A CHUNKING-AND-RAISING PARTIAL PARSER

Hsin-Hsi Chen
Yue-Shi Lee
Department of Computer Science and Information Engineering
National Taiwan University
Taipei, Taiwan, R.O.C.
E-mail: hh_chen@csie.ntu.edu.tw; leeys@nlg.csie.ntu.edu.tw

Abstract
Parsing is often seen as a combinatorial problem. It is not due to the properties of the natural languages, but due to the parsing strategies. This paper investigates a Constrained Grammar extracted from a Treebank and applies it in a non-combinatorial partial parser. This parser is a simpler version of a chunking-and-raising parser. The chunking and raising actions can be done in linear time. The short-term goal of this research is to help the development of a partially bracketed corpus, i.e., a simpler version of a treebank. The long-term goal is to provide high level linguistic constraints for many natural language applications.

1 Introduction
Recently, many parsers [1-10] have been proposed. Of these, some [1-7] belong to full parsers and some [8-10] partial parsers. Because the polycategory of a word and the use of the formal grammar, parsing is often seen as a combinatorial problem [11]. A feasible way to treat this problem is to separate the work of category determination from a parser and adopt a new parsing scheme. That is, automatic part-of-speech tagging serves as preprocessing of the parser. The tagging problem has been investigated by many researchers [12-18], and many interesting results have been demonstrated. Thus the remaining problem is how to construct a new non-combinatorial parser to increase the parsing efficiency and decrease the parsing ambiguity. This paper will propose a chunking-and-raising partial parser for such a goal. Section 2 introduces the framework of this parser. Section 3 specifies the training corpus - Lancaster Parsed Corpus, and Section 4 touches on how to extract Constrained Grammar from this corpus. Section 5 presents a simplified parsing algorithm based on Constrained Grammar. Before concluding the experimental results and the related works are shown.

2 Framework of a Chunking-and-Raising Parser

Fig. 1. The Chunking-and-Raising Scheme
In this scheme, parsing can be regarded as a sequence of actions of chunking and raising. Fig. 1 shows the configuration. An input sentence $W$ is input to a part-of-speech tagger and a (lexical) tag sequence $P$ is produced. The output of the tagger is the input of the parser. The chunking model of the parser groups some tags into chunks. The raising model assigns a (syntactic) tag to each chunk and generates a new tag sequence $P'$. The chunking and raising actions are repeated until no new chunking sequence is generated.

Consider an example. Let the input sentence be "Mr. Macleod went on with the conference at Lancaster House despite the crisis which had blown up ". The corresponding part-of-speech sequence is shown as follows.

$$NPT \ NP \ VBD \ RP \ IN \ ATI \ NN \ IN \ NP \ NPL \ IN \ ATI \ NN \ WDT \ HVD \ VBN \ RP$$

The chunking model produces a chunking sequence shown below.

$$[NPT \ NP] [VBD] [RP] IN \ ATI \ NN \ IN [NP \ NPL] IN \ ATI \ NN [WDT] [HVD \ VBN] [RP]$$

Seven parts-of-speech which cannot be formed into chunks at this step remain in the sequence. The raising model then generates the following chunking-and-raising sequence.

$$[N \ NPT \ NP \ N] [V \ VBD \ V] [R \ RP \ R] IN \ ATI \ NN \ IN [NP \ NPL \ N] IN \ ATI \ NN [Nq \ WDT \ Nq] [V \ HVD \ VBN \ V] [R \ RP \ R]$$

$[N \ NPT \ NP \ N]$ denotes that the chunk $[NPT \ NP]$ is raised to $N$. Similarly, the chunks $[VBD], [RP], [NP \ NPL], [WDT], [HVD \ VBN]$ and $[RP]$ are raised to $V, R, N, Nq, V$ and $R$, respectively. The seven syntactic tags and the remaining lexical tags form a new tag sequence and it is sent to the next chunking-and-raising cycle. If the word information is put back into the sequence, a partial parsed sentence is generated as follows.

$$[N \ Mr. \ NPT \ Macleod \ NP \ N] [V \ went_Im_VBD \ V] [R \ on_Im_RP \ R] \ with_Im IN \ the_Im_\text{ATI} \ \text{conference}_NN \ at_Im \ IN \ [N \ Lancaster_\text{NP} \ House_\text{NPL} \ N] \ despite_Im IN \ the_Im_\text{ATI} \ crisis_NN [Nq \ which_Im \ WDT \ Nq] [V \ had_Im \ HVD \ blown_Im \ VBN \ V] [R \ up_Im_RP \ R]}$$

After one more chunking-and-raising cycle, the partial parsed sentence is generated as follows.

$$[N \ Mr. \ NPT \ Macleod_\text{NP} \ N] [V \ went_Im_VBD_Im \ V] [R \ on_Im_RP_Im \ R] \ with_Im IN \ the_Im_\text{ATI} \ \text{conference}_NN \ [F \ at_Im \ IN \ [N \ Lancaster_\text{NP} \ House_\text{NPL} \ N] \ P] \ despite_Im IN \ the_Im_\text{ATI} \ \text{crisis}_NN [Fr \ [Nq \ which_Im \ WDT \ Nq] [V \ had_Im \ HVD \ blown_Im \ VBN \ V] [R \ up_Im_RP \ R]}$$

In other words, a new tag sequence “$V R IN ATI NN P IN ATI NN Fr."$ is generated. We repeat these two actions until no more chunking sequence is generated.

A Constrained Grammar is extracted from a Treebank and is applied in a simpler version of chunking-and-raising parser. The chunking and raising actions are applied only once in this parser. Thus it only produces a linear chunking-and-raising sequence, not a hierarchical annotated tree. The experimental framework is shown in Fig. 2.

![Fig. 2. The Experimental Framework](image-url)
3 Lancaster Parsed Corpus

The Lancaster Parsed Corpus is a modified and a condensed version of Lancaster-Oslo/Bergen (LOB) Corpus. It only contains one sixth of LOB Corpus, but involves more information than LOB Corpus. The corpus consists of fifteen kinds of texts (about 150,000 words). Each category corresponds to one file. The tagging set of Lancaster Parsed Corpus is extended and modified from LOB Corpus. The following shows a snapshot of Lancaster Parsed Corpus.

| A01 1 | [SP by_IN [N Trevor_NP Williams_NP N]P] ... S] |
| A01 2 | [S[N a_AT move_NN [Ti[Vi to_TO stop_VB Vi]]N 0Mr_NPT Gaitskell_NP N]P from_IN [Tg[Vg nominating_VBG Vg]N any_DT more_AP labour_NN life_NN peers_NNS N]Tg[P]Tg]N]V is_BEZ V][Ti[Vi to_TO be_BE made_VBN Vi]]P at_IN [N a_AT meeting_NN [Po of_INO [N labour_NN 0MPs_NPTS N]P]N]P[N tomorrow_NR N]Tg] ... S] |
| A01 3 | [S&N 0Mr_NPT Michael_NP Foot_NP N]V has_HVZ put_VBN V]R down_RP R] [N a_AT resolution_NN [P on_IN [N the_ATI subject_NN N]P]N]S+ and_CC [Na be_PP3AS Na]V is_BEZ V][Ti[Vi to_TO be_BE backed_VBN Vi]]P by_IN [N 0Mr_NPT Will_NP Griffiths_NP N] ... N 0MP_NPT [P for_IN [N Manchester_NP Exchange_NP N]P]N]P]Tg]S+] ... S&] |
| A01 4 | [S[Fa though_CS [Na they_PP3AS Na]V may_MD gather_VB V]N some_DT left­wing_JJB support_NN N]Fa] ... [N a_AT large_JJ majority_NN [Po of_INO [N labour_NN 0MPs_NPTS N]P]N]N]V are_BER V]J likely_JJ J]Tg[Vi to_TO turn_VB Vi]V]R down_RP R] [N the_ATI Foot-Griffiths_NP resolution_NN N]Tg] ... S] |
| A01 5 | *' *' [S[V abolish_VB V]N Lords_NPTS N] ***: ***: ... S] |

These are extracted from the first five sentences of category A. Before each sentence, a unique reference number, e.g., "A01 1", denotes its source. Each word is appended with a lexical tag, e.g., "by_IN", "Trevor_NP". The syntactic tag is shown by opening and closing brackets.

To indicate that phrases or clauses are coordinated, the symbols "&", "," or "+" will be used at the end of a phrase or a clause tag. An example is listed as follows.

\[
[ N & mothers_NNS _, [ N- children_NNS N- ] [ N+ and_CC sick_JJ people_NNS N+] 
\]

The first coordinated phrase is not labeled any tag. The second and the third coordinated phrases are labeled N- and N+, respectively. This is because N- or N+ tends to include ellipsis. Table 1 gives an overview of the Lancaster Parsed Corpus. In our experiment, those parsed sentences that don't begin with "[S" and end with "]S" are removed from the training corpus. Thus "A01 5" is deleted.

| Category | # of Sentences | # of Words |
|----------|----------------|------------|
| A        | 3403           | 9410       |
| B        | 3648           | 9999       |
| C        | 2870           | 8225       |
| D        | 3534           | 10110      |
| E        | 2990           | 9356       |
| F        | 2962           | 8562       |
| G        | 2185           | 6813       |
| H        | 2266           | 6524       |

Total 56162 157617
4 The Constrained Grammar

A Constrained Grammar is extracted from the Lancaster Parsed Corpus. Because the chunking and raising actions are applied only once in the preliminary experiment, only those rules that appear on the lowest level of the parsing trees form a Constrained Grammar.

Consider a sentence "Northern Rhodesia is a member the federation.". Its parsing tree is shown in Fig. 3. Three constrained rules shown below are extracted from this parsing tree.

(*) NP NP (BEZ) -> N
(NP) BEZ (ATI) -> V
(INO) ATI NN () -> N

Two constraints enclosed in parentheses, i.e., the left and the right constraints, are added into each constrained rule. For example, the constrained rule, (NP) BEZ (ATI) -> V, has the left constraint NP and the right constraint ATI. It means that chunk [BEZ] can be raised to V when its left tag is NP and its right tag is ATI. The other two rules have the similar interpretations. The asterisk marks the beginning of the sentence. A more complicated example is given as follows:

The following constrained rules are extracted from this example:

(NN) TO VB (NPT) -> Vi
(N) VBG (DTI) -> Vg
(NNS) BEZ (TO) -> V
(INO) NN NPTS (NR) -> N

Furthermore, the same constrained rules are grouped into one. Under this way, total 20,002 constrained rules are extracted from the Lancaster Parsed Corpus. All the constrained rules are examined and 219 conflict rules are found. The conflicts result from the inconsistent annotations in the corpus. Some conflict rules are listed below. The number enclosed in the parentheses denotes the frequency of the rule.

(NNS) VB (ATI) -> V (6), Vr (1)
(NNS) VBN (...) -> Vn (10), Vr (1)
(NNS) VBN (IN) -> Vn (54), V (3)
(NNS) VBN (RB) -> Vn (4), V (1)

In the above samples, (NNS) VB (ATI) -> V (6), Vr (1), means that VB can be raised to V (Vr) with frequency 6 (1). To avoid this inconsistency, some rules having lower frequencies are deleted. Finally, a decision tree is used to model the remaining unconflict rules. Fig. 4 shows the decision tree for the following rules:

(DT) BEDZ (WDT) -> V
(DT) BEDZ (WRB) -> V
(DT) BEG (PN) -> Vg
(DT) BEZ (,) -> V
(DT) BEZ (ABL) -> V
(DT) BEZ (ABN) -> V
A rule can be applied when its left constraints, chunks and its right constraints are satisfied. That is, a path is found in the decision tree.

5 The Partial Parsing Algorithm

The partial parsing algorithm based on Constrained Grammar is proposed below.

```
Partial_Parser(Tag_Sequence)
Begin
   C_Position = 1;
   While C_Position <= N Do
      Begin
         Find = 0;
         For Chunk_Length = 8, ..., 1 Do
            Begin
               If (C_Position + Chunk_Length - 1) <= N Then
                  Begin
                     If Search Decision Tree for
                        Tag_Sequence[C_Position - 1] as Left Constraint,
                        Tag_Sequence[C_Position .. (C_Position + Chunk_Length - 1)] as Chunk and
                        Tag_Sequence[C_Position + J] as Right Constraint
                        Is Successful Then
                           Begin
                              Output "[ ";
                              Output Raised Tag;
                              For Position = C_Position, ..., (C_Position + Chunk_Length - 1) Do
                                 Output Tag_Sequence[Position];
                              Output Raised Tag;
                              Output "]";
                              Find = 1;
                              Goto Done;
                           End
                  End
            End
         End
      End
   End
Done: If Find = 1 Then C_Position = C_Position + Chunk_Length - 1;
      Else Output Tag_Sequence[C_Position];
      C_Position = C_Position + 1;
End
```

Variable Find denotes whether a chunk is found in the decision tree or not and Variable C_Position means current position. Assume that the input sentence contains N words, and the symbol * is added to the beginning position (0) and the ending position (N+1) to facilitate the process. The processes for N=6, C_Position=1 and Chunk_Length=8 (7 and 6) are shown in Fig. 5.
6 The Experimental Results

The performance evaluation model compares the chunking-and-raising result \( P' \) with the corresponding syntactic structure \( T \). The evaluation criterion is to count how many tags are assigned correctly. For example, there is a parsing tree - say, \( [ A \ [ B \ [ C \ W_1 \ P_1 \ W_2 \ P_2 \ C \ ] \ W_3 \ P_3 \ [ D \ W_4 \ P_4 \ D \ ] \ B \ ] \ [ E \ W_5 \ P_5 \ W_6 \ P_6 \ E \ ] \ A \ ] \). If the parsing result is \( [ F \ W_1 \ P_1 \ W_2 \ P_2 \ F \ ] \ [ G \ W_3 \ P_3 \ G \ W_4 \ P_4 \ [ E \ W_5 \ P_5 \ W_6 \ P_6 \ E \ ] \) \), then 2 tags, i.e., \( P_5 \) and \( P_6 \), are assigned correctly. The tags \( P_1 \) and \( P_2 \) are wrong because the raised tag is wrong, i.e., it must be \( C \). The tag \( P_3 \) is wrong because \( P_3 \) cannot be a chunk. Similarly, the tag \( P_4 \) is wrong because it must be chunked and raised to \( D \). According to this criterion, the experimental results are shown in Table 2.

### Table 2. The Experimental results for Inside Test

| Category | Correct | Wrong | Total | Correct Rate (%) |
|----------|---------|-------|-------|------------------|
| A        | 7792    | 492   | 8284  | 94.06%           |
| B        | 8839    | 553   | 9392  | 94.11%           |
| C        | 7183    | 555   | 7738  | 92.83%           |
| D        | 9137    | 611   | 9748  | 93.73%           |
| E        | 8658    | 614   | 9272  | 93.38%           |
| F        | 7810    | 562   | 8372  | 93.29%           |
| G        | 5794    | 472   | 6266  | 92.47%           |
| H        | 5872    | 652   | 6524  | 90.01%           |
| J        | 7600    | 711   | 8311  | 91.45%           |
| K        | 10338   | 463   | 10801 | 95.71%           |
| L        | 11394   | 596   | 11990 | 95.03%           |
| M        | 6958    | 369   | 7327  | 94.96%           |
| N        | 11147   | 506   | 11653 | 95.66%           |
| P        | 11549   | 548   | 12097 | 95.47%           |
| R        | 7878    | 502   | 8380  | 94.01%           |
| Total    | 127949  | 8206  | 136155| 93.97%           |

If the inconsistency problem of the corpus does not occur, the performance can be better. When we remove one file from training corpus and use this file as the testing corpus, the experimental results are listed in Table 3.

### Table 3. The Experimental results for Outside Test

| Category | Correct | Wrong | Total  | Correct Rate (%) |
|----------|---------|-------|--------|------------------|
| K        | 8324    | 2477  | 10801  | 77.07%           |
| P        | 9366    | 2731  | 12097  | 77.42%           |
| N        | 9166    | 2487  | 11653  | 78.66%           |

In these experiments, \( K \), \( P \) or \( N \) are removed from training corpus. The performance is decreased. It reveals that the training corpus is still not large enough. Structure Mapping between different treebanks [19] provides a feasible way to obtain a larger corpus. In this way, much more reliable
statistic information can be trained from the large-scale treebanks, so that the feasibility of the parser is assured.

7 Related Works

Chen and Chen [20] propose a probabilistic chunker to decide the implicit boundaries of constituents and utilize the linguistic knowledge to extract the noun phrases by a finite state mechanism. Rather than using a treebank as a training corpus, Chen and Lee [21] also propose a probabilistic chunker based on parts-of-speech information only. However, the evaluation adopted in these two papers is not very strict. Consider the following parsed sentence, which is extracted from Susanne Corpus.

```
[ S [ Nns:s The_ATI [ Nns Fulton_NP County_NPL Nns ] Grand_JJ Jury_NN Nns:s ] [ Vd said_VBD Vd ] [ Nns: Friday_NR Nns:1 ] [ Fn:o [ Ns:s an_AT investigation_NN [ Po of_IN [ Ns [ G Atlanta's_NP$ G ] recent_JJ primary_JJ election_NN Ns ] Po ] Ns:s ] [ Vd produced_VBD Vd ] [ Ns:o +no_ATI evidence_NN [ Fn that_CS [ Np:s any_AT irregularities_NNS Np:s ] [ Vd took_VBD Vd ] [ Ns:o place_NPL Ns:o ] Fn ] Ns:o ] Fn:o ] S ]
```

By the method proposed by Chen and Chen [20], the result is shown as follows.

```
[ The_ATI Fulton_NP County_NPL ] [ Grand_JJ Jury_NN ] [ said_VBD ] [ Friday_NR ] [ an_AT investigation_NN ] [ of_IN Atlanta's_NP$ ] [ recent_JJ primary_JJ election_NN ] [ produced_VBD ] [ +no_ATI evidence_NN ] [ that_CS any_ATI irregularities_NNS ] [ took_VBD ] [ place_NPL ]
```

By their evaluation criterion, only chunk [ of_IN Atlanta's_NP$ ] is wrong. But, it is clear that some chunks are wrong. By our criterion, the correct output should be:

```
The_ATI [ Nns Fulton_NP County_NPL Nns ] Grand_JJ Jury_NN [ Vd said_VBD Vd ] [ Nns: Friday_NR Nns:1 ] an_AT investigation_NN of_IN [ G Atlanta's_NP$ G ] recent_JJ primary_JJ election_NN [ Vd produced_VBD Vd ] +no_ATI evidence_NN that_CS any_ATI irregularities_NNS [ Vd took_VBD Vd ] [ Ns:o place_NPL Ns:o ]
```

The key issue is: when the chunked results are erroneous on the lowest level, the effects will be propagated to the upper level. Besides, the interpretation of chunks is another problem. Consider a sequence of chunks, i.e., [ A ] [ B C ] [ D ]. There may be at least two possible interpretations shown in Figs. 6 and 7. That makes the chunker difficult to scale up to a full parser.

![Fig. 6. Interpretation 1](image1)

![Fig. 7. Interpretation 2](image2)

8 Concluding Remarks

This paper proposes a linear-time partial parser. It is a simple version of a chunking-and-raising parser, but it can be extended to a full parser easily by performing more chunking and raising actions. Basically, the Constrained Grammar is provided to each level of the chunking-and-raising parser. Because each rule in the Constrained Grammar has left and right constraints, the grammar is different from the LL(k) grammar although they have the similar concepts, i.e., left to right scanning and lookahead. In contrast to the Inside-Outside optimization algorithm [5] which is very computationally intensive, this kind of parser is very simple but effective. The short-term goal of this research is to help the development of a partially bracketed corpus, i.e., a simpler version of a treebank. The long-term goal is to provide high level linguistic constraints for many natural language applications.
References

[1] A. Corazza, et al., "Stochastic Context-Free Grammars for Island-Driven Probabilistic Parsing," Proceedings of International Workshop on Parsing Technologies, 1991, pp. 210-217.
[2] S.K. Ng and M. Tomita, "Probabilistic LR Parsing for General Context-Free Grammars," Proceedings of International Workshop on Parsing Technologies, 1991, pp. 154-163.
[3] D.M. Magerman and C. Weir, "Efficiency, Robustness and Accuracy in Picky Chart Parsing," Proceedings of ACL, 1992, pp. 40-47.
[4] R. Bod, "Using an Annotated Corpus as a Stochastic Grammar," Proceedings of EACL, 1993, pp. 37-44.
[5] Y. Schabes, M. Roth and R. Osborne, "Parsing the Wall Street Journal with the Inside-Outside Algorithm," Proceedings of EACL, 1993, pp. 341-347.
[6] J. Dowding, R. Moore, F. Andry and D. Moran, "Interleaving Syntax and Semantics in an Efficient Bottom-Up Parser," Proceedings of ACL, 1994, pp. 110-116.
[7] D. Tugwell, "A State-Transition Grammar for Data-Oriented Parsing," Proceedings of EACL, 1995, pp. 272-277.
[8] D.D. McDonald, "An Efficient Chart-Based Algorithm for Partial Parsing of Unrestricted Texts," Proceedings of Applied Natural Language Processing, 1992, pp. 193-200.
[9] C. Jacquemin, "Recycling Terms into a Partial Parser," Proceedings of Applied Natural Language Processing, 1994, pp. 113-118.
[10] C. Lyon and B. Dickerson, "A Fast Partial Parse of Natural Language Sentences Using a Connectionist Method," Proceedings of EACL, 1995, pp. 215-222.
[11] J. Vergne, "A Non-Recursive Sentence Segmentation, Applied to Parsing of Linear Complexity in Time," Proceedings of NeMLaP, 1994, pp. 234-241.
[12] K.W. Church, "A Stochastic Parts Program and Noun Phrase Parser for Unrestricted Text," Proceedings of Applied Natural Language Processing, 1988, pp. 136-143.
[13] S.J. DeRose, "Grammatical Category Disambiguation by Statistical Optimization," Computational Linguistics, 1988, 14(1), pp. 31-39.
[14] E. Brill, "A Simple Rule-Based Part-of-Speech Tagger," Proceedings of Applied Natural Language Processing, 1992, pp. 152-155.
[15] D. Cutting, et al., "A Practical Part-of-Speech Tagger," Proceedings of Applied Natural Language Processing, 1992, pp. 133-140.
[16] D. Elworthy, "Does Baum-Welch Re-Estimation Help Taggers?" Proceedings of Applied Natural Language Processing, 1994, pp. 53-58.
[17] B. Miraldi, "Tagging English Text with a Probabilistic Model," Computational Linguistics, 1994, 20(2), pp. 155-171.
[18] P. Tapanainen and A. Voutilainen, "Tagging Accurately - Do'n't Guess If You Know," Proceedings of Applied Natural Language Processing, 1994, pp. 47-52.
[19] E. Atwell, et al., "AMALGAM: Automatic Mapping Among Lexico-Grammatical Annotation Models," Proceedings of the Balancing Act - Combining Symbolic and Statistical Approaches to Language, 1994, pp. 11-20.
[20] K.H. Chen and H.H. Chen, "Extracting Noun Phrases from Large-Scale Texts: A Hybrid Approach and its Automatic Evaluation," Proceedings of ACL, 1994, pp. 234-241.
[21] H.H. Chen and Y.S. Lee, "Development of Partially Bracketed Corpus with Part-of-Speech Information Only," Proceedings of Workshop on Very Large Corpora, 1995.