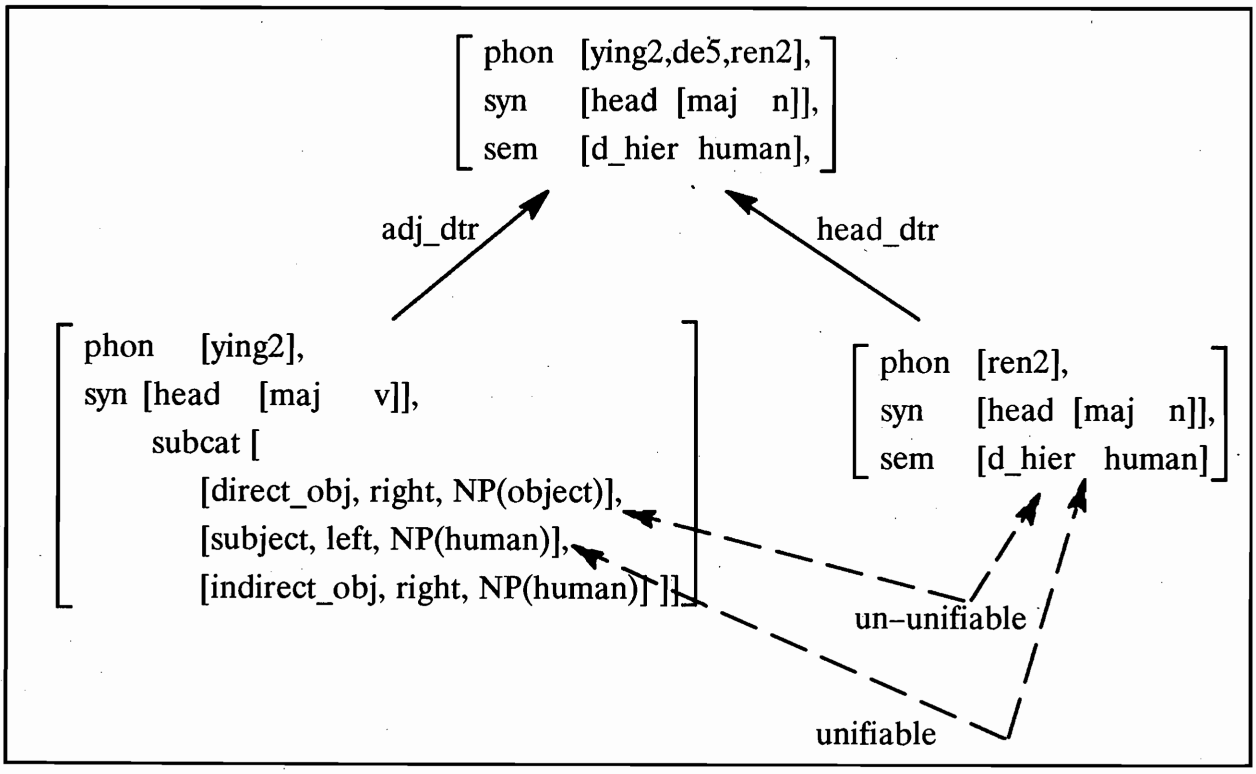


Figure 3. The parsing tree of *ying2 de5 ren2* (the winner).

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. *<u>mai4 gei3 Li3si4 de4 yi1fu2</u> 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 envirohment 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.

Acquisition of Syntactic Properties of English Unknown Words from a Corpus

Tzusheng Pei**, Wei-Chuan Li*, and Tung-Ming Koo*

**Advanced Technology Center
Computer and Communication Research Laboratories (CCRL)
Industrial Technology Research Institute (ITRI)
E000, Blg. 11, 195 Chung Hsing Road, Section 4,
Chutung, Hsinchu, Taiwan, R.O.C.

*Application Software Department, CCRL, ITRI
W300, Blg. 14, 195 Chung Hsing Road, Section 4,
Chutung, Hsinchu, Taiwan, R.O.C.

Abstract

In parsing English sentences, it happens quite often that parsers come across unknown words. A method for obtaining the syntactic properties of unknown English words from a big corpus is presented. To what extend can the syntactic properties of unknown words be obtained is investigated from the view-point of a machine translation system. The acquisition method is based on the contexts of unknown English words in the corpus.

1. Introduction

In parsing natural languages, it happens quite often that parsers come across words which are not listed in the system dictionary. Little can the syntactic properties of the unknown words be acquired if there is one input sentence only. Suppose there is a big corpus, and all sentences in the corpus are syntactically correct. Normally the corpus covers a large amount of sentences in a specific area. For each unknown word, there is no or at least one sentence in the corpus containing that unknown word. For example, a corpus which consists of the first four volumes of UNIX operating system manuals may be a good one. The corpus covers most words widely used in the field of operating system, where some of the words, especially some computer terminologies , may not be listed in the dictionary of a specific machine translation system. The syntax of the source language [BERWICK85] [FASS89] [RUQIAN89] and semantic knowledge of unknown words [BERWICK89] [VELARDI89] can, to some degree, be obtained from the various contexts of the words in the corpus. In this paper, the focus will be on the syntactic properties of unknown words, and the acquisition process will be on a practical machine translation system. In general, the acquisition method can be used in or adapted to other practical machine translation systems. The dictionary can be expanded based on the information obtained. The syntactic properties obtained in this way are subject to review by people before they are keyed into the system dictionary.

The English-Chinese machine translation system, CCRL-ECMT, formally ERSO-ECMT, has been developed in the Electronics Research and Service Organization (ERSO), ITRI, since July 1987. At present, the project is conducted in Computer and Communication Research Laboratories (CCRL), ITRI. In the machine translation system, an input sentence is preprocessed by the morphological module before being analyzed by the parser. Since there is no entry for unknown words in the dictionary, the morphological module can merely do primitive analysis for unknown words, such as eliminating "d" or "ed" for regular verbs, restoring the ending "ing" to "e" or simply eliminating the ending "ing", and eliminating the ending "s" or "es". Even such primitive analysis can not be done satisfactorily. The information obtained from the morphological processing will be used in later stages. The parser is based on Tomita's parsing algorithm [TOMITA86] with augmented syntactic and semantic tests under the context-free grammar rules. A contex-free rule in the system grammar is applied to construction of parsing trees when the parsing status passes the tests of the rule [TANG88]. The parsing trees constructed by the parser for an English sentence are all syntax trees which already have met the restrictions the tests impose upon.

The possible contexts of a word are the various situations, many of them recorded in the system dictionary, the word may be in. In general, the contexts of a word include information in dictionaries and acceptable sentences, such as semantic markers, verb-types, the grammar rules applied to the current segment of the input sentence containing the word, and the actual segments in acceptable sentences covering the word. In principle, the system dictionary should have abstracts of all possible syntactic and semantic contexts of words in sentences.

In the machine translation system, CCRL-ECMT, the relevant syntactic properties to be determined are as follows:

The first thing to do is to determine the part-of-speech of unknown words in each context. The part-of-speeches in the system are adjective, adverb, do-modal, article, be-verb, conjunction, have-verb, modal, noun, preposition, relative pronoun, "to" for infinitive, verb (other than be-verb and have-verb), subordinate conjunction, wh-word, wh-word as noun, and wh-word as conjunction. The verbs are further classified into 25 categories as shown in the Advanced Learner's Dictionary of Current English [HORNBY63]. A verb might be of several categories. Some categories of verbs may have prepositions, called particles, closely related to them. A verb in these categories and its related preposition normally appear at relative syntactic places in sentences at the same time. The entry of a verb in those categories should contain the related preposition(s). The verb and the related preposition(s) can be used as a restriction or a guide for parsing. The following patterns show the verb types of 18 and 24 .

VP18: Verb18 + NP + PP,
VP24: Verb24 + PP.

The prepositions for these two verb types should be put in the entries of the verbs of type 18 or 24 in the system dictionary.

For example, "I sold the book to my friend.", "sell" and "to" are closely related in this context. In the sentence, "He succeeded in taking the examination.", "succeed" and "in" are used together. "to" and "in" should be included in the entries for "sell" and "succeed" respectively in the system dictionary.

For some adjectives, there are also particles closely related to them. An adjective of this usage should be used together with its related preposition, so the entry of the adjective should have the related preposition. For example, the preposition "to" normally follow the adjective "available" as in the phrase "available to everyone".

Some nouns have the same property of the adjectives mentioned above. A noun of this category may have a preposition following right after it. For example, ticket "to" Paris, introduction "to" natural language understanding, relation "between" the book and the orange. Depending on the number of English usages the dictionary should reflect, it may contain some more properties of words. The properties show the relations between the word and its various possible contexts.

In practice, it is more convenient to look up a comprehensive dictionary to get syntactic properties of the word unknown to the system than to use this acquisition method, but actually it is not the case. There are many words and/or their usages not covered in any dictinary, and the words and/or their usages are used quite often by people, especially in specific areas, and, furthermore, the problem itself is interesting. Therefore, automation or semi-automation of syntactic knowledge acquisition is still worth study.

2. Determination of the part-of-speeches of an unknown word

Suppose that the unknown word could be any part-of-speech in the grammar, and then the parser tries all part-of-speeches for the unknown word to construct parsing trees for an input sentence containing that word. For each possible syntax, a parsing tree comes out. A sentence may have more than one parsing tree. The parser analyzes the syntax of all sentences containing the unknown word in the corpus. In fact, the unknown word does not necessarily have that many part-of-speeches. For some assumed part-of-speeches, the parser might fail to construct a parsing tree for the sentence. In this case, the word does not have these part-of-speeches.

In general, each parsing tree the parser has constructed does not necessarily correspond to an acceptable interpretation, because the syntactic structure might not be acceptable to people. Since the word is unknown, there is no way to judge the correctness of the sentence by the semantic information of the unknown word, but the semantic information of the context of the unknown word together with the part-of-speech of it is still useful for making judgement. The final judgement is always the job of people, not the computer system. Furthermore the etymological information of the unknown word, which can not be easily represented and processed on computer, is usually very important for making judgement. Actually, for some cases, the part-of-speech shown in a parsing tree is not a right one for that word. The possible reasons for this are some flaws of the context-free grammar of the parsing system which allows nonexistent syntax, even nonexistent usages, etc. Therefore the actual part-of-speeches of the unknown word will be subset of all the part-of-speeches of the unknown word in the parsing trees the parser has constructed from the sentences containing the unknown word in the corpus.

Suppose that there are N sentences in the corpus containing the unknown word W, and there are M part-of-speeches in the grammar. After having parsed the N sentences, a table is set up. The entry (i, j) of the table is 1 or a blank, where 1 indicates that the unknown word in the j sentence has the part-of-speech i, and a blank hasn't. A column which has two 1s in it indicates that the unknown word has two part-of-speeches and for each part-of-speech at least one parsing tree can be constructed. Each element of the last column is the result of the logical OR operation of all the elements to the left of it. This column shows the set of the possible part-of-speech(s) of the unknown word. An 1 at row i of the last column indicates that the word could have the part-of-speech i. The set of part-of-speeches indicated in the last column is defined as the maximum set.

The determination of part-of-speech(s) of an unknown word is not done alone. The other further properties, which will be discussed in the next section, are also acquired at the same time from the parsing trees constructed by the parser. All candidates for particles will be tested, and the verbs are divided into 25 types to be tested for each type.

Let's see an example. Suppose that there are 5 sentences in the corpus containing the unknown word W, and there are 5 possible part-of-speeches: n (noun), v (verb), a (adjective), z (adverb), p (preposition).

Sentence number \ Class	1	2	3	4	5	OR
1: n	1	1				1
2: v				1		1
3: a						
4: z	1					1
5: p					1	1

The last column is the logical OR operation of the preceding five columns. It shows that the part-of-speech of the unknown word in a sentence could be n, v, z, or p. For actual cases, the verb will be divided into 25 types, and there are various particles. Those usages are considered different and treated separately. In fact, this method gets more part-of-speeches than what the unknown word could have.

The approaching algorithm, which operates on the table mentioned above, is in the following. It picks a subset of the maximum set of part-of-speeches shown in the last column of the table, so, with the part-of-speeches in the subset for the unknown word, at least one parsing tree can be constructed for each sentence in the corpus containing the unknown word. It is based on the intuition that, in this way of selection, the subset, called minimum-covering set, will be as small as possible. A part-of-speech is said to cover a sentence with an unknown word if at least one parsing tree can be constructed for the sentence with the part-of-speech assigned to the unknown word. A minimum set is defined as a subset of the maximum set which has minimum number of part-of-speeches to cover all sentences in the corpus.

The procedure of the algorithm for obtaining minimum-covering set is as follows:

1). For sentences with only one possible part-of-speech for the unknown word, the part-of-speech should be selected to cover the sentence.

2). The part-of-speech which covers as many sentences as possible, excluding the sentences already covered, is selected.

3). Then select the second one and on until the part-of-speeches collected cover all sentences in the corpus containing the unknown word.

Some rarely used part-of-speeches might be lost in this process. In general, the difference between the number of elements of a minimum-covering set and that of a minimum set is small. Depending on the order of the part-of-speeches being selected, there could be more than one minimum-covering set and more than one minimum set also. Even the numbers of elements of two minimum-covering sets could be unequal.

Up to this point, the maximum set and a minimum-covering set have been obtained. The actual part-of-speeches of the unknown word should contain the minimum-covering set and possibly the whole maximum set. The set difference between the maximum set and the minimum-covering set is interpreted as the set of part-of-speeches rarely used from the view-point of the parsing system. Some of the rarely used part-of-speech(s) might not be acceptable to people.

The useful information acquired for review by people is as follows:

1. The unknown word.

2. The maximum set and the minimum-covering set.

3. A list of the related nodes of the parsing tree, related particle, if any, and the related segment in the sentence for each usage of each part-of-speech in the maximum set. For example, the related nodes for verb type 18 should include Verb18, NP, PP, and the related preposition is also listed.

The same usage of a word in the corpus can be combined. The number of sentences in the corpus showing the same usage is also a useful information. It reflects the frequency of the usage of the word in the corpus. If the corpus is big enough, it actually reflects the frequency of the usage being used by people.

All information obtained along with the parsing system should be reviewed by people before they are keyed into the system dictionary.

3. Further syntactic properties

For further detailed syntactic properties mentioned above, the algorithms for dealing with the properties look for the particles (prepositions) which follow the target syntactic elements, such as a verb, an adjective, and a noun, in input sentences. The parser in the system can be easily modified to do this. The acquisition of the detailed properties and determination of part-of-speech are done at the same time.

The algorithms for the various VP-types are as follows:

The parser tries all verb types in the system grammar to construct parsing trees. The templates from the verb type definition below are used for parser to identify the particles.

VP18: Verb18 + NP + PP,
VP24: Verb24 + PP.

The assumed preposition is treated as a particle, and the parser tries to construct parsing trees. If at least one tree comes out, then the preposition could be a particle. Particles for adjectives and nouns are identified in the same way. Actually, not all particles obtained are acceptable to people. The particles should be reviewed by people before they can be keyed into the system dictionary.

4. Discussion and Perspectives

In order to be meaningful, the corpus should be big enough. It means that unknown words appear once in the corpus should also appear in various contexts in sentences of the corpus that reflect the words' part-of-speech(s) and common usages. The proportion among all usages in the corpus reflects the relative frequencies of the usages of the unknown word. This is the idea case. For a practical corpus in general, it might not contain that many sentences which reflect all usages of the word. From the practical point of view, the processing time for acquisition, mainly the total parsing time, for unknown words is the only limit to the size of the corpus. No statistics has been suggested, since it makes sense only when there is a very big corpus which covers enough sentences containing unknown words.

Here only a few syntactic properties have been investigated. More properties can be incorporated into the system. A lot of further work, such as acquisition of syntax of the source language and semantic information of unknown words, can be done on the parsing system.

In general, the reason of failing to construct a parsing tree for a sentence is that the sentence is ungrammatical from the view-point of the system grammar or one of the word has a new usage not recorded in the system dictionary or there is at least one unknown word in the sentence. In the second case, the acquisition process can not do anything, since there is no way to determine which word has the new usage being used.

Not all results of the acquisition are right, as indicated before, so they should be reviewed by people. Further experiment may indicate the degree the process can be automated and clarify some problems of automatic acquisition. In practice, many unknown words are nouns. Therefore, to set all unknown words as nouns for doing machine translation may be a good policy.

The system grammar is not perfect. It is possible that there are some correct syntax not covered in the grammar, so some usages of the unknown words might not be reflected in the parsing trees constructed. It means that the parser fails to construct parsing trees for those usages. Actually, it is hard or even not possible to formulate a context free grammar which covers all acceptable English sentences.

There is an interesting phenomenon in English. In general, noun and preposition are not likely to be the part-of-speeches of a word at the same time, but, for noun and adjective pair, the likelihood is much higher. In this case, it is said that the interference between adjective and noun is high. Actually, for a practical machine translation system all nouns can be treated as adjectives. For some part-of-speech pairs, the interference among them is very small or negligible. For example, a word which can be used as a preposition is generally not to be used as other part-of-speeches. So, for some part-of-speeches, they can be eliminated in the syntactic contexts by the parser when a grammar rule is applied to construct a syntactic structure.

5. Acknowledgement

The paper is a partial result of the project No. 3131500 conducted by ITRI under sponsorship of the Ministry of Economic Affairs, R.O.C.

6. References

[BERWICK85] Robert C. Berwick, "The Acquisition of Syntactic Knowledge", The MIT Press, 1985

[BERWICK89] Robert C. Berwick, "Learning Word Meanings from Examples", Chaper three of "Semantic Structures", Edited by David Waltz, Lawrence Erlbaum Associates, Inc., Publisher, 1989, pp. 89-124.

[FASS89] Leona F. Fass, "Learnability of CFLs: Inferring Syntactic Models from Constituent Structure", SIGART Newsletter, April 1989, Number 108, Knowledge Acquisition Special Issue, pp. 175-176

[HORNBY63] A. S. Hornby, E. V. Gatenby, H. Wakefield, Oxford Advanced Learner's Dictionary of Current English, 2nd Edition, 1963

[RUQIAN89] Lu Ruqian, Liu Yinghui, Li Xiaobin, "Computer-Aided Grammar Acquisition in the Chinese Understanding System CUSAGA", IJCAI-1989, pp. 1550-1555

[TANG88] An-Ching Tang, Tzu-Cheng Huang, "Tomita 增強型 LR 剖析器在機器翻譯系統的實際應用",ROCLING I, 1988, pp. 241-256

[TOMITA86] M. Tomita, "Efficient Parsing for Natural Language", Kluwer Academic Publishers, 1986

[VELARDI89] Paola Velardi, Maria Teresa Pazienza, Stefano Magrini, "Acquisition of Semantic Patterns from a Natural Corpus of Texts", SIGART Newsletter, April 1989, Number 108, Knowledge Acquisition Special Issue, pp. 115-123

Parsing-Driven Generalization for Natural Language Acquisition

Rey-Long Liu & Van-Wun Soo

劉瑞瓏 蘇豐文

Institute of Computer Science

National Tsing Hua University

Proceedings of ROCLING III

中華民國第二屆計算語言學研討會論文集351～376頁

Parsing-Driven Generalization for Natural Language Acquisition

Rey-Long Liu and Von-Wun Soo

Institute of Computer Science

National Tsing-Hua University

ABSTRACT

Parsing is an important step in natural language processing. It involves tasks of searching for applicable grammatical rules which can transform natural language sentences into their corresponding parse trees. Therefore parsing can be viewed as problem solving. From this point of view, language acquisition can be generalized from problem solving heuristics. In this paper we show how learning methods can be incorporated into a wait-and-see parser (WASP), the problem solver. We call this approach parsing-driven generalization since learning (acquisition of parsing rules and classification of lexicons) is basically derived from the parsing process.

Three generalization methods are reported in this paper: a simple generalization mechanism, a mechanism of generalization by asking questions, and a mechanism of generalization back-propagations. The simple generalization mechanism generalizes from any two parsing rules whose action parts (right-hand sides) are the same while whose condition parts (left-hand sides) have a single difference. The mechanism of generalization by asking questions is triggered when a "climbing-up" move on a concept hierarchical tree is attempted and is necessary in avoidance of over-generalizations. The generalization back-propagation mechanism is to propagate a confirmed generalization of some later parsing rule back to its precedent rules in a parsing sequence and thus causes them to be generalized as well. This mechanism can save many questions to be asked. With the three generalization methods and a mechanism to maintain lexicon classification (the domain concept hierarchy), we have been able to show a plausible natural language acquisition model.

1. Introduction

Natural language acquisition has been an interesting and challenging research topic for both psycho-linguistists and computer scientists, although the former might be more interested in studying on human subjects while the latter on man-made machines. To understand how a natural language can be acquired by a machine can benefits from both studies. The psycho-linguistical study might be able to reveal many clues, constraints, and limitations regarding to natural language acquisition tasks that human actually face. These revelations may suggest us many guidelines for building computer systems that can automatically acquire a natural language. Our purpose in this paper, however, is not to show psycho-linguistic evidence on human natural language acquisition, rather is to build a computer model using machine learning techniques from artificial intelligence study [7] to demonstrate the feasibilities of natural language acquisition on machines.

There are many reasons to study natural language acquisition on a machine. One among them is to remove the current difficulty of having to construct and maintain a large set of lexicons and grammatical parsing rules in a complex natural language processing system. Since human natural language is ever-growing and evolutionary in nature, no natural language processing system with a complete set of both lexicons and grammatical rules can possibly be built. One solution to this is to seek ways of incrementally acquiring grammatical rules from training examples of sentences while keeping the coherence of the system. To achieve this, the system must have the capability of performing generalization over training examples. Traditional artificial intelligence researches have developed many techniques to perform generalizations [7].

Parsing involves tasks of searching for applicable grammatical rules which can transform natural language sentences into their corresponding parse trees. Therefore a successful parsing sequence can be treated as a solution to a parsing problem. By viewing natural language parsing as problem solving, the language acquisition tasks can be achieved to some extent by generalizing from problem solving heuristics. We call this approach parsing-driven generalization since learning (acquisition of parsing rules and classification of lexicons) is basically derived from the parsing process.

1.1 Related Works

Berwick [3] uses word features to perform conservative generalizations (the Subset Principle) but leaves their acquisition as an open problem. The acquisition of semantic and syntactic features of lexicons is important for the language development of children (Selfridge [14], Pustejovsky and Bergler [12], Berwick [2] and Zernik [21]). In the system, lexicons are classified in a concept description hierarchy. However, such a concept hierarchy is not built priorly. Instead, it is constructed during the learning process. This contrasts to Mitchell's version space approach (Mitchell [8]), where the domain concept hierarchy must be built before a generalization/specialization process can be carried out.

It has been argued by many researchers (Pinker [11], Wexler and Culicover [16], and Berwick [3]) that no negative examples are necessary in a language acquisition situation. That is, children usually acquire a language by imitating from parents' conversation. It is rare the case that parents teach illegal sentences to their children. However, children do sometimes generate illegal sentences due to overgeneralization from incomplete language acquisition. To avoid overgeneralization, it is necessary for a language acquisition system to be able to ask questions before performing generalization. This is similar to MARVIN's approach of learning (Sammut and Banerji [13]). The generalizer performs experiments by generating sentences and testing their validity by asking the trainer. When the system responses an invalid sentence (might be syntactically invalid such as *He eat an apple,* or might be semantically invalid such as *An apple runs),* the possible generalizations are prohibited. This generalization method shifts the burden of generating negative examples from trainers to learners. Trainers do not have to care about the current state of learners.

Traditional EBL (Utgoff [15], Ellman [4] and Mitchell et. al. [9] [10]) uses high-level domain knowledge to guide correct generalizations. Since parsing is viewed as problem solving, the powerful learning methods such as the Explanation-Based Learning (EBL) can be incorporated into the language acquisition system. In language acquisition, Zernik [20] and Zernik [19] employ the EBL method to acquire phrases and idioms respectively. However, it seems to be impossible to construct a complete knowledge base to perform EBL (Yu [18]). In our system, no such knowledge base is assumed. Rather, we incorporate the "back-propagation" concept of the EBL into the generalizer. The generalizations of the last fired rule are back-propagated when they are confirmed by the trainer.

2. The Parsing Device

The parsing device of this language acquisition system is based on the Wait-And-See strategy.

2.1 The Wait-And-See Strategy

Marcus [6] proposed the Wait-And-See strategy to parse natural languages. It is based on a "determinism hypothesis" which says that natural languages can be parsed by a computationally simple mechanism without backtracking. A Wait-And-See Parser (WASP) is like a production system, where the grammar and parsing heuristics are expressed in terms of rules (parsing rules) which are composed of condition and action parts. Two major data structures are required:

1. active node stack: a pushdown stack of incomplete constituents, and
2. lookahead buffer: a small constituent buffer containing constituents which are complete, but whose higher grammatical function is as yet uncertain.

The rules in a WASP are partitioned into rule packets. Each rule packet contains rules for a particular configuration of the constituent in the top of the node stack. For example, if the top of the node stack contains a VP, the corresponding rule packet for the VP is activated. However, the selection of rules to fire may depend on the contents of the lookahead buffer and the node stack.

The action operators in a WASP defined by Marcus [6] are:

1. ATTACH: attach the constituent at the top of the buffer stack (X) to the top of the node stack (Y). X is popped from the buffer and becomes the rightmost daughter of Y,
2. DROP: Y is popped from the node stack and pushed onto the buffer stack, and
3. CREATE: push a new active node onto the node stack.

More detailed descriptions for a WASP can be found in Liu and Soo [5] and Marcus [6].

2.2 Extending the WASP

In fact, the creation of an S, an NP, a VP, and a PP may be delayed until its first component is parsed and dropped. At that time, the creation action can be deterministically followed by an ATTACH action which attaches this parsed and dropped component. For example, consider the sentence *I ate an apple*. At the beginning, before creating an S node, the parser might first create an NP node which will attach the noun *I* and then be dropped to the buffer stack. Then, an S node is created, and the parser automatically attach this NP as its first son. Thus, we have an improved version of the action CREATE:

CREATE: push a new active node onto the node stack, **and** perform the ATTACH action.

This improvement has two advantages:

1. when the first component is parsed and dropped, the parser can lookahead more information, and

2. since the creation action is followed by an attach action, the total number of actions for transforming the input sentence to its parse tree is reduced.

For example, consider again the above sentence *I ate an apple*. If the traditional mechanism (as described in the above section) is employed, there are 15 actions in the solution path:

((CREATE S) (CREATE NP) (ATTACH) (DROP) (ATTACH)
(CREATE VP) (ATTACH) (CREATE NP) (ATTACH) (ATTACH)
(DROP) (ATTACH) (DROP) (ATTACH) (DROP)).

However, if the improved mechanism is employed, there are only 11 actions in the solution path:

((CREATE NP) (DROP) (CREATE S) (CREATE VP) (CREATE NP)
(ATTACH) (DROP) (ATTACH) (DROP) (ATTACH) (DROP)).

This will certainly reduce the number of acquired rules.

Since the rules in a WASP are uniform and suitable for language acquisition systems, it is adopted as the parsing device of the learning system. The parsing operators (ATTACH, DROP and CREATE) become the key action operators in the system. Each acquired rule contains one (and only one) of these operators as its action part.

3. The Learning Module

Fig.1 shows the flowchart of the language acquisition system. After accepting an input sentence and its corresponding parse tree, the system initially sets the node stack to be empty, fills the cells in the lookahead buffer (the number of cells of the lookahead buffer is discussed in later sections), and finds the solution path (sequence of actions that transforms the input sentence to its corresponding parse tree) deterministically.

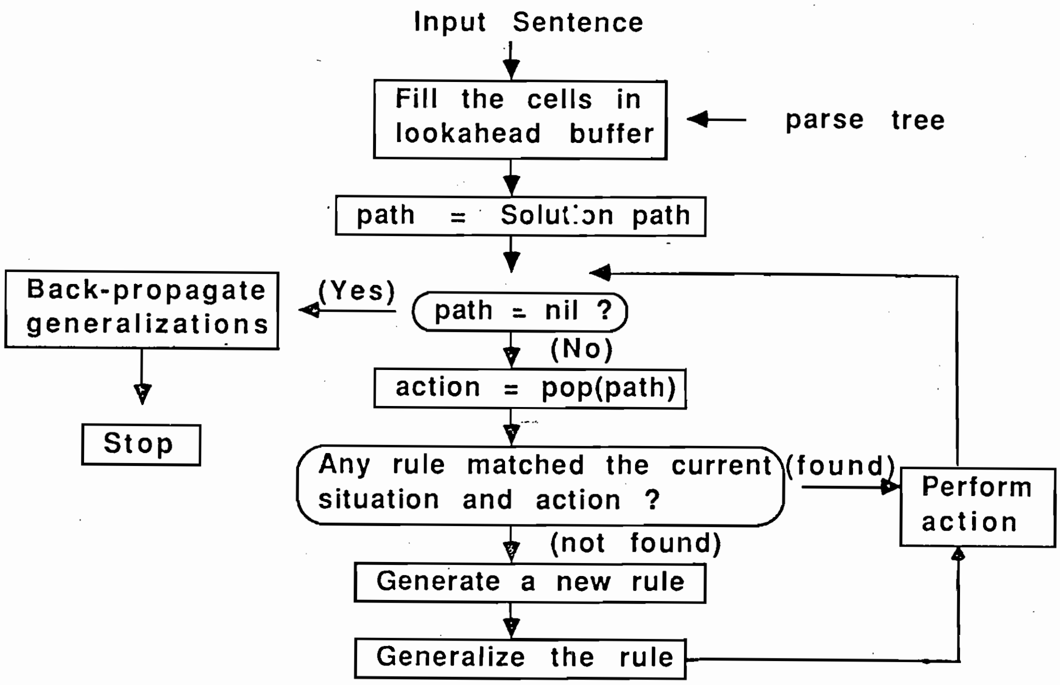


Fig. 1. The overview of the system.

Then, it iteratively pops up the first action in the solution path, tries to find a rule whose RHS (Right Hand Side) matches the action, and LHS (Left Hand Side) condition matches the current configuration of the node stack and the lookahead buffer. If a rule is found, it is fired, otherwise, a new rule is generated according to the current configuration of the node stack and the lookahead buffer (as the LHS of this rule) and the action (as the RHS of this rule). After trying to generalize this new rule, the action is performed, and the iterative process continues until all actions in the solution path are examined. At this time, the system will back-propagate the generalizations from the later acquired or fired rules to the previous acquired rules. Thus, the previous acquired rules are further generalized without causing over-generalizations.

3.1 The input

The semantic bootstraping hypothesis, which is employed in many language acquisition systems (Pinker [11] and Berwick [3]), states that when speaking to a child, parents refer to physical objects using nouns, and actions, which cause the change of the states, using verbs. Thus, although the child might not initially know what the grammatical categories (such as noun, verb, ... etc.) are in the target language, he or she might also learn them according to the syntax-semantic correspondences. Berwick [3] uses the thematic representations for these correspondences, and in Anderson [1], they are encoded as associated networks which describe the scene of the input sentence. Thus, the trainer must provide not only the training sentences but also their corresponding semantics. In our language acquisition model, the additional information is expressed in the parse trees. For example, for the input sentence *I eat an apple,* the corresponding parse tree is also input to the system:

(S (NP (N I)) (VP (V eat) (NP (DET an) (N apple)))).

In fact, the parse tree provides the learner much information including the categories of words and the structures of phrases. The provision of the category information is also a simplification made by Wexler and Culicover [16]. And as Pinker [11] pointed out, it is possible to derive from the parse tree the lexical entries and the phrase structure rules that generate the input sentence. However, since the derived phrase structure rules are too general, they might sometimes generate syntactically and semantically illegal sentences. What the system wants to acquire are the critical features of words in parsing situations to construct the correct parsing rules without causing over-generalizations. The derivations of general phrase structural rules are not helpful in our problem domain.

Also, from the viewpoint of problem solving, the input sentence can be treated as the initial state, while its corresponding parse tree as the goal state. By properly defining parsing operators (ATTACH, DROP, ... etc.) and representing the parse tree, we can design an algorithm to deterministically detect the whole solution path (the solution path can transform the initial state to the goal state) without resorting to searching or prior domain knowledge. Thus, the given parse tree is not only necessary (in the sense that it represents the syntax-semantic correspondences), but also powerful for the language acquisition system.

3.2 The Determination of a Solution Path

We view language parsing as a problem-solving task which is to transform the input sentence (the initial state) to its corresponding parse tree (the goal state). The LEX system (Mitchell et. al. [10] and Utgoff [15]) formulates the solving of the calculus as a search problem. After a solution is found, specializations and generalizations are then conducted to enhance the performance of the problem solver. The similar idea is used in our language acquisition system. However, by using the input parse tree and the parsing device described above, we can easily determine a solution path without resorting to searching the entire solution space or relying on the high-level domain knowledge (such as the X-bar theory used in Berwick [3]). For example, consider the sentence *I know the beautiful girl* and its parse tree (S (NP (N I)) (VP (V know) (NP (DET the) (ADJ beautiful) (N girl)))). A corresponding solution path can be easily determined (recall the parsing device described in section 2.2)

> ((CREATE NP) (DROP) (CREATE S) (CREATE VP) (CREATE NP) (ATTACH) (ATTACH) (DROP) (ATTACH) (DROP) (ATTACH) (DROP)).

The determination algorithm scans the parse tree from left to right. When a right parenthesis is encountered, a DROP operator is output, and when a left parenthesis is encountered, a CREATE operator should be output. However, this output of CREATE operator is delayed until its first component is parsed and dropped (section 2.2). The algorithm is as follows:

DetectAction (P)

Input: the parse tree P which is represented as a list.

Output: the solution path for constructing P.

Algorithm:

1. Let C = Car(P); D = Cdr(P).
2. If C is S, NP, VP, PP, ADJP, or ADVP then

2.1. Let CC = (pop D).

2.2. DetectAction (CC).

2.3. Output the action CREATE C.

2.4. For each element CC in D do

  2.4.1. DetectAction (CC).

  2.4.2. Output the action ATTACH.

2.5. Output the action DROP.

3. Return.

It should be noted that, when the first element (C) of the input tree (P) is not a phrase such as an S (Sentence), NP (Noun Phrase) , VP (Verb Phrase), PP (Prepositional Phrase), ADJP (Adjective Phrase), or ADVP (Adverbial Phrase), no action is output for constructing this element. For example, in step 2.4.1, when CC is "(N I)", the recursive call "DetectAction (CC)" produces nothing, and after this call, an ATTACH action is output in step 2.4.2. The structure "(N I)" only gives category information 'N' to the word 'I'. Since it is not a phrase, no actions are needed to construct it. The parser simply attaches it to the constituent at the top of the node stack. It should also be noted that, by the definition of the action CREATE in section 2.2, when a constituent C is created (step 2.3), it will automatically attach its first son (the constituent CC in step 2.2).

3.3 Generalizations

There are three types of generalizations in the system: simple generalizations, generalization by asking questions, and generalization back-propagations. We will describe them subsequently in this section.

3.3.1 Simple Generalizations

When a new rule is generated, the system will try to generalize it to its largest extent according to the current concept description. Initially, the concept description (the lexicon) is empty as shown in Fig.2.a. When training examples are provided, the concept description will accumulate the features of each word. For example, after the first sentence *I see the man* and its parse tree (S (NP (N I)) (VP (V see) (NP (DET the) (N man)))) are entered, there are 11 actions detected in its solution path, and thus 11 specific rules (one action for each rule) are acquired. Fig.2.b shows the current concept description.

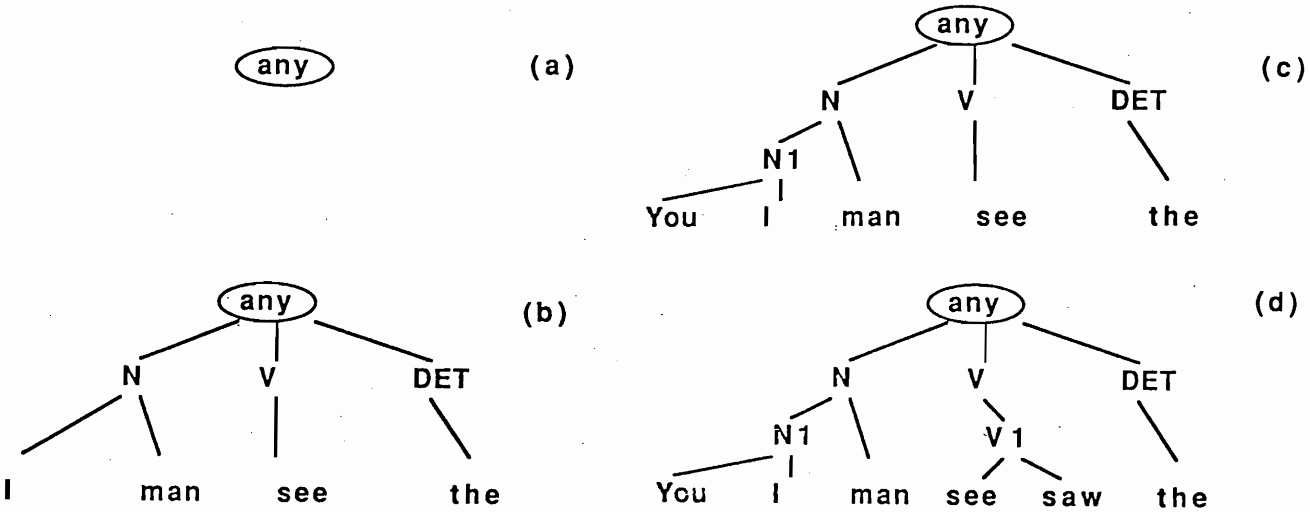


Fig. 2. The concept description.

Since the conception description currently has only specific information, no generalizations are possible in this case. When another sentence *You see the man* is entered, there is only one difference between the previously acquired rule in Fig.3.a and the newly generated rule in Fig.3.b. These two rules are deterministically generalized to the rule shown in Fig.3.c. Fig.2.c illustrates the current concept description (N1 is the acquired general node). Next, suppose the sentence *I saw the man* is entered, the generalizer will generalize from both the rule shown in Fig.3.c and the newly generated rule shown in Fig.3.d. Since there exists a difference (*see* and *saw*) and a more-general relationship (N1 is more general than I) between them, a question *You saw the man* will be asked to ensure the validity of the generalization shown in Fig.3.e. In this case, since this generated sentence is a valid one (confirmed by the trainer), the rule in

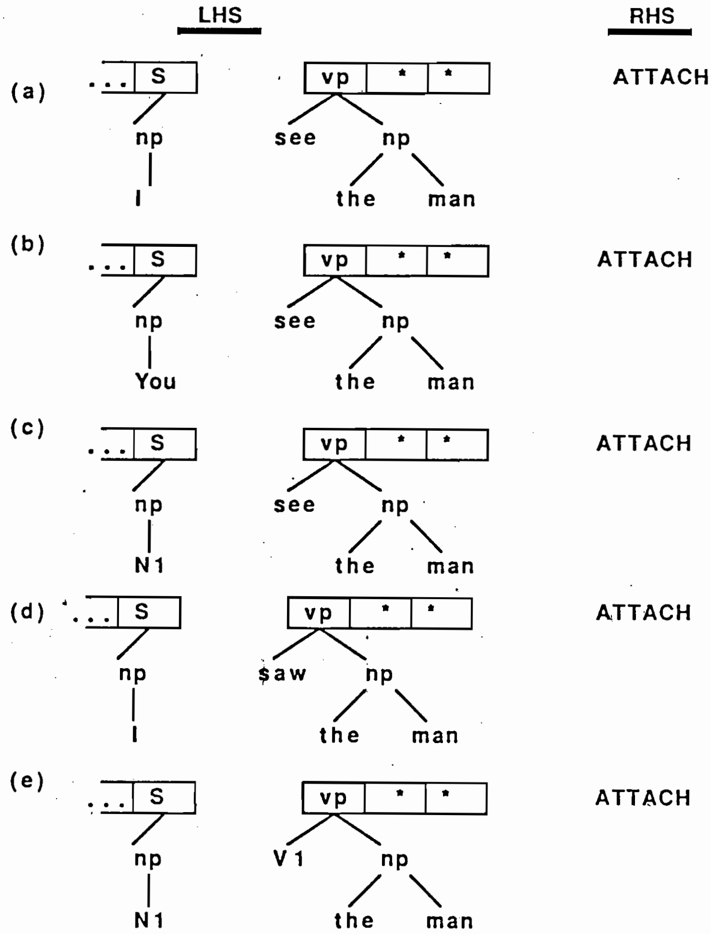


Fig. 3. Generalizations between two rules.

Fig.3.c and the rule in Fig.3.d are generalized to the rule in Fig.3.e. The conception description is also updated as shown in Fig.2.d (V1 is the newly acquired general node). The algorithm of this simple generalization mechanism is as follows:

SimpleGeneralization (R K)

Input: the new rule R and the activated rule packet K.

Output: the resulting rule after generalizing R.

Algorithm:

1. In the rule packet K, find a rule T which has the same action with R's, and there is only one difference between their LHSs.

   If found then

   2.1. If a more-general relationship is possible, ask a question to justify the generalization.

2.1.1. If the answer is "Yes" then

2.1.1.1. Generalize R and T. Denote the resulting rule by P.

2.1.1.2. Remove T from the rule packet K.

2.1.1.3. Return SimpleGeneralization (P K).

2.1.2. Otherwise, go to step 1 to find another rule.

2.2. Otherwise,

2.2.1. Generalize rule R and T. Denote the resulting rule by P.

2.2.2. Remove rule T from the rule packet K.

2.2.3. Return SimpleGeneralization (P K).

3. Otherwise, return R.

The above simple generalizations have the following features:

1. Generalizations are possible only when the action parts of the two rules are identical and there is at most one difference between their LHSs. If there are more-general relationships between them, questions must be asked to justify the generalizations.

2. When the asked sentences are indeed syntactically invalid (such as *He eat an apple*) or semantically invalid (such as *The apple runs),* the generalizations will be prohibited. With the capability to ask questions, it shifts the burden of generating near-miss (Winston [17]) examples from the trainer to the learner.

3. If a rule is successfully generalized to a new rule, this new rule will be generalized again by the same simple generalization mechanism.

4. There can be no over-generalizations. Each generalization is carefully-justified.

3.3.2 Generalizations by Asking Questions

Another type of generalization is to climb up the existing concept description hierarchy by asking questions. It is performed based on a single rule. For example, after the sentence *I give the man an apple* is processed, the concept description becomes a hierarchy as shown in Fig.4.a. Fig.4.b shows an acquired rule that can not be generalized by the above methods (since there are more than one differences between this new rule and the old ones).

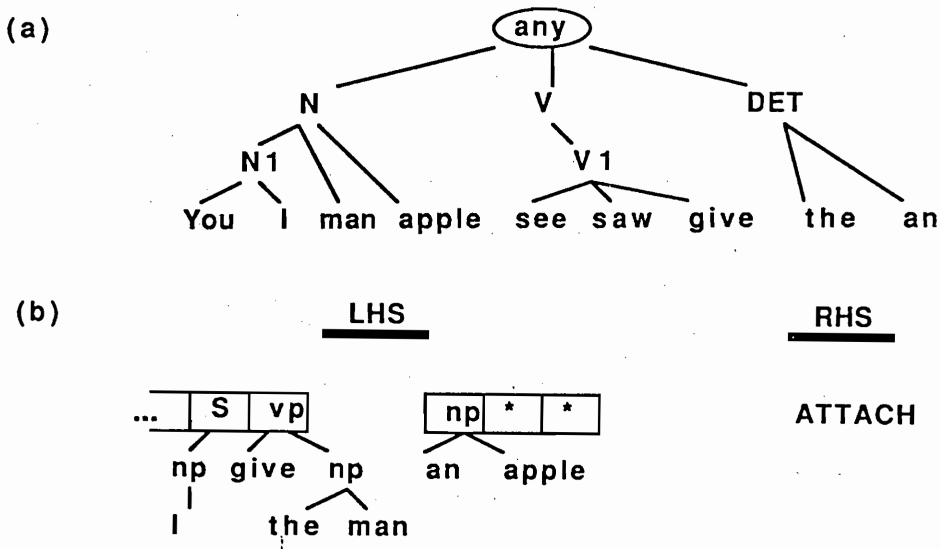


Fig. 4. Generalizations based on single rule.

The generalizer tries to generalize it by climbing the concept description hierarchy. Here, *see* and *saw* must be tested, and thus two questions are asked: *I see the man an apple* and *I saw the man an apple* which are all ungrammatical ones (since the verb *give* is dative, but the verb *see* and *saw* are not). The trainer answers "no" to the system, and thus, no generalizations are performed. The concept description remains unchanged.

Note that the learning system does not know what the newly generated nodes actually "mean". For example, syntactically, there may be Subject-Verb agreements between N1 and V1 in Fig.4.a, and semantically, the system can distinguish *apple* (eatable) from *desk* (un-eatable) by accepting *eat an apple* and not accepting *eat a desk.*

3.3.3 Generalization Back-Propagations

The above generalization by asking question mechanism, employed to climb the concept hierarchy, is very powerful. However, if the generalizer is allowed to climb the concept hierarchy each time when a new rule is generated, too many questions will be asked. The last type of generalizations is introduced to release this difficulty. It is performed when all necessary rules for parsing the current input sentence are fired. The system will generalize the latest fired rule by climbing the concept hierarchy, and then back-propagate the results of this generalization to previously fired rules. Consider the foilowing example: *I give the man I like an apple.* Assume that the current concept description is shown in Fig.5.a.

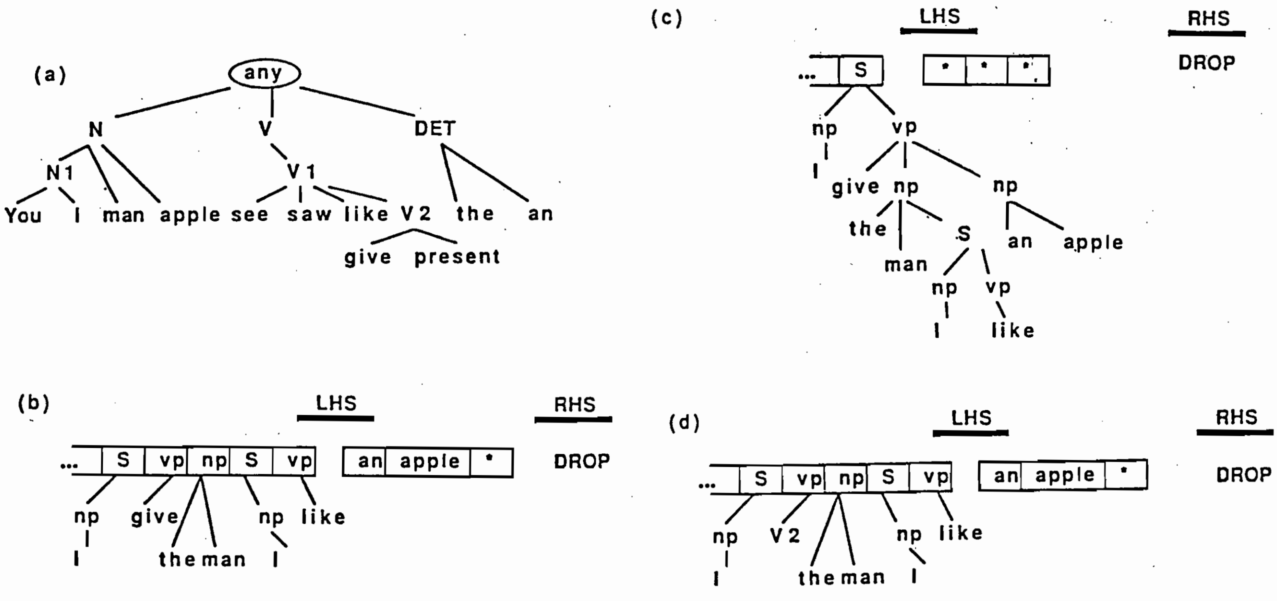


Fig. 5. Generalization Back-Propagations.

After processing this sentence, the set of fired rules for parsing this sentence will contain the rules shown in Fig.5.b and Fig.5.c. The rule shown in Fig.5.c is the last fired rule. The generalizer will try to generalize this rule by climbing the concept hierarchy. In this case, it try to climb the hierarchy from "give" to "V2". Thus, a question is asked: *I present the man I like an apple.* Since this sentence is syntactically and semantically valid (the trainer might answer "yes" to the system), the generalization from *give* to V2 is confirmed and back-propagated to the former one and the *give* in

Fig.5.b will be generalized to V2 which contains *give* and *present* as shown in Fig.5.a. Fig.5.d shows the result of this rule generalization by back-propagations.

Note that, by this method, all generalizations may be back-propagated to many rules. Since the generalizations of the last fired rules have been justified, the back-propagated generalizations will inherently be justified, and therefore will not cause over-generalizations. The number of differences between rules becomes irrelevant in this type of generalizations. This generalization mechanism is one kind of EBL. However, the generalizations are confirmed interactively by trainers (answering "yes" or "no" to the system), rather than justified by predefined domain knowledge. Thus, It is not limited by incomplete prior domain knowledge. Also, by this method, the number of questions asked by the system can be reduced. Thus, this is a very powerful method for conducting generalizations.

3.3.4 The Maintenance of The Concept Description

As described above, the concept description is simultaneously acquired while the system performs rule generalization. The system retrieves and updates the concept description very frequently. Therefore, as the concept description grows larger and larger, to efficiently maintain it becomes very important.

Currently, the concept description is represented as a hierarchy. When performing the generalization process, some concept nodes (such as "N1" and "V1" in Fig.5.a) might be created. The system keeps two information items for each node:

1. a list for recording its father nodes, and
2. a counter for recording how many times a node is referenced by rules.

The father list of a node is updated as follows:

> If a rule, which references a node T in its LHS, is generalized by the simple generalization mechanism, the corresponding new node F now referenced by the rule will be a father of the node T.

24A

While the counter of a node is updated as follows:

> It is incremented by one when a rule, which does not reference the node before, references the node now. On the other hand, it is decremented by one when a rule, which references the node before, does not reference the node any more.

In fact, the counters of concept nodes is updated only when the system generalizes rules:

1. If the system performs simple generalization, two rules are generalized to a more general rule. All concept nodes referenced by these two rules are not referenced by them now, thus their counters should be decremented by one. While for the nodes referenced by the new rule, their counters should be incremented by one.
2. If the system climbs the concept hierarchy from a node M to a node N, it will back-propagate the generalization. Each time a rule is generalized by this back-propagation, the counter of the node M is decremented by one, while the counter of the node N is incremented by one.

If the counter of a node is equal to zero (i.e. no rule references it), this node can be removed, and all its children nodes become the sons of all its father nodes.

4. Implementation

The language acquisition system proposed in this paper is implemented in GCLISP on an PC386 computer. There are about 1000 lines of lisp codes.

For efficiency, acquired rules are classified into the following rule packets: Srule, Nrule, Vrule, Prule, and Orule packets. If the first cell of the node stack in the condition part of a rule is an S, this rule is classified into the Srule packet. If this cell is a VP, it is classified into the Vrule packet, ... etc. Otherwise (such as the node stack is NIL), it is classified into the Orule packet. When searching for appropriate rules to fire, only those rules in the activated packets are examined.

4.1 A Simple Training Case

Table 1 shows a simple training case. Twelve training examples (12 pairs of input sentences and parse trees) are entered to the system.

Table 1. The training examples for illustrating the learning process.

s1:	(It is a book)
t1:	(S (NP (N It)) (VP (V is) (NP (DET a) (N book))))
s2:	(It is a desk)
t2:	(S (NP (N It)) (VP (V is) (NP (DET a) (N desk))))
s3:	(That is a pen)
t3:	(S (NP (N That)) (VP (V is) (NP (DET a) (N pen))))
s4:	(That is a chair)
t4:	(S (NP (N That)) (VP (V is) (NP (DET a) (N chair))))
s5:	(It is a pencil)
t5:	(S (NP (N It)) (VP (V is) (NP (DET a) (N pencil))))
s6:	(It is a pen)
t6:	(S (NP (N It)) (VP (V is) (NP (DET a) (N pen))))
s7:	(That is a desk)
t7:	(S (NP (N That)) (VP (V is) (NP (DET a) (N desk))))
s8:	(That is a pencil)
t8:	(S (NP (N That)) (VP (V is) (NP (DET a) (N pencil))))
s9:	(This is a pencil)
t9:	(S (NP (N This)) (VP (V is) (NP (DET a) (N pencil))))
s10:	(This is a book)
t10:	(S (NP (N This)) (VP (V is) (NP (DET a) (N book))))
p11:	(This is a pen)
t11:	(S (NP (N This)) (VP (V is) (NP (DET a) (N pen))))
s12:	(This is a desk)
t12:	(S (NP (N This)) (VP (V is) (NP (DET a) (N desk))))

The learner totally performs 1 time of climbing the concept hierarchy and back-propagating the generalization, and 4 times of simple generalization with asking questions. It successfully acquires 11 rules and classifies "THIS", "THAT", and "IT" into one category, and "PENCIL", "CHAIR", "PEN", "DESK", and "BOOK" into another category. Since the system is allowed to ask questions, it responses additional sentences which are confirmed by the trainer. In this case, the system has learned to parse 15 sentences.

It is interesting to note that:

1. If the learner is only allowed to perform simple generalization process without asking any questions, there are totally 20 rules acquired, and there are more concept nodes in the concept description.

2. If it is allowed to perform simple generalization process with asking questions, only 11 rules are acquired. And during processing these 12 training examples, training examples (s8, t8), (s10, t10), (s11, t11), and (s12, t12) actually contribute nothing to further generalization. That is, before processing them, the learner has already acquired capability to parse them. Thus, the learner could learn to parse 15 sentences from the given 8 (=12-4) training examples.

3. If it is also allowed to climb the concept hierarchy and back-propagate the generalization, training examples (s6 t6), (s7, t7), (s8, t8), (s10 t10), (s11, t11), and (s12, t12) would contribute nothing to further generalization. Thus, the learner actually learns to parse 15 sentences from the given 6 (=12-6) training examples.

4.2 A More Complex Training Case

We illustrate a more complex training case in Table 2 which is to show how the system learn proper PP-attachments (Prepositional phrases attachments). For processing these training examples, the system asked nine questions to perform simple generalizations, among which four were confirmed with "Yes", while five were denied with "No". Also, the system asked six questions to climb the concept hierarchy and to back-propagate generalizations, all of them are confirmed. There are totally 95 rules acquired.

After giving these 13 training examples, the system learned how to determine the proper attachment of the prepositional phrases (headed with *in, on,* and *with)* for the verbs *put, keep, want* and *see.* For verbs *put* and *keep,* if the prepositions are *in* or *on* (training sentence s1, s2, s3, and s4), the prepositional phrases should, although not

Table 2. More complex training examples (their corresponding parse trees are not listed).

s1: (I put the dog in the box)
s2: (I keep the dog on the box)
s3: (You put the dog in the house)
s4: (I put the dog on the table)
s5: (I want the dog in the house)
s6: (I want the dog on the house)
s7: (I see the dog with a telescope)
s8: (I see the dog with a bell)

necessary, be attached to the corresponding verb phrases (VP). On the other hand, if the verb is *want* (training sentence s5 and s6), it is better to attach these prepositional phrases to the corresponding noun phrases. For the verb *see* (training sentence s7 and s8), the way of attaching prepositional phrases depends on the object of this prepositional phrases. For example, in training sentence s7, the prepositional phrase *with a telescope* should be attached to the verb phrase *see the dog*. While for training sentence s8, the prepositional phrase *with a bell* should be attached to the noun phrase *the dog*.

4.3 A Chinese Training Case

In this training case, the following Chinese training sentences are entered to the system:

我 吃 一 個 蘋 果. (I eat an apple) (S1)

我 吃 一 個 梨 子. (I eat a pear) (S2)

我 吃 一 枝 冰 棒. (I eat a ice-rod) (S3)

我 有 一 個 蘋 果. (I have an apple) (S4)

我 吃 一 枝 甘 蔗.　　(I eat a cane)　　(S5)

他 吃 一 個 梨 子.　　(He eats a pear)　　(S6)

The system totally asked ten questions, and among which five questions are confirmed. After training with these sentences, the system has acquired 23 rules and successfully classified 蘋果 (apple) and 梨子 (pear) into a category whose associable unit is 個, and 冰棒 (ice-rod) and 甘蔗 (cane) into a category whose associable unit is 枝. Besides, it has learned that the subject 我 (I) and 他 (He) share the same verbs 吃 (eat, eats) and 有 (has, have). That is, unlike in English, these subjects and verbs can agree in Chinese. These associable relationships are acquired implicitly in parsing rules. Thus, in general, the system can be applied to not only English but also Chinese language acquisition situations.

5. Conclusion and Discussion

We have proposed a computationally plausible model for natural language acquisition by viewing natural language parsing as a problem solving task. The system can acquire parsing rules and classify lexicons simultaneously without saving training examples. The proposed language acquisition model has 5 major features including:

1. No negative examples are provided by trainers.
2. None of lexicon features are given. This language acquisition model does not require to define a concept hierarchy (the lexicons) priorly, rather, this hierarchy is built incrementally.
3. Since the parse trees are given as input, with the operators for constructing the parse trees, the solution path can be deterministically found.
4. Combining the mechanisms of the generalization by asking question and generalization back-propagation, the number of questions can be reduced, and the learning system is more flexible in the sense that it is not restricted by incomplete

prior domain knowledge.

5. Allowing the learner to ask questions can shifts the burden of generating negative examples from the trainer to the learner.

Since natural language acquisition involves more tasks than merely those in parsing, the parsing-driven generalization approach in this paper only emphasizes on parsing-related learning issues. Even this, there are still many future works remain to be explored:

1. This system has acquired lexicons and parsing rules from the input statement sentences. Other types of sentences (such as command sentences, wh-question sentences, ... etc.) are not yet considered.
2. Since the size of lexicons in a concept hierarchy might grow significantly, to effectively maintain it is an interesting and important problem.
3. Currently, the system cannot acquire the concept of inflections among lexicons. This is due to its incapability to distiguish the inflection relation between lexicons, for example, the lexicon "went" is an inflection of "go". To acquire this, features of inflections among lexicons should be provided.
4. There might be a trade-off in choosing the number of cells in the node stack and the lookahead buffer. Too many cells will cause too specific rules, while too few cells will cause many over-generalizations in rules. Although the Wait-And-See strategy is modified (section 2.2) to promote its ability for resolving ambiguities, the three-celled buffer might still be inadequate. In case this situation is encountered, other mechanisms such as suspension (Liu and Soo [5]) should be introduced.
5. Although the system is allowed to ask questions, the trainer might not answer it, and in this case, the learner can just give up the possible generalizations. How to ask smarter questions is one of the future extensions of our work, and in addition,

how to incorporate more sophisticated generalization back-propagation mechanisms into the learning system is also an interesting problem.

References

1. Anderson J. R., *A Theory of Language Acquisition Based on General Learning Principles,* IJCAI, 1981.
2. Berwick R. C., *Learning Word Meanings from Examples,* IJCAI, 1983.
3. Berwick R. C., *The Acquisition of Syntactic Knowledge,* The MIT Press, Cambridge, Massachusetts, London, England, 1985.
4. Ellman T., *Explanation-Based Learning: A Survey of Programs and Perspective,* The ACM Computing Surveys, Vol. 21, No. 2, June 1989.
5. Liu R. L. and Soo V. W., *Dealing with Ambiguities in English Conjunctions And Comparatives by A Deterministic Parser,* International Journal of Pattern Recognition and Artificial Intelligence, to appear.
6. Marcus M. P., *A Theory of Syntactic Recognition for Natural Language,* The MIT Press, Cambridge, Massachusetts, London, England, 1980.
7. Michalski R. S., Carbonell J. G., and Mitchell T. M. (Eds.), *Machine Learning: An Artificial Intelligence Approach,* Vol. 1, 1983, and Vol. 2, 1986.
8. Mitchell T. M., *Version Space: An Approach to Concept Learning,* Ph.D. diss., Standford University, 1978.
9. Mitchell T. M., Keller R. M., and Kedar-Cabelli S. T. *Explanation-Based Generalization: A Unifying View* Machine Learning 1:47-80, 1986.
10. Mitchell T. M., Utgoff P. E. and Banerji R. B., *Learning by Experimentation: Acquiring and Refining Problem-Solving Heuristics,* in Machine Learning: An Artificial Intelligence Approach, Vol. 1, R. S. Michalski, J. G. Carbonell, and T. M. Mitchell (Eds.), 1983.

11. Pinker S., *Language Learnability and Language Development,* The Harvard University Press, Cambridge, Massachusetts, London, England, 1984.

12. Pustejovsky J. and Bergler S., *The Acquisition of Conceptual Structure for The Lexicon,* AAAI, 1987.

13. Sammut C. and Banerji R. B., *Learning Concepts by Asking Questions,* in Machine Learning: An Artificial Intelligence Approach, Vol. 2, R. S. Michalski, J. G. Carbonell, and T. M. Mitchell (Eds.), 1986.

14. Selfridge M., *A Computer Model of Child Language Acquisition,* IJCAI, 1981.

15. Utgoff P. E., *Shift of Bias for Inductive Concept Learning,* in Machine Learning: An Artificial Intelligence Approach, Vol. 2, R. S. Michalski, J. G. Carbonell, and T. M. Mitchell (Eds.), 1986.

16. Wexler K. and Culicover P. W., *Formal Principles of Language Acquisition,* The MIT Press, Cambridge, Massachusetts, London, England, 1980.

17. Winston P. H., *Learning Class Descriptions from Samples,* in Artificial Intelligence, 2nd edition, Addison-Wesley Publishing Company, 1984.

18. Yu Jing-Chang, *Causal Models and Learning From Experiments in Imperfect Domain Theory,* M.S. thesis, Institute of Computer Science, National Tsing Hua University, R.O.C., 1989.

19. Zernik U., *Learning Idioms --- With and Without Explanation,* IJCAI, 1987.

20. Zernik U., *Language Acquisition: Learning a Hierarchy of Phrases,* IJCAI, 1987.

21. Zernik U., *Lexicon Acquisition: Learning from Corpus by Capitalizing on Lexical Categories", IJCAI, 1989.*

Parsing-Driven Generalization for Natural Language Acquisition

Rey-Long Liu and Von-Wun Soo

Institute of Computer Science

National Tsing-Hua University

ABSTRACT

Parsing is an important step in natural language processing. It involves tasks of searching for applicable grammatical rules which can transform natural language sentences into their corresponding parse trees. Therefore parsing can be viewed as problem solving. From this point of view, language acquisition can be generalized from problem solving heuristics. In this paper we show how learning methods can be incorporated into a wait-and-see parser (WASP), the problem solver. We call this approach parsing-driven generalization since learning (acquisition of parsing rules and classification of lexicons) is basically derived from the parsing process.

Three generalization methods are reported in this paper: a simple generalization mechanism, a mechanism of generalization by asking questions, and a mechanism of generalization back-propagations. The simple generalization mechanism generalizes from any two parsing rules whose action parts (right-hand sides) are the same while whose condition parts (left-hand sides) have a single difference. The mechanism of generalization by asking questions is triggered when a "climbing-up" move on a concept hierarchical tree is attempted and is necessary in avoidance of over-generalizations. The generalization back-propagation mechanism is to propagate a confirmed generalization of some later parsing rule back to its precedent rules in a parsing sequence and thus causes them to be generalized as well. This mechanism can save many questions to be asked. With the three generalization methods and a mechanism to maintain lexicon classification (the domain concept hierarchy), we have been able to show a plausible natural language acquisition model.

1. Introduction

Natural language acquisition has been an interesting and challenging research topic for both psycho-linguistists and computer scientists, although the former might be more interested in studying on human subjects while the latter on man-made machines. To understand how a natural language can be acquired by a machine can benefits from both studies. The psycho-linguistical study might be able to reveal many clues, constraints, and limitations regarding to natural language acquisition tasks that human actually face. These revelations may suggest us many guidelines for building computer systems that can automatically acquire a natural language. Our purpose in this paper, however, is not to show psycho-linguistic evidence on human natural language acquisition, rather is to build a computer model using machine learning techniques from artificial intelligence study [7] to demonstrate the feasibilities of natural language acquisition on machines.

There are many reasons to study natural language acquisition on a machine. One among them is to remove the current difficulty of having to construct and maintain a large set of lexicons and grammatical parsing rules in a complex natural language processing system. Since human natural language is ever-growing and evolutionary in nature, no natural language processing system with a complete set of both lexicons and grammatical rules can possibly be built. One solution to this is to seek ways of incrementally acquiring grammatical rules from training examples of sentences while keeping the coherence of the system. To achieve this, the system must have the capability of performing generalization over training examples. Traditional artificial intelligence researches have developed many techniques to perform generalizations [7].

Parsing involves tasks of searching for applicable grammatical rules which can transform natural language sentences into their corresponding parse trees. Therefore a successful parsing sequence can be treated as a solution to a parsing problem. By viewing natural language parsing as problem solving, the language acquisition tasks can be achieved to some extent by generalizing from problem solving heuristics. We call this approach parsing-driven generalization since learning (acquisition of parsing rules and classification of lexicons) is basically derived from the parsing process.

1.1 Related Works

Berwick [3] uses word features to perform conservative generalizations (the Subset Principle) but leaves their acquisition as an open problem. The acquisition of semantic and syntactic features of lexicons is important for the language development of children (Selfridge [14], Pustejovsky and Bergler [12], Berwick [2] and Zernik [21]). In the system, lexicons are classified in a concept description hierarchy. However, such a concept hierarchy is not built priorly. Instead, it is constructed during the learning process. This contrasts to Mitchell's version space approach (Mitchell [8]), where the domain concept hierarchy must be built before a generalization/specialization process can be carried out.

It has been argued by many researchers (Pinker [11], Wexler and Culicover [16], and Berwick [3]) that no negative examples are necessary in a language acquisition situation. That is, children usually acquire a language by imitating from parents' conversation. It is rare the case that parents teach illegal sentences to their children. However, children do sometimes generate illegal sentences due to overgeneralization from incomplete language acquisition. To avoid overgeneralization, it is necessary for a language acquisition system to be able to ask questions before performing generalization. This is similar to MARVIN's approach of learning (Sammut and Banerji [13]). The generalizer performs experiments by generating sentences and testing their validity by asking the trainer. When the system responses an invalid sentence (might be syntactically invalid such as *He eat an apple,* or might be semantically invalid such as *An apple runs),* the possible generalizations are prohibited. This generalization method shifts the burden of generating negative examples from trainers to learners. Trainers do not have to care about the current state of learners.

Traditional EBL (Utgoff [15], Ellman [4] and Mitchell et. al. [9] [10]) uses high-level domain knowledge to guide correct generalizations. Since parsing is viewed as problem solving, the powerful learning methods such as the Explanation-Based Learning (EBL) can be incorporated into the language acquisition system. In language acquisition, Zernik [20] and Zernik [19] employ the EBL method to acquire phrases and idioms respectively. However, it seems to be impossible to construct a complete knowledge base to perform EBL (Yu [18]). In our system, no such knowledge base is assumed. Rather, we incorporate the "back-propagation" concept of the EBL into the generalizer. The generalizations of the last fired rule are back-propagated when they are confirmed by the trainer.

2. The Parsing Device

The parsing device of this language acquisition system is based on the Wait-And-See strategy.

2.1 The Wait-And-See Strategy

Marcus [6] proposed the Wait-And-See strategy to parse natural languages. It is based on a "determinism hypothesis" which says that natural languages can be parsed by a computationally simple mechanism without backtracking. A Wait-And-See Parser (WASP) is like a production system, where the grammar and parsing heuristics are expressed in terms of rules (parsing rules) which are composed of condition and action parts. Two major data structures are required:

1. active node stack: a pushdown stack of incomplete constituents, and
2. lookahead buffer: a small constituent buffer containing constituents which are complete, but whose higher grammatical function is as yet uncertain.

The rules in a WASP are partitioned into rule packets. Each rule packet contains rules for a particular configuration of the constituent in the top of the node stack. For example, if the top of the node stack contains a VP, the corresponding rule packet for the VP is activated. However, the selection of rules to fire may depend on the contents of the lookahead buffer and the node stack.

The action operators in a WASP defined by Marcus [6] are:

1. ATTACH: attach the constituent at the top of the buffer stack (X) to the top of the node stack (Y). X is popped from the buffer and becomes the rightmost daughter of Y,
2. DROP: Y is popped from the node stack and pushed onto the buffer stack, and
3. CREATE: push a new active node onto the node stack.

More detailed descriptions for a WASP can be found in Liu and Soo [5] and Marcus [6].

2.2 Extending the WASP

In fact, the creation of an S, an NP, a VP, and a PP may be delayed until its first component is parsed and dropped. At that time, the creation action can be deterministically followed by an ATTACH action which attaches this parsed and dropped component. For example, consider the sentence *I ate an apple*. At the beginning, before creating an S node, the parser might first create an NP node which will attach the noun *I* and then be dropped to the buffer stack. Then, an S node is created, and the parser automatically attach this NP as its first son. Thus, we have an improved version of the action CREATE:

> CREATE: push a new active node onto the node stack, **and** perform the ATTACH action.

This improvement has two advantages:

1. when the first component is parsed and dropped, the parser can lookahead more information, and

2. since the creation action is followed by an attach action, the total number of actions for transforming the input sentence to its parse tree is reduced.

For example, consider again the above sentence *I ate an apple*. If the traditional mechanism (as described in the above section) is employed, there are 15 actions in the solution path:

((CREATE S) (CREATE NP) (ATTACH) (DROP) (ATTACH)
(CREATE VP) (ATTACH) (CREATE NP) (ATTACH) (ATTACH)
(DROP) (ATTACH) (DROP) (ATTACH) (DROP)).

However, if the improved mechanism is employed, there are only 11 actions in the solution path:

((CREATE NP) (DROP) (CREATE S) (CREATE VP) (CREATE NP)
(ATTACH) (DROP) (ATTACH) (DROP) (ATTACH) (DROP)).

This will certainly reduce the number of acquired rules.

Since the rules in a WASP are uniform and suitable for language acquisition systems, it is adopted as the parsing device of the learning system. The parsing operators (ATTACH, DROP and CREATE) become the key action operators in the system. Each acquired rule contains one (and only one) of these operators as its action part.

3. The Learning Module

Fig.1 shows the flowchart of the language acquisition system. After accepting an input sentence and its corresponding parse tree, the system initially sets the node stack to be empty, fills the cells in the lookahead buffer (the number of cells of the lookahead buffer is discussed in later sections), and finds the solution path (sequence of actions that transforms the input sentence to its corresponding parse tree) deterministically.

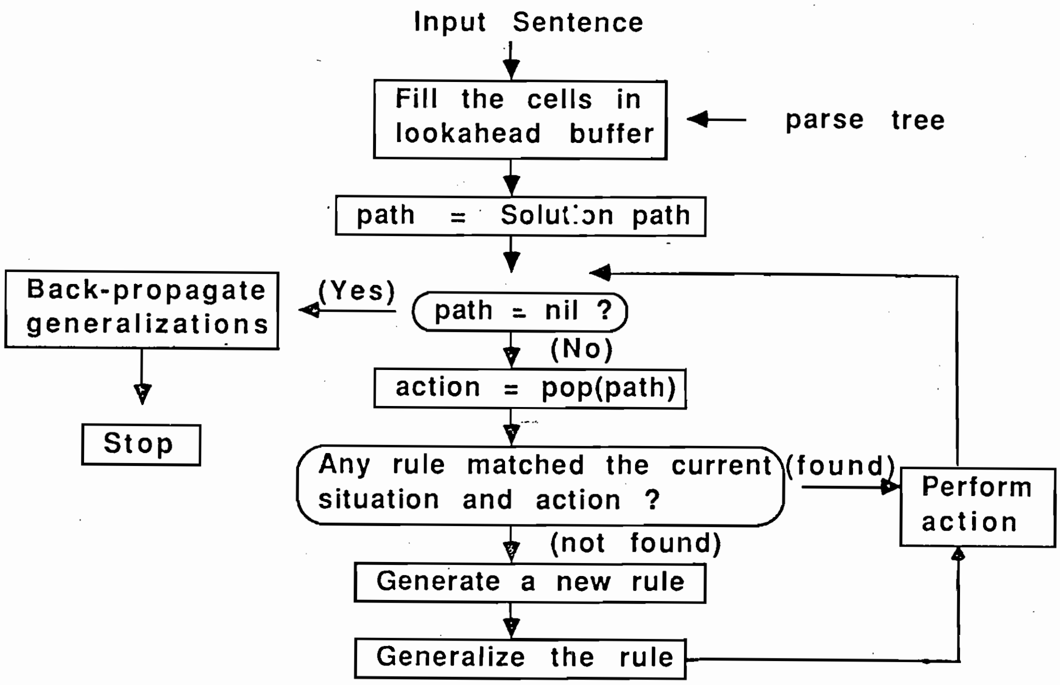


Fig. 1. The overview of the system.

Then, it iteratively pops up the first action in the solution path, tries to find a rule whose RHS (Right Hand Side) matches the action, and LHS (Left Hand Side) condition matches the current configuration of the node stack and the lookahead buffer. If a rule is found, it is fired, otherwise, a new rule is generated according to the current configuration of the node stack and the lookahead buffer (as the LHS of this rule) and the action (as the RHS of this rule). After trying to generalize this new rule, the action is performed, and the iterative process continues until all actions in the solution path are examined. At this time, the system will back-propagate the generalizations from the later acquired or fired rules to the previous acquired rules. Thus, the previous acquired rules are further generalized without causing over-generalizations.

3.1 The input

The semantic bootstraping hypothesis, which is employed in many language acquisition systems (Pinker [11] and Berwick [3]), states that when speaking to a child, parents refer to physical objects using nouns, and actions, which cause the change of the states, using verbs. Thus, although the child might not initially know what the grammatical categories (such as noun, verb, ... etc.) are in the target language, he or she might also learn them according to the syntax-semantic correspondences. Berwick [3] uses the thematic representations for these correspondences, and in Anderson [1], they are encoded as associated networks which describe the scene of the input sentence. Thus, the trainer must provide not only the training sentences but also their corresponding semantics. In our language acquisition model, the additional information is expressed in the parse trees. For example, for the input sentence *I eat an apple,* the corresponding parse tree is also input to the system:

(S (NP (N I)) (VP (V eat) (NP (DET an) (N apple)))).

In fact, the parse tree provides the learner much information including the categories of words and the structures of phrases. The provision of the category information is also a simplification made by Wexler and Culicover [16]. And as Pinker [11] pointed out, it is possible to derive from the parse tree the lexical entries and the phrase structure rules that generate the input sentence. However, since the derived phrase structure rules are too general, they might sometimes generate syntactically and semantically illegal sentences. What the system wants to acquire are the critical features of words in parsing situations to construct the correct parsing rules without causing over-generalizations. The derivations of general phrase structural rules are not helpful in our problem domain.

Also, from the viewpoint of problem solving, the input sentence can be treated as the initial state, while its corresponding parse tree as the goal state. By properly defining parsing operators (ATTACH, DROP, ... etc.) and representing the parse tree, we can design an algorithm to deterministically detect the whole solution path (the solution path can transform the initial state to the goal state) without resorting to searching or prior domain knowledge. Thus, the given parse tree is not only necessary (in the sense that it represents the syntax-semantic correspondences), but also powerful for the language acquisition system.

3.2 The Determination of a Solution Path

We view language parsing as a problem-solving task which is to transform the input sentence (the initial state) to its corresponding parse tree (the goal state). The LEX system (Mitchell et. al. [10] and Utgoff [15]) formulates the solving of the calculus as a search problem. After a solution is found, specializations and generalizations are then conducted to enhance the performance of the problem solver. The similar idea is used in our language acquisition system. However, by using the input parse tree and the parsing device described above, we can easily determine a solution path without resorting to searching the entire solution space or relying on the high-level domain knowledge (such as the X-bar theory used in Berwick [3]). For example, consider the sentence *I know the beautiful girl* and its parse tree (S (NP (N I)) (VP (V know) (NP (DET the) (ADJ beautiful) (N girl)))). A corresponding solution path can be easily determined (recall the parsing device described in section 2.2)

> ((CREATE NP) (DROP) (CREATE S) (CREATE VP) (CREATE NP) (ATTACH) (ATTACH) (DROP) (ATTACH) (DROP) (ATTACH) (DROP)).

The determination algorithm scans the parse tree from left to right. When a right parenthesis is encountered, a DROP operator is output, and when a left parenthesis is encountered, a CREATE operator should be output. However, this output of CREATE operator is delayed until its first component is parsed and dropped (section 2.2). The algorithm is as follows:

DetectAction (P)

Input: the parse tree P which is represented as a list.

Output: the solution path for constructing P.

Algorithm:

1. Let C = Car(P); D = Cdr(P).
2. If C is S, NP, VP, PP, ADJP, or ADVP then

2.1. Let CC = (pop D).

2.2. DetectAction (CC).

2.3. Output the action CREATE C.

2.4. For each element CC in D do

    2.4.1. DetectAction (CC).

    2.4.2. Output the action ATTACH.

2.5. Output the action DROP.

3. Return.

It should be noted that, when the first element (C) of the input tree (P) is not a phrase such as an S (Sentence), NP (Noun Phrase) , VP (Verb Phrase), PP (Prepositional Phrase), ADJP (Adjective Phrase), or ADVP (Adverbial Phrase), no action is output for constructing this element. For example, in step 2.4.1, when CC is "(N I)", the recursive call "DetectAction (CC)" produces nothing, and after this call, an ATTACH action is output in step 2.4.2. The structure "(N I)" only gives category information 'N' to the word 'I'. Since it is not a phrase, no actions are needed to construct it. The parser simply attaches it to the constituent at the top of the node stack. It should also be noted that, by the definition of the action CREATE in section 2.2, when a constituent C is created (step 2.3), it will automatically attach its first son (the constituent CC in step 2.2).

3.3 Generalizations

There are three types of generalizations in the system: simple generalizations, generalization by asking questions, and generalization back-propagations. We will describe them subsequently in this section.

3.3.1 Simple Generalizations

When a new rule is generated, the system will try to generalize it to its largest extent according to the current concept description. Initially, the concept description (the lexicon) is empty as shown in Fig.2.a. When training examples are provided, the concept description will accumulate the features of each word. For example, after the first sentence *I see the man* and its parse tree (S (NP (N I)) (VP (V see) (NP (DET the) (N man)))) are entered, there are 11 actions detected in its solution path, and thus 11 specific rules (one action for each rule) are acquired. Fig.2.b shows the current concept description.

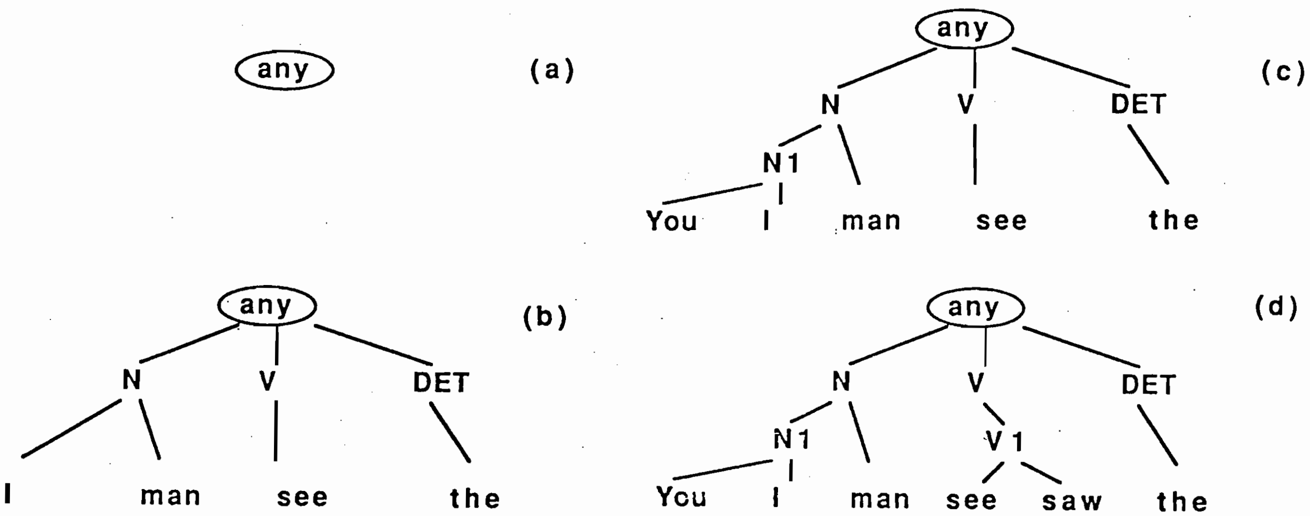


Fig. 2. The concept description.

Since the conception description currently has only specific information, no generalizations are possible in this case. When another sentence *You see the man* is entered, there is only one difference between the previously acquired rule in Fig.3.a and the newly generated rule in Fig.3.b. These two rules are deterministically generalized to the rule shown in Fig.3.c. Fig.2.c illustrates the current concept description (N1 is the acquired general node). Next, suppose the sentence *I saw the man* is entered, the generalizer will generalize from both the rule shown in Fig.3.c and the newly generated rule shown in Fig.3.d. Since there exists a difference (*see* and *saw*) and a more-general relationship (N1 is more general than I) between them, a question *You saw the man* will be asked to ensure the validity of the generalization shown in Fig.3.e. In this case, since this generated sentence is a valid one (confirmed by the trainer), the rule in

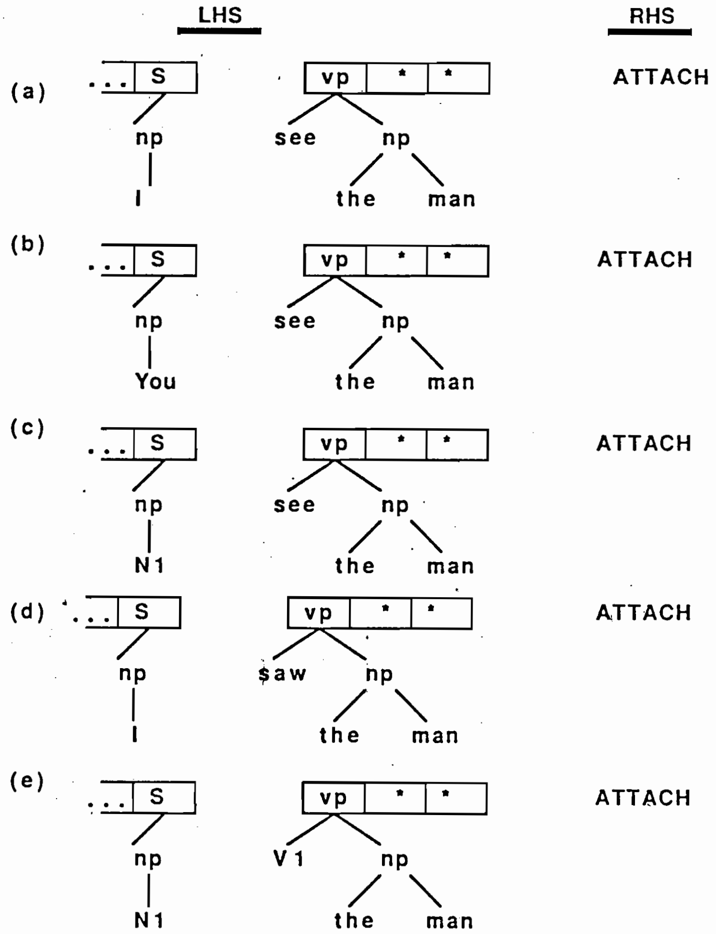


Fig. 3. Generalizations between two rules.

Fig.3.c and the rule in Fig.3.d are generalized to the rule in Fig.3.e. The conception description is also updated as shown in Fig.2.d (V1 is the newly acquired general node). The algorithm of this simple generalization mechanism is as follows:

SimpleGeneralization (R K)

Input: the new rule R and the activated rule packet K.

Output: the resulting rule after generalizing R.

Algorithm:

1. In the rule packet K, find a rule T which has the same action with R's, and there is only one difference between their LHSs.

If found then

2.1. If a more-general relationship is possible, ask a question to justify the generalization.

2.1.1. If the answer is "Yes" then

2.1.1.1. Generalize R and T. Denote the resulting rule by P.

2.1.1.2. Remove T from the rule packet K.

2.1.1.3. Return SimpleGeneralization (P K).

2.1.2. Otherwise, go to step 1 to find another rule.

2.2. Otherwise,

2.2.1. Generalize rule R and T. Denote the resulting rule by P.

2.2.2. Remove rule T from the rule packet K.

2.2.3. Return SimpleGeneralization (P K).

3. Otherwise, return R.

The above simple generalizations have the following features:

1. Generalizations are possible only when the action parts of the two rules are identical and there is at most one difference between their LHSs. If there are more-general relationships between them, questions must be asked to justify the generalizations.

2. When the asked sentences are indeed syntactically invalid (such as *He eat an apple*) or semantically invalid (such as *The apple runs),* the generalizations will be prohibited. With the capability to ask questions, it shifts the burden of generating near-miss (Winston [17]) examples from the trainer to the learner.

3. If a rule is successfully generalized to a new rule, this new rule will be generalized again by the same simple generalization mechanism.

4. There can be no over-generalizations. Each generalization is carefully-justified.

3.3.2 Generalizations by Asking Questions

Another type of generalization is to climb up the existing concept description hierarchy by asking questions. It is performed based on a single rule. For example, after the sentence *I give the man an apple* is processed, the concept description becomes a hierarchy as shown in Fig.4.a. Fig.4.b shows an acquired rule that can not be generalized by the above methods (since there are more than one differences between this new rule and the old ones).

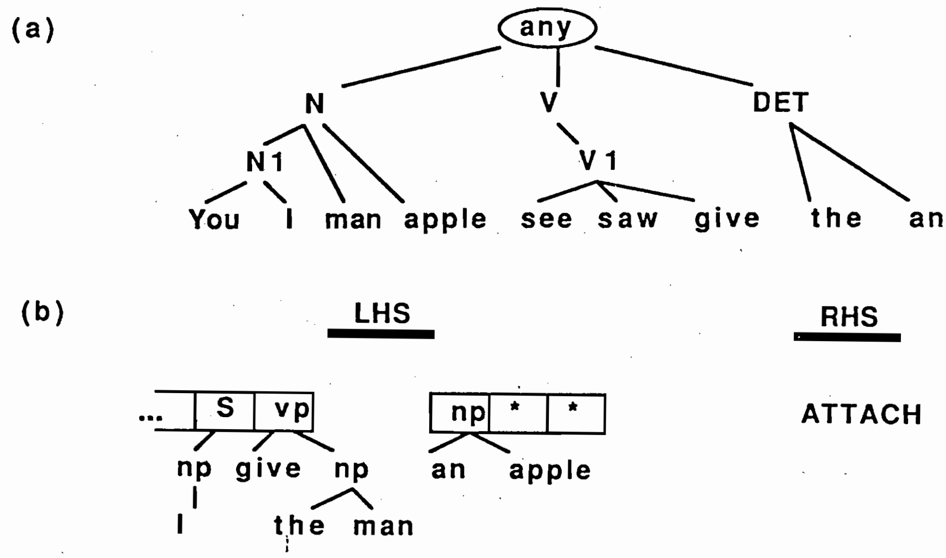


Fig. 4. Generalizations based on single rule.

The generalizer tries to generalize it by climbing the concept description hierarchy. Here, *see* and *saw* must be tested, and thus two questions are asked: *I see the man an apple* and *I saw the man an apple* which are all ungrammatical ones (since the verb *give* is dative, but the verb *see* and *saw* are not). The trainer answers "no" to the system, and thus, no generalizations are performed. The concept description remains unchanged.

Note that the learning system does not know what the newly generated nodes actually "mean". For example, syntactically, there may be Subject-Verb agreements between N1 and V1 in Fig.4.a, and semantically, the system can distinguish *apple* (eatable) from *desk* (un-eatable) by accepting *eat an apple* and not accepting *eat a desk*.

3.3.3 Generalization Back-Propagations

The above generalization by asking question mechanism, employed to climb the concept hierarchy, is very powerful. However, if the generalizer is allowed to climb the concept hierarchy each time when a new rule is generated, too many questions will be asked. The last type of generalizations is introduced to release this difficulty. It is performed when all necessary rules for parsing the current input sentence are fired. The system will generalize the latest fired rule by climbing the concept hierarchy, and then back-propagate the results of this generalization to previously fired rules. Consider the foilowing example: *I give the man I like an apple.* Assume that the current concept description is shown in Fig.5.a.

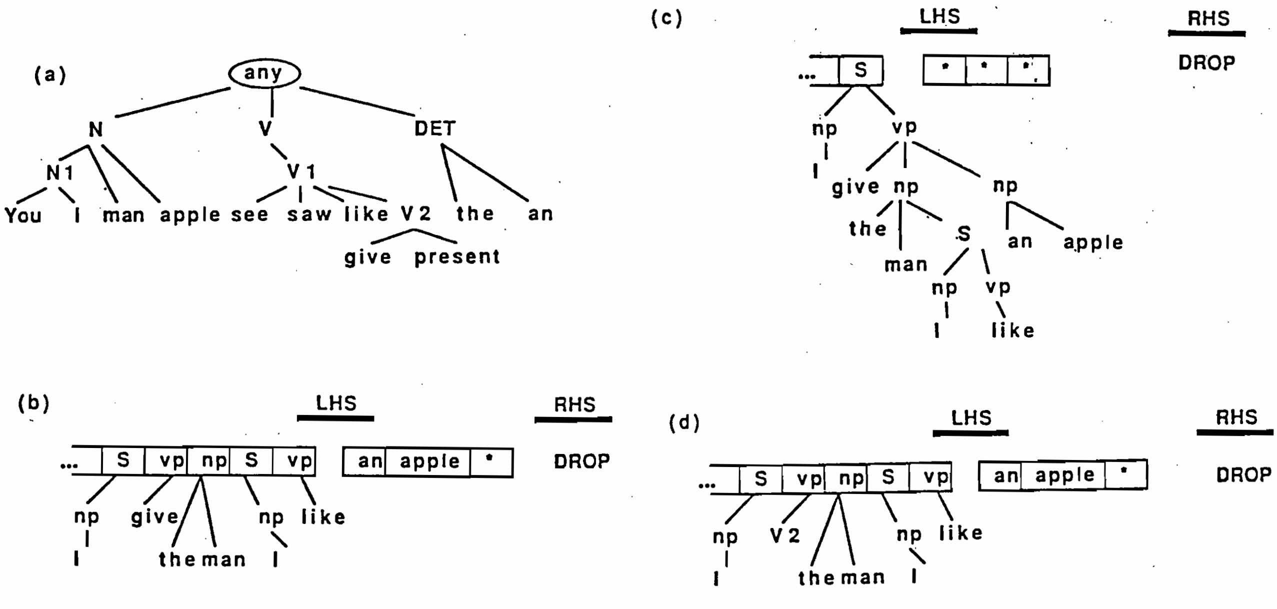


Fig. 5. Generalization Back-Propagations.

After processing this sentence, the set of fired rules for parsing this sentence will contain the rules shown in Fig.5.b and Fig.5.c. The rule shown in Fig.5.c is the last fired rule. The generalizer will try to generalize this rule by climbing the concept hierarchy. In this case, it try to climb the hierarchy from "give" to "V2". Thus, a question is asked: *I present the man I like an apple.* Since this sentence is syntactically and semantically valid (the trainer might answer "yes" to the system), the generalization from *give* to V2 is confirmed and back-propagated to the former one and the *give* in

Fig.5.b will be generalized to V2 which contains *give* and *present* as shown in Fig.5.a. Fig.5.d shows the result of this rule generalization by back-propagations.

Note that, by this method, all generalizations may be back-propagated to many rules. Since the generalizations of the last fired rules have been justified, the back-propagated generalizations will inherently be justified, and therefore will not cause over-generalizations. The number of differences between rules becomes irrelevant in this type of generalizations. This generalization mechanism is one kind of EBL. However, the generalizations are confirmed interactively by trainers (answering "yes" or "no" to the system), rather than justified by predefined domain knowledge. Thus, It is not limited by incomplete prior domain knowledge. Also, by this method, the number of questions asked by the system can be reduced. Thus, this is a very powerful method for conducting generalizations.

3.3.4 The Maintenance of The Concept Description

As described above, the concept description is simultaneously acquired while the system performs rule generalization. The system retrieves and updates the concept description very frequently. Therefore, as the concept description grows larger and larger, to efficiently maintain it becomes very important.

Currently, the concept description is represented as a hierarchy. When performing the generalization process, some concept nodes (such as "N1" and "V1" in Fig.5.a) might be created. The system keeps two information items for each node:

1. a list for recording its father nodes, and
2. a counter for recording how many times a node is referenced by rules.

The father list of a node is updated as follows:

> If a rule, which references a node T in its LHS, is generalized by the simple generalization mechanism, the corresponding new node F now referenced by the rule will be a father of the node T.

While the counter of a node is updated as follows:

> It is incremented by one when a rule, which does not reference the node before, references the node now. On the other hand, it is decremented by one when a rule, which references the node before, does not reference the node any more.

In fact, the counters of concept nodes is updated only when the system generalizes rules:

1. If the system performs simple generalization, two rules are generalized to a more general rule. All concept nodes referenced by these two rules are not referenced by them now, thus their counters should be decremented by one. While for the nodes referenced by the new rule, their counters should be incremented by one.
2. If the system climbs the concept hierarchy from a node M to a node N, it will back-propagate the generalization. Each time a rule is generalized by this back-propagation, the counter of the node M is decremented by one, while the counter of the node N is incremented by one.

If the counter of a node is equal to zero (i.e. no rule references it), this node can be removed, and all its children nodes become the sons of all its father nodes.

4. Implementation

The language acquisition system proposed in this paper is implemented in GCLISP on an PC386 computer. There are about 1000 lines of lisp codes.

For efficiency, acquired rules are classified into the following rule packets: Srule, Nrule, Vrule, Prule, and Orule packets. If the first cell of the node stack in the condition part of a rule is an S, this rule is classified into the Srule packet. If this cell is a VP, it is classified into the Vrule packet, ... etc. Otherwise (such as the node stack is NIL), it is classified into the Orule packet. When searching for appropriate rules to fire, only those rules in the activated packets are examined.

4.1 A Simple Training Case

Table 1 shows a simple training case. Twelve training examples (12 pairs of input sentences and parse trees) are entered to the system.

Table 1. The training examples for illustrating the learning process.

s1:	(It is a book)
t1:	(S (NP (N It)) (VP (V is) (NP (DET a) (N book))))
s2:	(It is a desk)
t2:	(S (NP (N It)) (VP (V is) (NP (DET a) (N desk))))
s3:	(That is a pen)
t3:	(S (NP (N That)) (VP (V is) (NP (DET a) (N pen))))
s4:	(That is a chair)
t4:	(S (NP (N That)) (VP (V is) (NP (DET a) (N chair))))
s5:	(It is a pencil)
t5:	(S (NP (N It)) (VP (V is) (NP (DET a) (N pencil))))
s6:	(It is a pen)
t6:	(S (NP (N It)) (VP (V is) (NP (DET a) (N pen))))
s7:	(That is a desk)
t7:	(S (NP (N That)) (VP (V is) (NP (DET a) (N desk))))
s8:	(That is a pencil)
t8:	(S (NP (N That)) (VP (V is) (NP (DET a) (N pencil))))
s9:	(This is a pencil)
t9:	(S (NP (N This)) (VP (V is) (NP (DET a) (N pencil))))
s10:	(This is a book)
t10:	(S (NP (N This)) (VP (V is) (NP (DET a) (N book))))
p11:	(This is a pen)
t11:	(S (NP (N This)) (VP (V is) (NP (DET a) (N pen))))
s12:	(This is a desk)
t12:	(S (NP (N This)) (VP (V is) (NP (DET a) (N desk))))

The learner totally performs 1 time of climbing the concept hierarchy and back-propagating the generalization, and 4 times of simple generalization with asking questions. It successfully acquires 11 rules and classifies "THIS", "THAT", and "IT" into one category, and "PENCIL", "CHAIR", "PEN", "DESK", and "BOOK" into another category. Since the system is allowed to ask questions, it responses additional sentences which are confirmed by the trainer. In this case, the system has learned to parse 15 sentences.

It is interesting to note that:

1. If the learner is only allowed to perform simple generalization process without asking any questions, there are totally 20 rules acquired, and there are more concept nodes in the concept description.

2. If it is allowed to perform simple generalization process with asking questions, only 11 rules are acquired. And during processing these 12 training examples, training examples (s8, t8), (s10, t10), (s11, t11), and (s12, t12) actually contribute nothing to further generalization. That is, before processing them, the learner has already acquired capability to parse them. Thus, the learner could learn to parse 15 sentences from the given 8 (=12-4) training examples.

3. If it is also allowed to climb the concept hierarchy and back-propagate the generalization, training examples (s6 t6), (s7, t7), (s8, t8), (s10 t10), (s11, t11), and (s12, t12) would contribute nothing to further generalization. Thus, the learner actually learns to parse 15 sentences from the given 6 (=12-6) training examples.

4.2 A More Complex Training Case

We illustrate a more complex training case in Table 2 which is to show how the system learn proper PP-attachments (Prepositional phrases attachments). For processing these training examples, the system asked nine questions to perform simple generalizations, among which four were confirmed with "Yes", while five were denied with "No". Also, the system asked six questions to climb the concept hierarchy and to back-propagate generalizations, all of them are confirmed. There are totally 95 rules acquired.

After giving these 13 training examples, the system learned how to determine the proper attachment of the prepositional phrases (headed with *in, on,* and *with)* for the verbs *put, keep, want* and *see.* For verbs *put* and *keep,* if the prepositions are *in* or *on* (training sentence s1, s2, s3, and s4), the prepositional phrases should, although not

Table 2. More complex training examples (their corresponding parse trees are not listed).

s1: (I put the dog in the box)
s2: (I keep the dog on the box)
s3: (You put the dog in the house)
s4: (I put the dog on the table)
s5: (I want the dog in the house)
s6: (I want the dog on the house)
s7: (I see the dog with a telescope)
s8: (I see the dog with a bell)

necessary, be attached to the corresponding verb phrases (VP). On the other hand, if the verb is *want* (training sentence s5 and s6), it is better to attach these prepositional phrases to the corresponding noun phrases. For the verb *see* (training sentence s7 and s8), the way of attaching prepositional phrases depends on the object of this prepositional phrases. For example, in training sentence s7, the prepositional phrase *with a telescope* should be attached to the verb phrase *see the dog*. While for training sentence s8, the prepositional phrase *with a bell* should be attached to the noun phrase *the dog*.

4.3 A Chinese Training Case

In this training case, the following Chinese training sentences are entered to the system:

我 吃 一 個 蘋 果. (I eat an apple) (S1)

我 吃 一 個 梨 子. (I eat a pear) (S2)

我 吃 一 枝 冰 棒. (I eat a ice-rod) (S3)

我 有 一 個 蘋 果. (I have an apple) (S4)

我吃一枝甘蔗. (I eat a cane) (S5)

他吃一個梨子. (He eats a pear) (S6)

The system totally asked ten questions, and among which five questions are confirmed. After training with these sentences, the system has acquired 23 rules and successfully classified 蘋果 (apple) and 梨子 (pear) into a category whose associable unit is 個, and 冰棒 (ice-rod) and 甘蔗 (cane) into a category whose associable unit is 枝. Besides, it has learned that the subject 我 (I) and 他 (He) share the same verbs 吃 (eat, eats) and 有 (has, have). That is, unlike in English, these subjects and verbs can agree in Chinese. These associable relationships are acquired implicitly in parsing rules. Thus, in general, the system can be applied to not only English but also Chinese language acquisition situations.

5. Conclusion and Discussion

We have proposed a computationally plausible model for natural language acquisition by viewing natural language parsing as a problem solving task. The system can acquire parsing rules and classify lexicons simultaneously without saving training examples. The proposed language acquisition model has 5 major features including:

1. No negative examples are provided by trainers.
2. None of lexicon features are given. This language acquisition model does not require to define a concept hierarchy (the lexicons) priorly, rather, this hierarchy is built incrementally.
3. Since the parse trees are given as input, with the operators for constructing the parse trees, the solution path can be deterministically found.
4. Combining the mechanisms of the generalization by asking question and generalization back-propagation, the number of questions can be reduced, and the learning system is more flexible in the sense that it is not restricted by incomplete

prior domain knowledge.

5. Allowing the learner to ask questions can shifts the burden of generating negative examples from the trainer to the learner.

Since natural language acquisition involves more tasks than merely those in parsing, the parsing-driven generalization approach in this paper only emphasizes on parsing-related learning issues. Even this, there are still many future works remain to be explored:

1. This system has acquired lexicons and parsing rules from the input statement sentences. Other types of sentences (such as command sentences, wh-question sentences, ... etc.) are not yet considered.
2. Since the size of lexicons in a concept hierarchy might grow significantly, to effectively maintain it is an interesting and important problem.
3. Currently, the system cannot acquire the concept of inflections among lexicons. This is due to its incapability to distiguish the inflection relation between lexicons, for example, the lexicon "went" is an inflection of "go". To acquire this, features of inflections among lexicons should be provided.
4. There might be a trade-off in choosing the number of cells in the node stack and the lookahead buffer. Too many cells will cause too specific rules, while too few cells will cause many over-generalizations in rules. Although the Wait-And-See strategy is modified (section 2.2) to promote its ability for resolving ambiguities, the three-celled buffer might still be inadequate. In case this situation is encountered, other mechanisms such as suspension (Liu and Soo [5]) should be introduced.
5. Although the system is allowed to ask questions, the trainer might not answer it, and in this case, the learner can just give up the possible generalizations. How to ask smarter questions is one of the future extensions of our work, and in addition,

how to incorporate more sophisticated generalization back-propagation mechanisms into the learning system is also an interesting problem.

References

1. Anderson J. R., *A Theory of Language Acquisition Based on General Learning Principles,* IJCAI, 1981.
2. Berwick R. C., *Learning Word Meanings from Examples,* IJCAI, 1983.
3. Berwick R. C., *The Acquisition of Syntactic Knowledge,* The MIT Press, Cambridge, Massachusetts, London, England, 1985.
4. Ellman T., *Explanation-Based Learning: A Survey of Programs and Perspective,* The ACM Computing Surveys, Vol. 21, No. 2, June 1989.
5. Liu R. L. and Soo V. W., *Dealing with Ambiguities in English Conjunctions And Comparatives by A Deterministic Parser,* International Journal of Pattern Recognition and Artificial Intelligence, to appear.
6. Marcus M. P., *A Theory of Syntactic Recognition for Natural Language,* The MIT Press, Cambridge, Massachusetts, London, England, 1980.
7. Michalski R. S., Carbonell J. G., and Mitchell T. M. (Eds.), *Machine Learning: An Artificial Intelligence Approach,* Vol. 1, 1983, and Vol. 2, 1986.
8. Mitchell T. M., *Version Space: An Approach to Concept Learning,* Ph.D. diss., Standford University, 1978.
9. Mitchell T. M., Keller R. M., and Kedar-Cabelli S. T. *Explanation-Based Generalization: A Unifying View* Machine Learning 1:47-80, 1986.
10. Mitchell T. M., Utgoff P. E. and Banerji R. B., *Learning by Experimentation: Acquiring and Refining Problem-Solving Heuristics,* in Machine Learning: An Artificial Intelligence Approach, Vol. 1, R. S. Michalski, J. G. Carbonell, and T. M. Mitchell (Eds.), 1983.

11. Pinker S., *Language Learnability and Language Development,* The Harvard University Press, Cambridge, Massachusetts, London, England, 1984.

12. Pustejovsky J. and Bergler S., *The Acquisition of Conceptual Structure for The Lexicon,* AAAI, 1987.

13. Sammut C. and Banerji R. B., *Learning Concepts by Asking Questions,* in Machine Learning: An Artificial Intelligence Approach, Vol. 2, R. S. Michalski, J. G. Carbonell, and T. M. Mitchell (Eds.), 1986.

14. Selfridge M., *A Computer Model of Child Language Acquisition,* IJCAI, 1981.

15. Utgoff P. E., *Shift of Bias for Inductive Concept Learning,* in Machine Learning: An Artificial Intelligence Approach, Vol. 2, R. S. Michalski, J. G. Carbonell, and T. M. Mitchell (Eds.), 1986.

16. Wexler K. and Culicover P. W., *Formal Principles of Language Acquisition,* The MIT Press, Cambridge, Massachusetts, London, England, 1980.

17. Winston P. H., *Learning Class Descriptions from Samples,* in Artificial Intelligence, 2nd edition, Addison-Wesley Publishing Company, 1984.

18. Yu Jing-Chang, *Causal Models and Learning From Experiments in Imperfect Domain Theory,* M.S. thesis, Institute of Computer Science, National Tsing Hua University, R.O.C., 1989.

19. Zernik U., *Learning Idioms --- With and Without Explanation,* IJCAI, 1987.

20. Zernik U., *Language Acquisition: Learning a Hierarchy of Phrases,* IJCAI, 1987.

21. Zernik U., *Lexicon Acquisition: Learning from Corpus by Capitalizing on Lexical Categories", IJCAI, 1989.*

An application of statistical optimization with dynamic programming to phonemic-input-to-character conversion for Chinese.

Richard Sproat

AT&T Bell Laboratories

Abstract

This paper describes a method for the conversion of Chinese text written in phonemic transliteration into Chinese character text. The method uses character bigram probabilities estimated from a large corpus of Chinese text to compute the most likely path through the lattice of possible character transliterations. Dynamic programming is used to prune the search. Thus the method is statistical, like previous work by Lin and Tsai (1987), but it is algorithmically simpler than their method. Performance scores are given which demonstrate that the proposed method produces good results.

An application of statistical optimization with dynamic programming to phonemic-input-to-character conversion for Chinese.

Richard Sproat

AT&T Bell Laboratories

1. Introduction: the problems of Chinese orthography.

It is a familiar point that inputing Chinese text is non-trivial because of the thousands of characters used in Chinese orthography; this makes the development of word-processing technology for Chinese much more difficult than for languages like English. While there have been a number of different classes of solutions proposed, there has been much interest in what is often referred to as *phonetic input,* more correctly *phonemic input.* Under these schemes one types Chinese text using one of several different transliteration schemes, usually either the Mandarin Phonetic system in Taiwan, or Hanyu Pinyin on the Mainland or abroad. The text is then converted to characters. Because the number of characters in the transliteration scheme is small, a normal keyboard can be used, and the user need only possess normal typing skills. Of course, to use such a system, one is required to know (one of) the transliteration schemes assumed, and be familiar with the particular Chinese language — usually Standard Mandarin — which the system assumes, but this is often argued to be a relatively small drawback since most Chinese speakers are familiar with one or another transliteration scheme and most literate Chinese are familiar with Standard Mandarin (Becker 1984b, Lin and Tsai 1987). Because phonemic input does not require special keyboards or the mastery of special coding schemes, it thus confers a number of advantages.

The major problem with phonemic input is of course ambiguity, since the number of syllable types in Chinese is significantly fewer than the number of characters. The syllable-to-character relation is therefore one-to-many, but since it is very skewed, highly ambiguous syllables like *shi4* are not unusual (I shall use the Hanyu Pinyin transliteration system in this paper, with numerals representing the tone):

是 事 市 式 世 示 士 視 識 試 適 室
勢 釋 氏 飾 侍 逝 誓 仕 嗜 恃 拭 噬
軾 弒 筮 柿 奭 舐 茌 伿 偍 倳

However, although syllables in isolation are quite ambiguous, most text consists of strings of many syllables, and one can significantly reduce ambiguity by taking the context into account. The most popular method is 'word-unit' based input, discussed in Becker (1984a,b). In all such systems some post-editing of the text is usually necessary, but the post-editing task is significantly reduced.

The present paper reports on a novel approach to the phoneme-to-character problem. The algorithm reported uses estimates of the conditional probability of a character given the previous character to find the optimal path through the lattice of possible characters. Dynamic programming is used to prune the search. The statistics are derived from a large corpus of Chinese text. In the next section we describe the algorithm and in the subsequent section we report on the implementation and the performance, and compare the latter with previous similar work.

2. The statistical method.

Church (1988, and see also Derose 1988) has proposed a statistical method for tagging English words with parts of speech. Conceptually, the problem can be laid out as follows: for any word in English text, there are a number of ways of tagging it with a part of speech label. So, *table* can be tagged as a noun in 'The suit is on the table' but as a verb in 'Let us table the suggestion.' Since each word may be many-ways ambiguous in this regard, a sentence of text may generally have a richly branching lattice of part of speech label possibilities associated with it. Church gives the following example (where 'UH' means 'interjection'):

I	see	a	bird
PRON	V	ART	N
PRON	V	P	N
PRON	UH	ART	N
PRON	UH	P	N
PROPNOUN	V	ART	N
PROPNOUN	V	P	N
PROPNOUN	UH	ART	N
PROPNOUN	UH	P	N

The general problem to be solved here is to find the most probable path through the lattice of parts given the lattice of words, or in other words, for a sentence of length *n*:

$$argmax \quad p(part_0 \cdots part_{n-} | word_0 \cdots word_{n-1})$$

Church approximates this conditional probability with a second order model as follows:

$$argmax \prod_{i=2}^{n-1} \frac{p(word_i | part_i) * p(part_i | part_{i-1} part_{i-2})}{p(word_i)}$$

Conceptually one enumerates all of the possible paths through the lattice and then picks the best path according to the above metric. However, we observe from the above formula that we are considering at most a trigram of part of speech labels, and that a dynamic programming solution is possible. So, at any point while traversing the lattice, for each triple of parts of speech $part_{i-2}part_{i-1}part_i$ we only need to remember the *best* path ending in that triple, since only the best such path can compete further down the lattice. This effects a significant reduction in the search space.

The relevance of this to the issue addressed in the present paper can be seen when we observe that the problem of deciding the best path through a lattice of part of speech labels is identical in relevant respects to the problem of deciding the best path through a lattice of Chinese characters generated from a phonemic input. So, consider the following example:

tai2	wan1	you3	tai2	feng1
(台)	(灣)	(有)	台	(風)
臺	彎	友	臺	豐
抬	蜿	黝	抬	封
颱	豌	酉	(颱)	峰
檯	剜	莠	檯	鋒
苔		牖	苔	蜂
抬			抬	瘋
檯			檯	楓
邰			邰	丰

The circled characters represent the most likely path through the lattice, that is, the sentence 'Taiwan has typhoons.'

The problem is approached in a way entirely analogous to Church, except that a first order model is used. (Until recently we did not have enough data to train a second order model.) Replacing 'part of speech' with 'character' (c) and 'word' with 'syllable' (σ) we arrive at the following expression:

$$argmax \prod_{i=1}^{n-1} \frac{p(\sigma_i | c_i) * p(c_i | c_{i-1})}{p(\sigma_i)}$$

We do not currently have a corpus of Chinese text with phonemic transcription so it is not currently possible to estimate $p(\sigma_i | c_i)$, i.e. the probability of a syllable σ given c, where σ is a possible pronunciation for c. However, since the majority of Chinese characters are unambiguous in their pronunciation (unlike the situation in Japanese) we can assume that this term is 1 in the general case. Furthermore, the denominator term does not effect the maximization since it is simply the probability of σ_i, which merely serves as a constant multiplier for probabilities of paths ending in that syllable. We can therefore simplify the above equation to:

$$argmax \prod_{i=1}^{n-1} p(c_i | c_{i-1})$$

In the current method, as in Church's method, dynamic programming is used to prune the search.

3. Performance issues.

3.1 Implementation and performance.

The statistics used by the version of the system which has been evaluated to date are derived from a corpus of 2.6 million characters of Chinese newspaper text (kindly provided by United Informatics, Inc., Taiwan). The current corpus contains 4846 distinct characters for which we have Pinyin transliterations. The bigram probabilities are estimated by dividing the frequency of occurrence of a sequence of characters by the size of the corpus. The unigram probabilities are estimated in a similar manner. (In practice the corpus size terms are omitted since they do not affect the maximization.) The algorithm has been implemented in C and runs on a Sun 3 at a rate of approximately 25 characters per second.

We turn now to a discussion of the performance of the method. Seven short samples of text of differing length were chosen, representative of various writing styles ranging from very classical to colloquial:

i) Ad [classical]

ii) Report [classical]

iii) Newspaper social commentary taken from the training corpus [semi-classical]

iv) Essay [semi-colloquial]

v) Narrative [semi-colloquial]

vi) Short story [colloquial]

vii) Exposition [colloquial]

The appendix contains the Pinyin along with the character renditions for the first text. Incorrect characters are circled.

The performance is given in the following table in terms of percentage correct by character (*hit rate*) for each of the styles; the hit rate from the algorithm is given in column three and should be compared with the hit rate achieved by merely picking the most common character given the pronunciation, which is given in the second column:

Style	Lexical Probabilities Only	Current Algorithm
i [class.]	76%	93%
ii [class.]	73%	90%
iii [semi-class.]	76%	98%
iv [semi-coll.]	69%	73%
v [semi-coll.]	72%	86%
vi [coll.]	71%	89%
vii [coll.]	71%	92%
Total	73%	90%

A trend evident in these data is that there is some dependency upon style: our current training corpus is heavily classical in style since it is mostly derived from newspapers.

As a consequence, texts (i-iii) which are classical in style are better rendered than the more colloquial texts, with the exception of (vii). One can expect that this style dependency would become less marked if the training corpus were expanded to include other styles.

3.2 Related research.

The only previous statistically based work on the phoneme-to-character problem of which we are aware is reported in Lin and Tsai (1987) and with subsequent modifications in Fan and Tsai (1988). Lin and Tsai employ the relaxation technique to obtain the optimal path through the lattice of possible characters, making use of the lexical probabilities of the characters given the syllables and the transition probabilities of adjacent characters and adjacent syllables. Their model was trained on a corpus of 196,573 characters of newspaper text written for elementary school students, and tested on similar materials. This text, unlike our corpus, contains phonemic transcriptions along with the character text. Thus Lin and Tsai are able to measure $p(\sigma_i | c_i)$, which is important in the few cases where a character has multiple pronunciations. In evaluating their results it is important to consider that elementary school texts represent a much simpler style than the texts with which we have been dealing, and are expected to be biased towards much more common words (and characters) than adult texts. One consequence of this is that they report an average hit rate of 77.6% by merely choosing the lexically most probable character, given the syllable, which is higher than our average rate of 73%. Their overall average hit rate is 95.4%, and the algorithm runs at a speed of about 2 characters per second. The speed appears to be slower than the one reported here, though this is at least in part due to a difference in hardware. As far as their hit rate is concerned, it is important again to realize that their training and test corpus was restricted to fairly elementary texts; one may expect not to get such a high score if the model were trained and tested on more mature Chinese texts.

4. *Summary*

The system reported here is clearly useful for reducing the amount of post-editing necessary for Chinese text entered with phonemic transliteration. The system is still under development and we have recently increased the size of the training corpus over the 2.6 million characters reported on above and are currently evaluating the behavior of the system under those conditions.

References.

Becker, J. 1984a. Multi-lingual word processing. *Scientific American* 251(1), 96-107.

Becker, J. 1984b. Typing Chinese, Japanese and Korean. *Computer* 18(1), 27-34.

Church, K. 1988. A stochastic parts program and noun phrase parser for unrestricted text. *Second Conference on Applied Natural Language Processing*, 136-143.

DeFrancis, J. 1984. *The Chinese Language*. Honolulu: University of Hawaii Press.

DeRose, S. 1988. Grammatical category disambiguation by statistical optimization. *Computational Linguistics* 14(1), 31-39.

Fan, C.-K. and Tsai, W.-H. 1988. Disambiguation of phonetic Chinese input by relaxation-based word identification. *Proceedings of ROCLING I*, 145-160.

Lin, M.-Y. and Tsai, W.-H. 1987. Removing the ambiguity of phonetic Chinese input by the relaxation technique. *Computer Processing of Chinese and Oriental Languages* 3(1), 1-24.

Appendix

Test text (i): circled characters are errors.

1. Pinyin:

pin3 zhi2 gao1 chao1 de0 guang3 gao4 yan2 liao4 shi4 you2 duo1 nian2 de0 jing1 yan4 . zai4 xian4 dai4 de0 ji4 shu4 she4 bei4 xia4 , cai3 yong4 gao1 ji2 yUan2 liao4 pei4 zhi4 er2 cheng2 . ge4 zhong3 se4 liao4 bu4 dan4 ke3 xiang1 hu4 jiao3 hun4 shi3 yong4 . qie3 se4 shang4 jia1 se4 , xiang1 die2 shi3 yong4 shi2 , yi4 wu2 tou4 lou4 xian3 chu1 di3 se4 zhi1 lU4 . ji2 shi3 hei1 se4 shang4 tu2 bai2 se4 yan2 liao4 , ye3 neng2 wan2 qUan2 yan3 gai4 .

2. Character rendition:

品質高超的廣告顏料是由多年的經驗。
在現代的計數設備下、
採用高級原料配置而成。
各種色料不但可相互皎混使用。
且色上加色、相疊使用食、
亦無透露顯出底色之律。
即使黑色上塗白色顏料、
也能完全掩蓋。

COMPUTER GENERATION OF CHINESE COMMENTARY ON OTHELLO GAMES*

Jen-Wen Liao and Jyun-Sheng Chang+

Department of Computer Science
National Tsing Hua University
Hsinchu, Taiwan 30043

Abstract

This paper describes the design and implementation of a text generation system which produces a mutil-sentence commentary in Chinese on a kind of board game call Othello. The system is built under a general framework of text generation. Discourse-related linguistic knowledge is encoded as rules and a production system is used to manipulate these rules to generate cohesive text. A previously constructed sentence generator based on systemic grammar is adopted to generate the final surface text.

1. Introduction

Natural language processing (*NLP*) is a main branch of *artificial intelligence*. There are many applications of NLP such as: *Machine Translation*, *Information Retrieval*, *Human-machine Interaction*, *Text Generation*, etc.

We choose a kind of board game called *Othello* as the application domain of text generation. We have implemented a text generation system creating commentary on Othello in **Chinese**. Despite the apparent simplicity of the task, the possibilities of producing text are rich and diverse. The commentary is intended to convey both the moves of the game and the significance of each move by showing errors and missed opportunities.

* This research is partially supported by National Science Council, grant No. NCS79-0408-E007-05 and NSC80-0408-E007-13.
+ To whom corresoondances should be addressed.

In next section, we present the description of Othello. In section 3, we introduce the general concepts of text generation. An overview of our system is also presented. In section 4, we introduce the first module of the system. In section 5, we give a detailed description about the second module of the system. In the last section, we show some examples generated by our system and summarize the paper.

2 Description of the game - Othello

Othello is a derivative of the *Go* family of board games; emphasizing capture of territory through the process of surrounding the opponent's pieces. It is played on an 8 x 8 board, with a set of dual-colored *discs* by two persons. Each disc is **Black** on one side and **White** on the other. Each color stands for one of the two players. The initial board configuration is shown in Figure 1.

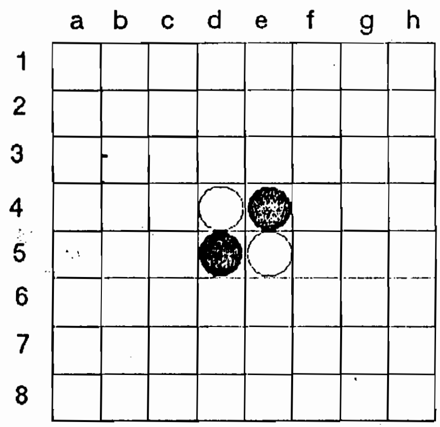


Figure 1. the initial Othello board configuration;

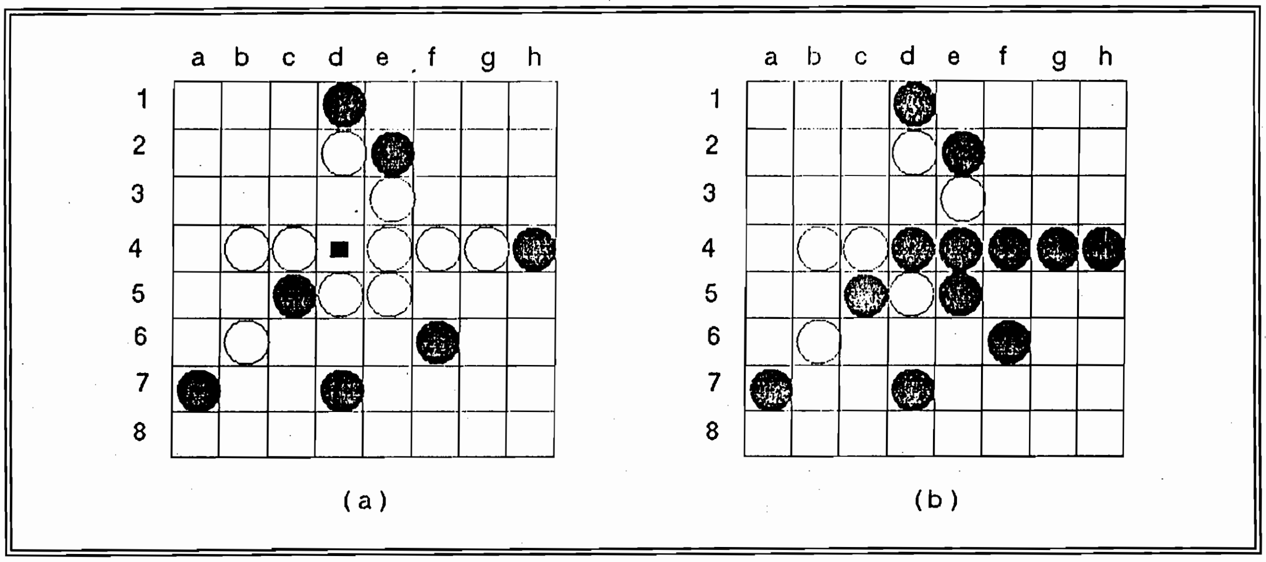


Figure 2. (a) Result of a legal play before BLACK's move at d4;
(b) after BLACK's play at d4.

Initially, WHITE owns the two central squares on the *major diagonal* (d4 and e5), and BLACK owns the two central squares on the *minor diagonal* (e4 and d5). BLACK plays first, and then the players take turns to play a move until neither side has a *legal* move. The player who has more disks at this point wins the game.

A move is made by placing a disc on the board, with the player's color facing up. In order for a move to be *legal*, the square must be empty prior to the move and it must result in *capturing* of the opponent's discs. The opponent's discs are captured by *bracketing* them between the disc being played and an existing disc belonging to the player. Bracketing can occur in a straight line in any of the eight directions(two vertical, two horizontal, and two in each diagonal direction). The captured disc (*discovered* discs) are flipped to the other color and become the opponent's. Bracketing and discovered discs does not further cause more discs to be flipped even if they result in a new bracket. Figure 2(a) and Figure 2(b) show an example of a legal move by BLACK on square d4, and the resulting change in disc ownership.

A number of simple strategies, such as maximizing the disc differential, have been suggested for Othello. For more details, see [Rosenbloom 1982] and [Liu, Huarng and Hsu 1987].

3 Text Generation

3.1 Introduction

Text generation is the reverse of natural language understanding. A generation system accepts a representation of linguistic information and goals and produces a sequence of sentences that realize the goals and convey the given information. The problem of text generation can be divided into two main areas which are not wholly independent: *planning the content of what to say* and *deciding how to say it*.

3.2 An Overview of Our System

Our system generates Chinese commentary on *Othello* games. After the user or the computer makes a move on the board, the system automatically produces a paragraph of text, analyzing the tactical pattern of the move, estimating the goodness of the move, comparing the move and the best move that the system knows of and showing the current board situation. The information of each step is stored in a list called *game history list* for later use. When the game is completed, this system displays a commentary on the whole game, including the mistakes the user or the computer has made, the information of the ownership of discs, and the winner. Our system consists of two principle modules (see Figure 3):

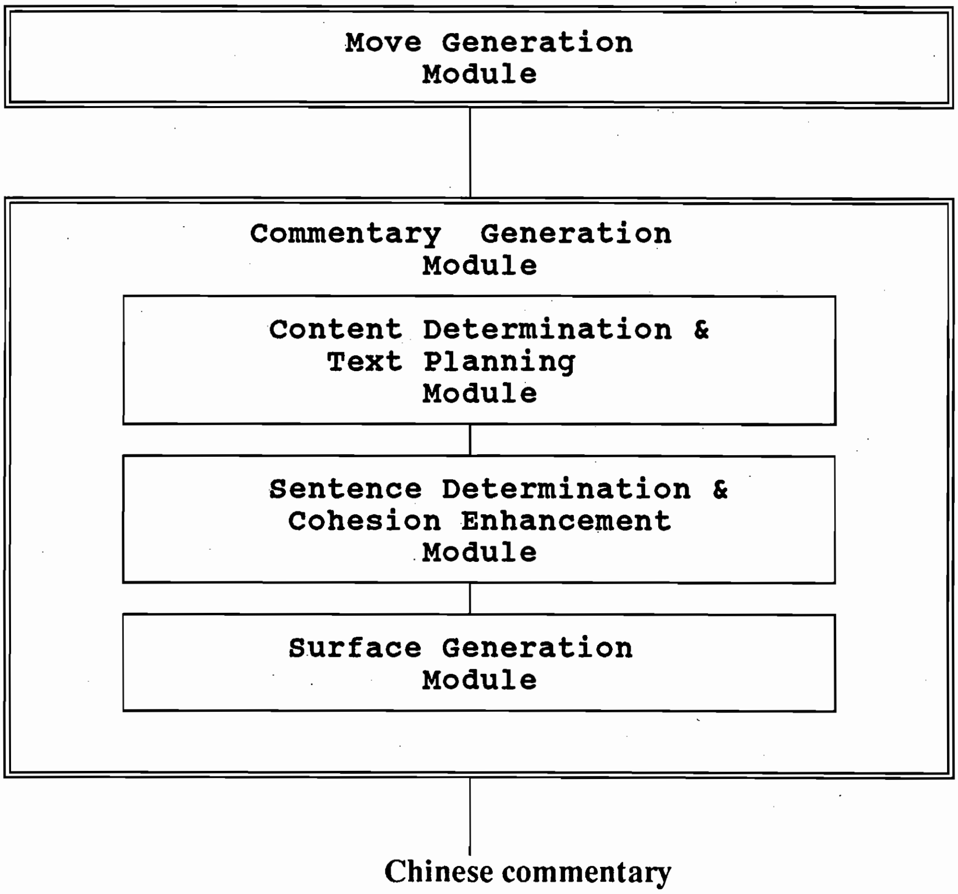


Figure 3. The text generation system

1. *Move Generation Module* which employs a minmax procedure and an alpha-beta cutoff procedure to select a move for the computer side or takes user's input as the move for the human side at each step. This module returns the *game history list* as output for the purpose of generating commentary on the whole game.
2. *Commentary Generation Module* which uses the methods described in subsequent sections to construct the commentary for one step or for the entire game. It can be further divided into three smaller modules:

   <A> *Content Determination & Text Planning Module* which determines what kind of information should be included in the output text.

   <B> *Sentence Determination & Cohesion Enhancement Module* which organizes the unordered, simple commentary into ordered, complex commentary.

<C> *Surface Generation Module* which transforms these commentary into Chinese sentences.

4. Move Generation Module

The purpose of *Move Generation Module* is to produce a legal move for the player (the computer side or the human side) and other relevant information such as game trees, the value of current evaluation function, etc. The move for the human side is taken from the input typed by the user and the move for the computer side is calculated by searching the game tree using a *minmax* procedure and an *alpha-beta cutoff* procedure. After each move of the game is generated, it is recorded in a so-called *game history list* for generating commentary on the entire game.

Usually the move returned by this module is not the best one since the depth of tree searching is limited. In order to generate versatile and meaningful commentary, the commentary generation module search at least one more level down the game tree to collect more knowledgeable information.

5. Commentary Generation Module

5.1 Content Determination and Text Planning Module

The first phase of text generation is to choose the information that is relevant and appropriate for inclusion in the output text. The decisions must be based on (1) what key objects involved that the system thinks is necessary to identify and (2) what aspects of situations are important and must be described. Also generation goals should be taken into account.

Text planning is the second phase in text generation. This phase accepts an unordered set of propositions generated by the content determination phase as input and organizes them into a series of ordered messages. The resulting messages must be appropriate (1) for the purpose of sentence generation and (2) to the intended audience. A knowledge base is also used to guide this process.

One of the problems that will occur if the content determination phase and text planning phase are totally separated is that some verbose or redundant message might be generated after the first phase such that the subsequent phase must waste time to remove them. This problem can be efficiently solved by integrating these two phases into one module as in our implementation. Besides, our system has a separate discourse module with explicit discourse structure which guide the content determination phase to generate propositions in a top-down, goal-directed fashion.

In our system, the input to this module includes the game tree, current board configuration, etc. We adopt the ATN (*Augmented Transition Networks*) framework to analyze these data and extract relevant information. The information generated from this phase is represented in frames like internal representation.

Transition Network formalism is commonly employed in NLP. A transition network consists of a network name, a set of nodes (states) and arcs. We express the transition network in a LISP-like *list* notation. The detailed description of representing ATN in list notation can be found in Charniak (1983). The registers in our notation is defined by beginning their names with character **?**. To define a network, we define a LISP function **def-net:**

(**def-net** *net-name* (*registers*) *commands)*

See Figure 4 for the commands for writing transition networks and Figure 5 for an example of network Generate-Partial-Comment in list notation.

We represent the internal structure of a sentence using a modified *slot-and-filler* representation, which reflects more the functional role of phrases in a sentence. This kind of internal representations is called *propositions*. In this notation, we assign a special semantic role (such as **Agent**, **Theme**, or **Focus**) to each major grouping of words. It identifies the properties of each semantic role as a set of *values*. A *slot name* distinguishes each value and indicates the role that value plays in the structure. Each slot statement is made up of the following components:

Commands	*Explanations*
(**traverse** (*network-name registers*))	Traverse the new network *network-name* with the values of *registers* are passed to the *network-name*.
(**seq** *actions*)	Perform the *actions* sequentially.
(**if** *test thenpart* **else** *elsepart*)	Check if *test* comes out true. If so, execute the commands of *thenpart*. Otherwise execute commands of *elsepart*.
(**either** *test1 action1 test2 action*2..)	If *test1* is true, execute commands of *action1*. If the test fails, try *test2* , *test3*, ... until one of them succeeds.
(**:=** *register value*)	Set the value of *register* to the value returned by the evaluation of *value*.
(**:=** (*registers*) *value-list*)	Set the value of *registers* according to the pattern returned by the evaluation of *value-list*.
(**jump** *tag*)	Jump to a *tag* of current network.
(**tag** *tag-name*)	Define a tag *tag-name*.
(**gen-comment** *comment*)	Generate *comment*.
=, <, >, and, or, dolist, first ...	some Lisp built-in functions

Figure 4.

--- a *slot name* (*operator*) indicating the type of the structure.

--- the *type* of the object.

--- the *modifiers* of the object, which may be a list of semantic role structures.

```
(def-net Generate-Partial-Comment (?Agent ?Opponent ?Move ?BestMove ?LastMove
                                   ?DiskFlipped ?GameTree ?Ev)
 (seq
  (traverse (Decide-Choices ?Agent ?GameTree ?Number-Of-Choices))
  (if    (= ?Number-Of-Choices 0)
         (jump NEXT)
  else (traverse (Show-Move ?Agent ?Opponent ?Move ?LastMove ?DiskFlipped)))
  (if    (= ?Number-Of-Choices 1)
         (jump NEXT)
  else (traverse (Measure-Goodness-Of-This-Move ?Agent ?Move ?BestMove
                         ?GameTree ?Other-Choices-Are-Equal ?MoveIsBest) ) )
  (if    (or ?Other-Choices-Are-Equal ?MoveIsBest)
         (jump NEXT)
  else (seq (traverse (Show-Best-Move ?BestMove))
            (traverse (Compare-BestMove-ThisMove ?Agent ?Move ?BestMove)))
 (tag NEXT)
  (traverse (Consequence-Of-This-Move ?Agent ?Opponent ?Move))
  (traverse (Summary ?Agent ?Opponent ?Ev))
 )
)
```

Figure 5.

1. The operators for simple NP will be used to indicate the determiner information. The possible combination for the operator of simple NPs are: DEF (for definite reference), INDEF (for indefinite reference), PRO (identifies an NP consisting of a pronoun), NAME (identifies an NP consisting of a proper name) and CONJ(represents two or more NPs connected by conjunction).

   The type for simple NPs will be as expected, and the modifiers will consists of adjectives, relative clauses, number information and so on.

2. The operators for clause structures include: PRES (for simple present tense), PAST (for simple past tense) and INF (for infinitive clause).

   The modifiers of a clause consist of: AGENT (the object that caused the event to happen), THEME (the thing that was acted upon), AFF-OBJ (the object that was affected by the event) and FOCUS (the focus of attention of the event).

3. The operators for simple sentences are simply ASSERT (for declarative sentence). As for compound sentences, the operators are the semantic relations between the clauses such as ALTHOUGH, CONJUNCTION, RESULT, etc.

   For example, the simple sentence "I have 3 choices" might be represented by the structure:

```
(ASSERT (PRES HAVE    (AGENT (PRO I))
                      (THEME (INDEF CHOICE (NUMBER 3)))
                      (AFF-OBJ nil)
                      (FOCUS (PRO I))))
```

Using the technique of transition networks, we can generate the internal representation for a versatile and comprehensive game commentary. During a network traversal, the command **gen-comment** creates a proposition and add it to a output list to be processed by the next module. If a register appears in the pattern of proposition, we must replace the register by its value at that position. To generate commentary for each step, we traverse the network *Generate-Partial-Comment*. When we traverse the network *Generate-Partial-Comment*, we also will traverse the subnetwork *Summary* (see Figure 6). Note that the symbol **$** means to evaluate the operation immediately following it and use the return value to fill the slot occupied by **$**.

```
(def-net Summary (?Agent ?Opponent ?Ev)
 (seq
    (:= ?AN (Count-disc-number '?Agent))
    (:= ?ON (Count-disc-number '?Opponent))
    (Gen-Comment '(assert (pres REMAIN (agent (pro ?Agent))
                                (theme (indef DISK (number ?AN))) (aff-obj nil)
                                (focus (pro ?Agent)))))
    (Gen-Comment '(assert (pres REMAIN (agent (pro ?Opponent))
                                (theme (indef DISK (number ?ON)))) (aff-obj nil)
                                (focus(pro ?Opponent)))
    (either (= ?AN ?ON)
            (Gen-Comment '(assert (pres TIE (agent (pro WE))
                                (theme nil) (aff-obj nil) (focus (pro WE))))
        otherwise
            (Gen-Comment '(assert (pres $(if (> ?AN ?ON) 'LEAD 'LAG)
                                (agent (pro ?Agent))
                                (theme (indef DISK (number $(- ?AN ?ON))))
                                (aff-obj (pro ?Opponent))
                                (focus (pro ?Agent)))))
    (either (= ?Ev 0)
            (Gen-Comment '(assert (pres HAVE (agent (pro NOBODY))
                                (theme (indef ADVANTAGE)) (aff-obj nil)
                                (focus (pro NOBODY))))
        otherwise
            (Gen-Comment '(assert (pres HAVE
                                (agent (pro $(if (> ?Ev 0) 'I 'You) ))
                                (theme (indef ADVANTAGE) (aff-obj nil)
                                (focus (pro $(if (> ?Ev 0) 'I 'You)))))
 )
)
```

Figure 6.

Suppose that the *?Agent* (the player who takes turn now) is 'You' (stands for the human side) and the *?Opponent* is 'I' (stands for the computer side). The current value of evaluation function (*?Ev*) is +5. After calculating the disc numbers on the board, we found that the human side owns 20 discs and the computer side owns 16 discs. The register *?AN* is set to 20 and the register *?ON* to 16. Therefore we will obtain the following propositions:

```
(assert (pres REMAIN   (agent (pro You))    (theme (indef DISK (number 20)))
                       (aff-obj nil)        (focus (pro You))))
(assert (pres REMAIN   (agent (pro I))      (theme (indef DISK (number 16)))
                       (aff-obj nil)        (focus (pro You))))
(assert (pres LEAD     (agent (pro You))    (theme (indef DISK (number 4)))
                       (aff-obj (pro I))    (focus (pro You))))
(assert (pres HAVE     (agent (pro I))      (theme (indef ADVANTAGE))
                       (aff-obj nil)        (focus (pro I))))
```

The corresponding commentary in Chinese is:

我 有 三 種 選 擇
你 剩 下 2 0 顆 棋 子
我 剩 下 1 6 顆 棋 子
你 領 先 我 4 顆 棋 子
我 佔 有 優 勢

5.2 Sentence Determination and Cohesion Enhancement Module

5.2.1 Introduction

After the content and overall ordered of the message in the text have been decided, there are still two problems that need to be resolved:

1. The first problem is deciding *how much information to put in a sentence*. For a set of propositions, it can be realized as a complex sentence or a few simple sentences. The factors influencing this decision include focus of attention [Derr and McKeown 1984], semantic and rhetorical relations between propositions [Davey 1979].
2. The second problem has to do with *cohesion in the text generated*. It is necessary in this phase to find and mark the cohesion links among propositions. These markings are subsequently used in surface generation for such activities as pronominal, demonstrative, verbal and clausal substitutions, ellipsis, selection of conjunction and lexical choice.

Some pragmatic knowledge is used to solve these two problems above. Under the consideration of easy understanding and modification, we have selected *rules* to represent the knowledge required for this module. Rule-based knowledge representation centers on the use of **IF** *condition* **THEN** *action* statements. Our rules are expressed with arrows (--->) to indicate the **IF** and **THEN** portions. For example,

the PH of the spill is less than 6 (**IF** part)

--->

the spill material is an acid. (**THEN** part)

5.2.2 Pattern Matching

To support rule-based approach described in the previous section, we need the technique of *pattern matching*. To enhance the capability of pattern matching scheme, we allow patterns to contain *pattern matching variables* beginning their names with the character ?. We also allow the user to attach an arbitrary *predicate* as a property of a pattern matching variable. Then when the matcher tries to match a variable against an item, it allows the match only if the item satisfies the attached predicate, if one exists. The pattern matching variable with a predicate is expressed as a list, in which the first element is the variable name and the second element is an arbitrary LISP predicate. For example, the expression

(age-of ?person ?(age (and (numberp ?age) (< ?age 20))))

would match

(age-of John 17)

with the result that ?person is bound to John and ?age is bound to 17.

It might be necessary to create a new pattern after we apply our pattern matcher to several patterns and get the variable bindings of a successful match. The pattern matching variables of the new pattern are required to be substituted with their corresponding value of bindings. In addition to replace variables, we might need to execute some other actions during the substitution to extend the new pattern such as:

!: Triggering another pattern matching process. The pattern following "!" is matched against a rule named by the first argument of the pattern. If it succeeds, the new pattern generated by that rule is used to fill the slot where "!" occupies. For example, consider the following rule Decide-Opening,

((3 3) (3 4)) ---> Parallel

the pattern before applying the rule: (The Opening is !(Decide-Opening (3 3) (3 4))) and the pattern after applying the rule: (The Opening is Parallel)

$: Evaluating the subsequent function and filling the slot with the return value. If a pattern matching variable is contained in the function, replace it with its binding value first and then evaluate that function. For example, if the bindings obtained after matching X, the **IF** part of a rule is ((?A 3) (?B 5)),

X ---> (The two variables are $(if (= ?A ?B) 'equal 'unequal))

the output pattern generated by the rule will be (The two variables are unequal)

@: Splitting the resultant pattern following it. It decomposes a pattern into a series of items. For example,

(a @(b c d) e) becomes (a b c d e)

5.2.3 Examples

The knowledge base in this module consists of two kinds of rules: one is called the <u>*discourse-rule*</u> and the other the <u>*pattern-rule*</u>. The former describes functional, semantic or rhetorical relation among propositions and the latter expresses facts in the knowledge base. The **IF** part of these rules is a list of one or more patterns. The main difference between them is that the **THEN** part of the discourse-rule has several patterns while the pattern-rule has only one. The pattern-rules can be applied in discourse-rules to check if a certain situation has happened, but the discourse-rules cannot be used in pattern-rules.

In this module, we encode *discourse-rules* to combine propositions. A discourse-rule consists of three components: *rule-name* , *rule-body* and *execution-priority*. The rule-name defines the unique name of a rule. The rule-body has an **IF** part and a **THEN** part. If a portion of the propositions matches the **IF** part of a rule-body, then it produces a new output proposition with every variables in the proposition replaced with its binding value. The execution-priority of rules are used to resolve the conflict where the **IF** parts of more than one rules matches a portion of propositions. When it happens, the rule with highest execution-priority fires. The output proposition of a rule can be tested for further combination with other propositions. We repeat this process until there is no possibility of combining propositions. The function **def-discourse-rule** is used to declare a discourse-rule.

A pattern-rule has two parts: *rule-name* and *rule-body*. The **IF** part of the rule-body consists of several patterns enclosed in a list and the **THEN** part specifies the new pattern to be generated if **IF** part of the rule matches the input patterns. The function **def-pattern-rule** is used to declare a pattern-rule.

Consider a discourse-rule <Although-but>. Suppose that a speaker makes an utterance that generates some expectation. But the expectation is contrary to the next utterance. Under such situation, combining these two utterances into a single complex sentence using conjunctions is better than producing two separate sentences since the former reflects the semantic relation while the later do not. To check if there is a certain relation among two sentences, the rule <Although-but> (See Figure 6) applies a pattern-rule Not-Expect:

There are other aspects of cohesion affecting the quality of output text. For instance, see input propositions below. Realizing these two propositions into two independent sentences (sequence 1) obviously is not a good idea since there are many redundant information. Instead we should join the two propositions by deleting the **PREDICATE, THEME** and **AFF-OBJ** of the proposition 2 and using conjunction to combine the two **AGENTs** (sequence 2). A discourse-rule named identity-delete-1 can be applied in such a condition by matching the values of **PREDICATE, THEME, AFF-OBJ** and **FOCUS** of one proposition with the corresponding arguments in the second proposition.

```
1. You lead me by 4 disks.
   I have advantage.

2. Although you lead me by 4 disks, I have the advantage.

(def-discourse-rule <Although-but>      7
 ( (assert (?T1 ?P1 ?A1 ?T1 ?O1 ?F1))
   (assert (?T2 ?P2 ?A2 ?T2 ?(O2
  !(Not-Expect (?P1 ?A1 ?T1 ?O1) (?P2 ?A2 ?T2 ?O2))) ?F2))
--->
 ((Although (assert (?T1 ?P1 ?A1 ?T1 ?O1 ?F1))(assert (?T2 ?P2 ?A2 ?T2 ?O2 ?F2))))
)

(def-pattern-rule Not-Expect
 ((LEAD ?P1 ?- ?P2) (HAVE ?P2 ?(Theme (= (Root ?Theme) 'ADVANTAGE)) ?- ))
---> True
)

input propositions:
     (assert (pres LEAD   (agent (pro You))      (theme (indef DISK (number 4)))
                          (aff-obj (pro I))      (focus (pro You))))
     (assert (pres HAVE   (agent (pro I))        (theme (indef ADVANTAGE))
                          (aff-obj nil)          (focus (pro I))))

output proposition:
     (Although (assert (pres LEAD (agent (pro You))  (theme (indef DISK (number 4)))
                                  (aff-obj (pro I))  (focus (pro You))))
               (assert (pres HAVE (agent (pro I))    (theme (indef ADVANTAGE))
                                  (aff-obj nil)      (focus (pro I)))) )
```

5.2.4 Discussion

Our discourse rules are inspired by the research of Derr and McKeown (1984) who concentrated on using focus to generate complex and simple sentences. They encode tests on functional information within *Definite Clause Grammar (DCG)* formalism to determine when to use complex sentences. These tests look like our discourse rules except that they are written in Prolog and the pattern matching mechanism is implicitly embedded in the grammar. Our discourse rules have an extra component '*execution-priority*' to resolve conflicts while the tests have not. Checking semantic relation between propositions to generate complex sentences are also not included in their paper.

1. You have 5 remaining disks.
   I have 5 remaining disks.

2. You and I both have 5 reamining disks.

```
(def-discourse-rule identity-delete-1 1
  ( (assert (?T ?P (agent ?A1) ?T ?O ?F1))
    (assert (?T ?P (agent ?A2) ?T ?O ?F2)) )
--->
  ( (assert (?T ?P (agent (conj ?A1 ?A2)) ?T ?O ?F1)) )
)

input propositions:
    (assert (pres REMAIN(agent (pro You))       (theme (indef DISK (number 5)))
                        (aff-obj nil)           (focus (pro You)))
    (assert (pres REMAIN(agent (pro I))         (theme (indef DISK (number 5)))
                        (aff-obj nil)           (focus (pro I)))

output proposition:
    (assert (pres REMAIN      (agent (conj (pro You) (pro I)))
                              (theme (indef DISK (number 5)))
                              (aff-obj nil)
                              (focus (pro You)))
```

The process of selecting and combining propositions in this module is similar to the *hill climbing* algorithm in Mann and Moore (1981). They designed so-called *aggregation rules* to combine sentences and then evaluate the resultant combinations. The best one is selected and the process is repeated. The difference between our method and theirs is that we always choose the best combination without generating many other alternatives. Another disadvantage of the aggregation rules is that they are language-dependent. Our discourse-rules are less dependent on language than theirs.

5.3 Surface Generation Module

The last module in the system is surface generation which takes the segmented and enhanced messages produced by last module as input and generates natural language sentences as output. It has to make such decisions as (1) what words and phrases to use in describing or referring to entities and their relationship, and (2) what syntactic structure to use in presenting these pieces of information.

Several grammatical formalisms have been used for surface generation: *Systemic grammar* [Halliday 1976], *Transformational grammar* [Chomsky 1965], *Augmented Transition Network (ATN) grammar* [Woods 1970], *Functional grammar*. Our surface generator use systemic grammar to produce Chinese sentences.

Our system employs the sentence generator implemented by Kuo (1989) as a part of the surface generation module for the production of versatile Chinese sentences. The form of input to the sentence generator looks like frames and has three parts:

1. *a frame name* --- specifies the constituent of a sentence that the systemic network is to generate.
2. *a list of features* --- provides the information about the functions that this constituent is intended to perform.
3. *an optional subframe list* --- gives the subconstituents that are to be handled by the lower level network. The subframes have exactly the same structure the we have described.

For example, to generate a sentence "You lead me by 4 disks" in Chinese, the input will be as follows:

```
(sentence (s-sentence)
    (clause (independent mood indicative transitivity transitive active double-obj)
            (agent (np head-noun pronoun (head-noun You)))
            (pred  (vp (verb lead)))
            (aff   (np head-noun pronoun (head-noun I)))
            (patient (np head-noun noun-mod class-phr (head-noun disk))
                        (classp (cp number (num 4) (class kir))))))
```

To prepare the input for the sentence generator, the propositions that we have produced need to be transformed into the form described in the last section. There are many decisions that we must make during this transformation such as the what features to include, how to fill the subframes, and which word to use. Instead of writing a subroutine to transform the propositions, we formalize the transformation by encoding these linguistic choices as pattern-rules. By matching the input propositions and the **IF** part of pattern-rules, the necessary form for the sentence generator can be obtained from the new pattern part of pattern-rules.

Consider the following pattern-rule for an example. The rule Transform-Clause transforms the clause element of a proposition into a subframe of the resultant input form according to the patterns specified in the **IF** part and the **THEN** part.

```
(def-pattern-rule Transform-Clause
  ( ?Type ?Pred (agent ?Agent) (theme ?Theme) (aff-obj ?Aff-Obj) (focus ?Focus))
--->
  ((clause !(Determine-Clause-type (?Agent))
     mood !(Determine-Mood (?Agent))
     transitivity @!(Determine-Transitivity (?Agent ?Theme ?Aff-Obj ?Focus))
   )
      $(if '?Agent        '(subj @!(Transform-NP ?Agent)))
      $(if '?Pred         '(pred !(Transform-VP ?Pred ?Type ?Agent ?Theme)))
      $(if '?Theme        '(patient @!(Transform-NP ?Theme)))
      $(if '?Aff-Obj      '(aff @!(Transform-NP ?Aff-Obj))) )
)
```

A word in propositions generated by content determination and text planning module denotes only a concept and does not necessarily correspond to the actual word to appear in the resultant sentence. But the word system in the sentence generator only consists of locative particle subsystem. We can either extend the word system in the sentence generator, or handle the problem of lexical choice during the transformation process described in the last section. We choose to implemented the lexical choice mechanism in the transformation phase for the following reasons: First of all, these choices can be easily formulated and encoded as rules without making the grammar too complicated. Secondly, these rules could be gracefully incorporated with other natural language formalism such as semantic networks to enhance its capability.

For example, the concept "HAVE" might have the following alternatives.

HAVE CHANCE	corresponds to	<u>GET</u> CHANCE
HAVE ADVANTAGE	corresponds to	<u>GAIN</u> ADVANTAGE

Through the help of pattern-rules, we are able to implement some simple word choices. The choices that we have made primarily concentrate on verbs. The rule accepts a verb and a direct object of the verb as inputs and generates a new word which is then used to replace the original verb in the input for the sentence generator. The pattern-rule Generate-Word for verb "HAVE" is described below.

```
(def-pattern-rule Generate-Word
  ( HAVE CHANCE )
---> GET )
(def-pattern-rule Generate-Word
  ( HAVE ADVANTAGE )
---> GAIN)
```

5.4 Examples of Text Generated by the System

The following two paragraphs are the commentary produced by our system. The first paragraph is the comment on a certain step during the game and the second is the commentary on the entire game.

<u>*Commentary on a certain step during the game*</u>

你 擁 有 10 種 選 擇 .
你 下 "(5,8)" 使 得 2 顆 棋 子 被 你 吃 掉 了 .

最 好 的 一 步 棋 是 "(2,6)" .
"(5,8)" 是 危 險 的 位 置 但 "(2,6)" 不 是 .
雖 然 你 可 能 會 吃 掉 右 邊 的 1 顆 棋 子 但 是 我 可 以 阻 止
你 的 攻 勢 .
你 獲 得 佔 領 左 邊 的 機 會 .
你 剩 下 1 1 顆 棋 子 , 我 剩 1 2 顆 棋 子 .
雖 然 你 落 後 我 1 顆 棋 子 但 是 你 佔 有 優 勢 .

Commentary on the entire game

你 開 始 這 一 盤 遊 戲 並 且 下 了 "(3,5)".
我 緊 接 著 下 了 "(3,4)".
這 種 開 局 法 稱 為 平 行 式 開 局 法
在 第 14 手.
我 阻 止 了 你 的 攻 擊.
在 第 16 手.
我 錯 下 一 步 棋 使 得 你 佔 領 了 下 邊 的 有 利 位 置.
在 第 19 手.
你 採 取 了 錯 誤 的 策 略 使 得 我 吃 掉 了 右 邊 的 1 顆 棋 子.
在 第 22 手.
我 下 一 步 錯 誤 的 的 棋 使 得 你 吃 掉 了 右 邊 的 3 顆 棋 子.
在 第 28 手.
我 沒 有 採 取 正 確 的 策 略 使 得 你 佔 領 了 下 邊 的 有 利 位 置.
在 第 34 手.
我 採 取 了 錯 誤 的 策 略 使 得 你 吃 掉 了 上 邊 的 1 顆 棋 子.
在 第 40 手.
我 下 一 步 錯 誤 的 的 棋 使 得 你 吃 掉 了 左 邊 的 3 顆 棋 子.
在 第 43 手.
你 佔 領 了 右 上 角.
在 第 46 手.
我 佔 領 了 右 下 角.
在 第 50 手.
我 錯 下 一 步 棋 使 得 你 佔 領 了 左 邊 的 有 利 位 置.
在 第 52 手.
我 採 取 了 錯 誤 的 策 略 使 得 你 吃 掉 了 左 邊 的 1 顆 棋 子.
在 第 53 手.
你 佔 領 了 左 下 角.
在 第 55 手.
你 沒 有 採 取 正 確 的 策 略 使 得 我 佔 領 了 下 邊 的 有 利 位 置.

在 第 61 手.
你 佔 領 了 左 上 角.
最 後 的 結 果 是.
我 剩 下 23 顆 棋 子 , 你 剩 下 41 顆 棋 子.
你 贏 得 這 一 盤 遊 戲.
本 程 式 謝 謝 你 的 使 用.

6 Conclusion

In this thesis, we have designed and implemented a text generation system for generating commentary on Othello games in Chinese. A framework for text generation is proposed and adopted in our system.

Our system is a complete text generation system that employes three independent phases to produce Chinese text. We hope that our work will give rise to other researches on the topic of text generation in Chinese.

References

[Charniak 1983]

E. Charniak, "A parser with something for everyone", in *Parsing Natural Language,* ed. M. King, Academic Press, London, 1983.

[Davey 1978]

A. Davey, *Discourse Production*,Edingburgh University Press,Edingburgh,1978.

[Derr and Mckeown 1984]

M.A. Derr and K.R. McKeown, "Using Focus to Generate Complex and Simple Sentences", *Proceedings of the 21st Annual Meeting of the ACL* (*COLING* 84), pp.319-326,1984

[Kuo 1989]

H.W. Kuo and J.S. Chang, "Systemic Generation of Chinese Sentences", *ROCLING II*, pp.187-212, 1989.

[Li and Thompson 1982]

C.N. Li and S.A. Thompson, *Mandarin Chinese - A Functional Reference Grammar*, University of California Press, California.

[Liu, Huarng and Hsu 1987]

P.F. Liu, D.H. Huarng and S.C. Hsu, "An Analysis and Implememtation of Othello", *Bulletin of The College of Engineering, National Taiwan University,* No.41, pp.17-26, 1987.

[Mann 1981]

W.C. Mann, "Two Discourse Generators", *Proceedings of the 19th Annual Meeting of the ACL*, pp.43-47, 1981.

[McKeown 1985]

K.R. McKeown, "Discourse Strategies for Generating Natural-Language Text", *Artificial Intelligence* 27, pp.1-41, 1985.

[Moore and Mann 1979]

J.A. Moore and W.C. Mann, "A Snapshot of KDS - A Knowledge Delivery System", *Proceedings of the 17th Annual Meeting of the ACL*, pp.51-52, 1979.

[Moore and Mann 1981]

J.A. Moore and W.C. Mann, "Computer Generation of Multiparagraph English Text", *American Journal of Computational Linguistics* 7:1, pp.17-29, 1981.

[Resenbloom 1982]

P.S. Rosenbloom, "A World-Championship-Level Othello Program", *Artificial Intelligence* 19, pp.279-320, 1982.

[Wilensky 1986]

R. Wilensky, *Common LISPCraft*, W.W.Norton & Company, 1986.

Bi-Lingual Sentence Generation*

Chung-Cherng Chen and Jyun-sheng Chang
Institute of Computer Science
National Tsing Hua University

Abstract

In this paper, we use a single internal representation for *bi-lingual* document generation (English and Chinese) to experiment on the feasibility of a language independent structure. The input is designed based on the *case relation* and the sentence generator is implemented using *systemic grammar*. We augment the systemic grammar with *procedural attachment* to deal with language dependent choices. The system is tested for the application domain of preparing technical manuals with satisfactory results.

1. Introduction

The aim of this paper is to experiment on bilingual generation of technical document.

The input is in some language independent internal representation and the output is sentences in a target language. There are two important issues in this process. One is the structure of internal representation and the other is the design of sentence generator. We will discuss these two issues in the following sections.

* This research is partially supported by National Science Council, grant No. NSC80-0408-E007-13.

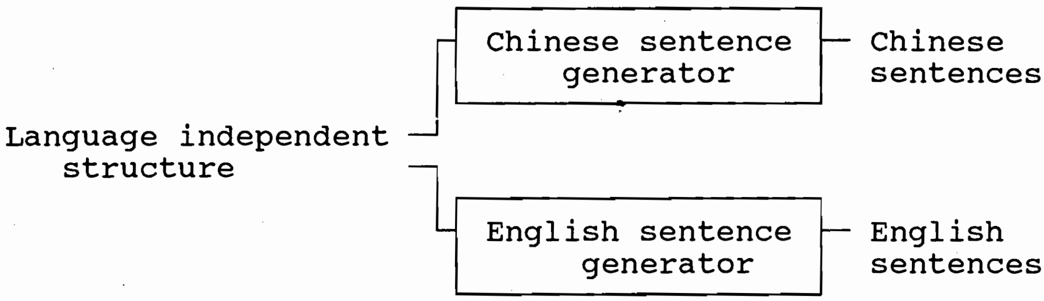


Figure 1. A Bi-lingual Sentence Generator

There are many advantages in using language independent structure in sentence generation. For example, in machine translation, the language-independent structure can be used as the interlingua so that an MT system decomposes into two modules, one for parsing and one for generation. Both modules can be used for translation of other language pairs with the same source or target language. For a text generation system, a knowledge base that yields language-independent structure can be used to generate multilingual document.

A sentence generator usually is based on some kind of grammactical formalism. Several grammactical formalisms have been used for sentence generation, including ATN [Goldman 1975], systemic grammar [Davey 1975, Mann 1973, Patten 1985 and Kuo 1989], and transformational grammar [Mckeown 1979 and Maudlin 1984]. All of these systems deal with one specific target language such as English and Chinese.

This paper continues previous work on Chinese sentence generation [Kuo 1989] and focuses on the following problems:

1. Making the representation more language independent.
2. Extending the scope of sentence generation.

3 Using a single internal representation for bilingual document generation to illustrate the feasibility of a language-independent representation.

This section introduces the problem that this paper sets out to solve and previous work on sentence generation. Section 2 presents detail description of the language-independent structure. Section 3 describes both the Chinese and English sentence generators. Section 4 presents examples to illustrate the entire process of generation; the differences between Chinese and English will be emphasized. Section 5 concludes the paper with a few remarks.

2. Internal Representation

The Chinese sentence generator described in [Kuo 1989] is not very language-independent. It has the following shortcomings.

1. The internal representation is insufficient to represent the events Considering the following two sentences

   張三用鑰匙打開門

   鑰匙打 開門

   The input for sentence (1) is

   Agent : 張三

   Adv : 用鑰匙

   The input for sentence (2) is

   Agent : 鑰匙

" 鑰匙 " plays the same roles but has different input forms in the above two sentences.

2. Users must specify some features particular to the target language. For example, in certain situation, users must specify *ba or passive* which has nothing to do with the semantics. We'll overcome this shortcoming with *procedural attachment.*
3. Considering the following two sentences

他放一本書在桌子上

他在桌子上睡覺

The location in the above sentences has different surface structures. If we are not careful, we'll generate an illegitimate sentence such as

* 他睡覺在桌子上

Removing these shortcomings and making the sentence generator language independent are the goals of this paper. The following sections will show how the problems are resolved.

In the following, we will describe the internal representation used in our sentence generator. The goal of the internal representation is to record the meaning of a sentence. A sentence with more than one meaning should have more than one internal representation. Similarly, sentences with different syntactic structures but the same meaning should get mapped to the same structure.

We adopt case grammar as the basis of our internal representation for the following two reasons:

1 It is fundamentally a theory of meaning.

2 The number of possible cases is quite small, so it is easy to manipulate and to use in internal representation.

Case Grammar and Case Relation

The essence of case grammar is that the semantics of a sentence can be expressed in terms of case relation which is the relationship between noun phrases and the verb. Typical cases are as follows.

1 *Agent* : A noun phrase fills the agent case if it describes the instigator of the action described by the sentence. For example,

John broke the window.

2 *Theme* : An NP that describes something undergoing some change or being acted upon will fill the theme case. For example,

John broke the rock.

3. *Instrument* : An NP is an instrument if it describes a tool, material or force used to perform some event. For example,

Jack saw the ship with the telescope.

4 *Experiencer* : An animate entity is an experiencer if the entity is in a desired psychological state or undergoes some psychological process such as perception. For example,

John saw the unicorn.

5 *Beneficiary* : The case is filled by the animate person for whom a certain event is performed, as in

I gave the book to Jack for Susan.

6 *At-Loc* : This case describes the location where the action happened as in

He sat on the chair.

7 *From* : The case can be further divided into two subcases *from-loc* and *from-poss*, representing the source of a movement action and the original owner respectively.

8 *To* : The case can be further divided into two subcases *to-loc* and *to-poss*, which mean the destination and the new owner respectively.

9 *Time* : This case indicate the time when the action happened.

10 *Direction* indicates the direction of the action.

Case grammar can be stated in the following three rules.

1 Sentence --> Modality + Proposition

2 Proposition --> $V + C_1 + C_2 + \ldots\ldots + C_i + \ldots\ldots + C_n$

3 C_i --> K + NP

The first rule indicates that a sentence consists of *modality* and *proposition*. Proposition is a tenseless set of case relations. *Modality* concerns about *mood* information such as *indicative*, *imperative* and *tense* information such as *present* or *past*.

The third rule states that each case consists of a case marker and a noun phrase. The case marker will be realized as a preposition ('in' and 'at' in English, or ' 把 ' and ' 在 ' in Chinese) or by the position in the surface structure. For example the *Agent* case is usually realized by the *subject* position in the surface structure.

The structure of Verbs

Some cases are intimately related to the verbs. We call them inner cases. Other cases are optional (called outer cases). There are two syntactical property associated with an inner case:

1 An inner case must always appear, For example the verb 坐 has the inner cases *Agent* and *Location*.

他坐在椅子上

2 There are different kinds of case marker associated with an inner case. For example,

老師坐在講臺上 ....normal order

講臺上坐著一位老師 ....locative inversion

老師在講臺上坐著 ....locative preposing

Internal Representation

According to case grammar, we classify the internal representation into three categories: *events, entities*, and *predicates*. Events concern about something that happened. It consists of *modality information* and *case information*. Case information specifies the kinds of information about this event. There are two kinds of case information. They are *semantic cases* and *discourse cases*. Semantic cases express the semantic meaning and the discourse cases conveys the discourse information such as *focus* and *understanding*. The discourse cases affects the syntactical choice and the surface structure of a sentence. Semantic cases can be further divided into two kinds. One is the *basic case* which we discuss in 3.1.1. The other is the augmented case which we used to express more complicated meanings. We added augmented cases to generate bounded sentences while maintaining generality. The following augmented cases are allowed:

1. *Time* : indicating that the subordinate event happened either before, after or at the time of the main event. For example,

   After you decided on a location for your printer, the first step in setting it up is to install the paper feed knob.

2 *Result* : indicating the resultant event of the main event. For example,

   Save it so that we can move the printer with it when we move the printer.

3 *Purpose* : indicating the purpose of the main event.

4 *And-then* : indicating the subsequent event of the main event.

5 *According* : indicating the method of the main event.

These three kinds of case are illustrated in Figure 2 through 4.

The corresponding linear representation is

```
( EventName    ( Event ( Modality Information )
               ( Agent ( Entity .......))
               ( Theme ( Entity .......))
               ( Pred ( Predicate .......))
               ( Experiencer ( Entity .....))
               ( At-Loc ( Entity .......))
               ( To-Loc ( Entity .......))
               ( To-poss ( Entity .......))
               ( From-Loc ( Entity .......))
               ( From-poss ( Entity .......))
               ( Instrument ( Entity ......))
               ( Time ( Entity .........))
               ( Direction ( mod .........))
          )
)
```

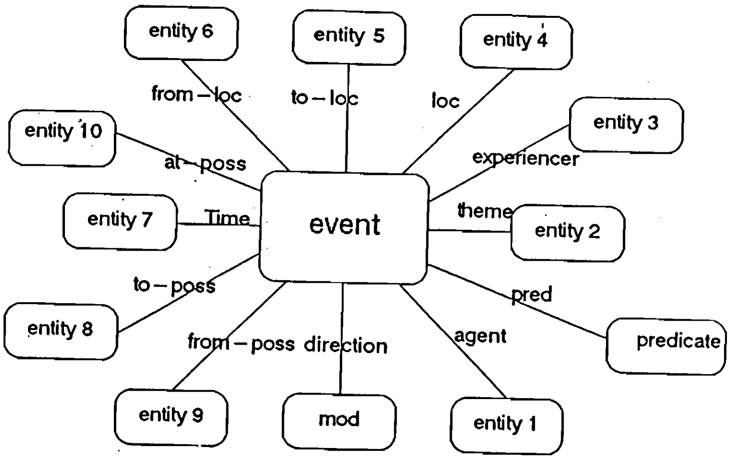


Figure 2. Basic Cases

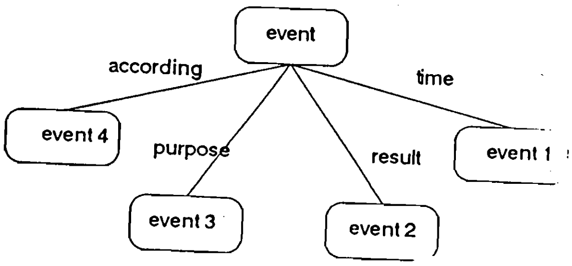


Figure 3. Augmented Cases

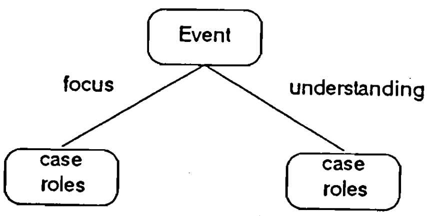


Figure 4. Discourse Cases

Entities can assume a case role such as *agent, location, theme* and *time* and is represented by a head noun, definite/nondefinite information and modifiers such as *adjective, location* and *genitive*. See Figure 5.

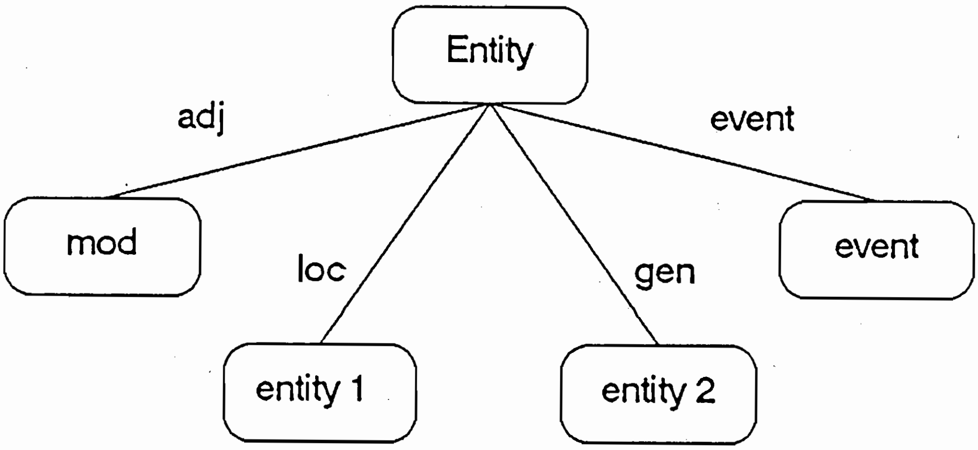


Figure 5. The Representation for an Entity

The corresponding representation is

```
( CaseName      ( Entity Singular/Plural Number Defnite/Nondefinite )
                (Loc ( Entity .......))
                (Gen ( Entity .......))
                (Event  ( Event ........))
                (Adj ( Mod .......))
```

The *Predicate* case represents the kind of action involved in an event. It may be accompanied by modifiers (adverb).

An Example

For example, a sentence from Epson Printer User Manual

Hold both ends of the tracter unit and slowly tilt the unit.

has the semantic representation as shown in Figure 6. And the semantic net can be linearized as follows.

(event1 (*event* imperative present)
(*pred(action* (verb hold)))
(*Theme* (*entity* definite third plural (hn end))
(adj (mod (mod both)))
(*gen (entity* definite third singular (hn tracter))))
(and-then *(event* present)
(*agent (entity* definite second singular pronoun (hn you)))
(*pred (action* (verb tilt))
(modifier (mod (mod slow))))
(*Theme (entity* definite third singular pronoun (hn unit)))))

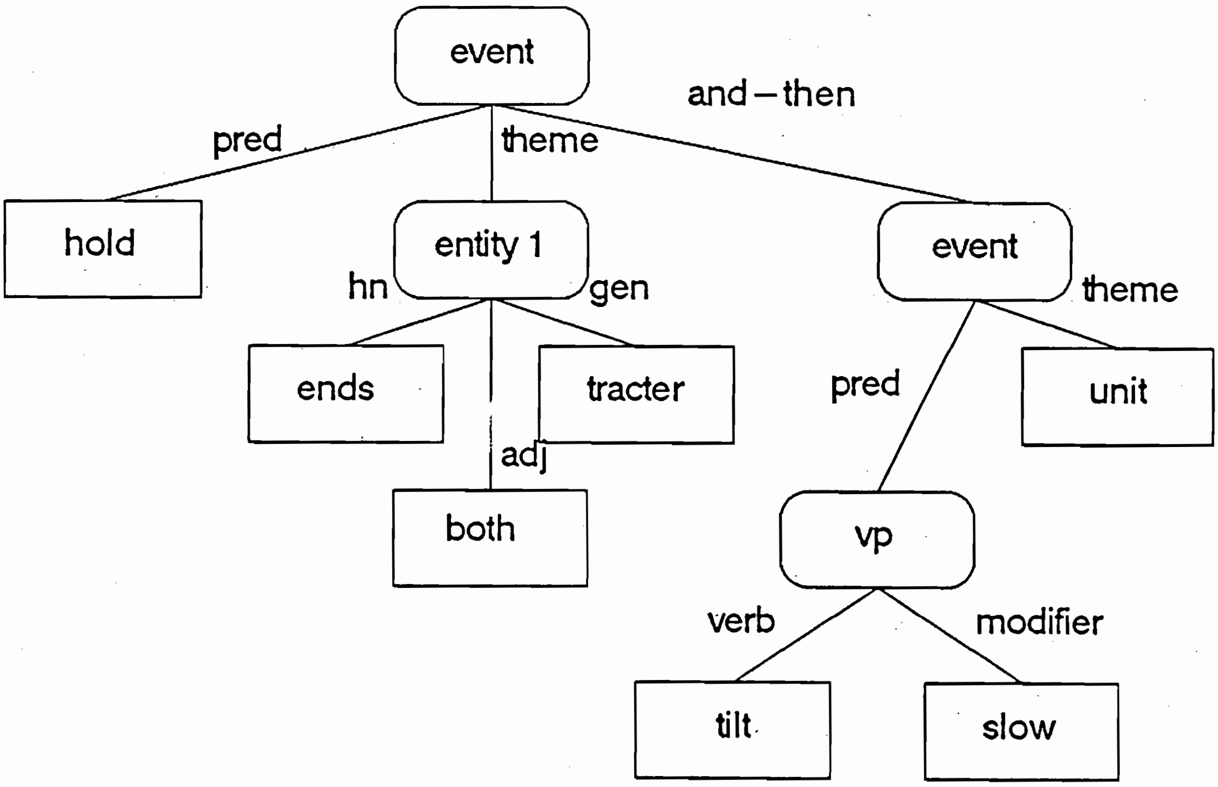


Figure 6. Representing "Hold both ends of the tractor unit and slowly tilt the unit"

3. Sentence Generation

The sentence generator accepts the *semantic cases* and *discourse cases* to generate an adequate sentence. The major mechanism to make syntactical choice is *procedural attachment*. In this paper, we'll show how the sentence generator manipulate these two kinds of cases through attached procedures.

The realization rules in the original systemic network are not sufficient for text generation in English and Chinese deterministically. We have proposed to add some notation to resolve the problems.

The ? Notation

While generating English sentences, we must enforce the *subject verb agreement*. So we need to know the number and singular/plural information about the subject. Here, we define a notation :

?(element feature)	to mean	whether the element has the feature
?('$ feature)	to mean	whether the current element has the feature
?(element 'self)	to mean	whether the element exists

For example *?('agent 'plural)* means whether agent has the *plural* feature; *?('$ 'present)* means whether the current clause has the present feature; *?('agent 'self)* means whether the agent exists.

The # Notation

During the generation of event, we should know whether the event is dependent or independent, whether it is a bounded or relative clause. This information must be obtained from the upper level. So we need a realization rule for *inserting feature*. We define realization rules (# *element feature-list*) as inserting *element* to *feature-list*. For example, when we generate the time- event, we have the realization of (# time (dependent bound time-bound)) which means the clause we generate next will have the feature *dependent, bound, time-bound.*

The Grammar

According to the internal representation, we classify the network into three levels : *event, object, predicate*. Each level of representation is handled by the corresponding network. We will sketch the grammar for both Chinese and English in the following. Refer to [Chen 1990] for more detailed description.

The Action-type System (Chinese)

The *action-type system* processes the action of the event. The philosophy that we take in action-type system is the **classification of verbs**. By classifying the verbs, we know the inner cases and the corresponding syntactical choice. We can manipulate the inner case and choose the adequate sentences according to the discourse function. The action-type system consists of four subsystems, and we will illustrate only the action and aux subsystems in the following.

The Action System (Chinese)

The action system distinguishes a *tran-action (transitive)* system from an *intran-action (intransitive)* system. The *intran-action* system consists of *vocal intran-action-location* and *intran-locomotion* system.

The *intran-location* system can have three syntactical choices, *normal-order, locative-inversion and locative-preposing*. For example

老師坐在講臺上　　　normal order

講臺上坐著一位老師 locative inversion

老師在講臺上坐著 locative preposing

The condition for the locative inversion is *agent nondefinite* and *loc definite*. This means that agent is the new information and location is the old information. It is often called *presentative clause*.

The condition for locative preposing is *agent* and *loc definite*. We realize these conditions by attaching a procedure to the system of *intran-location*. The following is the corresponding code:

```
( cond      ( ( and ?( 'Agent 'Definite ) ?( 'Loc 'Definite ))      '( Locative-Preposing ))
            ( ( and ?( 'Agent 'Nondefinite ) ?('Loc 'Definite )) '(  Locative-Inversion))
            ( T                                                     '( Normal-Order )))
```

Similarly, the *Tran-Action* consists of *Regular* and *Tran-Location* system. The *regular* system has the following four syntactical choices.

normal :

老闆開除了他

topic :

condition : Theme is definite

他老闆開除了

preverbal :

condition : 1 Theme is definite

2 Verbs contains disposal feature.

老闆把他開除了

passive :

condition : Focus is Theme

他被老闆開除了

and the procedure attached is:

```
( cond    ( ?( 'Focus 'Theme )                               '( passive ) )
          (( and ?( 'Theme 'Definite ) ?( '$ 'Disposal ))    '( preverbal)
          ( T                                                '( Normal )))
```

Similar work is done for the *tran-location system.*

The Event Network (English)

One of the main differences between the English network and the Chinese network is *voice* and *subject verb agreement*. The voice system will select the active or passive system according to *focus*. The *active* system consists of *select-subj* and *theme-obj* systems. The select-subj system will select the *subject* from *agent, experiencer* and *instrument*. If the *subject* is *the agent*, then there are two systems included: *agentnn and agentqq*. Agentnn has three alternatives, *subj-first, subj-second* and *subj-third*. We attach to agent-nn a procedure to choose one from the alternatives.

```
( cond    ( ?('agent 'first )       '( subj-first ) )
          ( ?('agent 'second )      '(subj-second ) )
          ( ?('agent 'third)        '(subj-third)))
```

Similar work is done for agent-qq which manipulates the singular/plural features.

4. Examples

In the following, we show a list of sentences used in the experiment. The sentences are listed in the sequence of (1) an original Chinese sentence (2) the generated Chinese sentence (3) the original English sentence (4) the generated English sentence.

1.1 當 你 打開 印表機 的 包裝材料 時 請依 下 圖 所示檢查 是否 所有 的配件 齊全了

1.2 當 你 打開 印表機 的 包裝材料 時 檢查 是否 你 有 下 圖 顯示 的 所有 配件

1.3 After you unpack the printer, check that you have all parts shown below .

1.4 when you open the packaging materials of the printer check whether you have all parts that the following figure shows

2.1 拿出 配件 後 將 包裝材料 保存 以便 將來 搬運 印表機 時 用 得 著

2.2 當 我們 拿掉 配件 以後 保存 包裝材料 使得 將來 我們 能 用 它 搬運 印表機

2.3 After removing the parts, store the packaging materials in case you ever need to transport your printe.

2.4 after we remove the PARTS store the packaging materials so that we can transport the printer with it later

3.1 拿掉 拖曳式 牽引器

3.2 拿掉 拖曳式 牽引器

3.3 Remove the pull tractor

3.4 remove the pull tractor

4.1 為了便利 自我測試功能 的執行 拿掉 拖曳式 牽引器

4.2 拿掉 拖曳式 牽引器 使得 將來 我們 能 執行 自我測試

4.3 In preparation for performing the self test later, remove the tractor.

4.4 remove the pull tractor so that we can perform the self test later

5.1 將牽引器 的 蓋子 直立 然後 向上 提起

5.2 將 牽引器 的 蓋子 直立 然後 將 它 向上 提起

5.3 Raise the tractor cover then lift it up.

5.4 raise the cover of a tractor and-then lift it up

6.1 拿掉 插 在 牽引器 的 兩 端 的 包裝材料

6.2 拿掉 插 在 牽引器 的 兩 端 的 包裝材料

6.3 Remove the packaging materials inserted between both sides of the tractor

6.4 remove the packaging materials that•is inserted between both sides of the tractor

7.1 保存 它 以便 再 搬運 印表機時 用得著

7.2 保存 它 使得 當 我們 搬運 印表機 時 我們 能 用 它 搬運 印表機

7.3 Store it in case you ever need to transport the printer.

7.4 store it so that when we transport the printer we can move the printer with it

8.1 握住 牽引器 的 兩 端

8.2 握住 牽引器 的 兩 端

8.3 Hold both ends of the tractor.

8.4 hold both ends of the tractor

9.1 慢慢地 將 它 向後 傾斜

9.2 將 它 向後 慢慢 傾斜

9.3 Slowly tilt it back

9.4 slowly tilt it back

10.1 牽引器 的 缺口 會 從 扣針 鬆開

10.2 牽引器 的 缺口 會 從 扣針 鬆開

10.3 The notches of the tractor will snap free from the mounting pins

10.4 the notches of the tractor will snap free from the mounting pins

11.1 把 牽引器 向上 提起

11.2 將 牽引器 向上 提起

11.3 Lift the tractor up

11.4 lift the tractor up

5. Conclusions

In this paper, we focus on the topic of language independent. We adopt the case relation to be our input and we use procedural attachment to deal with the language dependent choice. We implement both the English and Chinese Systemic Grammars to demonstrate that our input is in some way language independent.

The domain that we test our system is the technical manual, Epson user's guide for the LQ-500 printer.

Our system can be improved in the following respect:

(1) Extending the *scope* of the grammar : We implement just a subset of English and Chinese grammar. There are still many grammatical phenomena we didn't implement such as progressive, perfect tense.

(2) Improving the *procedural attachment* mechanism: We can improve the procedural attachment mechanism to make the sentences generated more elegant. For example, We can realize the focus of event using *cleft* sentences instead of using only *passive* sentences in Chinese sentence generation.

(3) Adding a *preprocessor*: Consider the sentences below.

    Unpack the printer.

    打開 印表機 的 包裝材料

    In order to generate this pair of sentences, we will need a preprocessor between the input and the sentence generator to deal with difference in lexical expression in different target language.

(4) Our system now generates a clause for the internal representation of an *event*. We can add variety to the text by generating a different kind of grammatical

constituent. For example, we can generate with a noun phrase in a prepositional phrase for a *result-event* as in Sentence 4.3.

References

[Chen 1990]

C.C. Chen, *A Multi-Lingual Sentence Generator Based on Systemic Grammar*, Master Thesis, Department of Computer Science, National Tsing Hua University, Hsinchu, Taiwan, 1990.

[Davey 1978]

A. Davey, *Discourse production*, Edinburgh University Press, Edinburgh, 1978.

[Fillmore 1971]

C.J. Fillmore, *Some Problems for Case Grammar*, Geogetowm University Monograph Series of Languages and Linguistics 22. PP 35-36.

[Goldman 1975]

N.M. Goldman, *Sentence Paraphrasing from a Conceptual Base*, CACM 18, pp. 96-106.

[Halliday and Hansan 1976]

M.K.A. Halliday and R. Hansan, *Cohesion in English*, Longman, New York, 1976.

[Mann 1983]

W.C. Mann, *An Overview of the Nigel Text Generation Grammar*, Proceedings of the 21st Annual Meeting of the ACL, pp.79-84, 1983.

[Maudlin 1984]

M.L. Maudlin, *Semantic Rule Based Text Generation*, COLING 84, pp. 376-380, 1984.

[Mckeown 1979]

K.R. McKeown, *Paraphrasing Using Given and New Information in a Question-Answering System*, Prooceedings of the 17th Annual Meeting of the ACL, pp.67-72, 1979.

[Kuo 1989]

H.W. Kuo and J.S. Chang, *Systemic generation of Chinese Sentences*, ROCLING II, pp. 187-212, 1989.

[Li-Thomson 1982].

C.N.Li and S.A. Thompson, *Mandarin Chinese - A Functional Reference Grammar*, University of California Press, California, 1982.

[Patten 1985]

T. Patten, *A Problem Solving Approach to Generating Text from Systemic Grammars*, Proceedings of the 2nd Conference of the European Chapter of the ACL, pp. 251-257, 1985

[Tang 1975]

T.C.C. Tang, *A Case Grammar Classification of Chinese Verbs*, Hai-Guo Book Company, Taipei, Taiwan 1975.

[Woods 1970]

W.A. Woods, *Transition Network Grammars for Natural Langual Analysis*, Computational Linguistics 13, No.10, pp. 591-606, 1970.

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

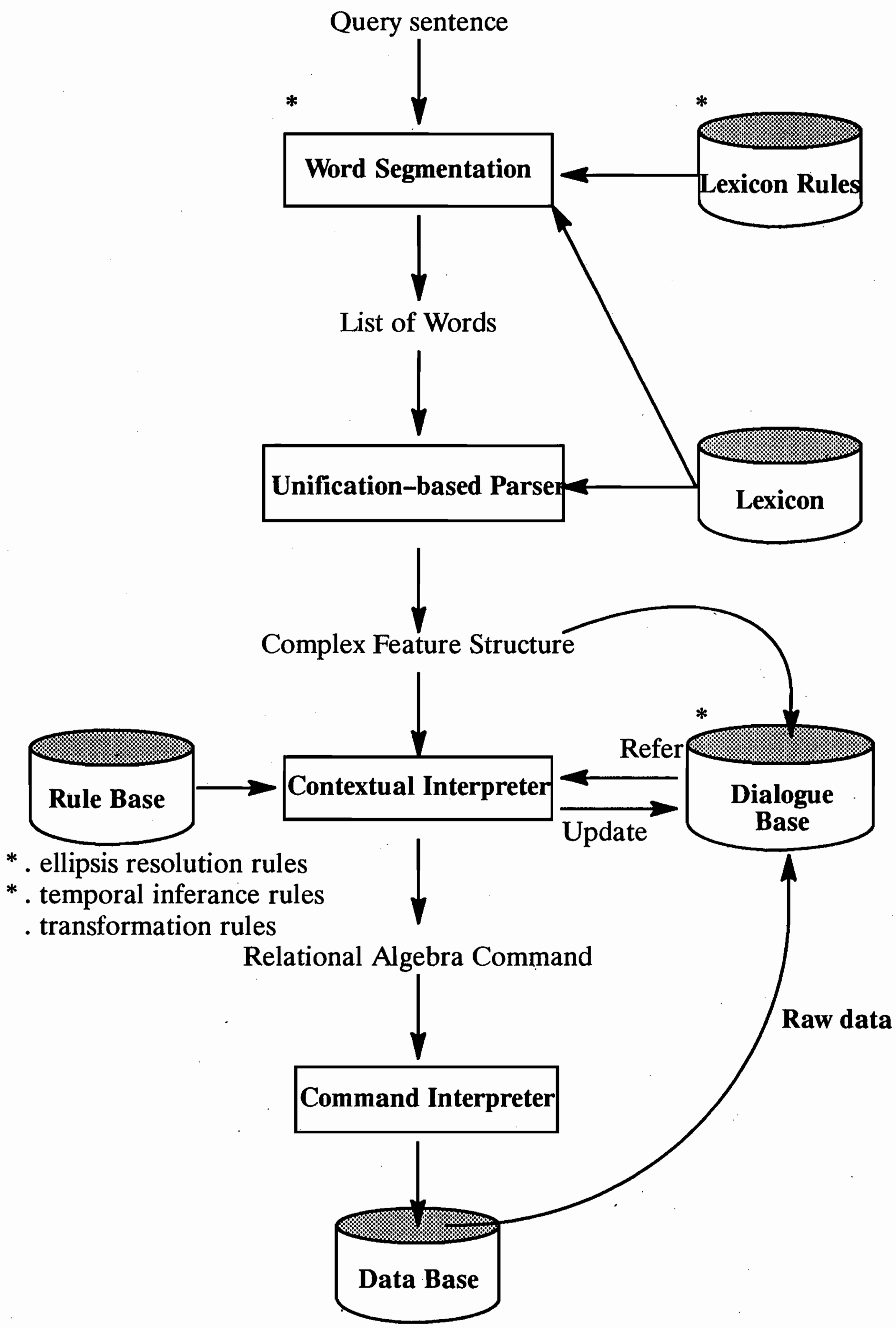


Figure 1. System architecture of CIDA.

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.

$$
\begin{array}{llll}
[\text{phon} & \alpha & & \\
\ \text{syn} & [\text{head} & [\text{maj} & \beta_1, \\
 & & \ \text{adjunct} & \beta_2, \\
 & \ \text{subcat} & \beta_3]], & \\
\ \text{sem} & \gamma] & &
\end{array}
$$

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 feature 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.