Chart Parser
For
Ill-formed Input Sentences

Kyongho Min • William H. Wilson

1 Introduction

Human beings often produce ill-formed sentences at various levels, typographical/morphological, syntactic, semantic, or pragmatic level, and can repair and understand them in a short time. When a sentence is not understood and anomalous, the meaning can be inferred by using a variety of linguistic and a real world knowledge. This paper focuses on the detection and correction of ill-formed sentences including a single syntactic error introduced by replacement of a valid word into a known/unknown word, insertion of an extra known/unknown word, and deletion of a word.

Ill-formed sentences can be handled in two ways: reject them or accept them after minor changes to make them acceptable. Many systems focus on the recovery of ill-formedness at the typographical level (Damerau & Mays, 1989), the morpho-syntactic level (Vosso, 1992), the syntactic level (Mellish, 1989; Weischedel & Sondheimer, 1983), the semantic level (Fass & Wilks, 1983), and pragmatic level (Granger, 1983). Those systems described how to identify a localised error and how to repair it using grammar independent rules (Mellish, 1989), grammar specific rules (meta-rules) (Weischedel & Sondheimer, 1983), semantic preferences (Fass & Wilks, 1983), and heuristic approaches (Damerau &

When detecting and correcting errors, a system can process both ill-formed and well-formed sentences with a single parser: Welschedel & Sondheimer (1988) called this one-stage error recovery. Such a system repairs an error as soon as it is identified. Two-stage error recovery strategy repairs an error using two parsers: a parser for well-formed sentences and a second parser for ill-formed sentences (Mellish, 1989). The second parser is invoked only if the first parser does not recognise an input sentence as well-formed. This strategy has the advantage that the recovery process cannot affect the performance of a parser on well-formed sentences in terms of speed or complexity of parsing strategies.

CHAPTER (CHart Parser for Two-stage Error Recovery) is a two-stage error recovery parser based on chart parsing and consists of a well-formed sentence chart parser (WFSCP) and an ill-formed sentence chart parser (IFSCP). The system invokes IFSCP only if the WFSCP cannot recognise the input string as well-formed. Compared to feature-unification-based grammar (Shieber, 1986), it employs a syntactic grammar incorporating syntactic selectional restrictions to reduce the production of categorial combinations (Ristad, 1990).

Specifically our system grammar gave different weights to constituents in a local tree, head constituents and non-head constituents, similar to a head-driven phrase structure grammar for head-driven parsing (Pollard & Sag, 1987), and uses the grammar for recovery from local errors which may include errors of lexical category, syntactic features, and of subcategorisation of verbs. CHAPTER can handle simple syntactic errors in sentences. This suggests that a generalised error detection and correction strategy without heuristic assumptions may be effective.

In section 2, we shall describe the structure of the system and in section 3, results obtained when parsing ill-formed sentences, and

chart parser for ill-formed input sentences

comparative results between the ACFG (Augmented Context-Free Grammar) version and an earlier straight CFG version. In section 4, we present problems, future developments and conclusions about the approaches taken.

2 Design of IFSCP

In this section, we describe both parser stages, but focus on the error-recovery stage. The system is based on chart parsing using a ACFG model. CHAPTER comprises two parsers, a syntactic grammar based on syntactic selectional restrictions using categorial features in GPSG (Gazdar, Klein, Pullum, & Sag, 1985), and a lexicon and verb subcategorisations based on the Oxford Advanced Learner’s Dictionary.

![Figure 1. The Structure of IFSCP.](image)

The WFSCP follows a bottom-up parsing strategy and parses left-to-right. The IFSCP employs three general strategies: goal processing and top-down expectation using syntactic rules based on ACFG, bottom-up satisfaction and producing a need-chart network using inactive arcs made by WFSCP, and an inference engine and retraction need-chart network for detection and repair of errors and reconstruction of local trees (Fig. 1). The bottom-up satisfaction process in IFSCP uses
both left-to-right and right-to-left parsing techniques.

2.1 Goal Processing and Top-down Expectation

2.2.1 Goal Processing

A goal is a local tree which cannot be found by bottom-up satisfaction process, because it contains syntactic errors (boxed constituents in Fig. 2). Such a goal is expanded by top-down expectation using augmented context-free grammar rules.

A goal is represented by the following information: (i) what phrasal constituent is needed for recovery in a sentence, (ii) what are its syntactic features, and (iii) what $need$-arc (see in section 2.2) generates this particular goal.

Such a goal is handled differently according to its characteristics (see Fig. 1): (a) a goal (e.g. $S$, NP1 NP2, ADJP in Fig. 2) is made for the top-down expectation process. (b) a goal (e.g. ADJ in Fig. 2) is achieved by inference engine to give a repaired local tree. (c) a goal (ADJP NOUN in Fig. 2) is processed by another bottom-up satisfaction process.

2.1.2 Top-down Expectation

With a goal, the relevant rules are invoked and expanded by a feature conjunction process between a goal and corresponding head daughters of rules, this is termed top-down expectation, and it identifies a lower local tree which may include syntactic errors.

When retrieving rules, first, the $MEL$ (Minimum Extension Length) and penalty score are considered. The MEL is the minimum number of lexical constituents from which the phrasal constituent can be constructed (e.g. the MEL of NP is 1 (pron "I")) and the penalty score is the number of words between two positions (from, to) of the goal minus MEL of each rule (e.g. the penalty score of a goal (NP from 0 to 3) with a rule, NP (det noun) is 1). Second, the syntactic features of the goal are conjoined with those of the invoked rules to pass down them to a head daughter of each rule.

Figure 2. goal production in a syntax.

2.2 Bottom-up Satisfaction & the Need-Chart Network

Bottom-up satisfaction comprises two distinct steps: looking up inactive arcs and creating the need-chart network. The need-chart network is composed of need-arcs which indicate what constituents are needed for the recovery of a specified phrasal constituent between two positions (i.e. active-arcs).

2.2.1 Looking up inactive arcs

After a rule is invoked by top-down expectation, if inactive arcs are found, that satisfy the leftmost or rightmost constituent of the rule’s rhs and the feature conjunction between the feature of an inactive constituent and the syntactic selectional restrictions of the rule, a need-arc is made using all the information from the rule and the found inactive arcs. The need-arc includes the following information: which constituents are already found and which constituents are needed for recovery (need-nodes, which become a goal) and syntactic features of the neednodes.

2.2.2 Producing Need-Chart Network

When producing a need-chart network, there are three possible
Chart Parser for Ill-formed Input Sentences

between its two positions - the deletion case), the need-arc is repaired by replacing it with inferred constituents.

The inference engine provides repaired constituents and their deviance notes and allows the recovery from specific errors in the bottommost local trees. The repaired constituents can retrace the need-chart network to get to the top-level (sentence) recovery from a bottommost recovery. This is termed a retraceable need-chart network.

3 Experimental Results

The experiments focused on the system's generality in detecting and correcting errors at the syntactic level without losing any possible repairs, as might occur as a result of employing a heuristic strategy. The test sentences used are ill-formed sentences with five types of errors: substitution of known/unknown word, addition of known/unknown word, and deletion of a word, and four lengths (3, 5, 7, and 11) of each error type. Errors occur at all positions in the ill-formed sentences. In the case of errors involving only known words, the inserted or substituted words are randomly chosen. The basic sample sentences are "I have dogs," "The boy takes a book," "He wants to see the beautiful girl," and "He believes that the big boy enjoys seeing the small baby." Results* of an ACFG-based version of CHAPTER are shown in Table 1 and 2.

The ACFG-based system has been compared with a basically similar CFG-based system. The version based on CFG has only 30 context-free rules. Whereas the version based on ACFG uses 57 rules, 32 types of verb subcategorization, 43 lexical category features, and 69 syntactic selectional restriction rules. Both systems employ a need-chart network for fast recovery of final S node.

* We used a Macintosh IIci with 9MB RAM and Mac Common Lisp 2.0 for the experiments.
The richer grammar detected ill-formedness of a sentence better than the poor grammar. The version employing CFG could not detect the ill-formedness of 19 sentences of 138 testing sentences. However, the ACFG-based system could detect 5 more sentences as ill-formed. In the cases of suggested recovery, the CFG-based system suggested 730 repaired sentences (6.2 repairs per ill-formed sentence) and the ACFG-based system suggested 598 repaired sentences (4.7 repairs per ill-formed sentence). The ACFG-based system suggested more accurate repaired sentences than CFG-based system (Table 1 & 2).

<table>
<thead>
<tr>
<th>Error type</th>
<th>Grammar type</th>
<th># of tests</th>
<th># of Subst.</th>
<th># of Del.</th>
<th># of Add.</th>
<th>Total Sols.</th>
<th>Avg. # repairs</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUW</td>
<td>CFG</td>
<td>26/26</td>
<td>65</td>
<td>16</td>
<td>81</td>
<td>3.1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ACFG</td>
<td>26/26</td>
<td>58</td>
<td>12</td>
<td>70</td>
<td>2.7</td>
<td></td>
</tr>
<tr>
<td>SKW</td>
<td>CFG</td>
<td>22/26</td>
<td>90</td>
<td>31</td>
<td>60</td>
<td>181</td>
<td>8.2</td>
</tr>
<tr>
<td></td>
<td>ACFG</td>
<td>23/26</td>
<td>65</td>
<td>11</td>
<td>20</td>
<td>96</td>
<td>4.2</td>
</tr>
<tr>
<td>AUW</td>
<td>CFG</td>
<td>30/30</td>
<td>51</td>
<td>40</td>
<td>91</td>
<td>3.0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ACFG</td>
<td>30/30</td>
<td>64</td>
<td>38</td>
<td>99</td>
<td>3.3</td>
<td></td>
</tr>
<tr>
<td>AKW</td>
<td>CFG</td>
<td>24/30</td>
<td>121</td>
<td>51</td>
<td>85</td>
<td>257</td>
<td>10.7</td>
</tr>
<tr>
<td></td>
<td>ACFG</td>
<td>24/30</td>
<td>87</td>
<td>48</td>
<td>33</td>
<td>188</td>
<td>7.8</td>
</tr>
<tr>
<td>DEL</td>
<td>CFG</td>
<td>19/26</td>
<td>54</td>
<td>23</td>
<td>49</td>
<td>126</td>
<td>7.4</td>
</tr>
<tr>
<td></td>
<td>ACFG</td>
<td>19/26</td>
<td>46</td>
<td>8</td>
<td>61</td>
<td>115</td>
<td>6.0</td>
</tr>
</tbody>
</table>

# of tests = 23/26 ("would mean we tested 26 sentences. Our system could not detect the ill-formedness of 3 of the 26 sentences.

# of Subst. = Suggested number of S repairs corrected by substitution of a syntactic category.

# of Del. = Suggested number of S repairs corrected by deletion of a syntactic category.

# of Add = Suggested number of S repairs corrected by addition of a syntactic category.

AVG = Average repairs per ill-formed sentence.

In terms of error types, when an extra known word is inserted, only 29% of repaired trees suggested the deletion of a word in an ill-formed sentence. CHAPTER suggested the best repairs when an unknown word is substituted for a known word. In that case, 83% of sentences were repaired by substitution of a word of the correct lexical category. The order of accuracy across ill-formed sentences in CHAPTER is: substitution of an unknown word > substitution of a known word > deletion of a word > addition of an unknown word > addition of a known word (c.f. Table 2).

<table>
<thead>
<tr>
<th>Error type</th>
<th>Grammar type</th>
<th>% of Subst.</th>
<th>% of Del.</th>
<th>% of Add.</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUW</td>
<td>CFG</td>
<td>80</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ACFG</td>
<td>83</td>
<td>17</td>
<td></td>
</tr>
<tr>
<td>SKW</td>
<td>CFG</td>
<td>50</td>
<td>17</td>
<td>33</td>
</tr>
<tr>
<td></td>
<td>ACFG</td>
<td>68</td>
<td>11</td>
<td>21</td>
</tr>
<tr>
<td>AUW</td>
<td>CFG</td>
<td>56</td>
<td>44</td>
<td>35</td>
</tr>
<tr>
<td></td>
<td>ACFG</td>
<td>65</td>
<td>35</td>
<td></td>
</tr>
<tr>
<td>AKW</td>
<td>CFG</td>
<td>47</td>
<td>20</td>
<td>33</td>
</tr>
<tr>
<td></td>
<td>ACFG</td>
<td>46</td>
<td>26</td>
<td>22</td>
</tr>
<tr>
<td>DEL</td>
<td>CFG</td>
<td>43</td>
<td>18</td>
<td>36</td>
</tr>
<tr>
<td></td>
<td>ACFG</td>
<td>40</td>
<td>7</td>
<td>53</td>
</tr>
</tbody>
</table>

CHAPTER employed left-to-right & right-to-left parsing strategy for the bottom-up satisfaction procedure. Therefore, leftmost or rightmost errors were repaired faster and more correctly than the errors in the middle of an ill-formed sentence. When the CHAPTER applied to the
data collected from electronic mails, ill-formed sentences were repaired
well, provided they were in the domain of CHAPTER's grammar and did
not include multiple errors.

4 Conclusions

4.1 Problems

If a semantically ill-formed sentence satisfies CHAPTER's syntactic
selectional restrictions (e.g. she becomes a car), it is parsed as a
well-formed sentence. In the case of IFSCP, the recovery is confined to
the syntactic structure. As a result, recovered versions of ill-formed
sentences may be meaningless. These problems can be solved: first,
CHAPTER's grammatical coverage needs extension of its syntactic
features and syntactic selectional restrictions. Second, CHAPTER is being
enhanced to handle surface and deep case using semantic selectional
restrictions.

If more information (e.g. case information or semantic information) is
applied to recovery of ill-formed sentences, the number of repaired
sentences will be less than in CHAPTER which has 4.7 repairs in the
ACFG version, on average.

4.2 Conclusion

CHAPTER makes some progress in recovering from ill-formedness
using syntactic selectional restrictions. To reduce the combinations of
categorial features, we employ head-driven parsing algorithm for
inheritance and instantiation of features. However, if category features
were fully implemented, more repairs of ill-formed sentences would be
possible.

The system based on a context-free grammar spent less time and
space than that based on ACFG in parsing well-formed and ill-formed
sentences. However, the ACFG-based system can suggest more accurate
corrections for ill-formed sentences.

Versions based on a need-chart network spent less time and space
than those without the need-chart network because the network
supported the fast searching for relevant need-arcs while reconstructing
the corrected sentence by retracing the need-chart network.

The need-chart network data structure allows the reduction of
searching time in processing higher constituent recovery after bottommost
error recovery. This idea can be extended to top-down chart parsing for
well-formed sentences or multiple errors recovery. In the case of multiple
errors, multiple goals for different local trees could be considered. For
example, the string might include multiple syntactic errors "a big boy
loves a big have." In this sentence, multiple goals are made as follows:
(1) searching for distinct words whose syntactic categories are noun,
pron, verb, etc. (e.g. "boy," "loves," and "have") (Stock, Falcone, &
Insinnamo, 1988), (2) generating multiple goals with the step (1)
information (eg. goal (NP "a big boy") & (VP "loves a big have"). With
these multiple goals, the system can recover from multiple errors
simultaneously and the repaired errors can reconstruct a final goal
(sentence) by retracing need-chart network.

Acknowledgments

We would like to express our gratitude to Professor Lee, Min's
supervisor and adviser during his linguistic studies at Chonnam National
University, and to Professor Shin, director of Language Research Centre,
for this opportunity to introduce our recent work on computational
linguistics to the readers of Language Education.
References


<Abstract>

This paper describes the syntactic repair of ill-formed sentences within an augmented context-free grammar using head-driven parsing. The system described here provides recovery of local trees and reconstruction of the sentence. It also produces a report on the details of the repair that has been carried out. The system incorporates generalised problem solving strategies for detecting and correcting several types of syntactic errors without any heuristic assumptions.

The paper focuses on a system for syntactic recovery of ill-formed sentences at the local tree and the final goal (sentence) level. The system is based on a chart parser and employs a mixed top-down/bottom-up strategy together with left-to-right and right-to-left parsing. The implementation is composed of two chart parsers: a well-formed sentence chart parser (WFSCP) and an ill-formed sentence chart parser (IFSCP).

A network called need-chart network underlies the IFSCP. The need-chart network is the collection of all need-arcs which are similar to active arcs. The need-arcs are linked together to give the history of goal expansion and fast reconstruction of the goal node from bottommost recovery. The subcategorization of verbs is based on Oxford Advanced Learner's Dictionary which uses 32 subcategorizations.

Our system grammar is based on a context-free grammar augmented with syntactic categorial feature constraints as in GPSG. The augmentations enforce syntactic selectional restrictions to reduce a potential explosion of categorial features when applying a top-down parsing strategy. The aim of our system, which is called CHAPTER (CHart Parser for Two-stage Error Recovery), is to maximise syntactic constraint without a potentially huge feature system.
Kyongho Min
School of Computer Science & Engineering
The University of New South Wales
SYDNEY NSW 2052 Australia
min@cse.unsw.edu.au

William H. Wilson
School of Computer Science & Engineering
The University of New South Wales
SYDNEY NSW 2052 Australia
billw@cse.unsw.edu.au

(Received: Sept. 10, 1994)