修士論文

A Chinese Deterministic Dependency Parser Augmented by Sentence Division

長文分割を用いた決定性の中国語依存構造パーサに関する研究

指導教員　江崎 浩 教授

東京大学大学院 情報理工学系研究科 電子情報学専攻

56444　**劉 浩**

平成19年2月2日作成

Abstract

Natural language syntactic parsing is a program that can perform the appropriate syntactic analysis of a language sentence. It is one of the important problems to many natural language processing tasks such as machine translation, information extraction, and question answering. There has been a steadily increasing interest in syntactic parsing based on dependency analysis in recent years. Many syntactic analyzers for English have been implemented and have demonstrated good performance [1] [2] [3]. However, implementation of Chinese syntactic structure analyzers is still limited, since the structure of the Chinese language is quite different from other languages. Therefore the experience in processing western languages can not be guaranteed that it can apply to Chinese language directly [4]. In addition, long sentence parsing is one of the remained problems of parsing for many languages, since if the sentence is longer, the more ambiguities will be generated during the parsing, especially for Chinese language. In this thesis, we present a Hierarchical Deterministic Dependency method augmented by sentence division for sloving the Chinese long sentence parsing problem and to improve dependency structure analysis of Chinese.

Our parser works as follows:

(1) We propose a sentence division method to divide long Chinese sentence into short clauses: After the usage and function of Chinese punctuations are studied in syntactic parsing, we find that the Chinese punctuations can be classified as divide punctuations and ordinary ones, in which divide punctuations can seem like the boundary or partition among the short clauses. That means long sentences which include divide punctuations can be broken into suitable units.

(2) Then it does deterministic dependency parsing for each divided clause by SVM classification, which is one of the binary

classifiers based on maximum margin strategy introduced by Vapnik [5].

(3) At last, the analyzed dependency structure of each clause is combined together with their respectively analyzed root node and the sentence main root node classified by the root finding classifier.

Except for the dependency parser, we also propose a morphological analysis system, which does the work of word segmentation and Parts-Of-Speech tagging for the Chinese sentence. We use the Penn Chinese Tree Bank 5.1 [6] for our main experiment data source and the experimental evaluation on the Chinese Tree Bank Corpus shows that the proposed extensions improve the parsing accuracy significantly.

Contents

Chapter 1

Introduction

1.1 Background

Natural language parsing is one of the important problems to many natural language processing tasks such as machine translation, information extraction, and question answering. The availabilty of high accuracy parsers will provide a platform for developing new applications and help such tasks achieve better performence. The process of determining one or more appropriate syntactic structures corresponding to an input sentence is commonly referred to as syntactic parsing. It is a hard problem. Many decades have passed since the parsing research started, different approaches have demonstrated some amount of success in different aspects, but the problem has not been solved very well.

The natural language is difficult because the natural language contains ambiguities. Natural language sentences have many structural ambiguities, and the language ambiguities effect on the language parsing significantly. Even human beings have difficulties in resolving these ambiguities, if they are reading a sentence such as "*He saw the man with the telescope*". Because the meaning of a natural language sentence varies depending on its context, human can not recognize the preposition phrase "*with the telescope*" modifies the "*man*" or the verb "*saw*", unless the context of the sentence is given.

Because a machine has little knowledge about the syntax of natural language, it may face much more difficulties in resolving the syntactic ambiguity of natural language sentences. In the recent development of parsing technology, the linguistic knowledge is encoded in the form of the grammar, lexicon, and ontology, and it is useful for resolving syntactic ambiguities by using the selectional restrictions defined in them. The selectional restrictions are specifications of the legal combinations of words that can co-occur and be appiled to emliminate wrong analysis constructed by a parser. However, selectional restrictions may have several problems, when it is used for parsing general-domain and wide-coverage tasks.

> (1) First of all, as the vocabulary size gets larger, the required linguistic information becomes larger. Thus, the incredible amount of manual labor is expected to build the whole necessary restriction set.
>
> (2) Another problem is that the all-or-nothing nature of selectional restriction makes it impossible to encode semantic preferences.
>
> (3) Further, in linguistics, the selectional restrictions have been known to have theoretical weakness.

Because of the reseachers' efforts and the development of computer technology, statistical techniques have been suggested for solving these problems. These approaches have become very popular, since a large syntactically annotated corpus or Tree Bank became available. With the advent of them, statistical approaches to natural language parsing have quickly developed. By providing a very large corpus of manually labeled parsing example, machines are trained and learn linguistic preferences from the annotated corpus, completely automatically. Some of these approaches were found to be successful and applied to play an invaluable role in enabling the broad analysis, automatic training and quantitative evaluation of parsing techniques.

Typically, the input unit that most parsers deal with is a sentence, and analysis is performed one sentence at a time. In some cases, parsers expect the input sentence to be tagged with parts-of-speech (POS) tags, such as Noun, Verb, Adjective, etc. Once a parser is given a sentence as input, it attempts to produce a syntactic structure of the sentence as output. The syntactic structure may take several forms, such as a constituent structure, and a dependency structure. Figure 1.1 shows the examples of each of these two syntactic representations using the same sentence "*I saw the boy in the park*". Dependency structure and constituent structure are two mainly different ways of representing the structure of sentences: while a constituent structure parse represents nesting of multi-word constituents, and a dependency structure parse represents dependencies between individual words. These various representations of a sentence may differ in the level of information they contain (parts-of-speech, morphology, case, syntactic function labels, etc.), but each of them describes in some way the structure of how words combine to form a sentence. The choice of a particular representation format depends mainly on the syntactic analyses will serve.

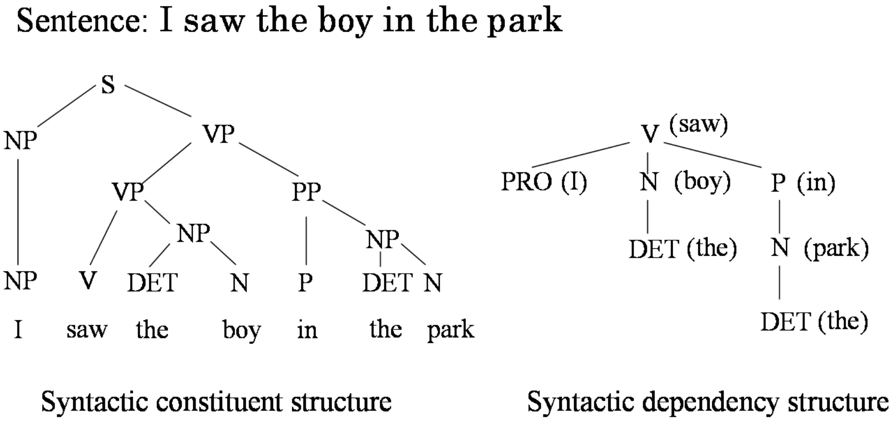


Figure 1.1: Constituent structure and dependency structure.

Dependency parsing has emerged as an attractive way of performing automatic syntactic analysis over the past decade [7] [8]. In dependency parsing, syntactic structure is represented by a dependency tree, where each word is a node, and the existence of an edge between two nodes indicates a syntactic relationship between the two words corresponding to the two nodes. Thus, a complete dependency structure provides a fully disambiguated analysis of a sentence, this analysis is typically less complex than in frameworks based on constituent analysis and can therefore often be computed deterministically with reasonable accuracy. Deterministic methods for dependency parsing have now been applied to a variety of languages, including Japanese by Kudo and Matsumoto [9], English by Yamada and Matsumoto [8], and Swedish by Nivre [10]. In Chinese, dependency analysis has been shown to be a powerful syntactic parsing method because the order of phrases in a Chinese sentence is relatively free compared with English. Zhou [11] proposed a rule based Chinese dependency parser. Ma [12] proposed a statistic dependency parsing approach by using probabilistic model. Nivre et al., [13] also use their deterministic dependency analysis algorithm on Chinese and demonstrate good performance.

1.2 Proposed Method

Long sentence parsing has been a critical problem because of high complexity. The longer a sentence becomes, the more ambiguities are generated during the parsing. Some researchers tend to use long distance information, which is different from only using local feature, for parsing [14][15]. But the incorrect previous parsing operations may bring severe affection to the following operations, and the later accuracy will be affected extraordinary consequently. In this thesis, we present a Hierarchical Deterministic Dependency method augmented by sentence division for sloving the Chinese long sentence parsing problem and to improve dependency structure analysis of Chinese. Our parser works as follows: (1) We propose a sentence division method to divide long Chinese sentence into short clauses: After the usage and function of Chinese punctuations are studied in syntactic parsing, we find that the Chinese punctuations can be classified as divide punctuations and ordinary ones, in which divide punctuations can seem like the boundary or partition among the short clauses. That means long sentences which include divide punctuations can be broken into suitable units. (2) Then it does deterministic dependency parsing for each divided clause by SVM classification. (3) At last, the analyzed dependency structure of each clause is combined together with their respectively analyzed root node and the sentence main root node classified by the root finding classifier.

Except for the dependency parser, we also propose a morphological analysis system, which does the work of word segmentation and Parts-Of-Speech tagging for the Chinese sentence in preprocess. We use the Penn Chinese Tree Bank 5.1 for our main experiment data source and the experimental evaluation on the Chinese Tree Bank Corpus shows that the proposed extensions improve the parsing accuracy significantly.

1.3 Paper Organization

This thesis is organized as following. In Chapter 2, the difficulties in parsing Chinese are described. We then explain the hierarchical deterministic dependency parsing strategy in Chaper 3, in this Chapter, we present our special solution to word segmentation & parts-of-speech tagging, sentence division, and parsing combination, which significantly affects the parsing accuracy. Experiment results and discussion are reported in Chapter 4. The conclusion and future work are given in Chapter 5.

Chapter 2

Difficulties In Parsing Chinese

China is the only country in the world with a literature written in one language for more than 3,000 consecutive years. This continuity results largely from the nature of the written language itself. Unlike English words which are composed of letters, written Chinese words are made up of characters and it is also popularly believed that Chinese characters represent words.

Many Chinese characters appear to have originated as depicting concrete objects. For example, the Chinese character "人" meaning "*person*", originated from a pictogram of a man; and the concepts "*trust*" is represented by the character "信", which is a combination of "*man*" and "*speech or words*". In Japan and Korea, Chinese characters were adopted and integrated into their languages and became "*Kanji*" and "*Hanja*", the names are Japanised and Koreanised pronunciations of "*Hanzi*". By now, Japan still uses "*Kanji*" as an integral part of its writting system.

Most Chinese words are composed of two characters, but this is not unique to Chinese, some are commonly made up of one, three, four or more characters. For example, the English word "*undoable*" is made up of three morphemes which mean "*not*", "*do*", "*able*", and in much the same way, Chinese word "作不完"(*undoable*) is composed of three characters or morphemes meaning "*do*", "*not*", "*finish*". In English, words are delimited by white space, but in Chinese, sentences are represented as strings of Chinese characters without natural delimiters illustrated in Figure 2.1. That brings the difficulty in Chinese morpholgical analysis, represented by the word segmentation.

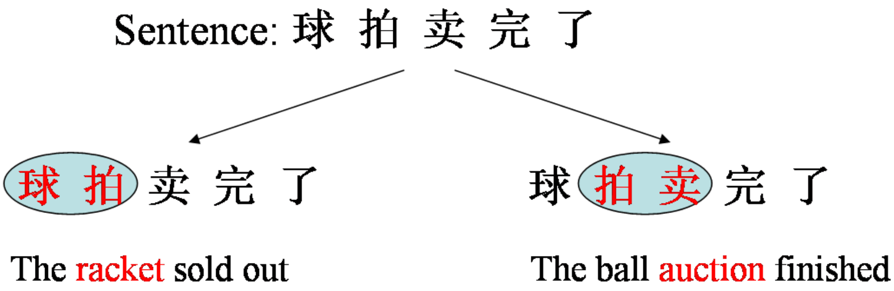


Figure 2.1: Difficulty in Chinese Word Segmentation.

Many works [16][17] on Chinese language have observed that Chinese has some very unique structural properties from the perspective of language like English, French and German. Chinese and English are both isolating languages: they rely primarily on relatively rigid phrase structure rather than rich morphological information to encode functional relations between elements. For purposes of statistical parsing, three salient differences distinguish the two languages.

(1) All words have only one grammatical form:

One key feature of Chinese grammer is that all words have only one grammatical form, as the Chinese language lacks conjugation, declension or any other inflection. Functions such as number in nouns or tense in verbs are expressed through word order or particles, while in English, nouns might to be distinguished by singular and plural just as "*woman*"(女人) and "*women*"(女人) or verbs distinguished by number or person as "*I go*"(我去) and "*He goes*"(他去). Chinese grammar may appear quite simple compared to that of many high-inflected western language or even the low-scale verb conjugations, for example the English words "*sing*", "*sang*" and "*sung*" are presented by a same Chinese word "唱歌". Because of the lack of inflections, Chinese words may serve different grammatical functions in different context. For example, the word "攻擊" may function as a subject in the sentence "攻擊並不成功" and also function as an adjective as in "攻擊火力不足" [18]. Therefore Chinese display a very high level of complexity in Syntax.

(2) Strong subject pro-drop tendency:

Secondly, the major difference is subject pro-drop: the null realization of uncontrolled pronominal subjects, which is widespread in Chinese, but rare in English. For example, in the sentence " 昨天上班, 明天休息"(Today work, tomorrow rest). It is an example of a pro-drop sentence. The subject of this sentence("*we*" or "*I*") would be determined by the context. This creates ambiguities between parses of subjectless structures as IP or as VP, and between interpretations of preverbal NPs as NP adjuncts or as subjects.

(3) The linear order of grammatical function roles are relatively free:

Chinese is a topic free language [16], and the topic can be either before or after subject. For example, the following three sentences have the same meaning "*I have written the novel.*".

a 我 写完了 小説 。
b 小説 我 写完了 。
c 我 小説 写完了 。

Chapter 3

Hierarchical Deterministic Dependency Parsing Strategy

3.1 Hierarchical Parsing Strategy

In this Chapter, we propose a hierarchical parsing strategy for dependency analysis of Chinese language sentence. Figure 3.1 described the flow chart of our method.

If a Chinese sentence “中国証券市場在改革開放中誕生, 経過探索和発展, 已経有了一定規模。” is given as the input, (1) the word segmentation and Parts-Of-Speech tagging of the input sentence should be done first by a morphological analysis system which does the input preprocess work and is introduced in the section 3.2. (2) During sentence segmentation, Chinese sentence will be divided into short clauses by the divide punctuations, which are classified out by a sentence segmentation analyzer proposed in the section 3.3. (3) During the clause parsing, the divided clauses are parsed separately by using the Chinese deterministic dependency analyzer whose design principle is introduced in section 3.4. (4) Finally, the analyzed dependency structure of each clause is combined together with their respective root node and the sentence main root node determined by the root node finding analyzer “$root\ finder$”, which is proposed in the section 3.5.

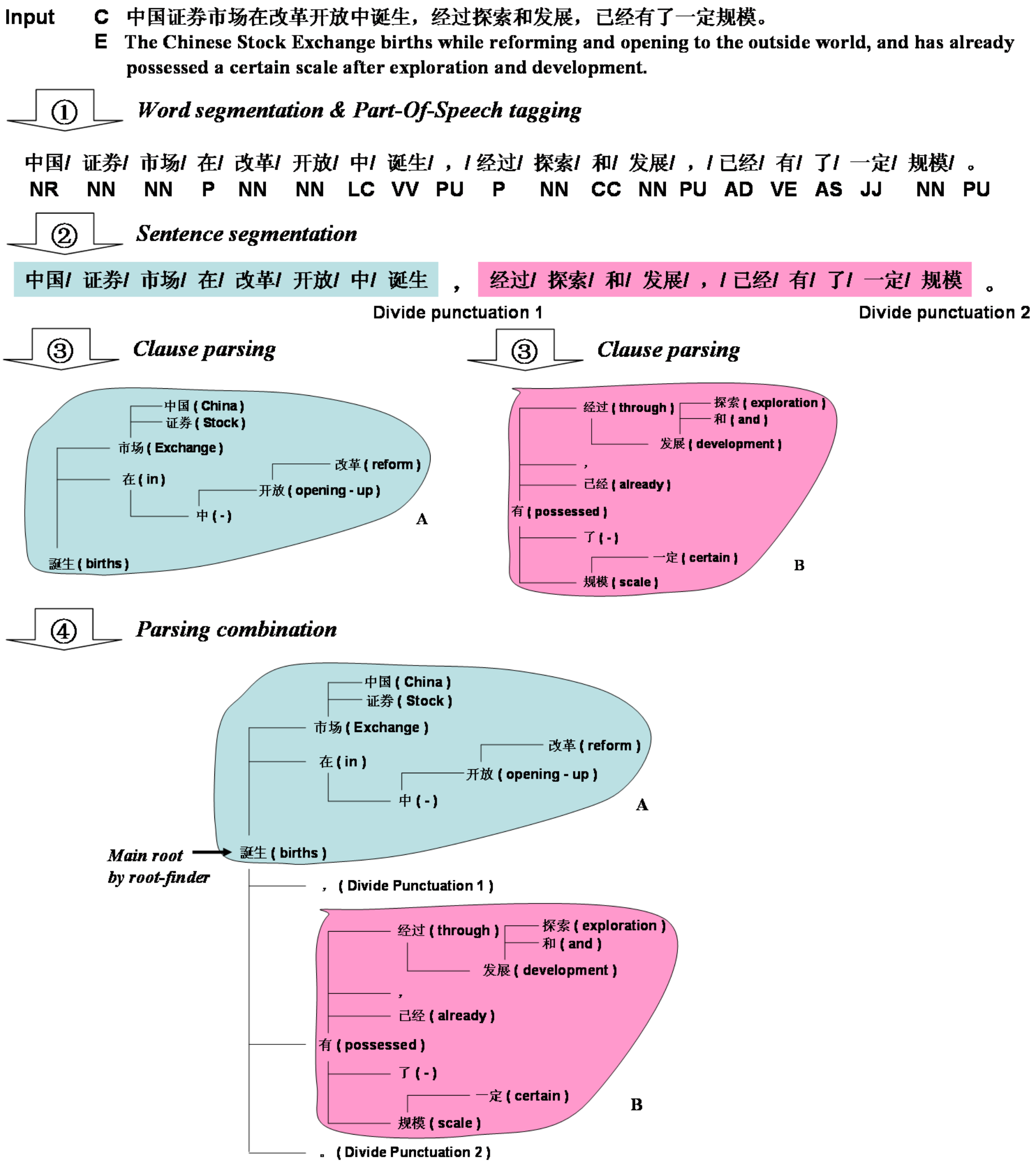
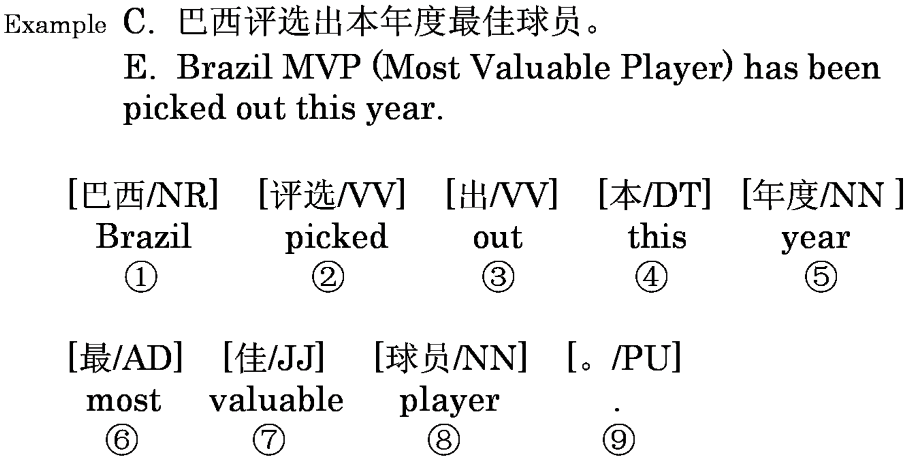


Figure 3.1: The flow chart of Hierarchical Parsing Strategy.

3.2 Word Segmentation & Parts-Of-Speech Tagging

Chinese word segmentation and Parts-Of-Speech tagging is the basic task for Chinese language processing. Generally, Word Segmentation is a system that identifies the sequence of words in a sentence and mark boundaries in appropriate places; and the Parts-Of-Speech tagging is a system that uses context to assign parts of speech to the words, it involes selecting the most likely sequence of syntactic categories for the words in a senetence. In English text, sentences are sequences of words delimited by white space, but in Chinese text, sentences are represented as strings of Chinese characters or *hanzi* without natural delimiters. Figure 3.2 illustrates the Word segmentation and Parts-Of-Speech tagging example of a Chinese sentence.



Example C. 巴西评选出本年度最佳球员。
E. Brazil MVP (Most Valuable Player) has been picked out this year.

[巴西/NR]	[评选/VV]	[出/VV]	[本/DT]	[年度/NN]
Brazil	**picked**	**out**	**this**	**year**
①	②	③	④	⑤

[最/AD]	[佳/JJ]	[球员/NN]	[。/PU]
most	**valuable**	**player**	**.**
⑥	⑦	⑧	⑨

Figure 3.2: Word segmentation and Pos tagging of Chinese sentence.

In our system, we treat word segmentation and Parts-Of-Speech tagging as one character tagging problem. Instead of segmenting a sentence into word sequences directly, characters are first assigned with position tags, and based on these position tags, the characters are converted into word sequences. Equation 3.1 shows the basic model for Chinese morphological analysis, with which the tag sequence T^* with the highest probability is found when given a sequence of characters or words $S = c_1c_2...c_n$.

$$T^* = \operatorname*{argmax}_T P(T|S) \tag{3.1}$$

We assume that the tagging of one character is independent of each other, so we modify Equation 3.1 to Equation 3.2.

$$\begin{aligned} T^* &= \operatorname*{argmax}_{T=t_1t_2...t_n} P(t_1t_2...t_n|c_1c_2...c_n) \\ &= \operatorname*{argmax}_{T=t_1t_2...t_n} \prod_{i=1}^{n} P(t_i|c_i) \end{aligned} \tag{3.2}$$

Our morphological analysis system uses SVM framwork and it defines the tag set as the cross of word boundary tag set and Parts-Of-Speech tag set from the surrounding context. Table 3.1 shows the tag set defined in our word segmentation and Parts-Of-Speech tagging system.

Word segmentation Tag		*Pos Tag*
B, I, E, S (4)	×	NN, CC, P, ... (33)

Table 3.1: Tags of word segmentation and pos tagging.

In the word segmentation, each Chinese character can be assigned one of 4 possible boundary tags as B, I, E, S shown in Table 3.2. Parts-Of-Speech tag set is based on the Penn Chinese Tree Bank Parts-Of-Speech definition, which includes 33 kinds of Parts-Of-Speech tags listed in Table 3.3.

Tag	*Description*
B	a character that leads a word and is followed by another character
I	a character that occurs in the middle of a word
E	a character that ends a word
S	a character that can serve as a single character word

Table 3.2: Position tags in word segmentation.

The feature templates used in our Word Segmentation and Parts-Of-Speech tagging analyzer are shown in Table 3.4. C refers to a Chinese character, which is contained in a *Character list* and W means a Chinese word which is contained in a *Lexiocn list*. C_0 denotes the focused character.

AD	adverb	还
AS	aspect marker	着
BA	把 in ba-construction	把, 将
CC	coordinating conjunction	和
CD	cardinal number	一百
CS	subordinating conjunction	虽然
DEC	的 in a relative-clause	的
DEG	associative 的	的
DER	得 in V-de const. and V-de-R	得
DEV	地 before VP	地
DT	determiner	这
ETC	for words 等, 等等	等, 等等
FW	foreign words	ＩＳＯ
IJ	interjection	啊
JJ	other noun-modifier	男, 共同
LB	被 in long bei-const	被, 给
LC	localizer	里
M	measure word	个
MSP	other particle	所
NN	common noun	书
NR	proper noun	美国
NT	temporal noun	今天
OD	ordinal number	第一
ON	onomatopoeia	哈哈, 哗哗
P	preposition excl. 被 and 把	从
PN	pronoun	他
PU	punctuation	、？。
SB	被 in short bei-const	被, 给
SP	sentence-final particle	吗
VA	predicative adjective	红
VC	是	是
VE	有 as the main verb	有
VV	other verb	走

Table 3.3: Parts-Of-Speech Tag Definition of Penn Chinese Tree Bank.

Feature C_n denotes the character in n positon on the left and right of the currently focused character. If the character sequence is given as “上海市 政府”(the goverment of ShangHai city), when the character “市”(city) is being considered, In feature template C_n, the following features $C_{-2} =$ 上, $C_{-1} =$ 海, $C_0 =$ 市, $C_1 =$ 政, $C_2 =$ 府 will be set. Feature $IsPu$ (C_0) judges whether the focused character is a punctuation enumerated in a *Punctuation list*.

Feature $W_n (n = -1, 0, 1)$ denotes all of the word in the lexicon, which contain the character C_0, and the words on the left and right of the chosen

$Feature$	$Description$
C_n	characters in different position, n=-2, -1, 0, 1, 2.
$IsPu(C_0)$	if C_0 is a punctuation character, then set it as 1, others set it as 0.
W_n	lexicon words in different positions, n=-1, 0, 1. (W_0 is the word containing the character C_0)

Table 3.4: Features description of word segmentation and Parts-Of-Speech tagging.

word in the character sequence. All of the features are binary features.

Three lists are used in this step.

(1) *Character list*
(2) *Punctuation list*
(3) *Lexicon list*

The *Character list* contains all of the distinct characters extracted from the *lexicon list*. The punctuations in the *Punctuation list* come from Penn Chinese Tree Bank 5.1, which is a segmented, parts-of-speech tagged, and fully bracketed corpus. To creat the *Lexicon list*, we use the NICT[1] as the basic lexicon. The NICT lexicon is extracted from Peking University Corpus of the second SIGHAN Bakeoff, Penn Chinese Tree Bank version 4.0 and part of the NICT Corpus. We also add all of the words in Penn Chinese Tree Bank version 5.1 into the lexicon list.

3.3 Long Sentence Division

Long sentence analysis has been a critical problem in language parsing because long sentences contain high complexity and ambiguities. Chinese is a language which has no case maker and less morphology. Usually in Chinese sentences, subordinate clauses and coordinate clauses are not connected with any conjunctions. Because of these characteristics, Chinese has a rather dif-

---

[1] National Institute of Information and Communications Technology, Japan

ferent set of salient ambiguities from the perspective of parsing. Adopting a shorter sentence unit than a long sentence as a parsing unit will achieves an efficient parsing result. In this section, the Long Sentence Division method to divide long sentences into clauses is proposed.

In written Chinese, the punctuation is used very frequently. In Penn Chinese Tree Bank 5.1 [6], the average use of punctuation in one sentence is 4.1[2]. Since Chinese has no case maker and less morphology, moreover the punctuation is used frequently, the punctuation becomes an important element in long Chinese segmentaion.

There are fourteen marks in common usage in Chinese language, only *period* ".", *quesrion mark* "?", *exclamation mark* "!", *comma* ",", *semicolon* ";", and *ellipsis* "......" are disscussed in this part. The former three are sentence terminators, and the latter three are segment separators.

> (1) *Period* "." is placed at the end of a sentence to indicate that the meaning of a sentence is complete.
>
> (2) *Quesrion mark* "?" is used to express the question, doubt, argument, or even astonishment.
>
> (3) *Exclamation mark* "!" shows the writer's feeling, e.g., happy, angry, or sad.
>
> (4) *Comma* "," has multiply functions. Its meaning is more difficult to identify. Nunberg [22] classified *Commas* in English into two categories, as a *delimiter comma* and a *separator comma*, by whether the *Comma* is used to separate the elements of the same type[3] or not. While the *delimiter comma* is used to separate different syntactic types, and a *separator comma* is used to separate members of conjoined elements. The *commas* in Chinese can also be classified into these two categories.
>
> (5) *Semicolon* ";" indicates the juxtaposed or the contrast clauses.
>
> (6) An *ellipsis* "......" indicates that words have been left out of the material that have been quoted.

---

[2] The calculation is based on Penn Chinese Tree Bank 5.1

[3] Same type means that it has the same syntactic role in the sentence, which can be a coordinate phrase or a coordinate clause.

Though there are fourteen marks in Chinese language, in our experiment, we found that not all of the punctuations can be used for segmentation. To ensure the accuracy, we define the punctuations introduced above, which are possible for segmentation as divide punctuation.

After analyzing the dependency structure of sentences in Penn Chinese Tree Bank 5.1, we find that, each sentence dependency structure has only one word which is not a punctuation as its root node, we call this root node *main root*. In most of Chinese sentences of Chinese Tree Bank 5.1, if we divide the long sentence into short clauses by part of the punctuations, the interesting phenomena is that each of the clauses possesses an independent dependency structure, in which there is only one node in the clause that indicates a syntactic relationship with the *main root*, and we call this kind of word node *sub root*. For example illustrated in Figure 3.3. Word "誕生"(births) is the *main root* of the example Chinese sentence and there are three punctuations *Punctuation A*, *Punctuation B*, *Punctuation C*, in which *Punctuation A* and *Punctuation C* depend on the *main root* "誕生"(births), but *Punctuation B* depends on the word "有"(possessed). If the sentence is divided by of the *Punctuation A* and *Punctuation C*, the divided clauses are independent of each others, that means we can parse the divided clauses separately. Because the deterministic parsing algorithm will avoid the error propagation and embodies the unique parsing mechanism, parsing the short clauses is more simple than parsing the long sentence and achieves better performance.

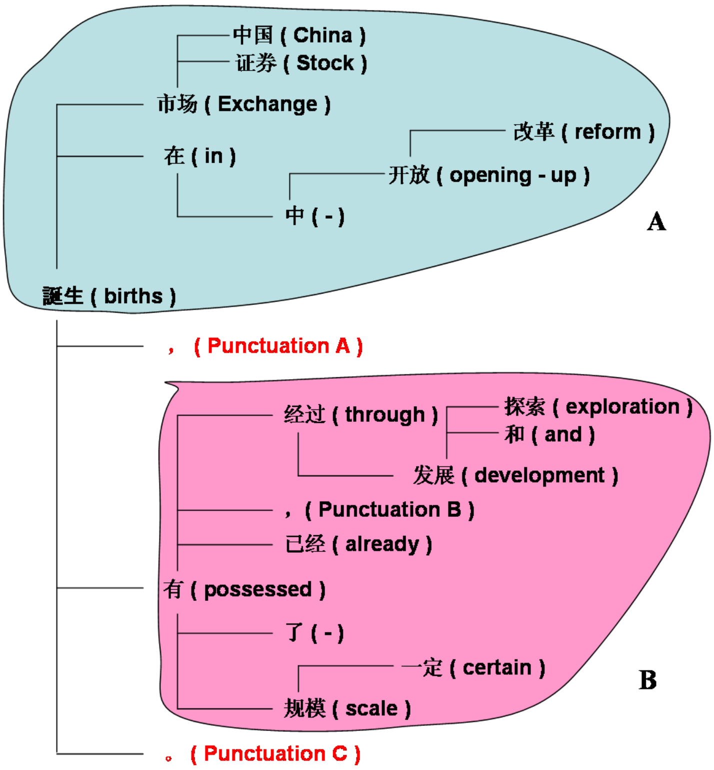


Figure 3.3: Chinese sentence dependency tree.

The feature templates used in our punctuation sentence segmentation are listed in Table 3.5 and described in Figure 3.4.

The punctuation in position n is the currently focused punctuation. And $Features$ (a) denote the words and corresponding Parts-Of-Speech tag in the position n $(-2 <= n <= 2)$. $Features$ (b) and $Features$ (c) denote the remaining words and corresponding Parts-of-Speech, which are surrounded by the nearest valid punctuations[4] on both sides and the nodes in the position $n-2$, $n+2$ separately. $Feature$ (d) and $Feature$ (e) denote the number of words between the focused punctuation and the first valid punctuation on the left or the right of the current punctuation, if the number is higher than 2, this feature will be set to 1, otherwise set as 0. And if there is a verb[5] between the focused punctuation and the first valid punctuation on the left or the right of the current punctuation, the feature $Feature$ (f) or

---

[4] Valid punctuation here means *comma* “,”, *period* “.”, *colon* “:”, *semicolon* “;”, *question mark* “?”, *exclamation mark* “!” and *ellipsis* “......”.

[5] The word whoes pos tag is defined as VA, VE, VC and VV is a verb in Chinese Tree

$Feature$		$Description$
(a)	W_n	words in different position, (n=-2, -1, 0, 1, 2).
	Pos_n	pos tag of words in different position, (n=-2, -1, 0, 1, 2).
(b)	W_left_n	words in different position, ($p < n < -2$).
	Pos_left_n	pos tag of words in different position, ($p < n < -2$).
(c)	W_right_n	words in different position, ($2 < n < q$).
	Pos_right_n	pos tag of words in different position, ($2 < n < q$).
(d)	$Num\_W\_left_n$	if the number of words in the position n is higher than 2, then set it as 1, others set it as 0. ($p < n < 0$).
(e)	$Num\_W\_right_n$	if the number of words in the position n is higher than 2, then set it as 1, others set it as 0. ($0 < n < q$).
(f)	$Verb\_left_n$	if the pos tag of words in the position n is (VA, VE, VC, VV), then set it as 1, others set it as 0. ($p < n < 0$).
(g)	$Verb\_right_n$	if the pos tag of words in the position n is (VA, VE, VC, VV), then set it as 1, others set it as 0. ($0 < n < q$).
(h)	$Noun\_left_n$	if the pos tag of words in the position n is (NN, NR, NT), then set it as 1, others set it as 0. (n=-1).
(i)	$Noun\_right_n$	if the pos tag of words in the position n is (NN, NR, NT), then set it as 1, others set it as 0. (n=1).
	P_right_n	if the pos tag of words in the position n is P, then set it as 1, others set it as 0. (n=1).
	CS_right_n	if the pos tag of words in the position n is CS, then set it as 1, others set it as 0. (n=1).

Table 3.5: Features description of sentence segmentation.

$Feature$ (g) will be set to 1. The $Feature$ (h) checks whether the Parts-Of-Speech tag firstly left nearby the focused node is a Noun[6], and $Features$ (i)

Bank 5.1.

[6] The word whoes pos tag is defined as NN, NR, NT is a noun in Chinese tree Bank 5.1.

check whether the Parts-Of-Speech tag firstly right nearby the focused node is a Noun, Preposition or Subordinating Conjuction.

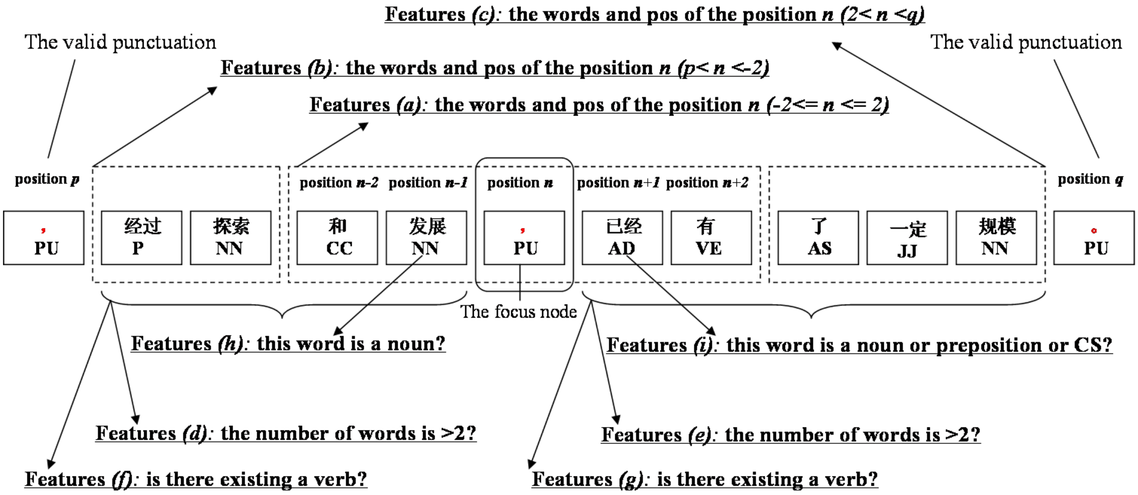

Features (c): the words and pos of the position n (2< n <q)
The valid punctuation
The valid punctuation
Features (b): the words and pos of the position n (p< n <-2)
Features (a): the words and pos of the position n (-2<= n <= 2)
position p
position n-2
position n-1
position n
position n+1
position n+2
position q
，
PU
经过
P
探索
NN
和
CC
发展
NN
，
PU
已经
AD
有
VE
了
AS
一定
JJ
规模
NN
。
PU
The focus node
Features (h): this word is a noun?
Features (i): this word is a noun or preposition or CS?
Features (d): the number of words is >2?
Features (e): the number of words is >2?
Features (f): is there existing a verb?
Features (g): is there existing a verb?

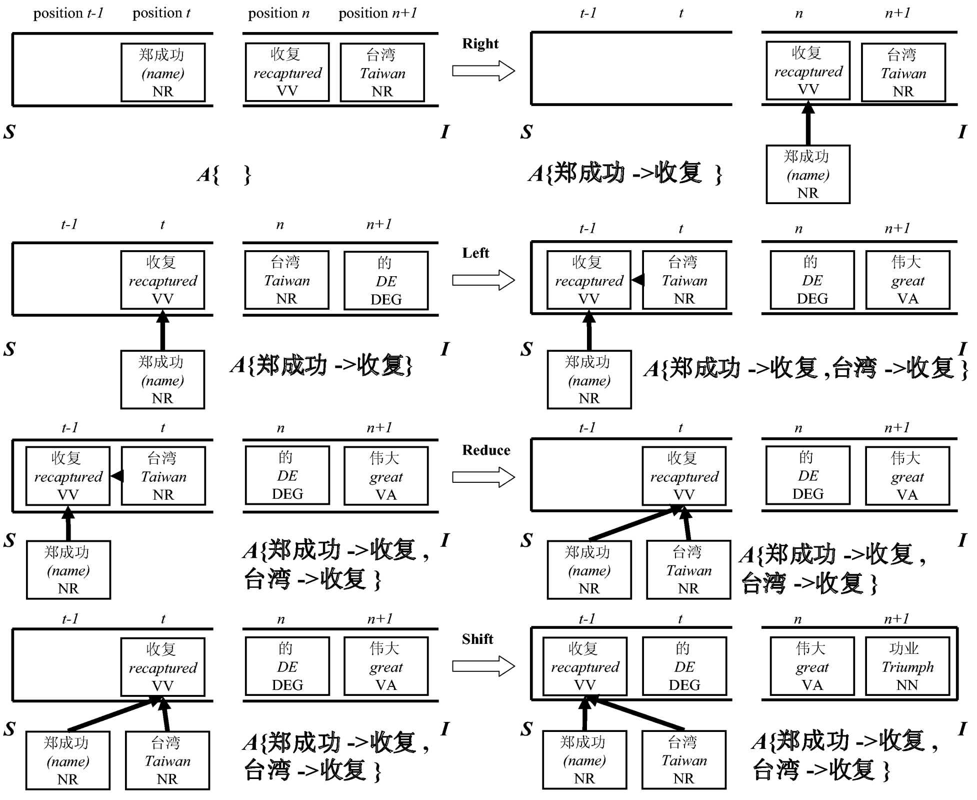
Figure 3.4: Features setting example of sentence segmentation.

3.4 Deterministic Dependency Parsing

The deterministic dependency parsing has recently been proposed as a robust and efficient method for syntactic parsing of unrestricted natural language text. Deterministic methods for dependency parsing have now been applied to a variety of languages, including Japanese [9], English [8] and Swedish [10]. In this Chapter, the bottom-up parsing algorithms represented by Nivre are to introduced as related research work.

In Nivre's algorithm, the configurations of analyzer are represented by a triple $\langle S, I, A\rangle$. S is a stack, in which the words are being in consideration, and I is the list of input words, in which the words to be processed stay. A is a list of dependency relations decided during the algorithm. Given an input word sequence W, the analyzer is initialized to $\langle nil, W, \Phi\rangle$.

The analyzer estimates the dependency relation between two words (the top element of stack S and the first element of stack I). The algorithm iterates until the list I becomes empty. Then, the analyzer outputs the word dependency relations A.



Example: 郑成功收复台湾的伟大功业 (*The great triumph that Cheng Cheng-Kung recaptured Taiwan.*)

Figure 3.5: The operations of Nivre's algorithm.

The parser constructs dependency trees in left to right word order of input sentences based on four parsing actions: *Right*, *Left*, *Shift*, *Reduce*. Figure 3.5 shows the example of the four actions.

> The *Right* action constructs a dependency relation between t and n (t and n are the top element of S and the first element of I respectively), where node t becomes a child of node n. Suppose the current triple is $\langle t \mid S, n \mid I, A\rangle$, if there is a dependency relation that the word t depends on word n , add the new dependency relation ($t {\rightarrow} n$) into A, and remove t from S. The configuration now becomes $\langle S, n \mid I, A \cup \{(t \rightarrow n)\}\rangle$.
>
> The *Left* action constructs a dependency relation between t and n, where the right node of target node becomes a child of the left one, opposite to the action *Right*. In the current triple is $\langle t \mid S, n \mid$

$I, A\rangle$, if there is a dependency relation that the word n depends on the word t, adds the new dependency relation ($n \rightarrow t$) into A, and push n onto the stack S. The configuration now becomes $\langle n \mid t \mid S, I, A \cup \{(n \rightarrow t)\}\rangle$.

The *Reduce* action follows by *Left* one, if there are no more words n' (n'$\in I$) which may depend on t, and t has a parent on its left side, the analyzer removes t from the stack S. The configuration now becomes $\langle S, n \mid I, A\rangle$.

The *Shift* action means no construction of dependencies between these target nodes, and the triple doesn't satisfy the conditions for *Reduce*, then push n onto the stack S. The configuration now becomes $\langle n \mid t \mid S, I, A\rangle$.

Nivre uses a simple left-right stack based shift-reduce algorithm. He used a memory-based learner and features include the word on top of the stack and its pos, the next word in the input string and its pos, the next n words in the input string(with pos), and the previously determined dependents of the word on the top of the stack. Nivre's parser is actually designed for labeled dependencies, where the classifier also decides the dependency labels, and the labels can also be used as features. This results in having several different actions for the classifier to choose from. Training of the classifier is also done by running the algorithm on sentences with known dependency structures, and associating the correct parser actions with features that correspond to the parser's current state.

Before the parsing of input sentence, we propose a baseNP chunker [27] to define NP chunk types which are used as a feature template during the parsing processing. The baseNP chunker, which looks chunking as boundary tagging, is developed to get the baseNP tags of input sentence word. The B, O, I tag definition is applied to indicate the boundaries for each baseNP shown in Table 3.6. "B" means one word is the beginning of baseNP, "I" means one word is inside of baseNP, and "O" means one word is outside of baseNP. The boundary tagging is performed by multi-class classification. The features used in baseNP chunking are current word and surrounding words and their corresponding Parts-of-Speech tags.

(a) $Word_n$ $(n = -2, -1, 0, 1, 2)$

$(b)\ Pos_n\ (n = -2, -1, 0, 1, 2)$

Tag	*Description*
B	the current word that starts a base NP
I	the current word that is inside a base NP
O	the current word that is outside a base NP

Table 3.6: Position tags in baseNP chunking.

Nivre's deterministic algorithm is used in our parser. But Nivre's work uses memory based learning in their parser, we utilize SVMs in ours, therefore, the features are different from the original Nivre's method.

Table 3.7 and Table 3.8 are feature templates used in our deterministic dependency parsing. We first define some features based on local context, which are often used in other determinstic parsers [8][10][15]. While deterministic parsing is a kind of bottom-up parsing. Lacking of the top-down information may result in local maximization during parsing process. Considering about this, we add some global features to give top-down information when making choice among actions.

Local Feature		*Description*
(a)	W_m	words in different position, (m=t-2, t-1, t, n, n+1, n+2).
	Pos_m	pos tag of words in different position, (m=t-2, t-1, t, n, n+1, n+2).
(b)	$Word\_child\ of\ W_m$	the child words of W_m, (m=t-2, t-1, t, n, n+1, n+2).
	$Pos\_child\ of\ W_m$	the pos tag of the child words of W_m, (m=t-2, t-1, t, n, n+1, n+2).
(c)	*Distance*	distance of the W_t and W_n in the sentence.

Table 3.7: Local feature description of parsing.

Global Feature		*Description*
(d)	$IsRoot_m$	if there exists a main root word in the position m, (m= t-2, t-1, t, n, n+1, n+2), set it as 1, otherwise set it as 0.
(e)	$BaseNP\_tag_m$	the baseNP tag of the word W_m, (m=t-2, t-1, t, n, n+1, n+2).

Table 3.8: Global feature description of parsing.

In Figure 3.6, $Features$ (a) denotes the words and corresponding Parts-Of-Speech tag in the position m $(t-2 <= m <= n+2)$. $Features$ (b) denotes the child words and corresponding Parts-Of-Speech tag of the words in the position m $(t-2 <= m <= n+2)$, and the dependency relations is decided during the parsing. $Features$ (c) denotes the distance between the two words in position t and n, and the distance feature is normalized between 0-1 by the length of this sentence. $Features$ (d) checks whether there exists sentence root node in the position m $(t-2 <= m <= n+2)$. $Features$ (e) denotes the corresponding BaseNP Tag of the words in position m $(t-2 <= m <= n+2)$ by a baseNP chunker.

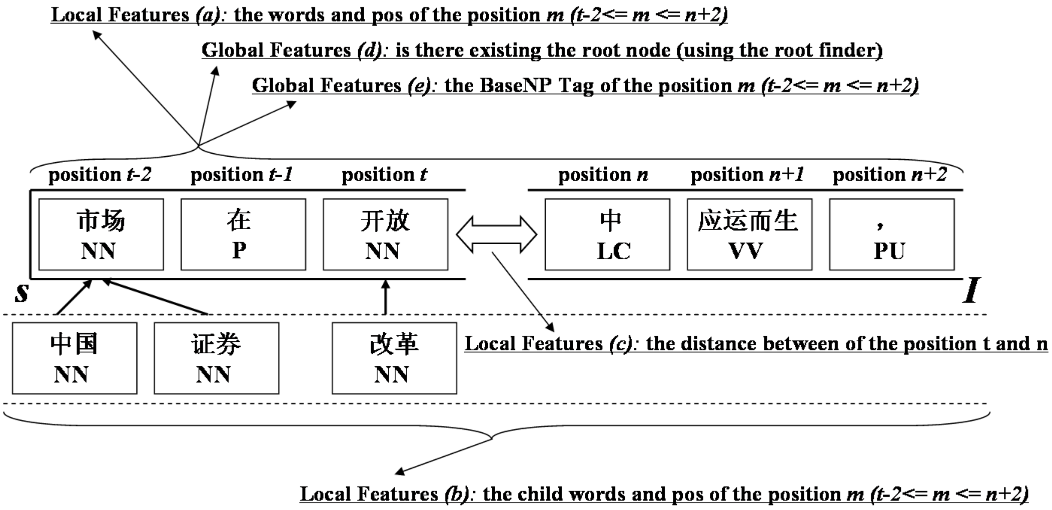

Local Features (a): the words and pos of the position m (t-2<= m <= n+2)
Global Features (d): is there existing the root node (using the root finder)
Global Features (e): the BaseNP Tag of the position m (t-2<= m <= n+2)
position t-2
position t-1
position t
position n
position n+1
position n+2
市场 NN
在 P
开放 NN
中 LC
应运而生 VV
， PU
S
I
中国 NN
证券 NN
改革 NN
Local Features (c): the distance between of the position t and n
Local Features (b): the child words and pos of the position m (t-2<= m <= n+2)

Figure 3.6: Features setting example of deterministic dependecy parsing.

3.5 Parsing Combination

As mentioned above, a long Chinese sentence can be divided into several clauses, and using the deterministic dependency parser they will be produced with dependency structure separately, but it is essential to combine the clauses' dependency structure trees together. There is a phenomenon that suggests the inner link among the clauses by the root node words, it means that all of the *sub root* should depend on the *main root* of the whole sentence. Based on this phenomenon, the parsing combination problem can be solved as finding the *main root* of the whole sentence, and representing dependencies between the *main root* and the *sub roots*.

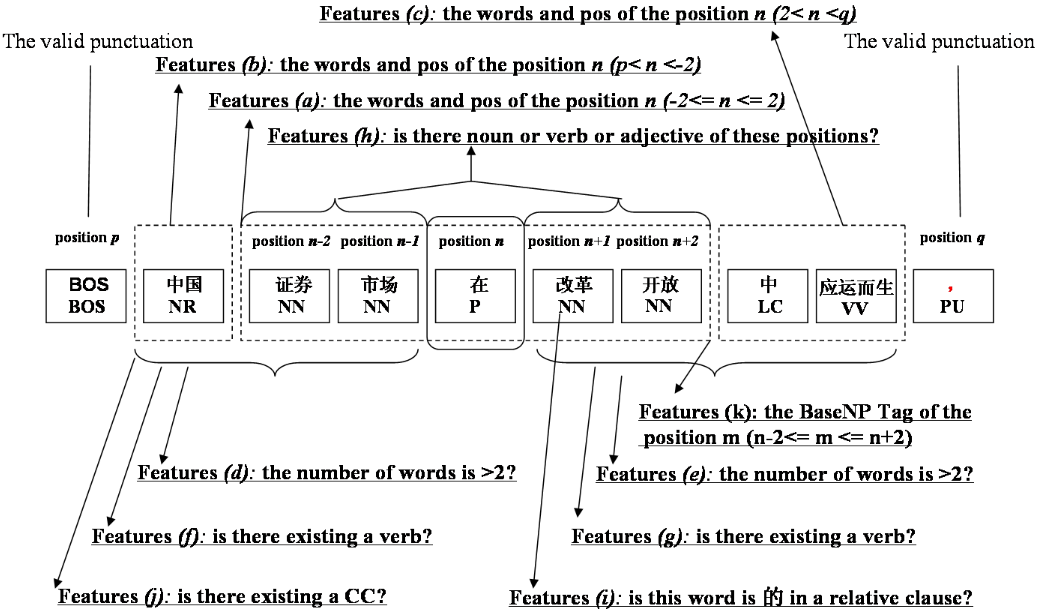

Features (c): the words and pos of the position n (2< n <q)
The valid punctuation
The valid punctuation
Features (b): the words and pos of the position n (p< n <-2)
Features (a): the words and pos of the position n (-2<= n <= 2)
Features (h): is there noun or verb or adjective of these positions?
position p
position n-2
position n-1
position n
position n+1
position n+2
position q
BOS BOS
中国 NR
证券 NN
市场 NN
在 P
改革 NN
开放 NN
中 LC
应运而生 VV
， PU
Features (k): the BaseNP Tag of the position m (n-2<= m <= n+2)
Features (d): the number of words is >2?
Features (e): the number of words is >2?
Features (f): is there existing a verb?
Features (g): is there existing a verb?
Features (j): is there existing a CC?
Features (i): is this word is 的 in a relative clause?

Figure 3.7: Features setting example of root finder.

A sentence root dependency node finding mechanism "*root finder*" is proposed. The main root would be found out by using the *root finder*. The feature templates used in our *root finder* are listed in Table 3.9 and described in Figure 3.7. About the feature set up used in root finding, feature templates (a)-(g) are adopted similarly with the features of sentence segmentation classifier. Also some extra features revolving on the Parts-Of-Speech tag information become feature members. Feature templates (h) determine

Feature		*Description*
(a)	W_n	words in different position, (n=-2, -1, 0, 1, 2).
	Pos_n	pos tag of words in different position, (n=-2, -1, 0, 1, 2).
(b)	W_left_n	words in different position, (p < n < -2).
	Pos_left_n	pos tag of words in different position, (p < n < -2).
(c)	W_right_n	words in different position, (2 < n < q).
	Pos_right_n	pos tag of words in different position, (2 < n < q).
(d)	$Num\_W\_left_n$	if the number of words in the position n is higher than 2, then set it as 1, others set it as 0. (p < n < 0).
(e)	$Num\_W\_right_n$	if the number of words in the position n is higher than 2, then set it as 1, others set it as 0. (0 < n < q).
(f)	$Verb\_left_n$	if the pos tag of words in the position n is (VA, VE, VC, VV), then set it as 1, others set it as 0. (p < n < 0).
(g)	$Verb\_right_n$	if the pos tag of words in the position n is (VA, VE, VC, VV), then set it as 1, others set it as 0. (0 < n < q).
(h)	$Noun_n$	if the pos tag of words in the position n is (NN, NR, NT), then set it as 1, others set it as 0. (n=-2, -1, 1, 2).
	$Verb_n$	if the pos tag of words in the position n is (VA, VE, VC, VV), then set it as 1, others set it as 0. (n=-2, -1, 1, 2).
	Adj_n	if the pos tag of words in the position n is (JJ), then set it as 1, others set it as 0. (n=-2, -1, 1, 2).
(i)	Dec_n	if the word in the position n is “的” in a relative clause, then set it as 1, others set it as 0. (n=1).
(j)	CC_left_n	if there is a word whose pos tag of words in the position n is CC, then set it as 1, others set it as 0. (p < n < 0).
(k)	$BaseNP\_tag_n$	baseNP tag of word in different position (n=-2, -1, 0, 1, 2).

Table 3.9: Features description of root finding.

whether the word in different position is a noun, a verb or a adjective. If the word on the right side of the focused word is the hanzi “的” in a relative clause, the feature (i) will be set as 1, otherwise set as 0. If there is a coordinating conjunction between the first valid punctuation and focus word, the feature (j) will be set as 1, otherwise set as 0.

Chapter 4

Experiment Results and Discussion

4.1 Experiment Results

4.1.1 Data Set and Experimental Evaluation

In this part, we introduce the experiment data and the description of experimental evaluation. We use the Penn Chinese Tree Bank 5.1 [6] for our main experiment data source.

The Penn Chinese Tree Bank(CTB) is an ongoing project, with its objective being to create a segmented chinese corpus annotated with Parts-Of-Speech tags and syntactic brackets. The first installment of the Chinese Tree Bank project consists of *xinhua* newswire between the years 1994 and 1998, totally 100,000 words, fully segmented, Parts-Of-Speech tagged and syntactically bracketed. Currently, the second installment of the project, the 400,000 words Chinese tree bank 5.1 follows the standards set up in the segmentation, Parts-Of-Speech tagging and bracketing guidelines, and it uses the articles from People's Daily, Hong Kong newswire and material translated into Chinese from other languages in addition to the *xinhua* newswire used in the first installment of Chinese tree bank in an effert to diversify the sources. Syntactic structure in the Chinese tree bank is represented as constituent trees, and has been converted to dependency graphs using essentially the same method as for the English data. In our experiment, we use Xia's head table [23] to transfer the constituent structure of Penn Chinese Tree Bank to dependency structure.

The same data distribution is set for the four classification modules:

(A) Word Segmentation & Parts-Of-Speech Tagging
(B) Sentence Division
(C) BaseNP Chunking
(D) Root Node Finding

We choose 90% and 10% from the Chinese Tree Bank corpus for training and testing respectively (In all of the module above, the training data contains 16984 sentences and the testing data contains 1800 sentences).

Module	*Training*	*Testing*
(A)-(D)	16984 sen (90%)	1800 sen (10%)

Table 4.1: Experiment data set of modules (A)-(D).

The performence of each module such as word segmentation & Parts-Of-Speech tagging analyzer, sentence segmenter, root finder and baseNP chunker are evaluated by the following three measures.

$$precision = \frac{\#\ of\ correctly\ classified\ result}{\#\ of\ classified\ result} \tag{4.1}$$

$$recall = \frac{\#\ of\ correctly\ classified\ result}{\#\ of\ in\ gold\ standard} \tag{4.2}$$

$$F\_score = \frac{2\ *\ precision\ *\ recall}{precision\ +\ recall} \tag{4.3}$$

In order to verify the superior activity of our parsing strategy based on sentence division in parsing the long sentence, we set up another experiment data for the parsing module (E). 8004 complex sentences[1] with more than 30 words are randomly selected as the training data and test data for experiment (7342 sentences for training and 662 sentences for testing from the training and testing data of modules (A)-(D) respectively). And the unchanging data set described above is still used for training the sentence segmenter, root

---

[1] A complex sentence consists of a main clause and several subordinate clauses

finder and baseNP chunker. The original segmented data annotated with Parts-Of-Speech tags in Chinese Tree Bank 5.1 are used for training and testing during the processings of sentence segmentation, root finding and the dependency parsing.

Module	*Training*	*Testing*
(E) Parsing	7342 sen (90%)	662 sen (10%)

Table 4.2: Experimental data set of module (E).

The performence of parsing module is evaluated by the following two measures: Dependency accuracy (Dep.Acc) and Root accuracy (Root.Acc), respectively.

$$Dep.Acc = \frac{\#\ of\ correctly\ classified\ dependency\ relation}{\#\ of\ classified\ dependency\ relation} \tag{4.4}$$

$$Root.Acc = \frac{\#\ of\ correctly\ classified\ root\ nodes}{\#\ of\ clauses} \tag{4.5}$$

4.1.2 Experiment Result of Word Segmentation & Parts-Of-Speech Tagging

The features description of word segmentation & Parts-Of-Speech tagging module are shown in the Table 4.3.

As the baseline experiment, we use a words list extracted from Chinese Tree Bank 5.1 corpus which contains 37043 words as the system lexicon. The experimental precision and recall are described in the Table 4.4. We know that using the W_n feature can improve the precison and recall of both word segmentation and pos tagging nearly by 1 percent, because using the word level feature can help detect the known words in the lexicon. When the lexicon size is enlarged into 188673, (The lexicon is extracted from Peking University Corpus of the second SIGHAN Bakeoff [24], Penn Chinese Tree Bank version 4.0 [6] and part of the NICT Corpus [25][26]. We also add all of the words in Penn Chinese Tree Bank version 5.1 into the lexicon list), both of the precision and recall also increase a little but no more than 1 percent.

$Feature$	$Description$
C_n	characters in different position, n=-2, -1, 0, 1, 2.
$IsPu(C_0)$	if C_0 is a punctuation character, then set it as 1, others set it as 0.
W_n	lexicon words in different positions, n=-1, 0, 1. (W_0 is the word containing the character C_0)

Table 4.3: Features of word segmentation and Parts-Of-Speech tagging module(Quotation of Table 3.4).

While the lexicon expands, more words contained in the large lexicon can be recognized with the large lexicon. Because there are about 6400 words are single character words. It made the system easy to segment one word into several words, for example, the words “経済組”(Economy Group) into “経済”(Economy) and “組” (Group). Even though, the large size of lexicon also brings errors of combining two words into one word, if the word was in the lexion, for example, words “只”(only) and “有”(have) are identified as one word, since there exists the word “只有”(only) in the lexicon. We should reduce our lexicon to a reasonable size to solve this problem.

$Lexicon$	$Feature$		$precision$	$recall$	F_score
small	w/o W_n	$Word\ seg$	93.78%	94.25%	94.01%
small	with W_n	$Word\ seg$	94.84%(+1.06)	94.94%(+0.69)	94.89%
big	with W_n	$Word\ seg$	95.58%(+0.74)	95.76%(+0.82)	95.67%
small	w/o W_n	$Pos\ tag$	87.48%	87.91%	87.70%
small	with W_n	$Pos\ tag$	89.55%(+2.07)	89.64%(+1.73)	89.60%
big	with W_n	$Pos\ tag$	89.61%(+0.06)	89.77%(+0.13)	89.69%

Table 4.4: Experiment result of word segmentation and Parts-Of-Speech tagging.

4.1.3 Experiment Result of Parsing

The features description of parsing module are shown in the Table 4.5 and Table 4.6.

Local Feature		*Description*
(a)	W_m	words in different position, (m=t-2, t-1, t, n, n+1, n+2).
	Pos_m	pos tag of words in different position, (m=t-2, t-1, t, n, n+1, n+2).
(b)	$Word\_child\ of\ W_m$	the child words of W_m, (m=t-2, t-1, t, n, n+1, n+2).
	$Pos\_child\ of\ W_m$	the pos tag of the child words of W_m, (m=t-2, t-1, t, n, n+1, n+2).
(c)	$Distance$	distance of the W_t and W_n in the sentence.

Table 4.5: Local features of parsing module(Quotation of Table 3.7).

Global Feature		*Description*
(d)	$IsRoot_m$	if there exists a main root word in the position m, (m= t-2, t-1, t, n, n+1, n+2), set it as 1, otherwise set it as 0.
(e)	$BaseNP\_tag_m$	the baseNP tag of the word W_m, (m=t-2, t-1, t, n, n+1, n+2).

Table 4.6: Global features of parsing module(Quotation of Table 3.8).

First, some features based on the local context are defined, which are often used in other deterministic parsers [8] [10] [15]. Because deterministic parsing is a kind of bottom-up parsing and the lacking of the top-down information may result in local maximization during parsing process, we define some local features to add the top-bottom information. The global feature information is determined by two classification analyzers: *root finder* and *BaseNP chunker*. Another classification analyzer *sentence segmenter* is

Module	*Precision*	*Recall*	*F_score*
(B)sentence segmentation	80.22%	89.71%	84.70%
(C)baseNP chunking	89.31%	88.67%	89.04%
(D)root finding	75.44%	80.59%	77.93%

Table 4.7: Experiment result of sentence segmentation, baseNP chunking and root finding.

used to divide long sentence into short clauses. Table 4.7 shows the experiment result of the three classification modules.

To verify the superior of parsing strategy augmented by sentence division, we do a comparison test: *Flat parsing* and *Hierarchical parsing*. The experiment result is shown in Table 4.8.

Flat parsing : parsing the whole sentence by the parser.

Hierarchical parsing :
(1) dividing the sentence into clauses by the sentence segmenter.
(2) parsing the clauses by the parser.
(3) combining the parsed dependency structure of each clause.

Parsing Strategy	*Dep.Acc*	*Root.Acc*
Flat parsing	77.46%	68.29%
Hierarchical parsing	79.38%(+1.92)	73.66%(+5.37)

Table 4.8: Experiment result of Flat parsing and Hierarchical parsing .

This result validates the effectiveness of our hierarchical deterministic parsing strategy augmented by sentence division. Reducing the length of sentences helps the parser get 1.92% improvement on dependency accuracy compared with flat parsing. The correct recognition of sentence structure and main root word by the sentence segmenter and root finder also help the parser get 5.37% improvement on root accuracy. Some previous parsing errors will be propagated during Flat parsing. For example in Figure 4.1, if the word "向"(to) is incorrectly parsed as depending on word "作好"(do) (marked by the red dotted line), the error will result in the incorrect parse of word “預備”(prepare) as the depending on the word “改正”(correct) (marked by the red dotted line). And during the hierarchical Parsing, some of these error propagation can be prevented. But during the hierarchical parsing, there exists the parsing error generated by the wrong sentence segmentataion. For example in Figure 4.2, theoretically, this sentence should be segmented by the divide point: *punctuation* 1 and *punctuation* 4. But the wrong recognition creates the *punctuation* 3 as another divide point, it leads to that the sentence is incorrectly divided into three clauses and the word “穩健”(firm) and *punctuation* 2 incorrectly depend on the word “起” (-). We should improve the accuracy of the sentence segmenter to solve this problem.

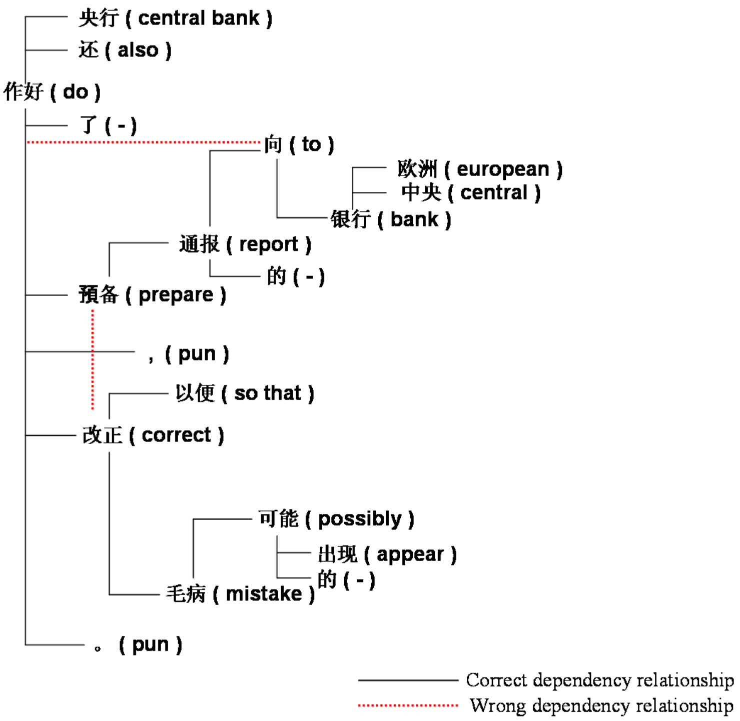


Figure 4.1: The error parsed example in Flat parsing.

We compare our proposed hierarchical parser with the result of Nivre’s parser reported in [13]. The different result showed in Table 4.9 illustrates that the dependency accuracy is improved by 2.17% by our parser compared with Nivre’s parser. Because among the testing data, there are 40% sentences with more than 30 words, that means relatively long sentences are in high proportion to the experiment data base. The experiment result shows the evidence that our parser can handle long sentence parsing better with the divide sentence parsing strategy.

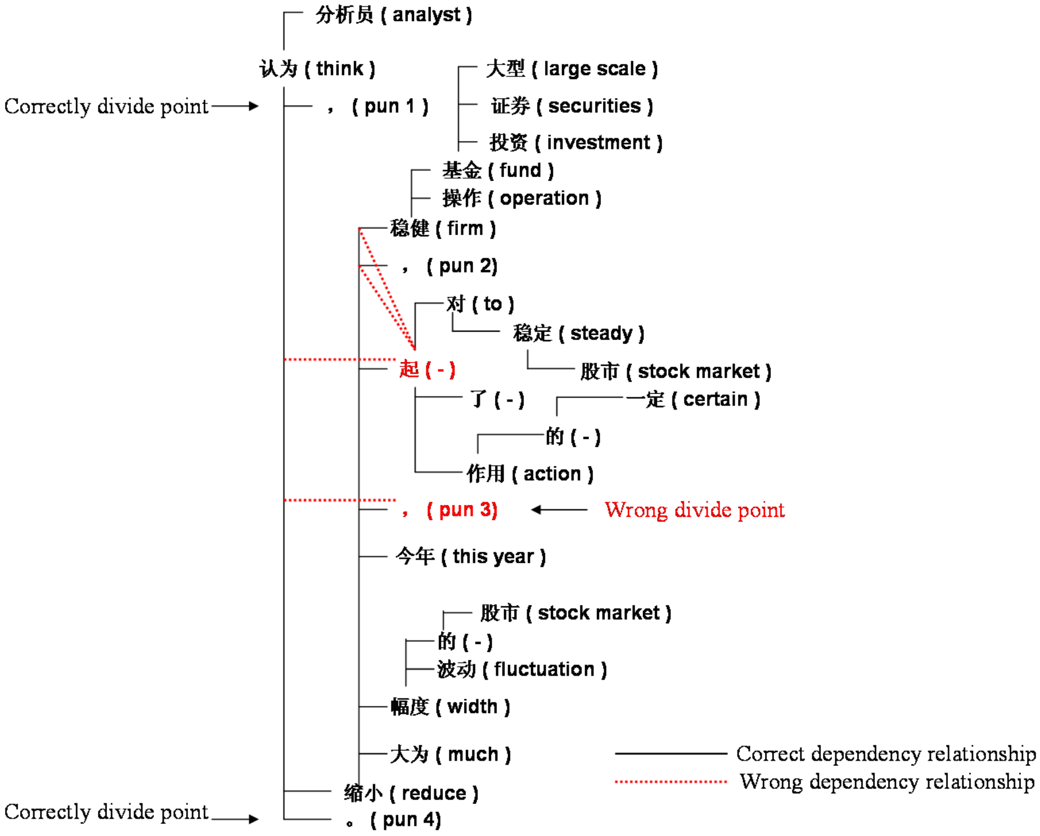


Figure 4.2: The error parsed example in Hierarchical parsing.

Parsing Strategy	*Dep.Acc*
Nivre's parser	84.30%
Our parser	86.47%(+2.17)

Table 4.9: Comparing with Nivre's parser.

4.2 Discussion

Theoretically, using our hierarchical determinstic dependency parser, each sentence can be completely parsed into a dependency tree with a node as root. With our parsing combination strategy, only when the main root found by the root finder is left unparsed, we use it to combine the clause parsing output. In other cases, we do not do combination. For example in the Figure 4.3, when the word “誕生”(births) and the word “有”(possessed) are the left unparsed words in the clause A and clause B separately (in other words, they are the sub roots of each clause), if the root finder finds the word “誕生”(births) the main root of whole sentence, in this case, we combine the dependency structure by making the main root word “誕生”(births) as the head of the word “有” (possessed) and punctuation 1, 2. With this strategy, the dependency accuracy is improved from 78.82% to 79.07%. But it also brings another problem that some sentences may have unparsed words which cannot be combined. Fortunately, such problem only appears in 10% sentences. We are considering using a reparse strategy to resolve this issue in the future.

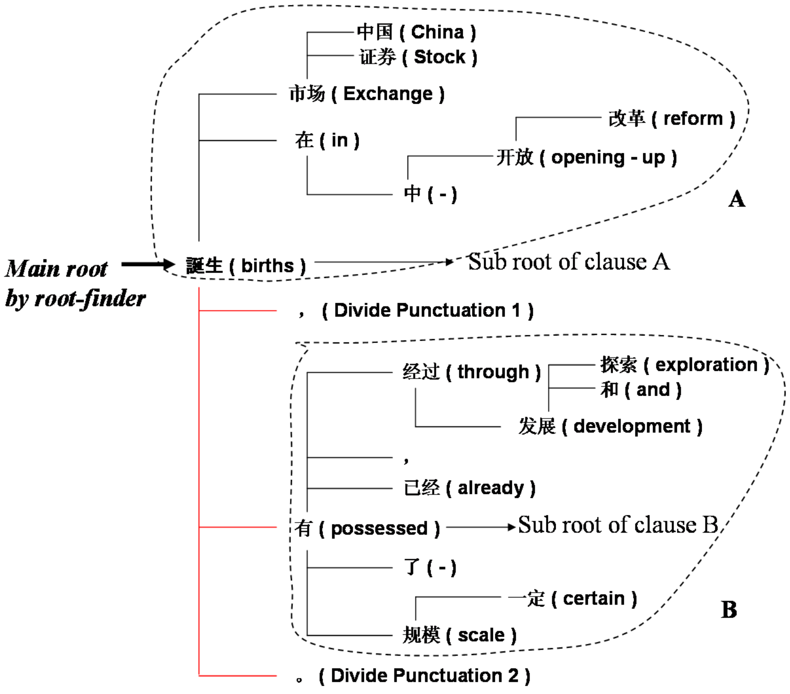


Figure 4.3: Parsing combination strategy.

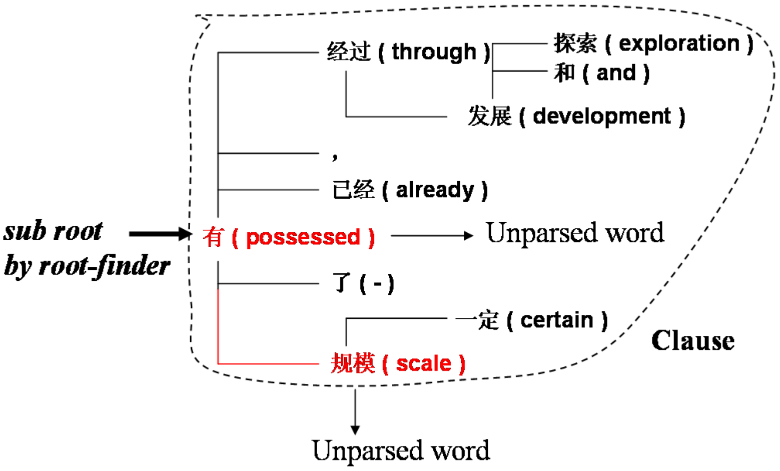


Figure 4.4: Parsing combination for unparsed words.

Our strategy requires that after parsing the divided clauses, each of them only has one unparsed word as the clause sub root. But the errors of clause parsing may lead to that more than one word left unparsed in the dependency structure of a clause. To solve this problem, we apply another parsing combination procedure. It means we find the sub root of each clause by the root finder, and then make other unparsed words depend on this sub root. For example in the Figure 4.4, the words “有”(have) and “規模”(scale) are unparsed words, in this case, root finder is used to find that the word “有”(have) is the sub root of the clause, then another unpared word “規模”(scale) is depended with the sub root “有”(have). With this additional step, the dependency accuracy is improved from 79.07% to 79.38%.

$Features$	$Dep.Acc$	$Root.Acc$
F_{local}	72.94%	61.46%
$F_{\text{local}} + F_{\text{global}}$	79.38%(+6.44)	73.66%(+12.02)

Table 4.10: Parsing result with different features.

Deterministic parsing is an important part of our proposed parser, and features are crucial to the SVM models used in this step. To get information from top to down direction, we used two global features instead of the basic local features . Table 4.10 lists the parsing results with different feature set. It shows that the use of global features bring 6.44% achievement on dependency accuracy 12.02% improvement on root accuracy.

Chapter 5

Conclusion and Future Work

5.1 Conclusion

Dependency parsing is one of the most important issues in Natural Language Processing community. Deterministic classifier based parsers offer attractive alternatives to generative statistical parsers, they are fast, efficient and simple to implement, but generally less accurate than optimal statistical parsers, in addition, how to prevent the error propagation when parsing long sentences deterministically is still a problem. To solve the error propagation problem and improve the dependency structure analysis of Chinese, we present a hierarchical deterministic Chinese dependency parser augmented by sentence division. Using this method, complicated and long sentences are properly divided into clauses, and these relatively simple, short clauses are parsed separately. The sentence division shortens the length of parsed sentence, which guarantee the accuracy of deterministic parsing at the same time.

5.2 Future Work

Although the parser proposed in this thesis has shown relatively satisfactory result, some difficulties for the Chinese language still remain. There are several research directions in which the work presented in this thesis can be extended. We conclude by disscussing some of these direction for future work. The performance of the sentence segmentation, root finding and unparsed word dealing should be improved in the future. Various features should be tried because of their effect on parsing accuracy and the relationship of the models to other work on statistical parsing should be used as a source of reference. This thesis has focused on language modeling and parsing of Chinese. However, because it used little language dependency feature, the work reported in this thesis can be extended to avariety of other languages. We believe that the parsing model can be easily adapted to other languages, which have similar syntactic characteristic as Chinese.

Acknowledgements

This thesis concludes my reseach work when I was a master candidate of the university of Tokyo. First, I would like to thank to my supervisers: Prof. Kurohashi and Prof. Esaki. They gave me detailed instructions and valuable suggestions during my work. I also want to thank Dr. Kun Yu for her helpful discussion. At last, I want to thank all the staffs and students of Kurohashi lab, who had given me great help during my master candidate life.

Reference

[1] E. Charniak. 2000. A maximum entropy inspired parser. In *Processings of the 1st Meeting of the North American Chapter of the Association for Computational Linguistics* , pages 132-139, Seattle, Washington, April 29 to May 4.

[2] M. Collins. 1997. Three generative lexicalised models for statistical parsing. In *Proceedings of ACL* , pages 16-23, Madrid, Spain.

[3] A. Ratnaparkhi. 1999. Learning to parse natural language with maximum entropy models. newblock In *Machine Learning* , pages 151-175.

[4] L. S. Lee, L. J. Lin, K. J. Chen, and J. Huang. 1991. An efficient natural language processing system specially designed for Chinese language. In *Computational Linguistics* , Volume 17, Number 4.

[5] C. Cortes, V. M. Vapnik. 1995. Support Vector Networks. In *Machine Learning* , 20: pages 273-297.

[6] N. W. Xue, F. Xia, F. D. Chioe, and P. Martha. 2005. Building a Large Annotated Chinese Corpus: the Penn Chinese Treebank. In *Journal of Computational Intelligence* , 21(3): pages 246-287.

[7] J. Hall, J. Nivre and J. Nilsson. 2004. Memory-Based Dependency Parsing. In *Proceedings of the 8th Conference on Computational Natural Language Learning (CONLL)* , pages 49-56.

[8] H. Yamada and Y. Matsumoto. 2003. Statistical dependency analysis with support vector machines. In *Processings of IWPT* , pages 195-206, Nancy, France.

[9] T. Kudo and Y. Matsumoto. 2000. Japanese Dependency Structure Analysis Based on Support Vector Machines. In *Proceedings of Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora 2000* , pages 18-25.

[10] J. Nivre. 2003. An efficient algorithm for projective dependency parsing. In *Processings of IWPT* , pages 149-160, Nancy, France.

[11] M. Zhou. 2000. A Block-based Dependency Parser for Unrestricted Chinese Text. In *ACL-2000 2nd Chinese Language Processing Workshop* .

[12] J. S. Ma, Y. Zhang, T. Liu, and S. Li. 2004. A statistical dependency parser of Chinese under small training data. In *IJCNLP 2004 Workshop: Beyond shallow analysis, formalisms and statistical modeling for deep analysis* .

[13] J. Hall, J. Nivre and J. Nilsson. 2006. Discriminative Classifiers for Deterministic Dependency Parsing. In *Proceedings of Coling-ACL 2006* , pages 316-323.

[14] Y. C. Cheng, M. Asahara, Y. Matsumoto. 2004. Deterministic Dependency Structure Analyzer for Chinese. In *Proceedings of IJCNLP 2004* , pages 135-140.

[15] Y. C. Cheng, M. Asahara, Y. Matsumoto. 2005. Chinese Deterministic Dependency Analyzer: Examining Effects of Global Features and Root Node Finder. In *fourth SIGHAN Workshop on Chinese Language Processing, proceedings of the Workshop* , pages 17-24, October 2005.

[16] Y. R. Chao. 1968. A Grammar of Spoken Chinese. Berkeley etc., University of California Press 1968.

[17] C. N. Li, S. A. Thompson. 1981. Mandarin Chinese, A function reference grammar. Berkeley etc., University of California Press 1981.

[18] Chen and Hong. 1995. Verb-Object phrase and modifier-head compounds in Mandarin VN Constructions. In *Proceedings of ROCLING VIII* , 1995.

[19] Zhou and Huang. 1994. An efficient syntactic tagging tool for Corpora. In *Proceedings of COLING* , pages 945-955, 1994.

[20] X. Li, C. Zong and R. Hu. 2005. A Hierarchical Parsing Approach with Punctuation Processing for Long Chinese Sentences. In *Proceedings of IJCNLP* , pages 7-12, 2005.

[21] M. Jin et al, 2004. Segmentation of Chinese Long Sentence Using Commas. In *ACL SIGHAN Workshop* , pages 1-8, 2004.

[22] G. Nunberg. 1990. The Linguistics of Punctuation. In *CSLI Lecture Notes* , No. 18, University of Chicago Press.

[23] F. Xia, M. Palmer. 2001. Converting Dependency structures to phrase structure. In *HLT-2001* .

[24] T. Emerson. 2005. The second international Chinese word segmentation bakeoff. In *Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing* , Jeju, Korea.

[25] K. Uchimoto. 2004. Multilingual Aligned Parallel Treebank Corpus Reflecting Contextual Information and its Applications . In *Proceedings of the MLR* , pages 63-70.

[26] Y. J. Zhang, Q. Ma and H. Isahara. 2005. Building an Annotated Japanese-Chinese Parallel Corpus - A part of NICT Multilingual Corpra. In *Proceedings of the MT SummitX* , pages 71-78.

[27] Hideki Isozaki, Hideto Kazawa and Tsutomu Hirao. 2004. A Deterministic Word Dependency Analyzer Enhanced With Preference Learning. In *Proceedings of COLING* , pages 275-281.