

Integrated Control of Chart Items for Error Repair

Kyongho MIN and William H. WILSON
School of Computer Science & Engineering
University of New South Wales
Sydney NSW 2052 Australia
{min,billw}@cse.unsw.edu.au

Abstract

This paper describes a system that performs hierarchical error repair for ill-formed sentences, with heterarchical control of chart items produced at the lexical, syntactic, and semantic levels. The system uses an augmented context-free grammar and employs a bidirectional chart parsing algorithm. The system is composed of four subsystems: for lexical, syntactic, surface case, and semantic processing. The subsystems are controlled by an integrated-agenda system. The system employs a parser for well-formed sentences and a second parser for repairing single error sentences. The system ranks possible repairs by penalty scores which are based on both grammar-dependent factors (e.g. the significance of the repaired constituent in a local tree) and grammar-independent factors (e.g. error types). This paper focuses on the heterarchical processing of integrated-agenda items (i.e. chart items) at three levels, in the context of single error recovery.

Introduction

Weischedel and Sondheimer (1983) described two types of ill-formedness: relative (i.e. limitations of the computer system) and absolute (e.g. misspellings, mistyping, agreement violation etc). These two types of problem cause ill-formedness of a sentence at various levels, including typographical, orthographical, morphological, phonological, syntactic, semantic, and pragmatic levels.

Typographical spelling errors have been studied by many people (Damerau, 1964; Peterson, 1980; Pollock and Zamora, 1983). Mitton (1987) found a large proportion of real-word errors were orthographical: *to* → *too*, *were* → *where*. At the sentential level, types of syntactic errors such as co-occurrence violations, ellipsis, conjunction errors, and

extraneous terms have been studied (Young, Eastman, and Oakman, 1991). In addition, Min (1996) found 0.6% of words misspelt (447/68966) in 300 email messages, leading to about 12.0% of the 3728 sentences having errors.

Various systems have focused on the recovery of ill-formed text at the morpho-syntactic level (Vosse, 1992), the syntactic level (Irons, 1963; Lyon, 1974), and the semantic level (Fass and Wilks, 1983; Carbonell and Hayes, 1983). Those systems identified and repaired errors in various ways, including using grammar-specific rules (meta-rules) (Weischedel and Sondheimer, 1983), least-cost error recovery based on chart parsing (Lyon, 1974; Anderson and Backhouse, 1981), semantic preferences (Fass and Wilks, 1983), and heuristic approaches based on a shift-reduce parser (Vosse, 1992).

Systems that focus on a particular level miss errors that can only be detected using higher level knowledge. For example, at the lexical level, in *I saw a man if the park*, the misspelt word *if* is undetected. At the syntactic level, in *I saw a man in the pork*, the misspelling of *pork* can only be detected using semantic information.

This paper describes the automatic correction of ill-formed sentences by using integrated information from three levels (lexical, syntactic, and semantic). The CHAPTER system (CHART Parser for Two-stage Error Recovery), performs two-stage error recovery using generalised top-down chart parsing for the syntax phase (cf. Mellish, 1989; Kato, 1994). It uses an augmented context-free grammar, which covers verb subcategorisations, passives, yes/no and WH-questions, finite relative clauses, and EQUI/SOR phenomena.

The semantic processing uses a conceptual hierarchy and act templates (Fass and Wilks, 1983), that express semantic restrictions. Surface case processing is used to help extract meaning (Grishman and Peng, 1988) by mapping surface cases to their corresponding conceptual cases. Unlike other systems that

have focused on error recovery at a particular level (Damerau, 1964; Mellish, 1989; Fass and Wilks, 1983), CHAPTER uses an integrated agenda system, which integrates lexical, syntactic, surface case, and semantic processing. CHAPTER uses syntactic and semantic information to correct spelling errors detected, including real-word errors.

Section 1 treats methodology. Section 2 gives test results for CHAPTER. Section 3 describes problems with CHAPTER and section 4 contains conclusions.

1 Methodology

The system uses a hierarchical approach and an integrated-agenda system, for efficiency in an environment where most sentences do not have errors. The first stage parses an input sentence using a bottom-up left-to-right chart parsing algorithm incorporating surface case and semantic processing. If no parse is found, the second stage tries to repair a single error: either at the lexical or syntactic level (§1.1) or at the semantic level (§1.2). The second parser uses generalised top-down strategies (Mellish, 1989) and a restricted bidirectional algorithm (Satta and Stock, 1994) for error detection and correction.

Errors at the syntactic level are assumed to arise from replacement of a word by a known or unknown word, addition of a known or unknown word, or deletion of a word. Real-word replacement errors may occur because of simple misspellings, or agreement violations. A semantic error is signalled if a filler concept violates the semantic constraints of the concept frame for a sentence.

1.1 Syntactic Recovery

CHAPTER's syntactic error recovery system employs generalised top-down and bidirectional bottom-up chart parsing (cf. Mellish, 1989) using an augmented context-free grammar. The system is composed of two phases: error detection and error correction (see section 4 in Min, 1996). A single syntactic error is detected by the following two processes:

- (1) *top-down expectation*: expands a goal using an augmented context-free grammar. (A goal is a partial tree, which may contain one or more syntactic categories, specifically a subtree of a syntax tree corresponding to a single context-free rule, and which might contain syntactic errors. For example, the first goal for the ill-formed sentence

I have a bif book is $\langle S \text{ needs from } 0 \text{ to } 5 \text{ with penalty score } 4 \rangle$.)

- (2) *bottom-up satisfaction*: searches for an error using a goal and inactive arcs made by the first-stage parser, and produces a *need-chart network*;

The error detected by this process is corrected by the following two processes:

- (3) a *constituent reconstruction engine*: repairs the error and reconstructs local trees by retracing the need-chart network; and
- (4) *spelling correction* corrects spelling errors (see Min and Wilson, 1995).

Because of space limitations, this paper focuses on (3) and (4).

Consider the sentence *I saw a man if the park*. The top-down expectation phase would produce the initial goal for the sentence, $\langle \text{goal } S \text{ is needed from } 0 \text{ to } 7 \rangle$, and expand it using grammar rules, $\langle (S \rightarrow NP VP) \text{ is needed from } 0 \text{ to } 7 \rangle$. Next, a bottom-up satisfaction phase uses the inactive arcs left behind by the first-stage parser to refine and localise the error by looking for the leftmost or rightmost constituent of the expanded goal in a bidirectional mode.

For example, given an inactive arc, $\langle NP("I") \text{ from } 0 \text{ to } 1 \rangle$, the left-to-right process is applied: for the expanded goal S , $NP("I")$ is found from 0 to 1 and VP is needed from 1 to 7: or, more briefly, $\langle S \rightarrow NP("I") \bullet VP \text{ is needed from } 1 \text{ to } 7 \rangle$. This data structure is called a *need-arc*. A need-arc is similar to an active arc, and it includes the following information: which constituents are already found and which constituents are needed for the recovery of a local tree between two positions, together with the arc's penalty score. From this need-arc, another goal, $\langle \text{goal } VP \text{ is needed from } 1 \text{ to } 7 \rangle$, is produced.

After detecting an error using the top-down expectation and bottom-up satisfaction phases, the detected error is corrected using two types of chart item: a goal and a need-arc, and the types of the goal's or need-arc's constituent and its *penalty score*. The penalty score ($PS(G)$) of a goal (or need-arc) G , whose syntactic category is L and whose two positions are $FROM$ and TO , is computed as follows:

$$PS(G) = RW(G) - MEL(L)$$

where $RW(G)$ is the number of remaining words to be processed, (ie. $TO - FROM$), and $MEL(L)$ is the minimal extension length of the category L .

MEL (Minimal Extension Length) is the minimum number of preterminals necessary to produce the rule's LHS category. For example, the MEL of S is 2, because of examples like "I go".

Using the penalty scores, three error correction conditions are as follows:

- The *substitution* correction condition is: the goal's label is a single lexical category, and its penalty score is 0 (there is a replaced word)
- The *addition* correction condition is: the goal's label is a single lexical category, and its penalty score is -1 (there is an omitted word).
- The *deletion* correction condition is: there is no constituent needed for repair, and the penalty score of the need-arc is 1 (there is an extra word).

The repaired constituent produced with these conditions is used to repair constituents all the way up to the original S goal via the need-chart network. This process is performed by the *constituent reconstruction engine*.

At the syntactic level, the choice of the best correction relies on two penalty schemes: error-type penalties and penalties based on the weight (or importance) of the repaired constituent in its local tree. The error-type penalties are 0.5 for substitution errors, and 1 for deletion or addition errors¹. The weight penalty of a repaired constituent in a local tree is either 0.1 for a head daughter, 0.5 for a non-head daughter, or 0.3 for a recursive head-daughter (e.g. NP in the right-hand side of the rule NP → NP PP). The weight penalty is accumulated while retracing the need-chart network. In effect, the system seeks a best repair with minimal length path from node S to the error location in the syntax tree.

Often more than one repair is suggested. The repaired syntactic structures are subject to surface case and semantic processing during syntactic reconstruction. If the syntactic repair does not violate selectional restrictions, it is acceptable.

1.2 Semantic Recovery

CHAPTER maps syntactic parses into surface case frames. These are interpreted by a mapping procedure and a pattern matching algorithm. The mapping procedure uses semantic selectional restrictions based on act templates and a concept hierarchy and converts the surface case slots into concept

slots, while the pattern matching algorithm constrains filler concepts using ACT templates which represents semantic selectional restrictions. Selectional restrictions are represented by a expressions like ANIMATE, or (NOT HUMAN). The latter represents any concept that is not a sub-concept of HUMAN. Surface cases are mapped to concept slots: subject → agent, verb → act, direct object → theme. Consider the sentence "I parked a car". The mapping of SENT1 into PARK1 is as follows:

```
SENT1: (subj (value "I"))
        (verb (value "parked"))
        (dobj (value "a car"))
⇒ PARK1: (agent (SPEAKER "I"))
          (act (PARK "parked"))
          (theme (CAR "a car"))
```

Semantic errors may be of two types:

- (1) there may be no full parse tree, so semantic interpretation is impossible;
- (2) the sentence may be syntactically acceptable, but semantically ill-formed (e.g. *I parked a bud* (bus)).

The first type of error is repaired from the spelling level up to semantic level (if a spelling error is detected). For errors of a semantic nature, semantic selectional restrictions may be forced onto the error concept to make it fit the template. For example, the sentence "*I parked a bud*" violates the semantic selectional restriction on the theme slot of *park*. The template of the verb *park* is (HUMAN PARK VEHICLE). However, the concept BUD, associated with 'bud', is not consistent with the restriction, VEHICLE, on the theme slot. As a result, the sentence is semantically ill-formed, with a semantic penalty of -1 (one slot violates a restriction). To correct the error, the filler concept BUD is forced to satisfy the template concept VEHICLE by invoking the spelling corrector with the word 'bud' and the concept VEHICLE. Thus the real word error *bud* would be corrected to *bus*.

The filler concept may itself be internally inconsistent. Consider the sentence *I saw a pregnant man*. The theme slot of SEE satisfies its restriction. However, the filler concept of the theme slot is inconsistent. In CHAPTER, the attribute concept *pregnant* is identified as the error rather than the head concept *man*. To correct it, the attribute concept is relaxed to any attribute concept that can qualify the MAN concept. It would also be possible to force *man* to fit to the attribute concept (e.g. by changing it to *woman*). There seems to be no general method to pick the correct component to modify with this type of error:

¹These penalties are somewhat arbitrary. Corpus-based probability estimates would be preferable.

we chose to relax the attribute concept. This problem might be resolved by pragmatic processing.

1.3 Integrated-Agenda Manager

CHAPTER is composed of four subsystems for parsing well-formed sentences and repairing ill-formed sentences: lexical, syntactic, surface case, and semantic processing. Each subsystem uses relevant chart items from other subsystems as its input and is invoked in a heterarchical mode by an agenda scheme, which is called the integrated-agenda manager. The manager controls and integrates all levels of information to parse well-formed sentences and repair ill-formed sentences (Min, 1996). Thus the integrated-agenda manager distributes agenda items to relevant subsystems (see Figure 1).

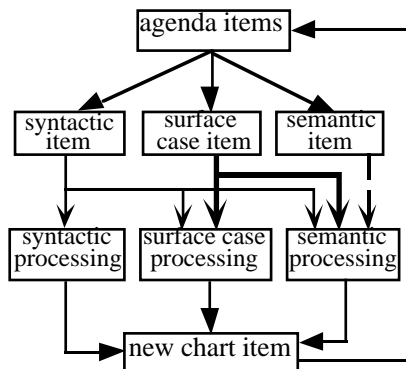


Figure 1. Integrated agenda manager

For example, if an agenda item is a repaired syntactic item, then it is distributed to syntactic processing for recovery, then to surface case and semantic processing. The invocation of the relevant subsystem depends on the characteristics of the chart item. Consider an agenda item which is a syntactic NP node. Syntactic and subsequently semantic processing are invoked. Surface case processing is not appropriate for an NP node. If an agenda item is a syntactic VP node, then syntactic, surface case, and semantic processing are all invoked. After subsystem processing of the item, the new chart item becomes an agenda item in turn. This continues until the integrated agenda is empty. The data structures of CHAPTER are based on a network-like structure that allows access to all levels of information (syntactic, surface case, and semantics). Some of the data are stored using associative structures (e.g. grammar rules, active arcs, and inactive arcs) that allow

direct access to structures most likely to be needed during processing.

2 Experimental Results

The test data included syntactic errors introduced by substitution of an unknown or known word, addition of an unknown or known word, deletion of a word, segmentation and punctuation problems, and semantic errors. Data sets we used are identified as: NED (a mix of errors from Novels, Electronic mail, and an (electronic) Diary); Appling1, and Peters2 (the Birkbeck data from Oxford Text Archive (Mitton, 1987)); and Thesprev. Thesprev was a scanned version of an anonymous humorous article titled "Thesis Prevention: Advice to PhD Supervisors: The Siblings of Perpetual Prototyping".

In all, 258 ill-formed sentences were tested: 153 from the NED data, 13 from Thesprev, 74 from Appling1, and 18 from Peters2. The syntactic grammar covered 166 (64.3%) of the manually corrected versions of the 258 sentences. The average parsing time was 3.2 seconds. Syntactic processing produced on average 1.7 parse trees², of which 0.4 syntactic parse trees were filtered out by semantic processing. Semantic processing produced 9.3 concepts on average per S node, and 7.3 of them on average were ill-formed. So many were produced because CHAPTER generated a semantic concept whether it was semantically ill-formed or not, to assist with the repair of ill-formed sentences (Fass and Wilks, 1983).

Across the 4 data sets, about one-third of the (manually-corrected) sentences were outside the coverage of the grammar and lexicon. The most common reasons were that the sentences included a conjunction ("He places them face down *so that* they are a surprise"), a phrasal verb ("I *called out* to Fred and went inside"), or a compound noun ("*PC development tools* are far ahead of *Unix development tools*"). The remaining 182 sentences were used for testing: NED (98/153); Thesprev (12/13); Appling1 (55/74); and Peters2 (17/18). Compound and compound-complex sentences in NED were split into simple sentences to collect 13 more ill-formed sentences for testing.

Table 1 shows that 89.9% of these ill-formed sentences were repaired. Among these, CHAPTER ranked the correct repair first or

²There are so few parse trees because of the use of subcategorisation and the augmented context-free grammar (the number of parse trees ranges from 1 to 7).

second in 79.3% of cases (see 'best repair' column in Table 1). The ranking was based on penalty schemes at three levels: lexical, syntactic, and semantic. If the correct repair was ranked lower than second among the repairs suggested, then it is counted under 'other repairs' in Table 1. In the case of the NED data, the 'other repairs' include 11 cases of incorrect repairs introduced by: segmentation errors, apostrophe errors, semantic errors, and phrasal verbs. Thus for about 71% of all ill-formed sentences tested, the correct repair ranked first or second among the repairs suggested. For 19% of the sentences tested, incorrect repairs were ranked as the best repairs. A sentence was considered to be "correctly repaired" if any of the suggested corrections was the same as the one obtained by manual correction.

Table 2 shows further statistics on CHAPTER's performance. CHAPTER took 18.8 seconds on average³ to repair an ill-formed sentence; suggested an average of 6.4 repaired parse trees; an average of 3 repairs were filtered out by semantic processing. During semantic processing, an average of 40.3 semantic concepts were suggested for each S node. An average 34.3 concepts per S node were classified as ill-formed. Twenty seven percent of the 'best' parse trees suggested by CHAPTER's ranking strategy at the syntactic level were filtered out by semantic processing. The remaining 73% of the 'best' parse trees were judged semantically well-formed.

In the case of the NED data set, 90 ill-formed sentences were repaired. On average: recovery time per sentence was 23.9 seconds; 9.8 repaired S trees per sentence were produced; 4.5 of the 9.8 repaired S trees were semantically well-formed; 95.1 repaired concepts (ill-formed and well-formed) were produced; 8.5 of 95.1 repaired concepts were well-formed; and semantic processing filtered syntactically best repairs, removing 22% of repaired sentences. The number of repaired concepts for S is very large because semantic processing at present supports interpretation of only a single verbal (or verb phrasal) adjuncts. For example, the template of the verb GO allows either a temporal or destination adjunct at present and ignores any second or later adjunct. Thus a GO sentence would be interpreted using both [THING GO DEST] and [THING GO TIME].

3 Discussion

3.1 Syntactic Level Problems

The grammar rules need extension to cover the following grammatical phenomena: compound nouns and adjectives, gerunds, TO+VP, conjunctions, comparatives, phrasal verbs and idiomatic sentences. For example, 'in the morning' and 'at midnight' are well-formed phrases. However, CHAPTER currently also parses 'in morning', 'in the midnight', and 'at morning' as well-formed.

CHAPTER uses prioritised search to detect and correct syntactic errors using the penalty scores of goals. However, the scheme for selecting the best repair did not uncritically use the first detected error found by the prioritised search at the syntactic level, because the best repair might be ill-formed at the semantic level. In fact, the prioritised search strategy did not contribute to the selection scheme, which depended solely on the error type and the importance of the repaired constituent in its local tree.

3.2 Semantic Level Problems

At present in CHAPTER's semantic system, the most complex problem is the processing of prepositions, and their conceptual definition. For example, the preposition 'for' can indicate at least three major concepts: time duration (*for a week*), beneficiary (*for his mother*), and purpose (*for digging holes*). If *for* takes a gerund object, then the concept will specify a purpose or reason (e.g. *It is a machine for slicing bread*).

In addition, the act templates do not allow multiple optional conceptual cases (i.e. relational conceptual cases - LOC for locational concepts, and DEST for destination concepts, etc.) for prepositional and adverbial phrases. This would increase the number of templates and the computational cost. If there is more than one verbal adjunct (PPs and ADVPs) in a sentence, then CHAPTER does not interpret all adjuncts.

³Running under Macintosh Common Lisp v 2.0 on a Macintosh II fx with 10 MB for Lisp

Data Set	Sentences tested	Number of repairs	Best repairs	Other repairs	No repairs suggested
NED (%)	98	90 (91.8)	64/90 (71.1)	26/90 (28.9)	8 (8.2)
Appling1 (%)	55	52 (94.5)	40/52 (76.9)	12/52 (23.1)	3 (5.5)
Peters2 (%)	17	17 (100)	14/17 (82.4)	3/17 (17.6)	0
Thesprev (%)	12	10 (83.3)	9/10 (90.0)	1/10 (10.0)	2 (16.7)
Average (%)	-	* 89.9%	79.3%	20.7%	10.1%

Table 1. Performance of CHAPTER on ill-formed sentences

*Peters2 data are not considered in the averages because Peters2 consists of only the sentences that were covered by CHAPTER's grammar, selected from more than 300 sentence fragments (simple sentences and phrases.)

Data set	Sentences repaired	Time (sec)	Repaired S trees	Semantically well-formed parse trees	Repaired concepts for S	Repaired well-formed concepts for S	% of syntactically-best parses filtered
NED	90	23.9	9.8	4.5	95.1	8.5	26/90 (22%)
Appling1	52	15.8	4.7	3.3	17.7	5.2	11/52 (20%)
Peters2	17	15.8	6.3	2.5	29.8	5.0	7/17 (41%)
Thesprev	10	19.4	4.9	3.4	18.7	5.3	2/10 (20%)
Average	-	18.8	6.4	3.4	40.3	6.0	46/169 (27%)

Table 2. Results on CHAPTER's performance (average values per sentence)

Conclusion

This paper has presented a hierarchical error recovery system, CHAPTER, based on a chart parsing algorithm using an augmented context-free grammar. CHAPTER uses an integrated-agenda manager that invokes subsystems incrementally at four levels: lexical, syntactic, surface case, and semantic. A sentence has been confirmed as well-formed or repaired when it has been processed at all levels.

Semantic processing performs pattern matching using a concept hierarchy and verb templates (which specify semantic selectional restrictions). In addition, procedural semantic constraints have been used to improve the efficiency of semantic processing based on a concept hierarchy. However, it increases computational cost.

CHAPTER repaired 89.9% of the ill-formed sentences on which it was tested, and in 79.3% of cases suggested the correct repair (as judged by a human) as the best of its alternatives. CHAPTER's semantic processing rejected 27% of the repairs judged "best" by the syntactic system.

References

- Anderson, S. and Backhouse, R. (1981). Locally Least-cost Error Recovery in Earley's Algorithm. *ACM Transactions on Programming Languages and Systems*, **3**(3) 318-347.
- Carbonell, J. and Hayes, P. (1983). Recovery Strategies for Parsing Extragrammatical Language. *American Journal of Computational Linguistics*, **9**(3-4) 123-146.
- Damerau, F. (1964). A Technique for Computer Detection and Correction of Spelling Errors. *Communications of the ACM*, **7**(3) 171-176.
- Fass, D. and Wilks, Y. (1983). Preference Semantics, Ill-formedness, and Metaphor. *American Journal of Computational Linguistics*, **9**(3-4) 178-187.
- Grishman, R. and Peng, P. (1988). Responding to Semantically Ill-Formed Input. *The 2nd Conference of Applied Natural Language Processing*, 65-70.
- Irons, E. (1963). An Error-Correcting Parse Algorithm. *Communications of the ACM*, **6**(11) 669-673.
- Kato, T. (1994). Yet Another Chart-Based Technique for Parsing Ill-formed Input. *The Fourth*

Conference on Applied Natural Language Processing, 107-112.

- Lyon, G. (1974). Syntax-Directed Least-Errors Analysis for Context-Free Languages: A Practical Approach. *Communications of the ACM*, **17**(1) 3-14.
- Mellish, C. (1989). Some Chart-Based Techniques for Parsing Ill-Formed Input. *ACL Proceedings, 27th Annual Meeting*, 102-109.
- Min. (1996). *Hierarchical error recovery based on bidirectional chart parsing techniques*. PhD dissertation, University of UNSW, Sydney, Australia.
- Min, K. and Wilson, W. H. (1995). Are Efficient Natural Language Parsers Robust?. *Eighth Australian Joint Conference on Artificial Intelligence*; 283-290
- Mitton, R. (1987). Spelling Checkers, Spelling Correctors and the Misspellings of Poor Spellers. *Information Processing and Management*, **23**(5) 495-505.
- Peterson, J. (1980). Computer Programs for Detecting and Correcting Spelling Errors. *Communications of the ACM*, **23**(12) 676-687.
- Pollock and Zamora (1983). Collection and characterisation of spelling errors in scientific and scholarly text. *Journal of the American Society for Information Science*. 34(1) 51-58.
- Satta and Stock (1994). Bidirectional context-free grammar parsing for natural language processing. *Artificial Intelligence* **69** 123-164.
- Vosse, T. (1992). Detecting and Correcting Morpho-Syntactic Errors in Real Texts. *The Third Conference on Applied Natural Language Processing*, 111-118.
- Weischedel, R. and Sondheimer, N. (1983). Meta-rules as Basis for Processing Ill-formed Input. *American Journal of Computational Linguistics*, **9**(3-4) 161-177.
- Young, C., Eastman, C., and Oakman, R. (1991). An Analysis of Ill-formed Input in Natural Language Queries to Document Retrieval Systems. *Information Processing and Management*, **27**(6) 615-622.