GLR Parser with Conditional Action Model (CAM)

Yong-Jae Kwak, Young-Sook Hwang, Hoo-Jung Chung, So-Young Park, Sang-Zoo Lee, Hae-Chang Rim

Dept. of Computer Science & Engineering, Natural Language Processing Lab, Korea University, Anam-dong 5-ga, Seongbuk-gu, Seoul, Korea

{yjkwak,yshwang,hjchung,ssoya,rim,zoo}@nlp.korea.ac.kr

Abstract

There are two different approaches in the LR parsing. The first one is the deterministic approach that performs the only one action using the control rules learned without any LR parsing resource. It shows good performance in speed. But it has a disadvantage that it cannot correct the previous mistakes, thus directly affects the parsing result. The second one is the probabilistic LR parsing approach, which uses annotated corpora and LR table.

In this paper, we propose a new probabilistic GLR parsing method that can solve the problems of conventional LR parsing approaches. The parser takes probabilistic shift or reduce action on the graph-structured stack, which is conditioned on the partially constructed parse. It chooses the most probable action sequences for the given input sentence when it reaches the final state. With the probabilistic model, it can overcome the shortcomings of the deterministic LR parsing scheme, and it uses richer information in the stack than the previous probabilistic methods.

1 Introduction

LR(Left-to-right scan, Rightmost derivation) parsing is regarded as one of the most efficient parsing methods, because it doesn't require any extra overheads to find the previously constructed substructure or to match each element of the grammar rule set. Many computational linguists were fascinated by its efficiency, and try to make NL syntactic parsers adapting the LR parsing technique. In the late 1980s, a generalized version of the LR parsing method was devised. It could overcome the limit of one-way linear stack of the traditional LR method. Generalized LR, so-called GLR parsing (Tomita, 1986) has been playing an important role of a "firm starting point" to generate other variations for NL parsing equipped with various mechanisms. Nowadays, many variations of this parsing method have been used in various NL applications that require an efficient phrase-structure generating parser, such as machine translation and speech recognition (Lavie, 1996; Ruland, 2000),etc.

In this paper, we introduce a new GLR parsing method which takes a probabilistic shift or reduce action on the graph-structured stack conditioned on the partially constructed parse. Section 2 shows some previous works that have tried to incorporate NLP techniques into LR parsing framework. In section 3, we introduce our conditional action model (CAM) including the basics of GLR parsing, and conditioning scheme of shift-reduce action. In section 4 we evaluate our parsing model through some experiments and discuss the results of them. And then the conclusion and planning of future works come in section 5.

2 Previous works

Several LR parsing techniques are proposed for resolving parsing ambiguities. It can be grouped into two major LR parsing approaches: the rule-based deterministic method, and the probabilistic method.

2.1 Deterministic LR Parsing

This approach points out that the computational complexity is the most critical factor of a parser. It takes shift-reduce actions deterministically using the learned control rules on the one-way linear-stack. Thus, this approach can achieve the considerable parsing efficiency. However, the approach doesn't have any mechanism for correcting the errors which can be propagated from the previous mistakes. If wrong actions were selected in the previous steps, these mistakes can impact on the next actions. As a result, wrong parse or parsing failure might be produced.

(Simmons and Yu, 1991) used action selecting rules controlled by "windowed context" in their deterministic LR parser. Windowed context is a limited length of sequence composed of syntactic symbols and part-of-speeches that is located both on a stack and an input buffer. It takes a follwing form

$$<stack\_Seq : input\_Seq \rightarrow action> \qquad (2.1)$$

, and it acts as conditional information for the action selection. Their method has shown a possibility of implementing context-sensitive NL parsers without having complex disambiguating schemes. (Kwak, 1999,2001) has tried to apply a similar approach to Korean. This framework was also extended by some subsequent works. (Hermjakob, 1997, 2000) has trained deterministic rules from rich contextual information with a machine learning technique and then applied the trained rules to parsing. Its training features are very specific and rich; Not only POS but also semantic and ontological information are used for features. Wong and Wu (Wong and Wu, 1999) did not depend on explicit CFG for tree production. Instead, they have labelled the constituents in stack using a decision tree.

2.2 Probabilistic LR Parsing Approach

In contrast to the rule-based deterministic approach, this approach tried to combine GLR parsing method and probabilistic method.It has some advantages such as flexibility, extensibility, and consistency.

The first work of GLR technique combined with a probabilistic method is (Wright and Wrigley, 1991)'s work that integrated PCFG into GLR technique. In their work, probabilities of each entry of PCFG are assigned to the reduce actions. Their parser could calculate the product of reduce probabilities of a parse in high speed, but the final score of the resulting parse were exactly same to the usual PCFG parser's output, when same CFG entries are used.

Such an absence of the context-awareness of Wright's method has been extended by subsequent works of others. (Su et al., 1991) introduced a method that uses left-right symbols surrounding the action decision points as a context for their scheme of parse scoring. They decomposed the given parse tree into a collection of parse phrase units, and redefined the parsing scheme as a transition of stack contents between two shift actions which have multiple shift-reduce actions compacted in one step. However, this method was not devised for GLR parsing originally, so it was too complex to be integrated into GLR framework.

After that, some probabilistic GLR parsers have been implemented, in which probabilities are assigned to actions of LR parsing tables by using lookahead symbols or words and LR states as simple context information (Briscoe and Caroll (B&C), 1992, 1993, 1995, 1996; Kentaro et al., 1997A, 1997B, 1998; Ruland, 2000). These works depend on LR tables and use insufficient information to resolve context-sensitive ambiguities.

3 GLR Parser with Conditional Action Model

In this section, we describe what GLR parsing is, what our conditional action model is, and how its parsing model is constructed. Additionally, differences between our model and previous approaches will be discussed.

3.1 Overview of GLR framework

Basically, GLR parsing framework (Tomita, 1986) is composed of **LR** (canonical or LALR) **table** and the **graph-structured stack** as represented in Figure 1.

LR table is a shift-reduce action fetcher which is made of an interpretation of the given CFG. Because the LR table is made of CFG grammar containing ambiguity, some entries in the table may have more than one acion. With the table, the GLR parser can reduce the overhead of searching existing grammar entries and avoid making some unnecessary phrase production, which can be allowed in the parsers which use the only learned rules from the annotated corpus.

Graph-structured stack stores the intermediate parse states while the GLR parser is working. It is functionally similiar to chart or trigonal table often used by other types of parser, but the stack maintains only the list of the positions of the state nodes called "active stack node" located in the tops of the stack, to which shift-reduce actions are applied; therefore, this can make LR parsers faster than any other type of PSG parser.

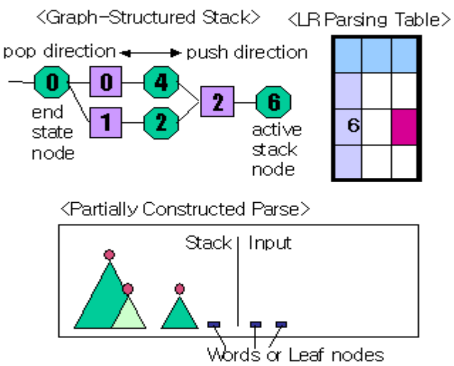

Figure 1. Basic Architecture of GLR Parser

Using the above-mentioned LR table and the graph-structured stack, GLR parser works as follows. For one word read from an input sentence, it takes the action(s) indicated by the LR table, and outputs packed-parse forest as the parse result, which reflects local ambiguity (packed nodes). Entire parsing is continued until there is no more input word left.

3.2 Conditioning in GLR Framework

Probabilistic GLR parsing is the method that uses probabilities on conditions of the LR parsing table or the stack. And some approaches for taking necessary context have been reported. (Briscoe and Caroll, 1992,1993,1995) took the transitions of LR state for the conditioning point, (Inui et al., 1997A,1997B,1998) and (Su et al., 1991) took the transition of the top of stack content, and (Ruland, 2000) modelled action sequence probability conditioned by the top of stack content and LR state.

We define a conditioning scheme controlled by more context than previous works. In the canonical GLR parsing framework, there are two major cases in which probabilities can be assigned to the current actions as follows.

Ambiguous Action: When the corresponding table entry has more than one action for the input word and state number of the active stack node, this is the ambiguous action. For example, when one reduce action and one shift action have to be applied simultaneously to a node, both the introduction of an input word into the stack by the shift action and the phrase production(s) by the reduce action can be occurred (see Figure 2).

Multiple Parse (phrase-production): In GLR parsing, an active stack node is often connected to more than one link from the previous nodes resulted by node combination whereas only one link exists in deterministic model. If a reduce action has to be applied to the active stack node, as many phrase productions as links would be generated according to the production rule (also see Figure 2) .

In proposed conditional action model (from now, we will call it CAM in short) proposed in this paper, we assign probabilities to the current action by considering the both cases described above. Our scheme assigns probability to the current action using structural information and input words which are extracted from partially constructed parse in the stack before the parser takes the current action

3.3 GLR Parsing with Conditional Action Model

3.3.1 Probabilistic Model

We define the entire parse of the given input sentence as the sequence of actions taken until GLR parser reaches the final state (accept action is taken). Each actions are performed with probability assigned to itself by CAM. First, probability of an action *P(a|cntx)* is defined as follows;

$$P(a \mid cntx) = P(a \mid \{f_1, \ldots, f_n\}) \qquad (3.1)$$

where the a is a prediction of either shift or reduce which is conditioned on feature set $\{f_1, \ldots, f_n\}$ is feature set extracted from partially constructed parse and stack.

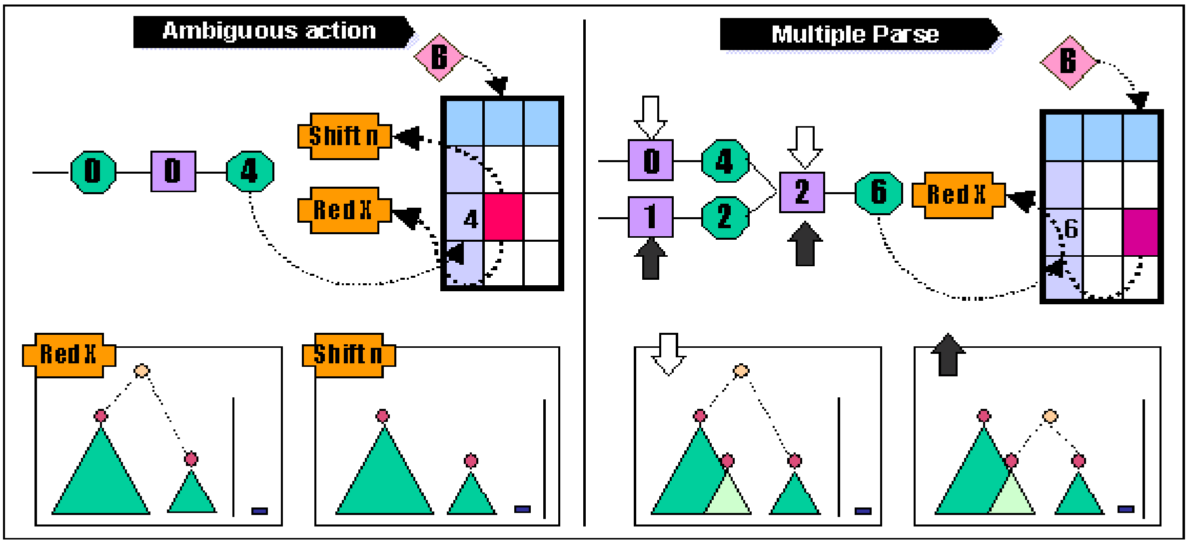

Figure 2 Situations required to be conditioned in GLR parser; Ambiguous Action and Multiple Parse. In this picture, symbol B represents an input word and the graph-structured stacks are fabricated for explanation. Icon "Red" and "Shift" represents a reduce action and a shift action, respectively.

Then, probability of a parse tree can be calculated with the "action probabilities", as follows;

$$P(T_i \mid S) = \left(\prod_{j=0}^{n} P(a_i^j \mid cntx_i^j) \right)^{\frac{1}{n}} \qquad (3.2)$$

In the above equation, T_i is the i-th parse tree generated from an input sentence S; a_i^j and $cntx_i^j$ represents the j-th action and context. Note that the probability of a parse is defined as a geometric mean of the all action probabilities, since the product of probabilities along a given event path is not appropriate for evaluating the parse results in some non-ergodic systems like GLR parsers (Caroll and Briscoe, 1992)[1].

As a result, "best-one" parsing using GLR with CAM is defined as searching the action sequence maximizing the parse probability as shown in (3.3).

$$P(T_{best} \mid S) = \arg\max_i P(T_i \mid S) \qquad (3.3)$$

3.3.2 Context definition for CAM

In this section, we describe the context features for CAM. The basic concept to use uses the more available information of the stack contents for selecting the proper action. It makes this model more context sensitive than any other previous probabilistic method.

For utilizing context feature, we considered the structure information of a given partial parse forest at the first time.

- **Reduced parse forests.** n parse forests that can be reduced by a certain production rule $X \rightarrow Y_1, \ldots, Y_n$. They are represented as $Red\_for_1, \ldots, Red\_for_n$.
- **Sibling parse forest.** a parse forest on the left of Red_for_1. It is represented as Sib_for[2].
- **Surrounding terminals.** m terminal symbols occurring in the left of Sib_for, and l input

[1] (Kentaro et al., 1997B) argued that such a scheme was caused by their imperfect probabilistic model, and he used the product of probabilities using his new formalism. But, in the case that same number of constituents are affected by the more than one reduce-rules with different number of right-hand side symbols (for example, a sequence of "V N PP" can be affected both $VP \rightarrow V\ N\ PP$ and $VP \rightarrow V\ NP$ / $NP \rightarrow N\ PP$), total transition counts of each action sequences cannot be same ; product of longer sequence might be lower than shorter one.

[2] The sibling forest (Sib_for) is naturally pointed by the end state node of given reduce path (in the pop-direction) . But, if more than one push-direction paths are connected to the end state, the sibling forest cannot be determined to only one. In that case, the forest with the smallest depth is selected heuristically; this can become a shortcoming that may influence on this model's behavior.

symbols occuring in the right of Red_for_n. The former is represented as $ts_1,\ldots,ts_m$ ($ts(m)$ in short), the latter $is_1,\ldots,is_l$ ($is(l)$ in short).

And then, we define the forest feature *ff(F)* as follows;

▪**Forest Feature.** *ff (F)* for a parse forest *F* is a triple, *<Root(F), S(F), Leaves(F,d)>,* where *Root(F)* stands for the non-terminal symbol of the *F*'s root node, *S(F)* stands for the span of *F,* represented as the number of leaf nodes. *Leaves(F,d)* stands for the subset of the leaf nodes dominated by *F,* with the cardinality of $d\ (\leq S(F))$. If *d* is same or greater than *S*, *d* is determined as *S.*

Finally, we define the feature set of CAM using the above notations(See Figure 3):

▪**Context for CAM .** *cntx* is an n-tuple $<ts(m), ff(Sib\_for), ff(Red\_for_1),\ldots,ff(Red\_for_n), is(l)>$, where each term is same as explained before.

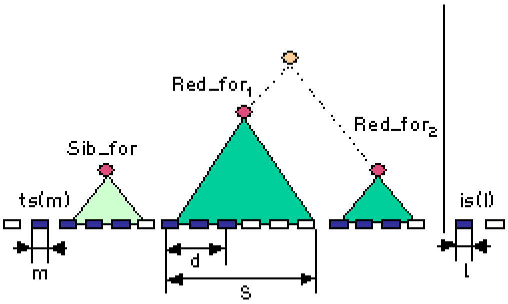

Figure 3. Partially constructed parse structure used as context for CAM. In this figure, small squares are the leaf nodes or input words. Solid squares are the nodes included in context, whereas hollow squares are excluded ones. In this figure, values of *d, m,* and *l* are set to 3,1, and 1, respectively.

To cope with unseen contexts when parsing with CAM, it assigns default probability to them without using any special special smoothing mechanism. In the case of ambiguous action, 1 divided by the number of actions is assigned to the action probabiliy, and in the case of multiple parse, fraction of the difference between 1 and the sum of probabilities of the seen contexts is used for action probabilities, as shown in (3.4):

$$P(a \mid unseen\_cntx) = \frac{1 - \sum_{i \in seen\_cntx} P(a_i \mid cntx_i)}{C(unseen\_cntx)} \qquad (3.4)$$

With the proposed model, our parser works like the common LR parser. When it is required to select an appropriate action among actions which are provided for the given context, it evaluates the probability of each action, adds the action to the existing sequence, and records the probability. Finally, we select the optimal parse with the best probability. For the action probability estimation, we adopted the statistical decision tree learning as described in section 4.

4 Experiments and Evaluation

4.1 Environments

We have experimented on the Korean tree-tagged corpus which consists of 12,084 sentences tagged with KFG scheme (Park et al. 1999) (the distribution of sentence length over the corpus is shown in Table 1). We used 11,932 sentences for training and used 152 sentences for testing.

Table 1 Characteristics of entire tree-tagged corpus. Mean of morpheme length is 22.51

Morpheme Length	Snts	Morpheme Length	Snts
0 – 9	573	30 – 39	2241
10 – 19	4244	40 – 50	350
20 – 29	4650	51 – 59	26

For training, we extracted 344,768 shift or reduce instances with predefined features from the training set, and trained CAM with the Quinlan's C4.5 decision tree learning (Quinlan, 1993). And then, we have converted the resulting decision trees to the set of rules, and estimated the probability of each rule as an action probability.

For the purpose of investigating the effect of various feature selections of CAM, we have defined the two feature sets according to whether the structure of the forest is considered or not, as shown below(currently, we have considered the part-of-speech tags alone for leaf nodes):

▪**F1 :** $m \geq 1$, $l \geq 1$ *(m=l)*, $d \geq 1$, and the span of the forest (*S*) is included; this set considers the structure of the forest.

▪**F2 :** $m \geq 1$, $l \geq 1$, $d = 0$ and *S* is ignored; this set does not consider the structure of the forest.

Our GLR parser uses canonical LR parsing table constructed from the binary CFG entries provided by the KFG (Park et al., 1999), which has 70 states. The parser takes a part-of-speech sequence as an input, and all of the part-of-speeches of input words were tagged with 100% accuracy by hand in advance. With CAM, our parser searches the rule that has the maximal number of features and uses its probability. To compare our parser with the existing PGLR methods, we implemented the model of (Briscoe and Carroll (B&C), 1993) and of (Inui et al.,1997B), and then tested them on the same condition where our parser was tested.

4.2 Experimental Results

Our parser can parse all the sentences of the treebank in 3204 seconds, producing every possible tree (0.26514 secs / snt) under the Intel machine with Celeron 566Mhz/256MB. The accuracy of our parser and of the existing two models are shown in Table 2. For evaluation, we used the criterion provided by (Goodman, 1996)[3]. Labelled recall (LR) rate and exact matching rate (EM) were calculated as follows:

$$LR\,(\%) = \frac{\#\,of\ correct\ parse\ constituents}{\#\,of\ treebank\ constituents} * 100 \qquad (4.1)$$

$$EM\,(\%) = \frac{\#\,of\ snts\ whose\ parse\ exactly\ match}{\#\,of\ treebank\ snts} * 100 \qquad (4.2)$$

Table 2 Result of parsing with GLR+CAM method.

	F2 feature		F1 feature		B&C	Inui et al.
	M=1 *l*=3	*m*=3 *l*=5	*M*=*l*=1 *d*=3	*M*=*l*=2 *d*=5		
LR	75.4	73.4	72.1	71.3	72.79	74.81
EM	5.2	3.2	3.9	3.2	3.2	4.6

[3] Because all the grammar rule entries used by our parser's LR parsing table are only binary branching, both Labelled Recall and Labelled Precision metric have the same value. Thus, we used the only Labelled Recall metic.

According to the above results, the model using F2 outperformed the model using F1, which was our originally intended model. It is because such a feature selection is not appropriate for taking the characteristics of structure of the parse forest. At first, the coverage and accuracy of the rules that includes the span (*S*) of the parse forest was relatively low (for example, when *m*=*l*=1 and *d*=3, the number of these rules were 11 of the total 134 rules, and the highest accuracy among the accuracies of these rule was 32.1 %). And, there was no key clues for disambiguating among the leaf nodes limited to a few front terminals, because most head words are usually located in the rear position of Korean sentences. Another problem is data sparseness, and currently the appropriate countermeasure is not provided for it.

We have also compared our method with the deterministic model, using the same training and test set (Table 3). For the comparison, we implemented a deterministic LR parser with a linear-stack. It performs shift-reduce action using the decision tree which was learned with the same feature set described in section 4.1 (It is simlar to Hermjakob's in that it uses decision tree, although it does not use a large number of linguistic resources), without any base grammar. The deterministic parser has selected correct actions for the given context with accuracy ranging from 75.9% to 85.4% for all of the test sentences. But, in some cases, previous selection mistakes influenced the next action selection steps, so the parser could not reach the accept state (popping all the items in the stack) consequently. This informs that the determinstic LR method with controlling rule set alone may not succeed in parsing many sentences.

Table 3 Result of parsing with Deterministic LR method.

	F2 feature		F1 feature	
	m=1 *l*=3	*m*=3 *l*=5	*m*=*l*=1 d=3	*m*=*l*=2 d=5
LR	68.9	67.2	61.3	58.5
Not Parsed	6.5 (10/152)	7.8 (12/152)	10.5 (16/152)	12.5 (19/152)
EM	9.21 (14/152)	9.86 (15/152)	3.94 (6/152)	3.28 (5/152)
Action Selecting	85.4%	83.1%	77.9%	75.9%

5 Conclusion and Future works

In this paper, we have introduced a probabilistic GLR parsing method which uses the structure of the partially constructed parse as context while the parsing procedure is in progress. We have depicted the cases in which probability assignment is necessary, and proposed the parsing model which can be operates according to the conditional information we defined.

For the future works, our context definition should be enhanced together with an appropriate smoothing mechanism, because experimental result showed that the current feature set we have defined as the partial parse structure was not enough to contribute to parser's performance. Additionally, we will develop the specific learning method that suits for our proposed model.

References

Tomita, Masaru. 1986. *Efficient Parsing for Natural Language: A Fast Algorithm for Practical Systems*. Kluwer Academic Publishers.

J. H. Wright and E. N. Wrigley. 1991. GLR Parsing with Probability. In *Generalized LR Parsing*. Kluwer Academic Publishers, 1991

Keh-Yi Su, Jong-Nae Wang, Mei-Hui Su, and Jing-Shin Chang. 1991. GLR Parsing with Scoring. In *Generalized LR Parsing*, Kluwer Academic Publishers, pages 93-112.

Alon Lavie. 1996. *GLR*: A Robust Parser For Spontaneously Spoken Language*. Ph.D. thesis, School of Computer Science, Carnegie Mellon University.

Inui Kentaro, Virach Sornlertlamvanich, Tanaka Hozumi and Tokunaga Takenobu. 1997A. A New Formalization of Probabilistic GLR Parsing. In *Proceedings of the 5th International Workshop on Parsing Technologies.1997.*

Inui Kentaro, Virach Sornlertlamvanich, Tanaka Hozumi and Tokunaga Takenobu. 1997B. *A New Probabilistic LR Language Model for Statistical Parsing*. Technical report, TR97-0004, Dept. of Computer Science, Tokyo Institute of Technology.

Inui Kentaro, Virach Sornlertlamvanich, Tanaka Hozumi and Tokunaga Takenobu. 1998. Probabilistic GLR parsing: a new formalization and its impact on parsing performance. In *Journal of Natural Language Processing*, Vol. 5, No. 3, pages 33-52.

John Carroll and Ted Briscoe. 1992. Probabilistic normalisation and unpacking of packed parse forests for unification-based grammars. In *Proceedings of the AAAI Fall Symposium on Probabilistic Approaches to Natural Language*, pages 33-38.

Ted Briscoe and John Carroll. 1993. Generalized Probabilistic LR Parsing of Natural Language (Corpora) with Unification-Based Grammars. In *Computational Linguistics,*19(1):pages 25-59.

Briscoe, E. and J. Carroll .1995 Developing and evaluating a probabilistic LR parser of part-of-speech and punctuation labels. In *Proceedings of the 4th ACL/SIGPARSE International Workshop on Parsing Technologies,* Prague, Czech Republic, pages 48-58

Ted Briscoe and John Carroll. 1996. Apportioning Development Effort in a Probabilistic LR-Parsing System through Evaluation. *ACL/SIGDAT Conference on EMNLP*, pages 92-100.

Tobias Ruland, 2000. A Context-Sensitive Model for Probabilistic LR Parsing of Spoken Language with Transformation-Based Postprocessing. In *Proceedings of the 18th International Conference on Computational Linguistics*, pages 677-683.

Robert F. Simmons and Yeong-Ho Yu. 1991. The Acquisition and Use of Context-Dependent Grammar for English. In *Proceedings of the 29th annual Meeting of the Association for Computational Linguistics*, pages 122-129.

Ulf Hermjakob. 1997. *Learning Parse And Translation Decisions From Examples with Rich Context.* PH.D thesis, University of Texas at Austin.

Ulf Hermjakob. 2000. *Rapid Parser Development: A Machine Learning for Korean.*In *Proceedings of the North American chapter of the Association for Computational Linguistics (NAACL-2000),* pages 118-123.

Yong-Jae Kwak. 1999. *Acquisition and application of Context Sensitive Grammar Rules for Deterministic Parsing of Korean*. Ms. thesis, Dept. of Computer Science and Engineering, Korea University.

Aboy Wong and Dekai Wu. 1999. Are Phrase Structured Grammars Useful in Statistical

Parsing?. In *Proceeding of 5th Natural Language Processing Pacific Rim Symposium*, pages 120-124.

J.Ross Quinlan. 1993. *C4.5: Programs for machine learning*. Morgan Kauffman Publishers.

Joshua Goodman. 1996. Parsing Algorithms and Metrics. In *Proceedings of the 34th Annual Meeting of the ACL*, pages 177-183.

So-Young Park, Young-Sook Hwang, Hoojung Chung, Yong-Jae Kwak, and Hae-Chang Rim. 1999. A Feature-based Grammar for Korean Parsing. In *proceedings of 5th Natural Language Processing Pacific Rim Symposium*, pages 167-171.