Empirical Evaluation of Probabilistic GLR Parsing

Virach Sornlertlamvanich, Kentaro Inui, Kiyoaki Shirai, Hozumi Tanaka, Takenobu Tokunaga
{virach,inui,kshirai,tanaka,take}@cs.titech.ac.jp
Department of Computer Science, Tokyo Institute of Technology
2-12-1, Oookayama, Meguro-ku, Tokyo 152
and Toshiyuki Takezawa
ATR Interpreting Telecommunications Research Laboratories
takezawa@itl.atr.co.jp

Abstract

This paper presents the results of experiments on probabilistic GLR (PGLR), a new probabilistic model proposed by (Inui et al., 1997). The model is formalized based on stack transition during parsing distinguishing it from the existing models proposed by Wright and Wrigley, and Briscoe and Carroll. Our model produces a remarkable improvement in syntactic parsing with probability. Associating probabilities directly to actions in an LR parsing table, and theoretically requiring only one probability for each action guarantee model trainability and potential applications to other related tasks.

1 Introduction

Probabilistic techniques have been introduced to various kinds of natural language processing tasks, due to the increasing availability of text corpora. In syntactical parsing, probabilistic techniques are utilized to rank the potentially high numbers of parses generated for natural language (NL) applications.

Several attempts have been made to prune meaningless parses and aid in the selection of the most likely parse from multiple parse candidates. Fujisaki et al. (Fujisaki et al., 1989) introduced the notion of a probabilistic context-free grammar (PCFG), with probabilities trained in the Forward/Backward manner. Wright and Wrigley (Wright and Wrigley, 1991) formalized a method of mapping PCFGs onto LR parsing tables (LR tables, for short) by way of distributing the probabilities originally associated with a given CFG to each corresponding LR parsing action. As a result, the parser can incrementally compute the probability of each parse. Nevertheless, under Wright and Wrigley's model the resultant probability of a parse is identical to that acquired from the original PCFG, despite the process of generating the LR table being greatly complicated.

Briscoe and Carroll (Briscoe and Carroll, 1993) proposed a simpler way of incorporating trained probabilities into each parsing action of the LR table. Probabilities are computed directly from the frequency of application of each action when parsing the training corpus. Their method seems to be able to exploit the advantages offered by the context-sensitivity of GLR parsing. GLR parsing is, indeed, context-sensitive in that reduce actions depend on the state and lookahead symbol. But, without the obvious formalization of their model, there is doubt as to the validity of their method of including the left context for reduce actions in an attempt to increase the accuracy of computation of parse probability.

In contrast to the former models proposed by Wright and Wrigley, and Briscoe and Carroll, our PGLR model is formalized according to stack transition during the parsing process. Parse probability is decomposed into a sequence of actions. Theoretically, our model requires only one probability for each action. Therefore, we can easily train our model by computing the frequency of application of each action when parsing the training corpus (correctly hand-annotated corpus), and directly associate a probability with each action in the LR table. The results of our experiments clearly show that our model outperforms the other two in all cases, and significantly reduces the per-word cross entropy of the task compared with the baseline model of PCFG.

Section 2 briefly reviews the probabilistic models proposed on GLR parsing, namely PCFG model, the model proposed by Briscoe and Carroll, and our PGLR model. We then clarify the context-sensitive nature of GLR parsing, and point out the dubious nature of Briscoe and Carroll's model in Section 3. Section 4 describes how a probability is allocated to each action of the table, through a simple example. Section 5 shows the results of experiments made on the three models.

2 Probabilistic Models in GLR Parsing

GLR parsing algorithm is associated with effective devices for handling nondeterminism in parsing, namely a *graph-structured stack*, and *packed shared parse forest* for representing numerous parse derivations in a space-efficient manner (Tomita and Ng, 1991). Inheriting the efficiency of GLR parsing is one of the common purposes in integrating probability into the parser. In this section, we briefly describe the existing probabilistic models, together with our newly proposed PGLR model.

Probabilistic Context-Free Grammar (PCFG) A probabilistic context-free grammar (PCFG) is an augmentation of a context-free grammar. Each production rule of the grammar (r_i) is of the form $< A \rightarrow \alpha, P(r_i) >$ where $P(r_i)$ is the associated probability, and the probabilities associated with all rules with a given nonterminal on the LHS must sum to one (Fujisaki et al., 1989; Wright and Wrigley, 1991). The probability of a parse derivation (T) is regarded as the product of probabilities of the rules which are employed for deriving that parse derivation, such that,

$$\sum_{\alpha} P(\alpha|A) = 1 \quad (1)$$

$$P(T) = \prod_{i} P(r_i) \quad (2)$$

Wright and Wrigley proposed a solution of integrating PCFG into the GLR parsing scheme (Wright and Wrigley, 1991). They distributed the original probabilities of the PCFG into the parsing action table by way of the parser generator. The successful application of their probabilistic LR table generator made a contribution in making word prediction available in the speech recognition process. With probabilities associated with shift actions as well as reduce actions, the parser can compute not only the total probability of a parse derivation but also the intermediate probability during the parse process. However, the resultant probability is identical to the original PCFG, which does not sufficiently capture the context-sensitivity of NL.

Briscoe and Carroll's Model (B&C) (Briscoe and Carroll, 1993) introduced probability to the GLR parsing algorithm in the light of the fact that LR tables do provide appropriate contextual information for solving the context-sensitivity problems observable in real world NL applications. They pointed out that an LR parse state encodes information about the left and right context of the parse. This results in distinguishability of context for an identical rule reapplied in different ways across different derivations. Briscoe and Carroll's method allows us to associate probabilities with an LR table directly, rather than simply with the rules of the grammar. They considered the LR table as a nondeterministic finite-state automaton. Each row of the LR table corresponds to the possible transitions out of the state represented by that row, and each transition is associated with a particular lookahead item[1] and a parsing action. Nondeterminism arises when more than one action is possible given a particular input symbol.

Briscoe and Carroll regard a parse derivation as a sequence of state transitions (T):-

$$s_0 \overset{l_1,a_1}{\Longrightarrow} s_1 \overset{l_2,a_2}{\Longrightarrow} \ldots \overset{l_{n-1},a_{n-1}}{\Longrightarrow} s_{n-1} \overset{l_n,a_n}{\Longrightarrow} s_n \quad (3)$$

where a_i is an action, l_i is an input symbol and s_i is the state at time t_i. The probability of the parse derivation T is estimated by equation (4):-

$$\begin{aligned} P(T) &\approx \prod_{i=1}^{n} P(l_i, a_i, s_i|s_{i-1}) \\ &= \prod_{i=1}^{n} P(l_i, a_i|s_{i-1}) \cdot P(s_i|s_{i-1}, l_i, a_i) \end{aligned} \quad (4)$$

Based on Briscoe and Carroll's model, the following is a summary of the scheme for deriving the action probabilities from the count of state transitions resulting from parsing a training set.

1. The probability of an action given an input symbol is conditioned by the state it originated from. The probabilities assigned to each action at a given state must sum to one. Therefore, the probability of an action ($p(a)$) is:-

$$p(a) = P(l_i, a_i|s_{i-1}) \quad (5)$$

such that:

$$\sum_{l \in La(s)} \sum_{a \in Act(s,l)} p(a) = 1 \quad (\text{for } \forall s \in S) \quad (6)$$

where $La(s)$ is the set of input symbols at state s, $Act(s, l)$ is the set of actions given a pair of state s and input symbol l, and S is the set of all states of the LR table.

2. In the case of a reduce action, the probability is subdivided according to the state reached after applying the reduce action. The reason for this is that Briscoe and Carroll considered associating probabilities with transitions in the automaton rather than with actions in the action part of the LR table.

The probability of a parse derivation is the product of the probabilities of the actions for state transitions through the whole parse derivation:-

$$P(T) = \prod_{i=1}^{n} p(a_i) \quad (7)$$

[1] The term "lookahead" originally used in (Aho et al., 1986) refers to the extra information that is incorporated into the state by redefining items to include a terminal symbol as a second component. In this case, however, Briscoe and Carroll refer to an "input symbol".

Probabilistic Generalized LR (PGLR) (Inui et al., 1997) have recently proposed a new formalization of a probabilistic model for GLR parsing. Unlike B&C, a parse derivation is regarded as a sequence of transitions between LR parse stacks (T) as shown in equation (8), where σ_i is the stack at time t_i, a_i is an action and l_i is an input symbol. Equations (3) and (8) show the inherent difference in the definition of parse derivation used in the two models.

$$\sigma_0 \stackrel{l_1,a_1}{\Longrightarrow} \sigma_1 \stackrel{l_2,a_2}{\Longrightarrow} \cdots \stackrel{l_{n-1},a_{n-1}}{\Longrightarrow} \sigma_{n-1} \stackrel{l_n,a_n}{\Longrightarrow} \sigma_n \qquad (8)$$

Based on the above definition, the probability of a complete stack transition sequence T can be represented with equation (9), by assuming that σ_i contains all the information of its preceding parse derivation:-

$$\begin{aligned} P(T) &= \prod_{i=1}^{n} P(l_i, a_i, \sigma_i | \sigma_{i-1}) \\ &= \prod_{i=1}^{n} P(l_i, a_i | \sigma_{i-1}) \cdot P(\sigma_i | \sigma_{i-1}, l_i, a_i) \end{aligned} \qquad (9)$$

Due to the two different types of actions in the LR table, namely *shift* and *reduce* actions (*accept* is an additional special dummy action to successfully end the parsing process), states are treated differently according to the type of action applied to reach that state. States reached after the application of a reduce action occupy the same input symbol as in the former state, whereas states reached after the application of a shift action read in a new input symbol. As a result, we classify states into two classes and estimate the probabilities of actions differently, corresponding to the class they belong to. By assuming that the stack-top state contains sufficient information of the current stack, we can estimate the probability of the current action a_i from the state s_i on top of the current stack, instead of the full stack σ_i. Therefore,

$$P(l_i, a_i, \sigma_i | \sigma_{i-1}) \approx \begin{cases} P(l_i, a_i | s_{i-1}) & (s_{i-1} \in S_s) \\ P(a_i | s_{i-1}, l_i) & (s_{i-1} \in S_r) \end{cases} \qquad (10)$$

such that:

$$\sum_{l \in La(s)} \sum_{a \in Act(s,l)} p(a) = 1 \quad (\text{for } s \in S_s) \qquad (11)$$

$$\sum_{a \in Act(s,l)} p(a) = 1 \quad (\text{for } s \in S_r) \qquad (12)$$

where $P(a)$ is the probability of an action, S_s is the class of states reached after applying a shift action, including the initial state, and Sr is the class of states reached after applying a reduce action.

3 Context-Sensitivity in GLR Parsing

GLR parsers (Tomita and Ng, 1991) are driven by a precompiled LR table, generated from a context-free grammar. Though the grammar used in generating the table is context-free, the nature of the table and the manner in which the GLR parser is driven, make the parser mildly context sensitive. Briscoe and Carroll raised this issue in (Briscoe and Carroll, 1993) but misinterpreted some aspects of the context sensitivity, causing the number of free parameters to increase unnecessarily and resulting in on unexpectedly complicated probabilistic model. We will come back to this point after describing the nature of context-sensitivity of GLR parsing.

In the generation of an LR table, each state in the table is generated by applying the *goto* function, (as described in (Aho et al., 1986)), to the previous state ($s_i = goto[s_{i-1}, X_i]$). Each new state (s_i) is generated by consulting the previous state (s_{i-1}) and a grammar symbol (X_i), where X_i is a terminal symbol in the case of the next input symbol being incorporated onto the stack, or a non-terminal symbol when the stack is reduced by way of an appropriate reduction rule. Each state in the LR table thus contains a local *left context* for the parser.

On the other hand, at parse time, actions in the LR table are determined by the pair of a state and an input symbol ($a_{i+1} = action[s_i, l_{i+1}]$). This means that at time t_i, when the parser has come to a state (s_i), the parser will consider the next input symbol (l_{i+1}) in determining the next action (a_{i+1}). The next input symbol here provides the parser with a *right context* to aid in the determination process. Basically, the context taken into account during parsing is limited to one viable state and one input symbol.

In parsing natural language, ambiguity inevitably occurs for a fixed state and input symbol. Based on GLR parsing, ambiguity occurs when there is more than one action corresponding to the given pair of state and input symbol. Two cases exist for potential action conflicts: reduce/reduce conflicts (Figure 1) and shift/reduce conflicts (Figures 2 and 3). Due to the properties of LR tables, shift/shift conflicts never occur.

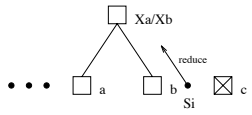

Figure 1: Reduce/reduce conflict

Let us consider the case of parsing with a grammar which constructs a binary tree. Reduce/reduce conflicts represent the dilemma of deciding which non-terminal label should be associated with the structure (for instance, X_a or X_b in Figure 1), and shift/reduce conflicts constitute the problem of deciding whether to incorporate the next input symbol (d) onto the stack and delay construction of hierarchical structure to the next step (see Figure 2) or to create a structure based on the previous stack (see Figure 3). In the case of parsing with PCFG, the most we can do is to assign a probability to each rule, based on the premise that the probabilities for all the rules that expand a common non-terminal must sum to one. Disambiguation in the case of Figure 1 is then made based on the comparison between the probabilities for $[X_a \rightarrow a\ b]$

and $[X_b \rightarrow a\ b]$. This same methodology is also used to disambiguate between Figures 2 and 3.

For both conflict types, the GLR parser at least provides the left/right context to distinguish probabilities for reducing to the same non-terminal. The overall probability for reducing to X_a may be higher than to X_b but in some context, such as Figure 1, the probability for reducing to X_b can be higher than for X_a. The GLR parser provides the context of the state number (s_i) and input symbol (c) to determine parse preference. Similarly, the context of state number (s_i) and input symbol (d) in Figures 2 and 3 can be used to give preference to either a shift or reduce action.

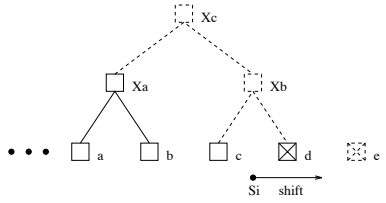

Figure 2: Shift preference for a shift/reduce conflict

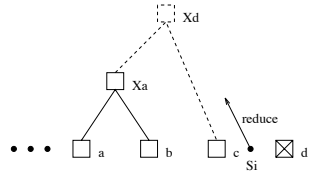

Figure 3: Reduce preference for a shift/reduce conflict

Briscoe and Carroll have pointed out some examples of NL phenomena that a GLR parser can inherently handle (Briscoe and Carroll, 1993). In the example of **he** *loves* **her**, a GLR parser can distinguish between the contexts for reducing the first pronoun and the second pronoun to NP, given that the next input symbol after **he** is *loves*, while that for **her** is the sentence end marker ($). However this does not work if the next input symbols are the same, such as the reduction of pronouns in the examples of **he** *passionately loves her* and *he loves* **her** *passionately*. As previously mentioned, Briscoe and Carroll proposed an approach of subdividing reduce actions according to the state reached after that reduce action is applied. The purpose of this approach is to distinguish between reductions using the left context of the reduction rule.

Subdividing reduce actions according to the state reached after the reduce action is one of the factors that leads to Briscoe and Carroll's model being unexpectedly complicated and including an unnecessarily large number of free parameters.

As we have described above regarding the left context of the GLR parsing scheme, every state is generated by consulting the previous state and a grammar symbol. The states contain some local context, with the degree of the context depending directly on the type of the table, namely SLR, LALR or CLR (see (Aho et al., 1986)). Furthermore, the states reached after reduce actions (for instance, s_i and s_x in Figure 4) are determined deterministically. This is accounted for in our probabilistic modeling in Section 2.

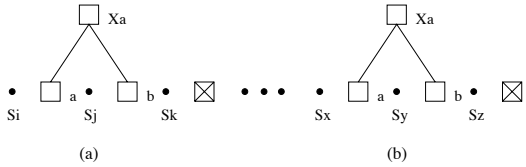

Figure 4: Reduction of $[X_a \rightarrow a\ b]$ in different contexts

The context sensitivity when parsing with either an LALR or CLR table is different, because during the process of generating an LALR table, states are merged together if they fulfill the requirement of have the same *core* in the LR item (Aho et al., 1986). As a result, the number of states in an LALR table is drastically decreased when compared a CLR table. Therefore, the left context contained in states in an LALR table is less than that for states in a CLR table. Despite this, however, the results of an experiment presented in Section 5 confirm that parsing with an LALR table does not significantly decrease accuracy.

4 Incorporating Probability into an LR Parsing Table

Our model described in Section 2 normalizes the probability differently depending on the class of the state, either S_s or S_r. States are distinguishable because of the property that the state classes for an LR table are mutually exclusive. This is because any state can be reached by only one type of grammar symbol. The states in the S_r class are those states referenced by transitions in the *goto* part of the table, and all other states are in the S_s class.

Suppose that we have a simple English context-free grammar as shown in Table 1, with LR table as shown in Table 2. We train our GLR parser in the supervised mode, using a correctly hand-annotated corpus to guide the parser in its extraction of the sequence of actions required to produce the correct parse. We sum the frequency of application of each action, and add a part of a count to each action that appears in the table, to smooth the probability for unobserved events. Finally, each action probability is computed according to the state class which the action belongs to.

Table 3 shows the artificial probability associated with each action. It is noticeable that the probabilities of actions in states in the S_s class (states 0, 1, 2, 4, 5, and 6) sum to one, but for states in the S_r class (states 3, 7, 8, 9, 10, 11 and 12), the probabilities of the actions in the slot of state and the input symbol sum to one. Table 3 does not show the goto part because actions in the goto part are deterministic and their probabilities are always one.

5 Experiments

We tested our model (PGLR) on two Japanese corpora. These two corpora were different in both their sources and

Table 1: A simple English context-free grammar

(1)	S	→	NP	VP
(2)	NP	→	det	n
(3)	NP	→	n	
(4)	NP	→	NP	PP
(5)	PP	→	p	NP
(6)	VP	→	v	NP
(7)	VP	→	VP	PP

Table 2: LR table for the English grammar in Table 1

State	Action					Goto			
	det	n	p	v	$	NP	PP	S	VP
0	s2	s1				3		12	
1			r3	r3	r3				
2		s4							
3			s5	s6			7		8
4			r2	r2	r2				
5	s2	s1				9			
6	s2	s1				10			
7			r4	r4	r4				
8			s5		r1		11		
9			r5/s5	r5	r5		7		
10			r6/s5		r6		7		
11			r7		r7				
12					acc				

Table 3: LR table with its associated probabilities

State	Action				
	det	n	p	v	$
0	s2	s1			
	0.6	0.4			
1			r3	r3	r3
			0.4	0.5	0.1
2		s4			
		1			
3			s5	s6	
			1	1	
4			r2	r2	r2
			0.3	0.6	0.1
5	s2	s1			
	0.7	0.3			
6	s2	s1			
	0.7	0.3			
7			r4	r4	r4
			1	1	1
8			s5		r1
			1		1
9			r5/s5	r5	r5
			0.7/0.3	1	1
10			r6/s5		r6
			0.4/0.6		1
11			r7		r7
			1		1
12					acc
					1

their licensed context-free grammar, but about the same size. Our experiments are designed not to be biased towards our model in terms of the degree of complexity of the task. For comparison of the performance of the different models, we conducted tests with the PCFG model as the baseline, and also with Briscoe and Carroll's model (B&C).

5.1 ATR Corpus and Grammar

The ATR corpus is a tree bank (a collection of trees annotated with a syntactic analysis, or "trees" for short) of Japanese dialogue. We randomly selected about 5% of the corpus to use as the test set and trained each parsing model with the remaining approximately 20,000 trees (see Table 4). The smallest sentence unit is a Japanese morpheme, and the length shown in the Table 4 is the number of morphemes. All trees are licensed by the Japanese phrase structure grammar developed at ATR. We generated an LALR table from the grammar of 360 production rules, comprised of 67 non-terminal symbols and 41 terminal symbols. 376 states were generated in the LALR table.

Table 4: ATR Corpus

		# of Morphemes	
ATR Corpus	*# of Sent.*	*Ave.*	*Range*
Training set	19,586	12.08	2-60
Test set	995	11.64	2-30

5.2 Parsing the ATR Corpus

In the test, we used the test set to generate a sequence of parts-of-speech for each sentence, to act as the input for each model. The evaluation data for each model is shown as the percentage of ranked candidate parses containing an exact match to the standard parse. Candidate parses were ranked based on the value of the parse probability. Table 5 shows the percentage of the exact matches in 4 classes. "Exact-1" is the percentage of most probable candidate parses that matched the standard parse. "Exact-5" is the percentage of parse outputs containing an exact matched parse ranked within top 5 parses and so on. To make sure that the models can rank their output parses accurately, we count the rank of the lowest parse for parses of the same probability.

Table 5: Performance on the ATR Corpus

Rank	*PCFG*	*B&C*	*PGLR*
Exact-1	69.35%	74.67%	83.22%
Exact-5	91.56%	89.05%	95.48%
Exact-10	94.97%	92.26%	97.49%
Exact-20	96.78%	95.08%	98.59%
Cross Entropy	2.72	N/A	2.41

Our model (PGLR) outperformed the other two models in every ranking class, and the per-word cross entropy is also less than that for the PCFG model. The per-word cross entropy of Briscoe and Carroll's model (B&C) is out of range because of the large number of free parameters used to re-predict the next input symbol after applying a reduce action. Our model returning the highest parsing accuracies and the lowest per-word cross entropy, clearly shows that it can efficiently use the context information encoded in the GLR parsing scheme.

5.3 EDR Corpus and Grammar

In a similar way, we prepared another testing environment using the EDR corpus, an entirely different type of corpus. The EDR corpus is a collection of Japanese documents extracted from various newspaper sources, with the size of the corpus given in Table 6. From the corpus, we selected data which could be constructed by a set of binary production rules, for the purpose of testing the distinguishability of the phrase dependency of each model.

Table 6: EDR Corpus

		# of Bunsetsu	
EDR Corpus	*# of Sent.*	*Ave.*	*Range*
Training set	21,602	8.24	2-26
Test set	993	8.11	3-20

The grammar is a set of production rules modeling inter-phrasal structure, consisting of 1,132 binary rules. The smallest sentence unit is the case-marked phrase (*bunsetsu*). From the grammar, we generated an LALR table consisting of 67 non-terminal symbols, 99 terminal symbols and 1,608 states.

5.4 Parsing the EDR Corpus

Though the complexity of the sentence structure increased for the EDR corpus and most parts of the structure were constructed differently depending on the context, our model still outperformed the other two models, (see Table 7). The complexity of the task can be ascertained from the per-word cross entropy, which is higher for the EDR test set than for the ATR test set.

Table 7: Performance on the EDR Corpus

Rank	*PCFG*	*B&C*	*PGLR*
Exact-1	41.39%	50.35%	60.22%
Exact-5	78.85%	80.87%	89.73%
Exact-10	86.51%	89.43%	94.86%
Exact-20	91.84%	92.95%	96.68%
Cross Entropy	2.94	N/A	2.80

5.5 Additional Experiment with CLR Table

We extended our experiment on the ATR corpus to use a CLR table, which is more distinctive in state assignment. The experiment was repeated for both Briscoe and Carroll's model and our PGLR model (note that the result for the PCFG model does not change for different table types). The CLR table occupied 814 states, more than twice the number of states in the corresponding LALR table. The overall performance using the CLR table was slightly higher than that for the LALR table, and the per-word entropy for the CLR table was also slightly less than that for the LALR table (see Table 8). States in a CLR table are more distinguishable than those in an LALR table, but the drastic increase in states for the CLR table causes data sparseness in training. Thus, whereas we would expect the predictability for a CLR table to be higher than that for an LALR table, LALR tables perform similarly to CLR tables given the same size of training data. Considering the cost of table generation and parsing time, using an LALR table is much more efficient.

6 Conclusion

We formalized a probabilistic model for GLR parsing and applied the method to the construction of a probabilistic LR table. The results of experiments clearly showed that our model (PGLR) is able to make effective use of both left and right context information within the GLR parsing scheme. As a result, our model outperformed both Briscoe and Carroll's model and the PCFG model in all tests. In addition, our model needs only the probability for each action in the LR table to compute the overall probability of each parse. It is thus tractable to training with the smallest amount of free parameters, and associates a probability directly to each action. Since the parse probability is incrementally calculated from action probabilities, the parser can compute the partial parse probability at any stage of the parse. We plan to extend the PGLR model to the parsing of sentences including unknown words, using the predictability of the model. The PGLR model is also expected to provide a means for recovering from ill-formed input sentences.

Table 8: Performance on the ATR Corpus

	B&C		*PGLR*	
Rank	*LALR*	*CLR*	*LALR*	*CLR*
Exact-1	74.67%	74.87%	83.22%	83.82%
Exact-5	89.05%	87.24%	95.48%	95.78%
Exact-10	92.26%	91.36%	97.49%	97.49%
Exact-20	95.08%	94.47%	98.59%	98.39%
Cross Entropy	N/A	N/A	2.41	2.32

7 Acknowledgments

We would like to thank Masahiro Ueki for support in using MSLR and kindly adapting his parser to be able to accumulate action counts during the training phase. Taiichi Hashimoto helped us to improve the probabilistic parser, and Toshiki Ayabe helped with LR table generation, especially in computing the state list for use with the B&C model. In addition, Thanaruk Theeramunkong of JAIST provided helpful discussions on accumulating probabilities into a GLR packed shared parse forest.

References

Aho, A., Sethi, R. and Ullman, J. 1986. *Compilers: Principles, Techniques, and Tools.* Addison-Wesley.

Briscoe, T. and Carroll, J. 1993. Generalized Probabilistic LR Parsing of Natural Language (Corpora) with Unification-Based Grammars. *Computational Linguistics*, Vol.19, No.1, pages 25-59.

Fujisaki, T., Jelinek, F., Cocke, J., Black, E. and Nishino, T. 1989. A Probabilistic Parsing Method for Sentence Disambiguation. *Proceedings of 1st International Workshop on Parsing Technologies*, Carnegie-Mellon University, Pittsburgh, PA, pages 85-94.

Inui, K., Sornlertlamvanich, V., Tanaka, H. and Tokunaga, T. 1997. A New Formalization of Probabilistic GLR Parsing. *Proceedings of the 5th International Workshop on Parsing Technologies.*

Tomita, M. and Ng, S-K 1991. The Generalized LR Parsing Algorithm. *Generalized LR Parsing*, edited by Tomita, M., Kluwer Academic Publishers, pages 1-16.

Wright, J. H. and Wrigley, E. N. 1991. GLR Parsing with Probability. *Generalized LR Parsing*, edited by Tomita, M., Kluwer Academic Publishers, pages 113-128.