

Integrated Correction of Ill-Formed Sentences

Kyongho Min and William H. Wilson
School of Computer Science and Engineering
The 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 recovery, and detects and corrects a single error in a sentence at the lexical, syntactic, and/or semantic levels. If the system is unable to repair an erroneous sentence on the assumption that it has a single error, a multiple error recovery system is invoked. The system employs a chart parsing algorithm and uses an augmented context-free grammar, and has subsystems for lexical, syntactic, surface case, and semantic processing, which are controlled by an integrated-agenda system. In the frequent case that there is a choice of possible repairs, the possible repairs are ranked by *penalty scores*. The penalty scores are based on grammar-dependent and grammar-independent heuristics. The grammar-independent ones involve error types, and, at the lexical level, character distance; the grammar-dependent ones involve, at the syntactic level, the significance of the repaired constituent in a local tree, and, at the semantic level, the distance between the semantic form containing the error, and normal act templates. This paper focuses on single error recovery.

1 Introduction

The ill-formedness may be at various levels, including the morphological (also typographical, orthographical, and phonological), syntactic, semantic, and pragmatic levels. Pollock and Zamora (1983) collected spelling errors from approximately 25 million words from seven scientific and scholarly databases, finding an error rate of 2%. These would have been mostly typographical errors. Mitton (1987) found that a large proportion of orthographical errors are real-word errors: *to* → *too*, *were* → *where*.

At the sentential level, Young, Eastman, and Oakman (1991) found that 27% of 426 queries typed into a library search interface had errors: conjunction errors (18%), punctuation errors (7%), and spelling errors (2%). In addition, spelling errors may raise a problem at the sentential level. In 300 email messages analysed by the authors, 0.6% of words were mis-spelt (447/68966), leading to about 12.1% of sentences having errors (451/3728).

Most systems focus on the correction of errors at a *particular level*. In many cases, however, it is impossible to detect the errors at the particular level because the detection and correction of the errors requires higher level knowledge. For example, at the lexical level (Damerau, 1964; Peterson, 1980), the misspelt word *it* is not detected in "I saw a man *it* the park". At the syntactic level, the misspelling of *pork* can not be detected or corrected using lexical and syntactic information in "I saw a man in the *pork*."

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), the semantic level (Fass and Wilks, 1983; Carbonell and Hayes, 1983), and the pragmatic level (Granger, 1983). Those systems described how to identify a localised error and how to repair it 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), conceptual dependency (Granger, 1983), and heuristic approaches based on a shift-reduce parser (Vosse, 1992).

This paper focuses on the automatic correction of ill-formed sentences by a method that integrates information from three levels (lexical, syntactic, and semantic). The system, called CHAPTER (CHArt Parser for Two-stage Error Recovery), performs two-stage error recovery, and employs generalised top-down chart parsing for the syntax phase (cf. Mellish, 1989; Kato, 1994). The system uses an augmented context-free grammar, which covers verb subcategorisations based on the Oxford Advanced Learner's Dictionary classification, passives, yes/no questions (direct), WH-questions, finite relative clauses, and EQUI/SOR phenomena but not conjunctions, comparatives, compound nouns, or topicalisations.

For the semantic phase, it uses a conceptual hierarchy and act templates (cf. Wilks, 1983), with some restrictions represented by a kind of boolean expression (e.g. (NOT HUMAN)). The surface case phase instantiates surface cases, which are used to help in extraction of meaning (cf. Grishman and Peng, 1988). CHAPTER uses an integrated agenda system, which integrates and controls its four levels of language processing: lexical, syntactic, surface case, and semantic.

In contrast to other systems (Mellish, 1989; Kato, 1994), CHAPTER uses syntactic *and* semantic information to correct spelling errors detected, including real-word errors. At the syntactic level, the detection and correction of errors are separated. In addition, suggested syntactic repairs are verified by the semantic processing.

In section 2, the method employed in CHAPTER will be described. Section 3 will describe the results of tests on the system using real world data. Section 4 will describe problems and issues related to CHAPTER and section 5 contains conclusions.

2 Methodology

The system uses separate error detection and correction phases and includes recovery subsystems at three levels: spelling, syntactic, and semantic. The subsystems work together to get the best repair for an ill-formed sentence, up to the semantic level. In case there are alternative repairs, heuristics are employed to select the best repair. In this paper, we will focus on description of the two subsystems (syntactic and semantic level) because of limitation of pages.

2.1 Syntactic Recovery

Syntactic errors may occur because of misspelt words, agreement violations (number, gender, case, and tense), extraneous words, and missing words. These errors are repaired using generalised top-down chart parsing (cf. Mellish, 1989) employing an augmented context-free grammar. Syntactic error recovery is based on the following four processes:

- (1) *top-down expectation*: expanding a goal¹ using syntactic rules based on grammar rules;
- (2) *bottom-up satisfaction*: searching for an error using a goal and inactive arcs made by the first parser for well-formed sentences, and producing a *need-chart network*;
- (3) a *constituent reconstruction engine*: repairing the error and reconstructing local trees by retracing the need-chart network; and
- (4) *spelling correction* (see Min and Wilson, 1995).

The first two phases are used to *detect* a single syntactic error and the others are used for *error correction*.

Consider the sentence *I saw a man at the park*. Syntactic error recovery would be as follows:

- **Top-down expectation**

The initial goal for the sentence is a constituent of category S from 0 to 7. From this first goal, the system can infer the needed constituents for the (repaired) parse of the sentence; this is called top-down expectation. Each goal is expanded using grammar rules, refining information about the error location. For example, the first goal, <goal S is needed from 0 to 7> is expanded to <S needs (NP VP) from 0 to 7> using the grammar rule $S \rightarrow NP VP$.

When expanding goals, the MEL (Minimal Extension Length) of the rule is considered, to avoid useless goal production. For example, if there is a goal <NP is needed from 0 to 1>, then NP rules with an MEL > 2 are not retrieved as only a single error is being considered. However, the rule $NP \rightarrow DET NOUN$ would be retrieved, as it would be assumed that one category is missing: a *deletion error* \rightarrow *addition correction*.

- **Bottom-up satisfaction**

After a goal is expanded, the leftmost or rightmost constituent of the expanded goal is looked for among the inactive arcs left behind by the first-stage parser, which is designed to parse a well-formed sentence. Thus this is a bidirectional technique. If inactive arcs for the leftmost constituent are found, then the default process, left-to-right, is employed. Otherwise, if inactive arcs for the rightmost constituent are found, the right-to-left process is employed. If inactive arcs for leftmost/rightmost constituents are found and both constituents are phrasal, then both processes are employed. For example, the expanded goal, <S needs (NP VP) from 0 to 7>, would be processed with two inactive arcs: {NP("I") from 0 to 1} and {VP("park") from 6 to 7}. With the first inactive arc, 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, <S \rightarrow NP("I") • VP is needed from 1 to 7>. This data structure is called a *need-arc* (see footnote 4). From this need-arc, we can produce another goal, <goal VP is needed from

¹A goal is a partial tree (which may contain one or more syntactic categories), specially 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 <S needs from 0 to 5 with its penalty score 4>.

1 to 7>, which will require new top-down expectation and bottom-up satisfaction phases. With the second inactive arc, the right-to-left process is applied: <S → VP ("park") • NP is needed from 0 to 6>.

These two need-arcs are linked to the objects to which they refer, and also passed to the goal production process. The structure created is called a need-chart network; it represents the history of the chart items (i.e. goals, need-arcs², and repaired constituents).

• **Constituent Reconstruction Engine**

After performing top-down expectation and bottom-up satisfaction phases, a localised error is corrected using two types of chart item: a goal and a need-arc. With the goal <goal: needs PREP from 4 to 5>, a substitution error is detected because the goal's constituent is a single lexical category and the number of words which covers the category is 1. Thus this error would be handled by substitution of the constituent {PREP ("in") from 4 to 5} after spelling correction. In terms of the need-chart network, the goal is linked to its parent data structure, a need-arc. Via the need-chart network links, the repaired constituent is used to repair constituents all the way up to the first S goal. The subprocess of CHAPTER that does this is called the *constituent reconstruction engine*.

At the syntactic level, the choice of the best correction depends on two penalty schemes: grammar-independent error-type penalties and grammar-dependent penalties based on the weight (or importance) of the repaired constituent in its local tree. The error-types penalties are 0.5 for substitution errors, and 1 for deletion and addition errors³. The weight penalty of a repaired constituent in a local tree is either a head daughter penalty of 0.1, a non-head daughter of 0.5, or a recursive head-daughter (e.g. NP in the right-head side of the rule NP → NP PP) penalty of 0.3. The weight penalty is accumulated while retracing the need-chart network. In effect, the system looks for a best repair with minimal length path from the S goal to the error location in the syntax tree.

Often there is more than one repair suggested. The repaired syntactic structures are interpreted by surface case and semantic processing during syntactic reconstruction. If a syntactic repair does not violate selectional restrictions, then it is acceptable.

2.2 Semantic Recovery

CHAPTER interprets the meaning of a sentence using semantic selectional restrictions based on a concept hierarchy and act templates. Some selectional restrictions are represented by a boolean expression like (NOT HUMAN), which represents any concept that is not a subconcept of HUMAN. This permits fast computation and update. The interpretation of a surface case frame is based on a mapping procedure and a pattern matching algorithm using a concept hierarchy. The mapping procedure converts

²A 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 with its penalty score.

³These penalties are somewhat arbitrary. A corpus-based probability estimate would be preferable.

the surface case slots into concept slots, while the pattern matching algorithm constrains filler concepts using ACT templates. The surface cases are mapped to concept slots: subject \rightarrow agent, verb \rightarrow act, direct object \rightarrow theme⁴. Consider the sentence "I parked a car". The mapping is as follows:

SENT1	(subj (value "I"))	\rightarrow	PARK1	(agent (SPEAKER))
	(verb (value "parked"))	\rightarrow		(act (PARK))
	(doobj (value "a car"))	\rightarrow		(theme (CAR))

Semantic errors come in two forms:

- (1) an input sentence may be ill-formed at the syntactic level so that there is no complete parse tree and semantic interpretation is not possible;
- (2) the sentence may be syntactically acceptable, but semantically ill-formed (e.g. "I parked a *bug* (bus)").

The first type of error is incrementally repaired from the spelling level (if a spelling error is detected) up to semantic level. 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 bug" violates the semantic selectional restrictions on the theme slot of *park*. The template of the verb *park* is (HUMAN PARK VEHICLE). However, the concept BUG, associated with *bug*, 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 BUG is forced to satisfy the template concept VEHICLE by invoking the spelling corrector with the word *bug* using the concept VEHICLE. Thus the real word error *bug* would be corrected to *bus*.

In another case, the filler concept itself can be inconsistent. Suppose the sentence "I saw a *pregnant man*." The theme slot of SEE satisfies restrictions. However, the filler concept of the theme slot is inconsistent. In CHAPTER, the attribute concept *pregnant* is identified as an error rather than the head concept *man*. In this case, the attribute concept is relaxed to any attribute concept that can qualify the *man* (i.e. a male person) concept. It is also possible to force the *man* concept 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: we chose to relax the attribute concept. This problem might be resolved by pragmatic processing.

2.3 The Integrated-Agenda Manager

CHAPTER has two integrated-agenda managers (figure 1): one for the first-stage system that deals with well-formed sentences and one for the error recovery system. The integrated-agenda manager controls all types of parsing items: lexical, syntactic, surface case, and semantic. Thus the agenda system aims to distribute each agenda item to relevant subsystems. For example, if an agenda item is a repaired syntactic node, then the item is distributed to syntactic recovery, to surface case interpretation (if the item can instantiate a surface case frame) and to semantic recovery.

⁴See footnote 1.

```

To Perform integrated-agenda control in constituent reconstruction engine
loop
  while there is an agenda item (i.e. a repaired constituent)
    perform get the next agenda item
    if the agenda item is a syntactic node
      then perform syntactic node control with the item
    else if the agenda item is a surface case frame
      then perform surface case frame control with the item
    else if the agenda item is a semantic concept
      then perform semantic concept control with the item
    end if
  end if
end while
end loop
to perform syntactic node control with the item
  store the item in syntactic memory.
  case1: the category of the item is S
    perform syntactic, surface case, and semantic recovery
  case2: the category can be used to instantiate a surface case frame
    perform surface case and syntactic recovery
  case3: the category is phrasal
    perform syntactic recovery, then semantic recovery
  case4: the category is lexical
    perform syntactic recovery
  end case
end perform
to perform surface case frame control with the item
  if the frame is a surface case frame and its label is one of {STMT, WHQ,
    INV, RELS}
  then perform semantic recovery
  else if the frame is a surface case frame
    then store the item in surface-case memory.
  end if
end if
end perform
to perform semantic concept control with the item
  store the item in semantic memory.
end perform
end perform

```

Fig. 1. The integrated-agenda manager algorithm

3 Experimental Results

CHAPTER was implemented and tested using Macintosh Common Lisp 2.0, with 10MB working memory, on a Macintosh IIfx. Real world data were collected, much of it from email messages: it included syntactic errors introduced by substitution of

unknown/known word, addition of unknown/known word, deletion of a word, segmentation, apostrophe problems, and semantic errors. The data sets used as a testbed for CHAPTER are referred to as: Wilson (a mix of errors found in novels, electronic mail, and on-line diary); Appling1, and Peters2 (the Birkbeck data from Oxford Text Archive (Mitton, 1987)); and Thesprev⁵.

In all, we had 258 ill-formed sentences from various sources as described above: 153 in the Wilson data, 13 in Thesprev, 74 in Appling1 data, and 18 in Peters2. We tested the coverage rate for our grammar on manually corrected versions of these 258 sentences. The syntactic grammar covered 166 (64.3%) of these: 85/153 on Wilson, 9/13 on Thesprev, 54/74 on Appling1, and 18/18 on Peters2. The average parsing time was 3.2 seconds. Syntactic processing produced on average 1.7 parse trees⁶, of which 0.4 syntactic parse trees were filtered out by semantic processing. Semantic processing produced 9.3 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 sentence.

Across the four data sets, 34.1% of the well-formed sentences were classified as ill-formed because of a lack of syntactic and semantic information in the grammar and lexicon. The most common reasons they were not parsed as well-formed were that the sentences included a conjunction (e.g. "He places them face down *so that* they are a surprise"), a phrasal verb (e.g. "I *called out* to Fred and went inside"), or a compound noun (e.g. "*PC development tools* are far ahead of *Unix development tools*"). From the 258 original sentences, we selected 182 sentences: Wilson (98/153)⁷; Thesprev (12/13); Appling1(55/74); and Peters2 (17/18).

Data Set	No. of sents tested	Number of repairs	Best repairs	Next repairs	No repairs suggested
Wilson (%)	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

Table 1 shows that 89.9% of these ill-formed sentences were repaired⁸. Among these, CHAPTER ranked the correct repair first or second in 79.3% of cases

⁵A scanned version of "Thesis Prevention: Advice to PhD Supervisors: The Siblings of Perpetual Prototyping" 18 October 1991.

⁶There 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).

⁷Compound and compound-complex sentences were split into simple sentences to collect 13 more ill-formed sentences for testing.

⁸The results for the Peters2 data are not considered here because we selected only the sentences that were covered by our grammar, from more than 300 sentence fragments which were either a simple sentence or a phrase. The Peters2 sentences tested were all repaired (100%).

(see 'best repair' column in Table 2). 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 'next repairs' in Table 1. In the case of the Wilson data, the 'next repairs' include 11 cases of incorrect repairs (12.2%), introduced by: segmentation error (7 sentences); apostrophe error (1); semantic error (2); and a phrasal verb (1). 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 tested sentences, incorrect repairs⁹ were ranked as the best repairs.

Table 2 shows further results of repairing ill-formed sentences. CHAPTER took 18.8 seconds on average to repair an ill-formed sentence, and suggested an average of 6.4 repaired parse trees and an average of 3 repairs were filtered out by semantic processing. In the case of semantic processing, an average of 40.3 semantic concepts were suggested for each S node and 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 Wilson dataset, 90 ill-formed sentences were repaired. On average: recovery time per sentence was 23.9 seconds; 9.8 repaired S trees per sentence was 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 allowed single interpretation of verbal (or verb phrasal) adjuncts. For example, the template of the verb GO allows either a temporal (PRD) or destination (DEST) adjunct only to its template at present: [THING GO DEST] or [THING GO PRD].

Data set	Total sent repaired	Time (sec)	Repaired S trees	Semantically well-formed parse trees	Repaired concepts for S	Repaired well-formed concepts for S	% of syntactical-ly-best parses filtered
Wilson	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)

4 Discussion

4.1 Problems at the syntactic level

The grammar rules need extension to cover the following grammatical phenomena: Compound noun and adjectives; Gerund and TO+VP (infinitive); Conjunctions and

⁹A sentence was considered to be "correctly repaired" if any of the suggested corrections was the same as the one obtained by manual correction.

comparatives; and Phrasal verbs and idiomatic sentences. For example, 'in the morning' and 'at midnight' are the well-formed strings as syntactic idioms. However, CHAPTER currently also parses the ill-formed strings like 'in morning', 'at the morning', and 'at morning' as well-formed at the syntactic and semantic levels.

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.

4.2 Problems at the semantic level

In the current state of 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 both prepositional and adverbial phrases. This would increase the number of templates and the computational cost as well. If there is more than one verbal adjunct (PPs and ADVPs) in a sentence, then CHAPTER does not interpret all adjuncts. For example, the sentence "We are supposed to be going *to the restaurant for dinner*" has two PPs. The first PP would be mapped to the optional conceptual case DEST but the second PP would not be mapped to another conceptual case, for example, PURPOSE, because the act templates of GO do not at present allow multiple optional conceptual cases for an EVENT.

5 Conclusions

This paper has presented a hierarchical error recovery system, CHAPTER, based on a chart parsing algorithm using an augmented context-free grammar. CHAPTER performs a heterarchical procedure, based on an integrated-agenda manager, that invokes subsystems incrementally at four levels: lexical, syntactic, surface case, and semantic. Thus the well-formedness of a sentence has been confirmed and/or repaired after finishing parsing the sentence at four levels.

Semantic processing performed pattern matching using a concept hierarchy and verb templates (which specify semantic selectional restrictions). In addition, procedural semantic constraints represented using a type of boolean expression 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 filtered out 27% of the best repairs

suggested by syntactic recovery system. So it suggested the first correct repair in 73% of cases at both the syntactic and semantic level.

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, 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.
- 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.