On the Complexity of Parsing Ill-formed Inputs

Thanaruk Theeramunkong Hozumi Tanaka

Dept. of Computer Science, Tokyo Institute of Technology

Abstract

This paper gives an analysis of time and space complexities of parsing ill-formed inputs. So far it is known that any context-free langauge can be parsed by an efficient algorithm, e.g., Earley's method, in $O(n^3)$ time complexity. This complexity seems not to be realized for parsing ill-formed inputs. Two modified efficient approaches based on chart parsing methodology are considered: a simple method with no consideration of full context and a method with consideration of full context. We show that inputs with the following common types of errors: extra-word, omitted-word and substituted-word, can be parsed by these two algorithms in $O(n^3\epsilon^3)$ and $O(n^{2(\rho+\epsilon)})$ respectively, where n is the length of the input string, ρ is the length of the longest production and ϵ is the number of errors in the input. The space complexities of these approaches are shown to be propositional to $n^3\epsilon^2$ and $n^{2(\rho+\epsilon)}$ respectively.

1 Introduction

Robustness is an essential property of practical natural language processing. In the area of parsing, many researchers emphasize the importance of handling syntactical ill-formedness. Several inputs to natural language systems are unparsable due to not only the mistakes (or the ungrammatical use) done by users, but also the limitation of the systems, such as misspellings, telegraphic ellipsis, fragmentary inputs, omitted words, incomplete input, punctuation errors, unknown word problem, and so forth.

In the past decade, developmental robustness approaches have been forwarded in several ways: pattern-matching oriented approach, relaxation approaches, parse tree fitting and case-frame oriented approaches (See [Mat93]). One common feature of these researches is the use of grammar-specific rules (with respect to syntactic and semantic characteristics of the language) to recover ill-formedness. Though the solution to processing ill-formed inputs must give consideration to semantic and pragmatic factors, it is important to comprehend the recovery strategies which are based entirely on syntax and also their computational difficulty issues. So far it is known that any context-free language can be parsed in $O(n^3)$ with an efficient algorithm. However this complexity might not be realized for parsing ill-formed inputs.

The aim of this work is therefore to explore purely syntactic and grammar-independent methods for parsing ill-formed inputs, and their complexities. To achieve the ability of coping ill-formedness, a straightforward method is to modify an efficient parsing algorithm. Taking advantage of recording substructures during parsing, a chart parser or tabular parser(e.g., CYK method, Earley's method) can avoid repeating their existing work and analyze any input efficiently via an appropriate indexing mechanism. In this paper, two approaches of modifying chart parsing to deal with the following primitive kinds of lexical errors (extra noise words, omitted words, unknown/substituted words) are taken into account. The first approach, we call *bottom-up hypothesis* approach, is to propose some hypothesizing edges and then to execute an extended active bottom-up chart parsing after the standard parser has terminated unsuccessfully. These edges correspond to the lexical errors at any position in the input.

The second approach, *top-down hypothesis* approach, is to run a bottom-up chart parser, producing complete parses as possible, and then to fit the result parses using some extra rules which are applied in order to find lexical errors and generate interpretations in top-down fashion. This approach is taken in several researches[Mel89][Kat91][Mek93] due to its ability of taking account of full context in the determination of the best interpretation of an ill-formed input.

In this paper, we give an analysis of time and space complexities of these approaches. Upon examination, these approaches are found to have general time complexities of $O(n^3\epsilon^3)$ and $O(n^{2(\rho+\epsilon)})$ respectively, where n is the length of the input string, ρ is the length of the longest production and ϵ is the number of errors. The space complexities of these approaches are propositional to $n^3\epsilon^2$ and $n^{2(\rho+\epsilon)}$ respectively.

2 The Two Modified Chart-based Approaches

Exploiting a working structure to keep track of the parsing progress, a chart parser(or tabular parser) can not only avoid repeating the existing works but also utilize the track for hypothesizing errors. In addition, chart parsing methodology[1] is a flexible parsing framework to enable the distinction among essential bookkeeping

[1] Chart parsing methodology is a general framework to construct a parser.

mechanism, scheduling issues (e.g., agenda), and details of grammatical formalisms. In this respect, the chart parsing method is superior to others in efficiency and flexibility of dealing with ill-formed inputs.

In general, when a robust parser encounters an ill-formed input, a plausible solution is to produce all approximate (partial) parses of the ill-formed input and then to apply some meta rules (hypotheses of ill-formedness) to glue them together to form a complete parse. In this section, we describe two modified chart-based approaches occupying two different types of hypotheses: *bottom-up hypothesis* and *top-down hypothesis*. These approaches are based on active chart parsing and are capable of diagnosing the following simple errors: extra-word, omitted-word and substituted/unknown-word. The former approach is a simple extension of active chart parsing by placing some hypothesis edges (corresponding to errors) at any position in the input, and then restarting the parsing process. The latter approach is to hypothesize that the input sequence constructs a sentence, and then to fit the derived approximate (partial) parses in top-down fashion. Though the former approach is simpler, it cannot exploit the context to choose the best interpretation of the ill-formed input. The latter approach is introduced by Mellish[Mel89] to enable the parser to take account of full context in the determination of the best interpretation of an input. In the rest of this section, we give a brief introduction of active chart parsing methodology (See [Kay80][Win83] for more detail) and then describe these two approaches.

2.1 Active Chart Parsing Methodology

The data structures occupied in active chart parsing methodology are so-called *chart* and *agenda*. *Chart* is a record of nodes and edges, which are numbered 0 to n. Nodes are connected by *active* and *inactive* edges, which record the states of parsing process. *Agenda*, a queue of pending edges, enables parsing process to be controlled. The basic operation of chart parsing is to get an edge from agenda (pending edge), apply *fundamental* rule, and then add the pending edge to the chart. The fundamental rule states that if there is a pending active edge $\{i, j, A \rightarrow A_1 A_2 \cdot A_3 A_4\}$[2] and an inactive edge of category A_3 from j to k ($\{j, k, A_3 \cdot\}$) in the chart, a new active edge $\{i, k, A \rightarrow A_1 A_2 A_3 \cdot A_4\}$ is added into the agenda. However, just only the *fundamental* rule is not sufficient to realize parsing process. A source of empty active edges is required. Such edges typically arise in response to the addition of either inactive edges (data-driven and bottom-up direction) or active edges (hypothesis-driven and top-down direction) to the agenda. This process, so-called *rule invocation*, occurs with respect to the grammar rules. The fundamental rule and rule invocation are shown in Fig.1.

Bottom-up Rule Invocation Procedure:
pending_edge : $\{i, j, A \rightarrow \alpha \cdot, Pnl_p\}$
$\forall (B \rightarrow A\gamma) \in Grammar \wedge \{i, j, B \rightarrow A \cdot \gamma, \_\} \notin Chart$
$\Rightarrow$ place $\{i, j, B \rightarrow A \cdot \gamma, Pnl_p\}$ in *Agenda*

Fundamental Rule Procedure 1 :
pending_edge : $\{i, j, A \rightarrow \alpha \cdot, Pnl_p\}$
$\forall\, k$ s.t. $0 \leq k \leq i$, $\{k, i, B \rightarrow \beta \cdot A\gamma, Pnl_e\} \in Chart \wedge \{k, j, B \rightarrow \beta A \cdot \gamma, \_\} \notin Chart$
$\Rightarrow$ place $\{k, j, B \rightarrow \beta A \cdot \gamma, Pnl_p + Pnl_e\}$ in *Agenda*

Fundamental Rule Procedure 2 :
pending_edge : $\{i, j, B \rightarrow \beta \cdot A\gamma, Pnl_p\}$
$\forall\, k$ s.t. $j \leq k \leq len$, $\{j, k, A \rightarrow \alpha \cdot, Pnl_e\} \in Chart \wedge \{i, k, B \rightarrow \beta A \cdot \gamma, \_\} \notin Chart$
$\Rightarrow$ place $\{i, k, B \rightarrow \beta A \cdot \gamma, Pnl_p + Pnl_e\}$ in *Agenda*.

Fig. 1: Active Chart Parsing Methodology

2.2 Chart Parsing with Bottom-up Hypothesis

When the input is unparsable, the straightforward treatment is to propose some hypothesis edges at any position in the input. A hypothesis edge represents the position of the error and its type. Fig.2 illustrates the primitive types of hypothesis edges corresponding to three types of errors: (a) extra-word, (b) omitted-word and (c) substituted-word.

An edge of type (a) is proposed to make disregard of the word which this edge covers. ϵ expresses *empty* category of the word. A type (b)'s edge is for assuming a word disappearing from the position the edge located. An edge of type (c) offers that there is the category confusion of the word covered by this edge. Among these

[2] $A \rightarrow A_1\ A_2\ A_3\ A_4$ is a production and the symbols $A_1\ A_2$ of this production have been matched to words $i+1$ through j of the input.

three types, types (b) and (c) can be treated with the original active chart algorithm (Fig.1). However, type (a) is different. To handle this type, in addition to the conventional fundamental rule, another rule is needed. Fig.3 shows this additional fundamental rule. A hypothesis edge and an edge derived from this hypothesis edge is given a penalty. Using this penalty, one can enumerate the order of parsing steps.

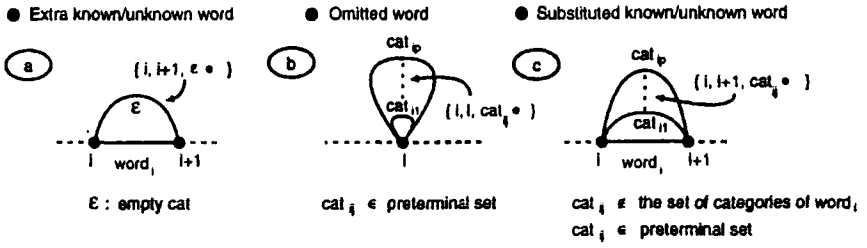

Fig. 2: Types of hypothesis edges

Additional Fundamental Rule Procedure
$\forall \{i, j, B \rightarrow \beta \cdot \gamma, Pnl\}, \{j, j+1, \varepsilon \cdot, Pnl_\varepsilon\} \in Chart \wedge \{i, j+1, B \rightarrow \beta \cdot \gamma, \_\} \notin Chart$
$\Rightarrow$ place $\{k, i, B \rightarrow \beta \cdot \gamma, Pnl + Pnl_\varepsilon\}$ in *Agenda*.

Fig. 3: The Additional Fundamental Rule

2.3 Chart Parsing with Top-down Hypothesis

The top-down hypothesis chart parsing is introduced by Mellish[Mel89]. The merit of this approach is that it allows full syntactic context to be exploited in the determination of the best parses for an ill-formed input. The basic algorithm is to run a left-corner bottom-up parser with no top down filtering and failing which to execute a modified top-down parser (error recovery process). The top-down parser recovers errors by exploiting the edges generated during bottom-up parsing and constructs the interpretations of the input. An improvement for Mellish's method has been proposed by Kato[Kat91]. Kato offers an intermediate step between bottom-up and top-down parser, named *edge completion phase* to exploit more edges generated in bottom-up fashion to reduce searching space in top-down parser. In these methods, the following notation of a generalized edge is introduced.

$$\{S, E, Cat, [(S_1, E_1, Cats_1), ..., (S_l, E_l, Cats_l)]\}$$

where $S(E)$ specifies the starting(ending) position of the edge. Cat is the edge's category and [$(S_1, E_1, Cats_1)$, ..., $(S_l, E_l, Cats_l)$] is the list of the unparsed part of the edge. $Cats_i$ are lists of categories. S, E, S_i, and E_i are denoted by an integer for determined position, or by '$*$' for undetermined position[3].

The error recovery process starts with a hypothesis assuming that all words of an input are finally constructed a sentence. To illustrate this, a data structure is introduced for representing each state during the search as follows.

$$< \text{hole:}N \text{ err:}M \ [(S_1, E_1, Cats_1), \ldots, (S_k, E_k, Cats_k)] >$$

where N is the total number of categories in $Cats_i$; M is the number of errors detected before reaching this state ; S_i and E_i are positions in the chart ; $Cats_i$ is a set of categories needed between S_i and E_i.

The initial searching state is < hole:1 err:0 [(0,n,[S])] >, where S is the start symbol (goal category) and n is the final position in the input. In addition to top-down rule and fundamental rule of conventional top-down chart parsing, some extra rules are utilized to detect some primitive types of errors: extra-word, omitted-word, substituted/unknown-word errors. Fig.4 shows the generalized top-down parsing rules[4]. The first two rules are used to refine the undetermined part of the edge while the last three rules are used to take account of three kinds of primitive errors (extra word, omitted word, and unknown/substituted word). One interpretation of the ill-formed input can be obtained when the process reaches a state < hole:0 err:Err [ ] >. In both Mellish's and Kato's methods, each edge(in the chart) is given a score based on some heuristics. These heuristics are based on purely syntactical information, i.e., minimal penalty scores. By this score, searching direction during parsing can be controlled.

[3]For instance, {0,*,S,[(2,4,[Adv]),(7,*,[Pred])]} stands for an edge which covers from 0 to undetermined position, and needs [Adv] from 2 to 4 and [Pred] from 7 to the same position, when S→ Subj,Adv,Bep,Pred is a production, Subj is found from 0 to 2 and Bep is found from 4 to 7.

[4]These rules are analogous to ones defined in [Kat91].

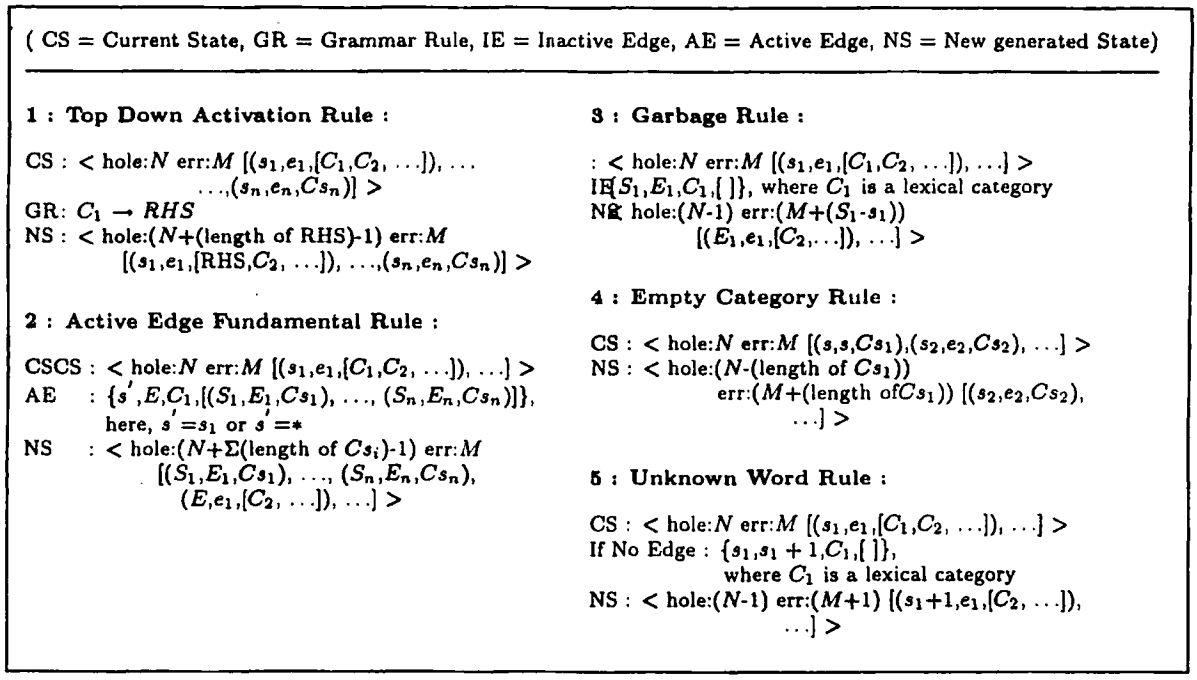

Fig. 4: Rules in P-ETD

3 Complexity of Parsing Ill-formed inputs

In this section, we present the time and space complexities of the two modified parsing methods described in the previous section.

3.1 Complexity of Chart Parsing with Bottom-up hypothesis

The main complexity factors involved in chart parsing with bottom-up hypothesis are the applications of fundamental rule and rule invocation. Based on this fact, the time and space complexities of bottom-up hypothesis chart parsing will be proven. These complexities are shown as the function of input length(n), grammar size ($|G|$) and the number of errors (ϵ).

Lemma 1 *The number of edges generated is at most $O(|G|n^2\epsilon)$.*

Proof :

Each edge of chart parsing can be represented in the form of $\{S, E, A \rightarrow \alpha \bullet \beta, Penalty\}$. S and E represent positions which are integers ranging from 0 to n. There are at most O(|G|) edges between each pair of positions. *Penalty* is bounded to the number of errors in the input (ϵ). Therefore, the number of possibilities is $O(|G|n^2\epsilon)$.

Lemma 2 *The number of partial parses occupied in each edge is at most $O(n\epsilon)$.*

Proof :

Each edge($\{S, E, A \rightarrow \alpha B \bullet \beta, Penalty\}$) is generated by applying either fundamental rule or rule invocation. An edge (*Edge*$'$) generated by the bottom-up rule invocation keeps a pointer to the parses[5] of the inactive edge from which *Edge*$'$ is derived. On the other hand, an edge (*Edge*$''$) generated by fundamental rule, holds two pointers to the parses of the pair of the active edge and the inactive edge from which *Edge*$''$ is derived. This pair is in the form of $\{S, I, A \rightarrow \alpha \bullet B\beta, Penalty'\}$ and $\{I, E, B\bullet, Penalty\text{-}Penalty'\}$. Here, the number of the possible pairs depends on I and *Penalty*$'$. Therefore, it is $O(n\epsilon)$.

Lemma 3 *The space comsumed by the chart is at most $O(|G|n^3\epsilon^2)$.*

Proof :

According to lemma 1 and 2, the space comsumed by all edges in the chart is the multiplication of $O(|G|n^2\epsilon)$ and $O(n\epsilon)$. It is $O(|G|n^3\epsilon^2)$.

Lemma 4 *The number of actions occurred by applying rule invocation is at most $O(|G|n^2\epsilon)$.*

[5]These parses can be kept as a list.

Proof :

The number of applications of rule invocation is proportional to the total number of inactive edges. Each inactive edge can be represented in the form of $\{S, E, B\bullet, Penalty\}$. The number of inactive edges is related to $S, E, |G|$ and $Penalty$. It is $O(|G|n^2\epsilon)$.

Lemma 5 *The number of actions occurred by applying fundamental rule is at most $O(|G|n^3\epsilon^2)$.*

Proof :

Fundamental rule involves any pair of active edge and inactive edge. Thus, the number of the applications of fundamental rule corresponds to the number of possible pairs. At this point, each pair of edges can be represented by $\{S, I, A \rightarrow \alpha \bullet B\beta, Penalty\}$ and $\{I, E, B\bullet, Penalty'\}$. The number of pairs is related to $S, I, E, |G|, Penalty$ and $Penalty'$. Therefore, there are $O(|G|n^3\epsilon^2)$ pairs of edges which are needed to be calculated.

Lemma 6 *Checking whether an edge is previously proposed or not, costs no more than $O(|G|\epsilon)$.*

Proof :

To avoid calculation duplication, when an edge is generated, it is necessary to check whether the edge has already generated or not. Since each edge can be represented in the form of $\{S, E, A \rightarrow \alpha \bullet \beta, Penalty\}$, the number of edges which have S and E as its starting and ending position respectively, is not greater than $O(|G|\epsilon)$. By indexing edges with their starting and ending position, the number of steps of checking a newly generated edge with pre-existing edges is not greater than $O(|G|\epsilon)$.

Lemma 7 *The efficiency chart parsing algorithm with bottom-up hypothesis for ill-formed input has $O(|G|^2n^3\epsilon^3)$ time complexity.*

Proof :

The actions in chart parsing are the applications of rule invocation and fundamental rule, and duplication checking. According to lemma 4-7, the total complexity is $(O(|G|n^2\epsilon) + O(|G|n^3\epsilon^2)) \times O(|G|\epsilon) = O(|G|^2n^3\epsilon^3)$.

3.2 Complexity of Chart Parsing with Top-down hypothesis

From the viewpoint of complexity issue, the main difference of chart parsing with top-down hypothesis from chart parsing with bottom-up hypothesis is the sequence of the parsing process. In order to exploit the context in determining the best interpretation, the top-down hypothesis chart parser processes any edge with no respect to left-to-right order. This feature makes the top-down hypothesis chart parsing occupy more complexity than the bottom-up hypothesis chart parsing. In this part, we explore how much complexity this type of chart parsing has.

The top-down hypothesis chart parsing has two phases: bottom-up parsing and top-down parsing (error recovery process). Instead of Mellish's parser, we consider Kato's parser because it is more explicit than Mellish's parser without losing generalization. To find out the parsing complexity, the actions of each phase have to be taken into account. The complexity factors of top-down hypothesis chart parsing are involved with the number of generated edges. The time and space complexities are shown as the function of input length(n), grammar size ($|G|$), the number of errors(ϵ) and the length of the longest production rule (ρ).

Lemma 8 *The number of edges generated in bottom-up parsing is at most $O(|G|n^{2\rho})$.*

Proof :

As described in section 2.3, each edge in top-down hypothesis chart parsing can be represented in the form of

$$\{S, E, Cat, [(S_1, E_1, Cats_1), ..., (S_k, E_k, Cats_k)]\}$$

where S, E, S_i and E_i are integers ranging from 0 to n. The maximum number of elements in [....] is correspondent to the length of longest production rule - 1 (ρ -1). The number of generalized edges is related to S, E, S_i, E_i and Cat. Therefore, there are $O(|G|n^{2\rho})$ different edges.

Lemma 9 *The time and space complexity of bottom-up parsing is $O(|G|n^{2\rho})$.*

Proof :

The main action of bottom-up parsing is generating the generalized edges which are kept to utilize in the next step, the error recovery process. Both time and space complexities are related to the number of generalized edges. It is $O(|G|n^{2\rho})$.

Lemma 10 *The number of searching states of top-down parsing (error recovery process) is at most $O(|G|^\epsilon n^{2\epsilon})$.*

Proof :

A searching state in top-down parsing, can be represented as follow.

$$\{Hole, Err, [(S_1, E_1, Cats_1), ..., (S_m, E_m, Cats_m)]\}$$

The number of states is concerned with the following variables: S_1, E_1, ..., S_m, E_m, and $Cats_1$, ..., $Cats_m$. S_i (E_i) ranges between 0 to n while $Cats_i$ depends on the number of categories, that is at most $|G|$. Moreover, the number of items in [...], at most, corresponds to the existing error (ϵ). Based on this fact, the number of states is $O(|G|^{\epsilon} n^{2\epsilon})$.

Lemma 11 *The number of actions per one state is at most $O(|G|n^{2\rho})$.*
Proof :

In top-down parsing, there are six basic actions (rules): Top-down, Fundamental, Simplification, Garbage, Empty and Unknown-word rules as shown in Fig.4. Among these rules, the Fundamental rule is the main source of the complexity. Focused on one state, the number of actions is correspondent to the number of generalized edges. It is $O(|G|n^{2\rho})$.

Lemma 12 *The number of actions of top-down parsing is at most $O(|G|^{\epsilon+1} n^{2(\rho+\epsilon)})$.*
Proof :

According to lemma 10-11, the total number of actions of top-down parsing corresponds to the multiplication of the number of states and the number of edges. That is $O(|G|^{\epsilon} n^{2\epsilon})$ and $O(|G|n^{2\rho})$. It is $O(|G|^{\epsilon+1} n^{2(\rho+\epsilon)})$.

Since the results of actions have to be kept to represent the parses, the space complexity is also bould to the number of actions. From lemma 9 and 12, the time (space) complexities of chart parsing with top-down hypothesis are the sum of the actions (space) in bottom-up and top-down parsing. It is $O(|G|^{\epsilon+1} n^{2(\rho+\epsilon)})$.

4 Conclusion

In this paper, we have given an analysis of time and space complexities of parsing ill-formed inputs. We show two approaches based on chart parsing for dealing with certain common types of ill-formedness: extra noise words, omitted words, unknown/substituted words. The first approach is a simple extension of active chart parsing by proposing some hypothesizing edges corresponding to the lexical error at any position in the input after a standard parser has terminated unsuccessfully, and then restarting the parsing process. The second approach is to run a bottom-up chart parser producing complete parses as possible, and then to fit the resultant parses using some extra rules to find lexical errors and generate interpretations in top-down fashion. From the results of this paper, we can conclude that these two approaches can parse an ill-formed input in $O(n^3\epsilon^3)$ and $O(n^{2(\rho+\epsilon)})$, respectively. The space complexities of these approaches are shown to be propositional to $n^3\epsilon^2$ and $n^{2(\rho+\epsilon)}$ respectively. From the results, the time complexity of the former is not worse than that of the conventional chart parsing. On the other hand, to achieve context-dependent error recovery strategies, the latter may require time bound to more than the cube of the input length.

References

[Kat91] Kato, T.: Yet Another Chart-based Technique for Parsing Ill-formed Input, in *Natural Language Processing*, pp. 83–100, Information Processing Society of Japan, 1991, in Japanese.

[Kay80] Kay, M.: Algorithm Schemata and Data Structures in Syntactic Processing, Technical report, Xerox PARC, 1980, Research Report CSL-80-12.

[Mat93] Matsumoto, Y.: Current Status and Perspective of Robust Natural Language Processing, in *Proc. of the 8th Annual Conference of JSAI*, pp. 75–78JSAI, 1993, S-3, (in Japanese).

[Mek93] Meknavin, S., T. Theeramunkong, and H. Tanaka: Parsing Ill-Formed Input with ID/IP rules, in *The Fourth International Workshop on Natural Language Understanding and Logic Programming(NLULP4)*, pp. 158–171, 1993.

[Mel89] Mellish, C.: Some Chart-based Techniques for Parsing Ill-formed Input, in *Proceeding of 27th Annual Meeting of the ACL*, pp. 102–109, 1989.

[Win83] Winograd, T.: *Language as a Cognitive Process*, Vol. 1:Syntax, Addison-Wesley, 1983.