社団法人 電子情報通信学会
THE INSTITUTE OF ELECTRONICS.
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE.
SP94-63, NLC94-32(1994-12)

音素環境依存 LR テーブル作成法とその連続音声認識システムへの応用

李 輝
田中 穂積

竹沢 寿幸
シンガー ハラルド
林 輝昭

東京工業大学情報工学科
〒 152 東京都目黒区大岡山 2-12-1

ATR 音声翻訳通信研究所
〒 619-02 京都府相楽郡精華町光台 2-2

li@cs.titech.ac.jp

あらまし

本論文では，音素環境依存 GLR パージング実現方法及び連続音声認識システムへの応用について述べる．異音接続表に基づいた制約伝播法による，音素環境依存 LR 表の新しい生成アルゴリズムを提案する．

日本語文節音声認識を対象として，認識性能の評価実験を行なった．さらに，SLR と canonical LR による，異音に基づく音声認識システムの性能の比較実験を行なった．

和文キーワード 異音モデル，一般化 LR 法，制約伝播，音声認識

A Method for Generating Phoneme-Context-Dependent LR Table and its Applications in Continuous Speech Recognition System

Hui Li
Hozumi Tanaka

Toshiyuki Takezawa
Harald Singer
Teruaki Hayashi

Department of Computer Science
Tokyo Institute of Technology
2-12-1 Ookayama Meguro-ku Tokyo 152

ATR Interpreting Telecommunications
Research Laboratories
2-2 Hikaridai, Seika-cho, Soraku-gun, Kyoto 619-02

Abstract

This paper describes a method of realizing phoneme-context-dependent GLR parsing and its applications in a continuous speech recognition system. A new algorithm for generating a phoneme-context-dependent (allophonic) LR table on the basis of an allophone connection matrix through constraints propagation is given.

For a Japanese phrase recognition task, the evaluations of recognition performance is given. The performance of SLR and canonical LR(CLR) table for allophone-based speech recognition system is also discussed.

英文 key words allophone model, GLR parser, constraints propagation, speech recognition

1 Introduction

In the phone-based speech recognition system, a GLR parser has been successfully employed as a phoneme predictor, which provides efficient search of phones during the process of speech recognition[3]. A GLR parser is guided by an LR parsing table automatically created from a set of context-free grammar(CFG) rules, and proceeds left-to-right without backtracking.

On the other hand, in continuous speech recognition, the performance of recognition systems has been improved by using allophones as recognition units instead of phones[2][4][7][8]. Allophone models (such as triphone models) are context-dependent phone models that take into consideration the left and right neighboring phones to model the major coarticulatory effects in continuous speech.

The combination of allophone models and a GLR parser is desirable to achieve better performance in continuous speech recognition. One of the difficulties of integrating GLR parser with an allophone-based recognition system is how to solve the word juncture problem, because for the phones at word boundaries, their left or right context depends on the preceding or succeeding words.

There has been some approaches of realizing a phoneme-context-dependent GLR parsing[2][6][7]. To solve the word juncture problem, these methods either need dynamic processing during the speech recognition process, or have the problem of explosion in LR table size.

In this paper, we will describe a method, called CPM (constraints propagation method), for generating an allophone-based LR table which can solve the word juncture problem effectively.

The organization of this paper is as follows: Section 2 describes the method for generating an allophone-based LR table. Section 3 discusses the two different type of LR table (SLR table and canonical LR table) applied to an allophone-based speech recognition system. Section 4 describes some improvements of CPM method. Section 5 presents the GLR parsing algorithm based on the allophone-based LR table generated by CPM. Section 6 provides some experiment results for a Japanese phrase recognition task. Section 7 concludes with discussions.

2 Algorithm for generating an allophone-based LR table

In this algorithm, by introducing a set of allophone rules into a set of syntactic and lexical rules(CFG), an allophone-based LR table is generated first, then this LR table is modified by applying CPM.

We use a simple Japanese grammar with lexical rules shown in Fig. 1 to illustrate the CPM algorithm of generating allophone-based LR table.

(1) S → N P
(2) N → a n i(brother)
(3) N → a n e(sister)
(4) P → g a
(5) P → n i

Fig. 1 A simple Japanese grammar with lexical rules

2.1 Allophone connection matrix

The allophone connection matrix provides the connectability between two adjacent allophones.

For example, assume that the context sets of allophones "i1" for phone "i" and "g2" for phone "g" are as follows.

$$\mathbf{i1}: \left\{ \begin{array}{c} \vdots \\ n \\ \vdots \end{array} \right\} \mathrm{i} \left\{ \begin{array}{c} \vdots \\ g \\ \vdots \end{array} \right\}, \quad \mathbf{g2}: \left\{ \begin{array}{c} \vdots \\ i \\ \vdots \end{array} \right\} \mathrm{g} \left\{ \begin{array}{c} a \\ \vdots \\ \vdots \end{array} \right\}$$

We say "g2" can follow "i1" because the right context of "i1" contains the phone "g" and the left context of "g2" contains the phone "i". A connection matrix is expressed as an array of *Connect*[left_allophone, right_allophone], whose value is "1" (two allophones are connectable) or "0" (two allophones are not connectable).

Fig. 2 shows an example of allophone connection matrix.

		RIGHT										
		a1	a2	n1	n2	i1	i2	e1	e2	g1	g2	$
LEFT	a1			1	1							0
	a2			0	0							1
	n1					1	0	0	0			
	n2					1	0	1	1			
	i1			1	0					1	1	1
	i2			1	0					0	0	1
	e1			1	1					1	0	
	e2			0	0					1	0	
	g1	0	0									
	g2	1	1									

Fig. 2 An example of connection matrix

2.2 Initial allophone-based LR table

At first, for the lexical rules, we change the phones within a word into the allophones according to the allophone contexts. In Fig. 3, rule (2)' to (5)' corresponds to the converged lexical rules of rule (2) to (5) in Fig. 1.

(1) S → N P
(2)' N → a n1 i
(3)' N → a n2 e
(4)' P → g a
(5)' P → n i
(6) a → a1
(7) a → a2
(8) i → i1
(9) i → i2
(10) e → e1
(11) e → e2
(12) g → g1
(13) g → g2
(14) n → n1
(15) n → n2

Fig. 3 A set of syntactic, lexical and allophonic rules

To solve the word juncture problem, the allophone rules is introduced into the set of syntactic and lexical rules(CFG). Assume the following: {a1, a2} for "a", {i1, i2} for "i", {e1, e2} for "e", {g1, g2} for "g", {n1, n2} for "n", a set of allophone rules can be produced. In Fig. 3, rules (6) to (15) are allophone rules.

From the rule set in Fig. 3, an allophone-based canonical LR table shown in Fig. 4 is generated.

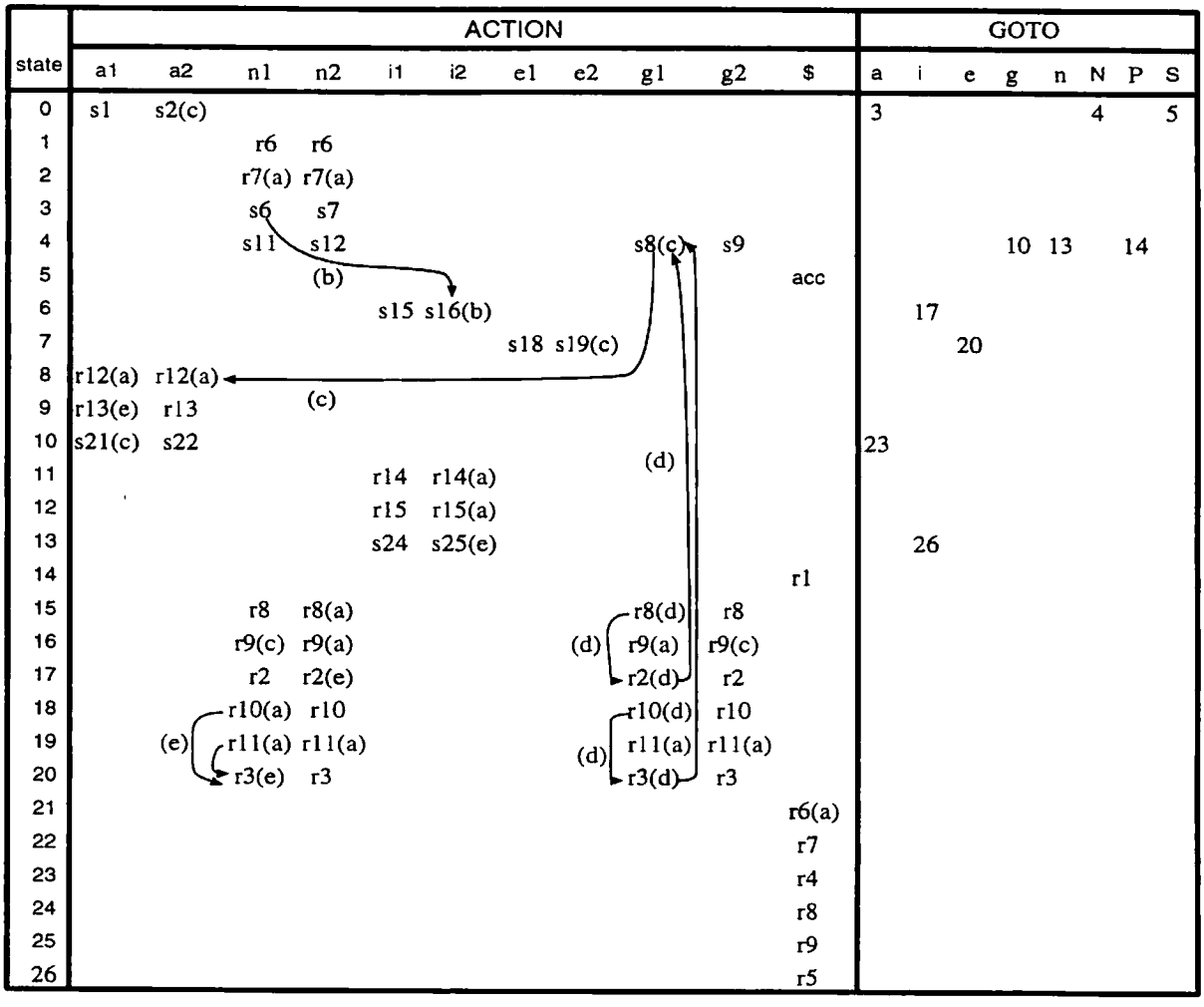


Fig. 4 An allophone-based LR table from the rules in Fig. 3

2.3 Modifications of LR table

In the LR table in Fig. 4, the connectability information represented in Fig. 2 is not included for the phones at word boundaries. For example, for the beginning phone "a" of word "a n1 i", the action "re7" with lookahead symbol "n1" in state 2 allows the connection between "a2" and "n1", but this connection is illegal since *Connect*[a2,h1]=0" as shown in Fig. 2. In order to incorporate the connection constraints into the LR table, we modify the LR table on the basis of allophone connection matrix through constraints propagation.

2.3.1 Connection check

At first, we use connection matrix to remove the illegal actions in LR table in Fig. 4 in a similar way as in [9].

(a). Remove the illegal reduce actions of allophone rules

For example, "re14"(lookahead "i2") in state 11 should be removed because "n1" (RHS of rule 14) and "i2" are not connectable(*Connect*[n1,i2] = 0).

(b). Remove the illegal shift actions whose predecessors are shift actions

For example, consider "sh16"(lookahead "i2") in state 6, which is transferred from "sh6"(lookahead "n1") in state 3, since "n1" and "i2" are not connectable, "sh16" in state 6 should be removed.

2.3.2 Constraints propagation

Secondly, we remove all other illegal actions in Fig. 4 through constraints propagation.

(c). Remove the empty states and the shift actions that lead to the empty states

An empty state is defined as a state whose all actions have already been removed, or all the preceding actions of this state have been removed. The shift actions that lead to the empty state should be removed. For example, state 8 is an empty state, so "sh8"(lookahead "g1") in state 4 should be removed.

(d). Remove the reduce actions that lead to removed shift actions

Consider "re8"(lookahead "g1") in state 15, the

parser will transfer to state 17 after "re8". In state 17, the next action is "re2"(lookahead "g1"), after "re2", the parser will transfer to state 4, but in this state, "sh8"(lookahead "g1") has already been removed by the step (c). Thus, "re8" with lookahead "g1" (in state 15) and "re2" with lookahead "g1" (in state 17) should be removed too.

(e). Remove the actions whose all predecessors are removed actions

Consider "re3" (lookahead "n1") in state 20, this action is transferred from a goto action after "re10" (lookahead "n1") in state 18 or a goto action after "re11" (lookahead "n1") in state 19 . Since "re10" in state 18 and "re11" in state 19 (lookahead "n1") have been removed by step(a), re3 (lookahead "n1") in state 20 should be removed.

The algorithm will repeat this constraints propagation (step(c)-(e)) until no more actions are removed, and then compress the LR table to reduce the table size.

In Fig. 4, the actions that were marked (a), (b), (c), (d), (e) mean they were removed by step(a), (b), (c), (d) and (e), respectively.

3 Incorporation of connection constraints into the generation process of LR table

As explained above, we introduced the allophone rules into a set of syntactic and lexical rules and generated an initial allophone-based LR table first, then removed all the illegal actions and states on the basis of allophone connection matrix through constraints propagation. Although this method has the advantage of enabling us using already existing LR table generation method to get the initial LR table, the states and actions of initial LR table often explode as the number of CFG rules and allophone models increases.

In order to rectify this situation, we can incorporate the connection constraints(step(a) and (b) in Section 2.3.1) into the LR table generation process by removing the illegal items that violate connection constraints for each set of items.

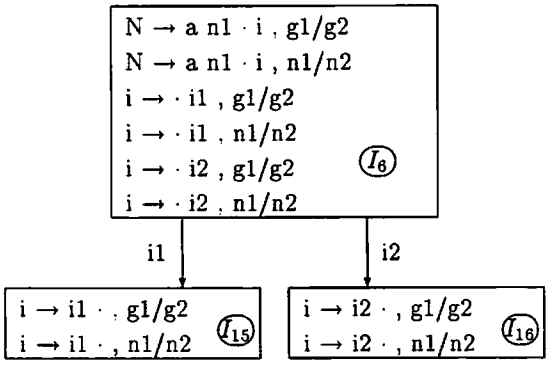


Fig. 5 Part sets of items of LR table in Fig 4

For example, consider the sets of items I_6, I_{15} and I_{16}(Fig. 5), which corresponds to the state 6, 15 and 16 of LR table in Fig. 4, respectively.

- The following items of I_6

  i → · i2, g1/g2
  i → · i2, n1/n2

  are due to the closure "N → a n1 · i", but according to the connection matrix in Fig. 2, $Connect[n1, i2] = 0$, so the above four items should be be removed, which corresponds to step(b) of CPM. Since the above four items were removed, the set of items I_{16} will vanish automatically.

- According to Fig. 2, $Connect$[i1, n2] = 0, so the following item of I_{15}

  i→ i1 · , n2

  can be removed, this corresponds to step(a) of CPM.

By the above two steps, three sets of items in Fig. 5 will decrease to two sets of items as shown in Fig. 6.

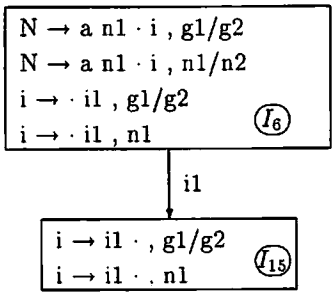


Fig. 6 Item sets after connection check

The size of initial allophone-based LR table can be reduced greatly through this processing.

4 SLR and Canonical LR Table

In this section, we would like to discuss the types of LR tables. Until now all methods about phoneme-context-dependent GLR parsing[2][6][7] have used the SLR or LALR table.

Compared with the canonical LR table, the SLR and LALR table can not provide us precise phoneme predictions because the SLR and LALR table have fewer states due to merging several states in an LR table [1], and merging several states brings many actions in a state that produces many predictions.

Consider the simple grammar shown in Fig. 7.

(1) <s> → <koko> <ni> <hon> <ga> <aru>
(2) <s> → <hon> <da>
(3) <koko> → k o k o
(4) <hon> → h o N
(5) <ni> → n i
(6) <ga> → g a
(7) <aru> → a r u
(8) <desu> → d a

Fig. 7 An example grammar with lexical rules

According to this grammar, there are only two correct sentences "h o N / d a (本だ)" and "k o k o / n i/ h o N /g a/ a r u (ここに本がある)".

state	ACTION						GOTO	
	k	h	o	N	g	d	<koko>	<hon>
0	s1	s3					2	4
3			s9					
8		s3						14
9				s16				
15					r4	r4		

Fig. 8 Part of SLR table from Fig 7

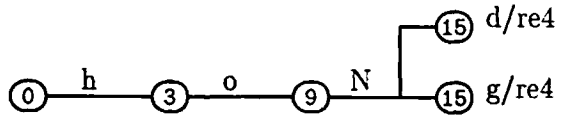
state	ACTION						GOTO	
	k	h	o	N	g	d	<koko>	<hon>
0	s1	s3					2	4
3			s9					
8		s14						15
9				s16				
14			s19					
16						r4		
19				s23				
23					r4			

Fig. 9 Part of CLR table from Fig 7

The part of SLR table and canonical LR table generated from Fig. 7 are shown in Fig. 8 and Fig. 9, respectively.

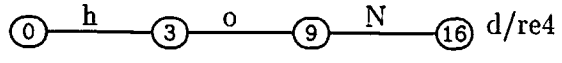
In this example, the SLR table merged two states (state 16 and state 23 in Fig. 9) of the canonical LR table into one state (state 15 in Fig. 8). This merge will bring useless phone predictions in speech recognition.

For example, if the correct sentence is "h o N / d a", using the SLR table, we have a path:



In state 15, an useless prediction "g" occurred.

But if using the CLR table, the path is:



The useless prediction "g" did not occur.

In particular, for the phoneme-context-dependent LR parser, this useless prediction may affect the context of the preceding and succeeding phones. In the above example, for "N" of "h o N / d a", using SLR table, we will get two triplets "o/N/d" and "o/N/g", where "o/N/g" is an useless triplet, but using CLR table, "o/N/g" does not occur.

Generally, the canonical LR table has more states than the SLR or LALR table.

To compare the size of LR table, two ATR phrase grammar shown in Table 1 have been used.

Table 1: Size and perplexity of grammar

Grammar	mset_phrase	eset_phrase
Rules	979	2813
Vocabulary	456	1588
Perplexity(phone)	2.66	3.43

Table 2 shows a comparison of table size between SLR and CLR table.

Table 2: Table size of SLR and CLR table

Grammar	mset_phrase		eset_phrase	
Table type	SLR	CLR	SLR	CLR
states	2171	2317	6556	7411
shifts	1825	1920	5967	6578
reduces	3474	3564	14628	15428
gotos	691	700	1409	1463
predictions	2.44	2.36	3.14	2.97

Compared with the SLR table, the number of states of CLR table increased by 6.7% (mset_phrase) and 13% (eset_phrase), but the average phone predictions for each state decreased. This will result in the precise phone prediction for speech recognition. We will discuss the performance of these two type of LR table applied in allophone-based speech recognition in Section 6.

5 Parsing Algorithm

As phoneme context has been compiled in the LR table in advance, the GLR parsing algorithm is principally the same as Tomita's GLR parsing algorithm. There is only slight change about GLR parsing algorithm with the allophone-based LR table generated by CPM. According to the modified LR table in Fig. 4, we can construct the parsing tree shown in Fig. 10

In state 15, for the end phone "i" of word "a n1 i", the lookahead symbols of the allophone reduce action "re8"(i → i1) are "g2" and "n1". After carrying out this reduce action, the parser transfers to state 17. In state 17, the reduce action "re2"(N → a n1 i) with the same lookahead symbols "g2" and "n1" is carried out, and goto state 4. However, in state 4, there are three shift actions with lookahead "g2", "n1" and "n2". Since "n2" can not succeed "i1", for the word "a n1 i", sh12 is illegal. But we can not delete this action from the LR table, because after "re3"(N → a n2 e), the parser will transfer to this state too, and "n2" can succeed "e1".

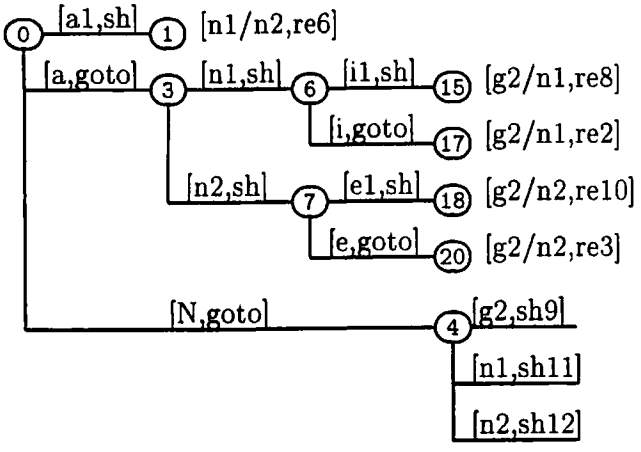


Fig. 10 Parsing tree for LR table in Fig. 4

To avoid this useless allophone prediction, we have to modify the GLR parsing algorithm slightly. For the reduce action with allophone rule, its lookahead symbols should be stored and carried to the new state after reduce, and at the new state, only those actions which applicable for the stored lookahead symbols are carried out.

In the above example, if the path is "a n1 i", the lookahead symbols "g2" and "n1" of action "re8"(i → i1) in state 15 will be carried to state 17, and then to state 4. In state 4, only sh9 with lookahead "g2" and sh11 with lookahead "n1" are carried out.

The parsing algorithm for allophone prediction is summarized below:

[LR Parsing Algorithm:]

As same as [3][6], a data structure (named cell) with information about one possible parse is used. The following information is kept in the cell:

- LR stack, with information for parsing control.
- Prediction allophone set P, which includes the next allophones the parser will predict, is used to remove the ambiguity of allophone predictions when the actions are transferred from a goto action.
- Probability array, which includes end point candidates and their probabilities.

1. Initialization

Create a new cell C, push the LR initial state 0 on the top of LR stack of C.

2. Ramification of cells

Construct a set.

$S = \{(C, s, a, x) \mid C, a, x(C$ is a cell $\&C$ is not accepted & s is the top state of C & ACTION[s,a]=x $\&x \neq$ error.)$\}$

For each element $(C, s, a, x) \in S$, do the following. If set S is empty, parser is completed.

3. If x = "shift s'",

- if state s is transferred from a shift action, verify the existence of allophone "a", and let the prediction allophone set P be an empty set.
- if state s is transferred from a goto action and the input "a" is included in the prediction set P, verify the existence of allophone "a", otherwise the cell C is abandoned.

4. If x = "reduce n",

- if rule n is an allophone rule, do this reduce action, and add the allophone "a" into P.
- if rule n is not an allophone rule, but "a" is included in P, do this reduce action. Otherwise, the cell C is abandoned.

5. If x = accept and the probability of cell C exceeds a certain threshold, cell C is accepted.

Otherwise, cell C is abandoned.

6. Return to 2.

6 Speech Recognition Experiments

6.1 Speech Data

The recognition experiments were carried out using 345 phrases uttered by one professional announcer (MAU in the ATR database).

The speech was sampled at 12 kHZ, quantized to 16 bits, preemphasized by $(1 - 0.98z^{-1})$, and windowed using a 20 msec Hamming window with a 5 msec shift, 34 coefficients which consists of log-power, delta log-power, 16-channel cepstrum coefficients, 16-channel delta cepstrum coefficients were used as the feature parameters, a diagonal-covariance single Gaussian distribution was used as an output probability density distribution of each state. Isolated 2620 Japanese words (even words of ATR 5240-word database) were used for training data. Allophone model (called HMnet) is generated by a SSS algorithm[6]. The number of states for the HMnet are 200, 400 and 600, which corresponds to 283, 1026 and 1759 allophones, respectively.

6.2 Grammar

A phrase grammar for international conference secretarial service (eset_phrase in Table 1) is used to the speech recognition experiments.

6.3 Recognition Results

The proposed allophone-based GLR parsing approach was evaluated by the recognition rates, CPU time, average allophone verifications and average accepted phrase candidates.

Table 3: Recognition rates

allophone number	Table type	beam=50		beam=100		beam = 250	
		top 1	top 5	top 1	top 5	top 1	top 5
26(*)		82.90	90.43	85.51	94.20	87.83	97.10
283	SLR	83.48	90.72	85.80	95.07	87.83	97.10
	CLR	84.06	91.30	86.09	95.65	87.83	97.39
1026	SLR	88.12	95.07	90.72	97.68	91.01	99.13
	CLR	88.70	95.94	90.72	98.26	91.01	99.13
1759	SLR	86.96	94.49	89.28	97.68	89.28	98.84
	CLR	87.54	95.36	89.28	97.68	89.28	98.84

6.3.1 Recognition rates

The recognition rates are listed in Table 3 for 283, 1026 and 1759 allophone models. For the comparison, the recognition rates for phoneme-context-independent model(26 phones) is given too.

It is easy to see that when beam width is small, canonical LR table gives slightly better recognition rates. This indicates that the CLR table gives more precise allophone predictions than the SLR table.

6.3.2 CPU time

The average CPU time (measured on an HP735 machine) for each phrase utterance is shown in Fig. 11. For the comparison, the CPU time of GLR parsing method realized in parser level[6] is shown too.

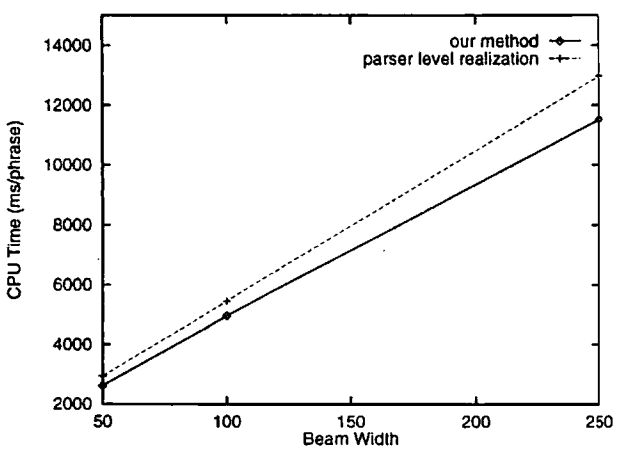


Fig. 11 Average CPU time for each utterance

Since the phoneme context is compiled into the LR table in advance, less CPU time is required with our method.

6.3.3 Average allophonic verifications

The average allophone verifications for each utterance are listed in Table 4.

From Table 4, compared with the SLR table, CLR table needs less verifications.

6.3.4 Average accepted phrase candidates

The average accepted phrase candidates for each utterance are listed in Table 5.

Table 4: Average phonetic verifications

allophone number	table type	beam width		
		50	100	250
283	SLR	2552	4315	9639
	CLR	2444	4105	9126
1026	SLR	2054	3654	8134
	CLR	1998	3536	7840
1759	SLR	1880	3381	7567
	CLR	1839	3286	7334

Table 5: Average accepted phrase candidates

allophone number	table type	beam width		
		50	100	250
283	SLR	73	152	386
	CLR	88	181	454
1026	SLR	63	133	340
	CLR	81	165	417
1759	SLR	60	128	330
	CLR	78	160	406

From Table 5, compared with the SLR table, CLR table gives more accepted phrase candidates for the same beam width.

7 Conclusions

In this paper, we have described a phoneme-context-dependent GLR parsing method. First, we introduce a set of allophone rules into a set of syntactic and lexical rules from given allophone models, and construct an initial canonical LR table, then modify the initial LR table on the basis of an allophone connection matrix through constraints propagation. With this modified allophone-based LR table, only slight change is needed for the GLR parsing algorithm used generally. It was found that CLR table can provide us more precise allophone predictions than SLR table. The proposed method was evaluated by the continuous speech recognition experiments for a task of Japanese phrase recognition with 1225 CFG rules and 1588 words.

One problem about our method is that generating an allophone-based LR table is time-consuming for a large grammar compared with the phoneme-context-independent case.

The future work will include the following:

- Evaluation of CPM algorithm with sentence recognition.
- Application to stochastic CFG grammar.

Acknowledgments

The authors wish to thank Dr. Yamazaki, the president of ATR Interpreting Telecommunications Research Laboratories for giving us this research opportunity. The authors are also grateful to all the members of ATR-ITL and Tanaka & Tokunaka Laboratory, Tokyo Institute of Technology for their constant help and encouragement.

We are especially grateful to Dr.Suresh(Tokyo Institute of Technology) for supplying us the Prolog program of generating canonical LR table.

References

[1] Aho,A.V., Sethi,R. and Ullman,J.D. *Compilers: Principles, Techniques, and Tools.* Massachusetts, Addison-Wesley, 1986.

[2] Itou,K., Hayamizu,S. and Tanaka,H. *Continuous Speech Recognition by Context Dependent Phonetic HMM and an Efficient Algorithm for Finding N-best Sentence Hypotheses.* ICASSP92, pp. 21-24, 1992

[3] Kita,K., Kawabata, T. and Saitou,H. *HMM Continuous Speech Recognition Using Predictive LR Parsing.* ICASSP89, 1989

[4] Lee,K.F. *Automatic Speech Recognition: The Development of the SPHINX System.* Kluwer Academic Publishers, Norwell, MA, 1989

[5] Li,H., Takezawa,T., Singer,H., Hayashi,T. and Tanaka,H. *An Efficient Phoneme-Context-Dependent LR Table and its Applications in Continuous Speech Recognition.* 音学講論 (秋), pp.125-126, 1994

[6] Nagai,A., Takami,J., Sagayama,S. *The SSS-LR Continuous Speech Recognition System: Integrating SSS-derived Allophone Models and a Phoneme-Context-Dependent LR Parse.* 信学技報, SP92-33. 1992

[7] Nagai,A., Sagayama,S., Kita,K. and Kikuchi,H. *Three Different LR Parsing Algorithms for Phoneme-Context-Dependent HMM Based Continuous Speech Recognition.* IEICE Trans. Inf. & Syst., vol. E76-D, No.1, pp.29-37, January, 1993

[8] Schwartz,R., Chow,Y., Kimball,O., Roucos,S. Krasner,M. and Makhoul,J. *Context Dependent Modeling for Acoustic Phonetic Recognition of Continuous Speech.* ICASSP85, 1985

[9] Tanaka,H., Tokunaga,T. and Aizawa,M. *Integration of Morphological and Syntactic Analysis Based on LR Parsing Algorithm.* International Workshop on Parsing Technologies, Tilburg, pp.101-109, 1993

[10] Tanaka,H., Li,H. and Tokunaga,T. *Incorporation of Phoneme-Context-Dependence into LR table through Constraints Propagation Method.* Proc. of AAAI-94 Workshop on the Integration of Natural Language and Speech Processing, Seattle, pp.15-22, 1994

[11] Tomita,M. *Efficient Parsing for Natural Language: A Fast Algorithm for Practical systems.* Kluwer Academic Publishers, pp.201, 1986