The SAU Report for the 1st CIPS-SIGHAN-ParsEval-2010

Qiaoli Zhou **Wenjing Lang** **Yingying Wang** **Yan Wang** **Dongfeng Cai**

Knowledge Engineering Research Center,Shenyang Aerospace University,Shenyang,China

Qiaoli_z@yahoo.com.cn

Abstract

This paper presents our work for participation in the 2010 CIPS-SIGHAN evaluation on two tasks which are Event Description Sub-sentence (EDSs) Analysis and Complete Sentence (CS) Parsing in Chinese Parsing. The paper describes the implementation of our system as well as the results we have achieved and the analysis.

1 Introduction

The paper describes the parsing system of SAU in 1st CLPS-SIGHAN evaluation task 2. We participate in two tasks - EDS Analysis and CS Parsing. The testing set only provides segmentation results, therefore, we divide our system into the following subsystems: (1) Part-of-Speech (POS) tagging system, we mainly make use of Conditional Random Fields (CRFs) model for POS tagging; (2) parsing system, the paper adopts divide-and-conquer strategy to parsing, which uses CCRFs model for parsing and adopts searching algorithm to build trees in decoding; (3) head recognition system, which also makes use of CCRFs model.

The rest of the paper is organized as follows: Section 2 describes the POS tagging system; Section 3 describes the structure of our parsing system; Section 4 describes head recognition system in parsing tree; Section 5 presents the results of our system and the analysis; Section 6 concludes the paper.

2 Part-of-Speech Tagging

We use CRFs model and post-processing method for POS tagging. In the first step, we tag POS based on CRFs. The second step is the post-processing after tagging, which is correcting by using dictionary drawn from training set. The system architecture of POS tagging is shown in Figure 1.

2.1 Features

Feature selection significantly influences the performance of CRFs. We use the following features in our system.

Atom Template
word(-2) , word(-1) , word(0) , word(1) , word(2) prefix(word (0)) ,suffix(word(0)) includeDot1(word (0)) includeDot2(word (0))
Complex Template
word(-1)& word(0) ， word(0)& word(1) word(0)& prefix(word (0)) word(0)& suffix(word(0)) word(0)& includeDot1(word (0)) word(0)& includeDot2(word (0))

Table 1: Feature templates used in POS tagger. word(i) represents the ith word, prefix(word (i)) represents the first character of the ith word, suffix(word (i)) represents the last character of the ith word, ncludeDot1(word (i)) represents the ith word containing '· ' or not, and includeDot2(word (i)) represnts the ith word containing '.' or not.

2.2 Post-processing

The post-processing module adopts the following processing by analyzing the errors from tagging result based on CRFs. We firstly need to build two dictionaries which are single class word dictionary and ambiguity word dictionary before the post-processing. The single class word dictionary and ambiguity word dictionary are built by drawing from training set.

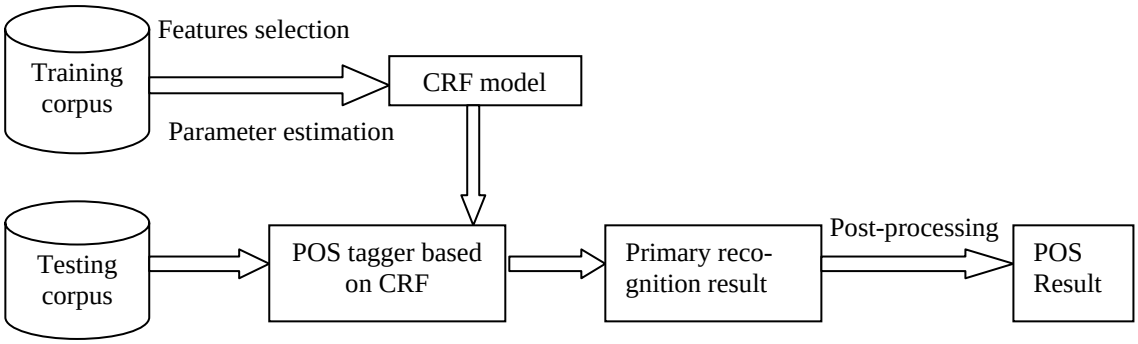

Figure 1: System architecture of POS tagging

The single class word is the word having single POS in training set, and the ambiguity word is the word having multi POS in training set. Besides, we build rules for words with distinctive features aiming at correcting errors, such as “的”, numbers and English characters, etc.

Figure 2 shows the post-processing step after POS tagging by CRFs model. As shown in Figure 2, we respectively post-process single class words and ambiguity words according to CRF score.

(1) Single class word processing module

The post-processing of single class words consults the single class word dictionary and CRFs score. When the score from CRFs is higher than 0.9, we take the POS from CRFs as the final POS; otherwise, POS of the word is corrected by the POS in the single class word dictionary.

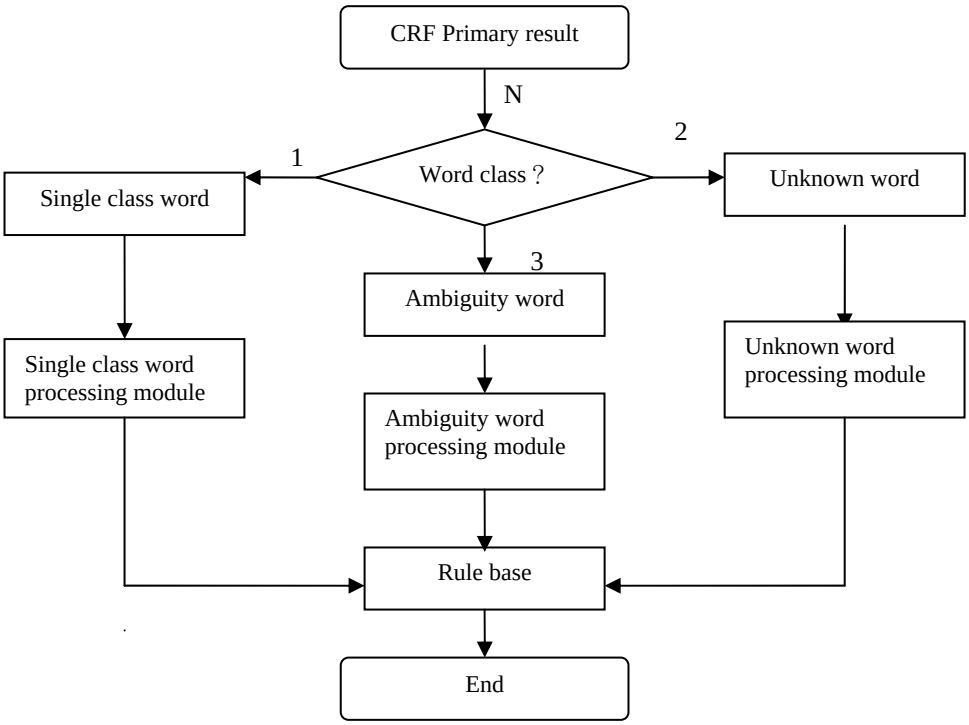

Figure2: Post-processing architecture after CRF labeling

(2) Ambiguity word processing module

The post-processing of ambiguity words consults the ambiguity word dictionary and CRFs score. When the POS from CRFs belongs to the POS of the word in the ambiguity word dictionary, we take the POS from CRFs as the final POS; otherwise, we examine the score of CRF, if the score is less than 0.4, the final POS of the word is the POS who has the highest score (has highest frequency), or else taking POS from CRF as the final POS.

(3) Unknown word processing module

The unknown words are the words not in training set. By analyzing the examples, we find that there are great deals of person names, location names, organization names and numbers, etc. And the words have characteristics when building word, therefore, we set up rules for processing.

2.3 Experiment results

Table 2 shows the comparative experimental results of POS tagging using two methods.

Method	EDSs precision	CS precision
CRF	92.83%	89.42%
CRF + post-processing	93.96%	91.05%

Table 2: Comparative POS tagging results

3 Parsing system

The paper uses divide-and-conquer strategy (Shiuan 1996 et al., Braun 2000 et al., Lyon 1997 et al.)for parsing. Firstly, we recognize MNP for an input sentence, which divide the sentence into two kinds of parts. One kind is MNPs, and the other one is frame which is a new sentence generating by replacing MNP using its head word. Secondly, we use parsing approach based on chunking (Abney, 1991, Erik Tjong and Kim Sang, 2001) and a searching algorithm in decoding. Thirdly, we combine the parsing trees of MNPs and frame, which obtains the full parsing tree of the original sentence. Figure 3 shows the architecture of paring system.

3.1 MNP recognition

Maximal Noun Phrase (MNP) is the noun phrase which is not contained by any other noun phrases. We use Berkeley parser (2009 1.0) for MNP recognition. We first use Berkeley parser to parse sentences after POS tagging, and then we tag MNPs from the parsing results. As the following example:

Berkeley parser result: dj[中国/nS vp[重视/v vp[发展/v np[pp[与/p np[欧洲/nS 国家/n]] 的/uJDE 关系/n]]]]

MNP recognition result: 中国/nS 重视/v 发展/v np[与/p 欧洲/nS 国家/n 的/uJDE 关系/n]

The results of MNP recognition EDSs analysis and CS parsing are as table3:

	P	R	F
EDSs	85.3202%	85.998%	85.6578%
CS	77.7102%	79.2782%	78.4864%

Table 3: Results of MNP recognition

3.2 Head recognition of MNP and generation of frame

In this paper, the new sentence in which MNPs are replaced by their head word is defined as the sentence's frame. The head of MNPs is identified after MNP recognition and then they are used to replace the original MNP, and finally the sentence's frame is formed. We use the rules to recognize the head of MNP. Usually, the last word of MNP is the head of the phrase, which can represent the MNP in function. For example: "[该/r 学派/n] 同样/ad 主张/v 消除/v [干预/v 造成/v 的/u 阻碍/n]。" In this sentence" 该/r学派/n" and " 干预/v 造成/v 的/u 阻碍/n" are MNPs. If we omit the modifier in MNP, for example "[学派/n] 同样/ad 主张/v 消除/v [阻碍/n]。", the meaning of the sentence will not be changed. Because the head can represent the syntax function of MNP, we can use the head for parsing, which can avoid the effect of the modifier of MNP on parsing and reduce the complexity of parsing.

However, the components of MNP are complicated, not all of the last word of MNP can be the head of MNP. The paper shows that if MNP has parentheses, we can use the last word before parentheses as the head. When the last word of MNP is "等", we use the second last word as the head.

3.3 Chunking with CRFs

The accuracy of chunk parsing is highly dependent on the accuracy of each level of

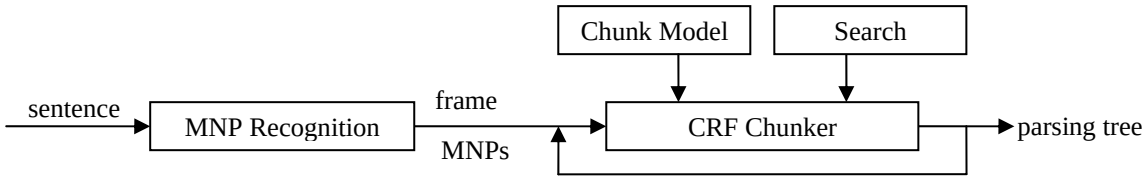

Figure3: Parsing system architecture

chunking. This section describes our approach to the chunking task. A common approach to the chunking problem is to convert the problem into a sequence tagging task by using the "BIEO" (B for beginning, I for inside, E for ending, and O for outside) representation.

This representation enables us to use the linear chain CRF model to perform chunking, since the task is simply assigning appropriate labels to sequence.

3.3.1 Features

Table 4 shows feature templates used in the whole levels of chunking. In the whole levels of chunking, we can use a rich set of features because the chunker has access to the information about the partial trees that have been already created (Yoshimasa et al., 2009). It uses the words and POS tags around the edges of the covered by the current non-terminal symbol.

Word Unigrams	W_{-2} , W_{-1}, W_0, W_1, W_2,
Word Bigrams	$W_{-2}W_{-1}$, $W_{-1}W_0$, W_0W_1, W_1W_2, W_0W_{-2}, W_0W_2,
Word Trigrams	$W_0W_{-1}W_{-2}$, $W_0W_1W_2$
POS Unigrams	P_{-3}, P_{-2} , P_{-1} , P_0 , P_1, P_2, P_3,
POS Bigrams	$P_{-3}P_{-2}$, $P_{-2}P_{-1}$, $P_{-1}P_0$, P_0P_1, P_1P_2, P_2P_3, P_0P_{-2}, P_0P_2,
POS Trigrams	$P_{-3}P_{-2}P_{-1}$, $P_{-2}P_{-1}P_0$, $P_{-1}P_0P_1$, $P_0P_1P_2$, $P_1P_2P_3$
Word & POS	W_0P_0, W_0P_{-1}, W_0P_1,
Word & WordCount	W_0C_0
Word & FirstWord	W_0F_0 , $W_{-1}F_0$
Word & LastWord	W_0L_0, W_1L_0
Word & Symbol	W_0S_0

Table 4: Feature templates used in parsing system. W represents a word, P represents the part-of-speech of the word, C represents the sum of the chunk containing the word, F represents the first word of the chunk containing the word, L represents the last word of the chunk containing the word, S represents that the word is a non-terminal symbol or not. W_j is the current word; W_{j-1} is the word preceding W_j, W_{j+1} is the word following W_j.

3.4 Searching for the Best Parse

The probability for an entire parsing tree is computed as the product of the probabilities output by the individual CRF chunkers:

$$score = \prod_{i=0}^{h} p(y_i / x_i)$$

We use a searching algorithm to find the highest probability derivation. CRF can score each chunker result by A* search algorithm, therefore, we use the score as the probability of each chunker. We do not give pseudo code, but the basic idea is as figure 4.

```
1: inti parser(sent)
2: Parse(sent, 1, 0)
3:
4: function Parse(sent, m, n)
5:  if sent is chunked as a complete sentence
6:     return m
7:  H = Chunking(sent, m/n)
8:   for h∈H do
9:    r = m * h.probability
10:     if r > n then
11:        sent2 = Update(sent, h)
12:        s = Parse(sent2, r, n)
13:        if s > n then n = s
14:    return n
15: function Chunking(sent, t)
16: perform chunking with a CRF chunker and return a set of chunking hypotheses whose
17: probabilities are greater than t.
18: function Update(sent, h)
19:  update sequence sent according to chunking hypothesis h and return the updated sequence.
```

Figure 4: Searching algorithm for the best parse

It is straightforward to introduce beam search in this search algorithm—we simply limit the number of hypotheses generated by the CRF chunker. We examine how the width of the beam affects the parsing performance in the

experiments. We experiment beam width and we adopt the beam width of 4 at last.

3.5 Head Finding

Head finding is a post process after parsing in our system. The paper uses method combining statistics and rules to find head. The selected statistical method is CRF model. The first step is to train a CRF classifier to classify each context-free production into several categories. Then a rule-based method is used to post process the identification results and gets the final recognition results. The rule-based post-processing module mainly uses rule base and case base to carry out post-processing.

3.6 Head finding based on CRFs

The head finding procedure proceeds in the bottom-up fashion, so that the head words of productions in lower layers could be used as features for the productions of higher layers (Xiao chen et al. 2009).

Atom template	Definition
CurPhraseTag	The label of the current word
LCh_Word	The left most child
RCh_Word	The right most child
LCh_Pos	The POS of the left most child
MCh_Pos	The POS of the middle child
RCh_Pos	The POS of the right most child
NumCh	The number of children
CurPhraseTag $\pm$ 1	The labels of the former phrase and the latter

Table 5: Atom templates for Head finding

Complex Template
CurPhraseTag/ NumCh, CurPhraseTag/ LCh_Word, CurPhraseTag/LCh_Pos, CurPhraseTag/LCh_Pos/RCh_Pos, CurPhraseTag/NumCh/LCh_Pos/ RCh_Pos, CurPhraseTag/NumCh/LCh_Word/LCh_Pos/MCh_Pos/RCh_Word/RCh_Pos, LCh_Word/LCh_Pos, CurPhraseTag/MCh_Pos, NumCh/LCh_Pos/ MCh_Pos/ RCh_Pos, CurPhraseTag/ NumCh/ MCh_Pos, CurPhraseTag/LCh_Word/LCh_Pos/MCh_Pos/RCh_Word/RCh_Pos, LCh_Word/ LCh_Pos, LCh_Pos/ MCh_Pos, CurPhraseTag/NumCh, RCh_Word/RCh_Pos, NumCh/LCh_Word/LCh_Pos/MCh_Pos/RCh_Word/RCh_Pos

Table 6: Complex templates for Head finding

The atom templates are not sufficient for labeling context; therefore, we use some complex templates by combining the upper atom templates for more effectively describing context. When the feature function is fixed, the atom templates in complex templates are instantiated, which will generate features.

The final feature templates are composed of the atom templates and the complex templates. The feature templates of the head recognition in phrases contain 24 types.

3.7 Head Finding based on rules

Through the analysis of error examples, we found that some CRFs recognition results are clearly inconsistent with the actual situation; we can use rules to correct these errors, thus forming a rule base. Example-base is a chunk-based library built through analysis and processing on the training corpus. The Example-base is composed of all the bottom chunk and high-level chunk in training corpus. High-level phrases are the bottom chunk replaced by heads.

3.8 Experiment results of head finding

Table 7 shows the comparative experiment results of head recognition.

	Total Num	Wrong Num	Precision
CRFs	7035	93	98.68%
CRFs + rule-base+ case-base	7035	74	98.95%

Table7: Comparative results of head recognition

4 Experiment of parsing system

We perform experiments on the training set and testing set of Tsinghua Treebank provided by CIPS-SIGHAN-ParsEval-2010. For the direct influence of parsing result by the length of sentence, we count the length distribution of corpus.

Table 8 shows that the length of training set and testing set of EDSs is mostly less than 20 words. The length of training set of CS is evenly distributed, while the length of testing set is between 30 and 40 words.

The paper adopts divide-and-conquer strategy to parsing; therefore, we conduct the

comparative experiment of MNP parsing and frame parsing. In addition, the results of MNP parsing and frame parsing depend on the length largely, so we list the length distribution of MNP and frame of EDSs and CS as table 9 and table 10.

	EDSs		CS	
length	training set	testing set	training set	testing set
[0, 10)	50.68%	64.30%	10.59%	0
[10,20)	37.27%	29.50%	27.55%	0
[20,30)	8.64%	5.40%	26.37%	79.9%
[30,40)	2.31%	0.60%	16.63%	20.1%
40≤	1.10%	0.20%	18.86%	0

Table 8: Length distribution of EDSs and CS

We define Simple MNP (SMNP) whose length is less than 5 words and Complete MNP (CMNP) whose length is more than 5 words.

	EDSs		CS	
length	training set	testing set	training set	testing set
[0,5)	55.30%	62.46%	55.42%	59.45%
[5,10)	32.66%	29.69%	32.57%	30.77%
[10,20)	10.03%	6.75%	10.03%	8.65%
20≤	2.00%	1.09%	1.98%	1.12%

Table 9: Length distribution of MNP

Table 9 shows the length distribution of MNP in training set and testing set of sub-sentence is consistent in basic, but the SMNP distribution of EDSs is 3.01% less than CS, which illuminates the complexity of MNP in CS is higher than in EDSs.

	EDSs		CS	
length	training set	testing set	training set	testing set
[0,5)	45.84%	47.20%	10.17%	1.00%
[5, 10)	43.58%	44.00%	24.14%	10.80%
[10, 20)	9.98%	8.70%	41.31%	62.20%
20≤	0.60%	0.10%	24.38%	26.00%

Table 10: Length distribution of frame

Table 10 shows the length distribution of frame in training set and testing set of EDSs is consistent in basic, while the CS is non-consistent. For the frame whose length is less than 5 words, the frame length distribution of training set is 9.17% higher than the testing set; for the frame whose length is more than 5 words and less than 10 words, the training set is 7.65% lower than testing; and for the frame whose length is between 10 words and 20 words, the testing set is 20.09% higher compared with the training set. From another aspect, in testing set, CS is 46.2% lower compared with EDSs for frame whose length is less than 5. Therefore, the complexity of frame in CS is higher than in EDSs.

As shown in Table 8, 9 and 10, the length distribution of testing set shows that the paring unit length of EDSs is reduced to less than 10 from less than 20 in original sentence and CS is reduced to less than 20 from between 30 and 40 after dividing an original sentence into MNPs parts and frame part. The above data indicate the divide-and-conquer strategy reduces the complexity of sentences significantly.

We can conclude that the parsing result of CS is lower than EDSs from Table 11, which is due to the higher complexity of MNP and frame in CS compared with EDSs from the results of Table 9 and Table 10. In addition, we obtain about 1% improvement compared with Berkeley parser in MNP and Frame parsing result in EDSs from Table 11 and Table 12, which indicates that our method is effective for short length parsing units. In particular, Table 12 shows that our result is 1.8% higher than Berkeley parser in the frame parsing of CS. Due to the non-consistent frame length distribution of training set and testing set in CS from Table 10, we find that Berkeley parser largely depends on training set compared with our method.

To more fairly compare the performance of our proposed method, the comparative results are shown as Table 13, the first one (Model01) is combination method of MNP pre-processing and chunk-based, and the chunk-based result which adopts CCRFs method with searching algorithm; the second one (Berkeley) is the parsing result of Berkeley parser; the third one (Model02) also is combination method of MNP pre-processing and chunk-based, and the chunk-based result which adopts CCRFs method only; and the lase one (Model03) is the chunk-based result which adopts CCRFs method with searching algorithm.

	method	P	R	F
EDSs	Berkeley	87.5746%	87.8365%	87.7053%
	Proposed Method	88.5752%	88.6341%	88.6047%
CS	Berkeley	84.4755%	84.9182%	84.6963%
	Proposed Method	84.7535%	85.046%	84.8995%

Table 11: Comparative results of MNP parsing

	method	P	R	F
EDSs	Berkeley	91.3411%	91.1823%	91.2617%
	Proposed Method	92.4669%	92.0765%	92.2713%
CS	Berkeley	85.4388%	85.3023%	85.3705%
	Proposed Method	87.3357%	87.0357%	87.1854%

Table12: Comparative results of Frame parsing

	P	R	F
Model 01	85.42%	85.35%	**85.39%**
Berkeley	84.56%	84.62%	84.59%
Models 02	85.31%	85.30%	85.31%
Models 03	83.99%	83.77%	83.88%

Table13: Comparative results of EDSs

	dj constituent			fj constituent			overall F
	P	R	P	R	F	F	F
Model 01	78.64%	78.73%	78.69%	70.22%	71.62%	70.91%	**74.80%**
Berkeley	78.37%	78.16%	78.26%	69.43%	72.42%	70.89%	74.58%
Models 02	78.18%	78.30%	78.24%	70.20%	70.98%	70.59%	74.41%
Models 03	77.38%	77.41%	77.39%	70.39%	70.01%	70.24%	73.82%

Table14: Comparative results of CS

From Table 13, we can see that Model01 performance in EDSs is improved by 0.08% than Model02, and the searching algorithm helps little in EDSs analysis. From Table 14, we can see that Model01 performance in CS is improved by 0.4% than Model02, better than Berkeley parser result with search algorism. Overall, in EDSs analysis, Model01 performance is improved by 0.8% than Berkeley parser, and in overall F-measure of CS, Model01 performance is 0.22% higher than Berkeley parser. From Table 13 and 14, We can see that Model01 performance in EDSs is improved by 1.51% than Model03 and the Model01 in CS is improved by 0.98% than Model03, and the MNP pre-processing helps.

5 Conclusions

We participate in two tasks - EDS Analysis and CS Parsing in CLPS-SIGHAN- ParsEval-2010. We use divide-and-conquer strategy for parsing and a chunking-based discriminative approach to full parsing by using CRF for chunking. As we all know, CRF is effective for chunking task. However, the chunking result in the current level is based on the upper level in the chunking-based parsing approach, which will enhance ambiguity problems when the input of the current level contains non-terminal symbols, therefore, the features used in chunking is crucial. This paper, for effectively using the information of partial trees that have been already created, keeps the terminal symbols in the node containing non-terminal symbols for features. Our experiments show that these features are effective for ambiguity problems.

We suppose that MNP pre-processing before statistical model can significantly simplify the analysis of complex sentences, which will have more satisfatory results compared with using statistical model singly. The current results

show that the MNP pre-processing does simplify the complex sentences. However, the performance of MNP recognition and the parsing of MNP need to be improved, which will be our next work.

References

Yoshimasa Tsuruoka, Jun'ichi Tsujii, Sophia Anaiakou. 2009. Fast Full Parsing by Linear-Chain Conditional Random Fields. In *Proceedings of EACL'09*, pages 790-798.

Xiao chen, Changning Huang, Mu li, Chunyu Kit. 2009. Better Parser Combination. In *CIPS-ParsEval-2009*, pages 81-90.

Abney, S.. 1991. Parsing by chunks, Principle-Based Parsing, Kluwer Academic Publishers.

Erik Tjong, Kim Sang. 2000. Transforming a chunker to a parser. In J.Veenstra W.daelemans, K Sima' an and J. Zavrek, editors, *Computational Linguistics in the Netherlands 2000*, Rodopi, page 177-188.

P.L. Shiuan, C.T.H. Ann. 1996. A Divided-and-Conquer Strategy for Parsing. In *Proc. of the ACL/SIGPARSE 5th International Workshop on Parsing Technologies.* Santa Cruz, USA, 1996, pages 57-66

C. Braun, G. Neumann, J, Piskorski. 2000. A Divide-and-Conquer Strategy for Shallow Parsing of German Free Texts. In *Proc. of ANLP-2000*. Seattle, Washington, 2000, pages 239-246.

C.Lyon, B.Dickerson. 1997. Reducing the Complexity of Parsing by a Method of Decomposition International Workshop on Parsing Technology, 1997, pages 215-222.

Qiaoli Zhou, Xin Liu, Xiaona Ren, Wenjing Lang, Dongfeng Cai. 2009. Statistical parsing based on Maximal Noun Phrase pre-processing. In *CIPS-ParsEval-2209.*

P.L. Shiuan, C.T.H. Ann. A Divide-and-Conquer Strategy for Parsing. In: Proc. of the ACL/SIGPARSE 5th International Workshop on Parsing Technologies. Santa Cruz, USA, 1996. 57-66.

C. Braun, G. Neumann, J. Piskorski. A Divide-and-Conquer Strategy for Shallow Parsing of German Free Texts. In: Proc. of ANLP-2000. Seattle, Washington, 2000. 239-246.

C. Lyon, B. Dickerson. Reducing the Complexity of Parsing by a Method of Decomposition. International Workshop on Parsing Technology. 1997. 215-222.