Systems and Computers in Japan, Vol. 33, No. 13, 2002
Translated from Denshi Joho Tsushin Gakkai Ronbunshi, Vol. J83-D-II, No. 11, November 2000, p p. 2407–2417

Construction and Evaluation of Language Models Based on Stochastic Context-Free Grammar for Speech Recognition

Chiori Hori,[1] Masaharu Katoh,[2] Akinori Ito,[2] and Masaki Kohda[2]

[1]Graduate School of Information Science and Engineering, Tokyo Institute of Technology, Tokyo, 152-8550 Japan

[2]Faculty of Engineering, Yamagata University, Yonezawa, 992-8510 Japan

SUMMARY

This paper deals with the use of a stochastic context-free grammar (SCFG) for large vocabulary continuous speech recognition; in particular, an SCFG with phrase-level dependency rules is built. Unlike n-gram models, the SCFG can describe not only local constraints but also global constraints pertaining to the sentence as a whole, thus making possible language models with great expressive power. However, the inside–outside algorithm must be used for estimation of the SCFG parameters, which involves a great amount of calculation, proportional to the third power of the number of nonterminal symbols and of the input string length. Hence, due to problems in dealing with extensive text corpora, the SCFG has hardly been applied as a language model for very large vocabulary continuous speech recognition. The proposed phrase-level dependency SCFG allows a significant reduction of the computational load. In experiments with the EDR corpus, the proposed method proved effective. In experiments with the Mainichi corpus, a large-scale phrase-level dependency SCFG was built for a very large vocabulary continuous speech recognition system. Speech recognition tests with a vocabulary of about 5000 words showed that the proposed method could not compare with the trigram model in performance; however, when it was used in combination with a trigram model, the error rate was reduced by 14% compared to the trigram model alone. © 2002 Wiley Periodicals, Inc. Syst Comp Jpn, 33(13): 48–59, 2002; Published online in Wiley InterScience (www.interscience.wiley. com). DOI 10.1002/scj.1172

Key words: speech recognition; language model; stochastic context-free grammar; dependency grammar; inside–outside algorithm.

1. Introduction

Recently, extensive attempts have been made to implement large vocabulary continuous speech recognition (LVCSR), and much research has been devoted to the recognition of newspaper articles read aloud, or of broadcast news [1, 2]. When recognizing such spoken texts, unique identification is complicated when only acoustic information is used. Therefore, it is important to compensate for the uncertainty of the acoustic data by imposing linguistic constraints based on language models, and avoiding ungrammatical strings.

In recent systems for continuous speech recognition, n-gram models have been used as the language models. An n-gram is widely used because, despite its simplicity, it is effective in improving recognition performance and allows easy setting of the model parameters. However, n-gram models express local constraints between words and do not express global text-level constraints such as are presented by structural grammar.

Other language models available for use include stochastic context-free grammars (SCFG), based on context-free grammars used for syntactic parsing of natural languages. Compared to n-gram models, SCFGs can express global sentence-level constraints; however, because of the grammatical uncertainty of natural languages, the inside–outside algorithm [3, 4] must be used for learning. As a result, model parameter estimation involves a very heavy computational load, which is proportional to the third power of the number of nonterminal symbols and of the input sentence length. There are fast SCFG learning algorithms using a bracketed corpus with added syntactic parsing data, such as the Penn Treebank [5, 6], but syntactically parsed texts are not normally available. In addition, in SCFGs, probabilities have usually been assigned to production rules formulated in accordance with linguistic grammar; but natural languages include many ungrammatical sentences, and such sentences could not be analyzed. For such reasons, SCFG has hardly been applied to LVCSR, but has been restricted to small-vocabulary recognition.

The authors previously proposed a method of reducing the computational complexity from the third to the second power of the number of nonterminal symbols by means of dependency grammar, a feature of the Japanese language [8]. Consequently the SCFG involved only the setting of nonterminal symbols and the vocabulary size, without explicit specification of the grammatical functions of nonterminal symbols, and probabilities were assigned to the production rules obtained by various combinations of nonterminal symbols. This model provided analyses for all evaluation sentences while achieving the same level of performance as the trigram model with appropriate initial setting of the parameters. However, a very large number of nonterminal symbols was required in order to attain such a performance level; therefore, we could not then achieve a reduction of processing sufficient for the construction of a large-scale SCFG to be used in LVCSR.

In this study, an SCFG using phrase-level dependency grammar is proposed for further reduction of computational complexity. In experiments with the EDR corpus, we compared the perplexity and processing load for parameter determination in several types of SCFGs that incorporated the proposed method. In addition, a large-scale phrase-level dependency SCFG was built from the newspaper corpus, and was compared to the trigram model in terms of recognition performance.

2. Stochastic Context-Free Grammar

2.1. Definition of SCFG

In an SCFG, the productions of context-free grammar (CFG) are assigned probabilities as follows:

$$\alpha \to \theta, \quad P(\alpha \to \theta) \in [0,1]$$
$$\alpha \in \mathcal{N}, \qquad \theta \in (\mathcal{N} \cup \mathcal{T})^*$$
$$\sum_{\theta} P(\alpha \to \theta) = 1$$

Here $\mathcal{N}$ is the set of nonterminal symbols, $\mathcal{T}$ is the set of terminal symbols, and $P(\alpha \to \theta)$ is the production probability of the rule $\alpha \to \theta$.

The probability that a word sequence $w_1, w_2, \ldots, w_L$ ($w_i \in \mathcal{T}$) is produced by the SCFG is the probability that such a word sequence is obtained from the start symbol S ($S \in \mathcal{N}$) by applying arbitrary rules, and thus is the product of all production probabilities:

$$P(S \to w_1 w_2 \ldots w_L)$$

When the derivation tree cannot be determined uniquely, the probability of a word sequence is defined as the sum of the probabilities for all trees.

The probability of an SCFG production rule can be estimated if the derivation trees for all sentences are found from a language corpus, and the number of production applications is counted. However, SCFG derivation trees cannot be defined uniquely in most cases, and hence the production probability must be estimated by the inside–outside algorithm. The grammar is normally generated from the linguistic standpoint, so that the computational complexity increases with the number of nonterminal symbols. In this study, existing grammars are not employed; instead, only the number of nonterminal symbols is defined, and the probabilities are estimated under the assumption that production rules exist for all possible combinations of nonterminal symbols.

2.2. The inside–outside algorithm

The inside–outside algorithm is designed for learning by SCFG maximum likelihood estimation. The grammar must be represented in Chomsky normal form as follows:

$$\alpha \to \beta\gamma$$
$$\alpha \to w$$
$$\alpha, \beta, \gamma \in \mathcal{N}, \quad w \in \mathcal{T}$$

The calculation formulas for the inside and outside probabilities and the reestimation of SCFG parameters are explained below, with $a(\alpha \to \beta\gamma)$ standing for the production probability of a production rule for a nonterminal symbol, expressed by $\alpha \to \beta\gamma$, and $b(\alpha \to w)$ standing for

the production probability of a terminal symbol (word), expressed by $\alpha \rightarrow w$.

(1) The inside probability

For an input text string $w_1, w_2, w_3, \ldots, w_L$, the inside probability is the probability that nonterminal symbol α appears in the word sequence from the i-th through the j-th position; it is shown in Fig. 1 as the probability of the shaded area of the derivation tree. The inside probability is defined as follows:

$$
\begin{aligned}
& e(i,j|\alpha) \\
& \quad = P(\alpha \rightarrow w_i \ldots w_j) \\
& \quad = \begin{cases} \displaystyle\sum_{k=i}^{j-1} \sum_{\beta\gamma} a(\alpha \rightarrow \beta\gamma) e(i,k|\beta) e(k+1,j|\gamma) & \text{if } \ i < j \\ b(\alpha \rightarrow w_i) & \text{if } \ i = j \end{cases}
\end{aligned} \tag{1}
$$

(2) The outside probability

The outside probability is the generation probability of the shaded area of the derivation tree shown in Fig. 2. Its definition is as follows:

$$
\begin{aligned}
& f(i,j|\alpha) \\
& \quad = P(S \rightarrow w_1 \ldots w_{i-1} \alpha w_{j+1} \ldots w_L) \\
& \quad = \sum_{k=1}^{i-1} \sum_{\beta\gamma} a(\beta \rightarrow \gamma\alpha) e(k,i-1|\gamma) f(k,j|\beta) \\
& \qquad + \sum_{k=j+1}^{L} \sum_{\beta\gamma} a(\beta \rightarrow \alpha\gamma) e(j+1,k|\gamma) f(i,k|\beta)
\end{aligned} \tag{2}
$$

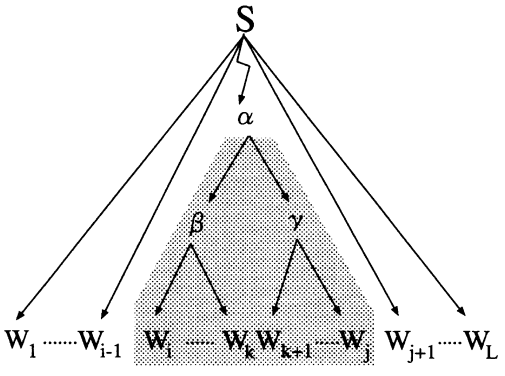


Fig. 1. The inside probability.

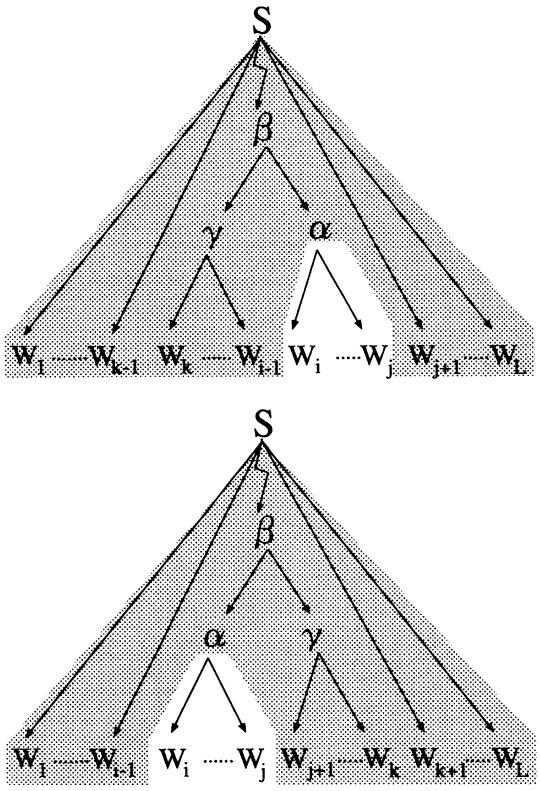


Fig. 2. The outside probability.

(3) Parameter reestimation

When a word sequence w_1, w_2, w_3, . . . , w_L is generated, the production probability of rule $\alpha \rightarrow \beta\gamma$ in an arbitrary location in the derivation tree is calculated in the following way:

$$
\hat{a}(\alpha \rightarrow \beta\gamma) = \frac{\displaystyle\sum_{i=1}^{L-1} \sum_{j=i+1}^{L} \sum_{k=i}^{j-1} g(i,k,j;\alpha \rightarrow \beta\gamma)}{e(1,L|S)} \tag{3}
$$

Here $g(i, k, j; \alpha \rightarrow \beta\gamma)$ represents the probability that $\beta\gamma$ is derived from α, that $w_i \ldots w_k$ is derived from β, and that $\omega_{k+1} \ldots \omega_j$ is derived from γ. It is calculated as follows, using the inside and outside probabilities:

$$
\begin{aligned}
& g(i,k,j;\alpha \rightarrow \beta\gamma) \\
& \quad = e(i,k|\beta) e(k+1,j|\gamma) a(\alpha \rightarrow \beta\gamma) f(i,j|\alpha)
\end{aligned} \tag{4}
$$

The production probability of a word w is

$$
\hat{b}(\alpha \rightarrow w) = \frac{\displaystyle\sum_{i;w_i=w} b(\alpha \rightarrow w) f(i,i|\alpha)}{e(1,L|S)} \tag{5}
$$

Learning involves initial setting of the production probabilities for all production rules, calculation of the inside–outside probabilities for all learning sentences, and reestimation of the production probabilities. After that, the reestimated probabilities are used as new initial settings, and so on. Appropriate SCFG parameters are found by such repetitive learning. However, a large amount of calculation is necessary in order to estimate the SCFG by the inside–outside algorithm; specifically, the computational complexity is $O(N^3L^3)$, with N and L denoting, respectively, the number of nonterminal symbols and the number of words per text.

3. Phrase-Level Dependency SCFG

The following three types of SCFG are usually considered in order to reduce the computational complexity of SCFG learning.

(1) Dependency SCFG

The production rules between nonterminal symbols are transformed in terms of dependency grammar:

$$\alpha \rightarrow \beta\alpha$$

$$\alpha \rightarrow w$$

Dependency grammar is a phrase structure grammar, characteristic of the Japanese language, in which the preceding phrase modifies the following phrase. By transforming production rules among nonterminal symbols in terms of dependency grammar, the computational complexity of SCFG learning becomes $O(N^2L^3)$, which is smaller by a factor of N than $O(N^3L^3)$. Below such an SCFG is referred to as K-SCFG.

(2) Phrase-level SCFG

In this case, the production rules for nonterminal symbols are applied on the phrase level, and the stochastic normal grammar applies inside a phrase:

(*interphrase*) $\alpha \rightarrow \beta\gamma$
(*intraphrase*) $\alpha \rightarrow w_c$
$\alpha \rightarrow \beta w_f$

Here w_c and w_f denote content and function words, respectively.

By applying SCFG on the phrase level, the computational complexity of learning is reduced from $O(N^3L^3)$ to $O(N^3M^3)$, with M denoting the number of phrases in one sentence. Assuming that one phrase contains two words on average, the complexity is reduced by a factor of 2^3. Below, such an SCFG is referred to as P-SCFG.

(3) Phrase-level dependency SCFG

Here the production rules among nonterminal symbols are applied on the phrase level and are transformed in terms of dependency grammar. Inside a phrase, the stochastic normal grammar applies:

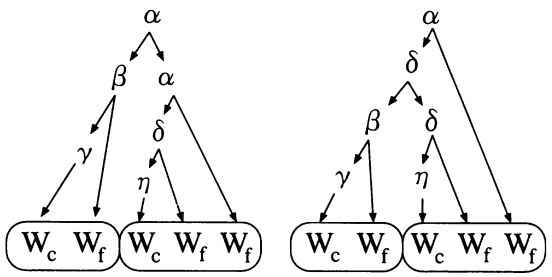
(*interphrase*) $\alpha \rightarrow \beta\alpha$
(*intraphrase*) $\alpha \rightarrow w_c$
$\alpha \rightarrow \beta w_f$

This is the model proposed in our study. The computational complexity of learning is reduced from $O(N^3L^3)$ to $O(N^2M^3)$, that is, by a factor of $1/N\,(1/2)^3$. The proposed SCFG is referred to as PK-SCFG.

Previously, the authors proposed to make the transformation $\alpha \rightarrow \beta\gamma$ in terms of dependency grammar, and the transformation $\alpha \rightarrow w$ in terms of normal grammar [8], which is referred to as K-SCFG2. The same rules were used as in PK-SCFG, but the phrase partitioning was made indeterminate. The PK-SCFG proposed in this paper is different from K-SCFG2 in that the application of the $\alpha \rightarrow \beta\alpha$ rule is restricted to phrase level. Figure 3 presents examples of derivations allowed in both PK-SCFG and K-SCFG2, and derivations allowed only in K-SCFG2. Thus, the transformations $\alpha \rightarrow \delta w_f$ and $\delta \rightarrow \beta\delta$ in Fig. 3(b) are not allowed in PK-SCFG.

When SCFG is applied on the phrase level as in P-SCFG and PK-SCFG, phrase partitioning is specified explicitly. In the proposed method, a phrase is defined as a word sequence of one content word and zero or more function words. That is, a phrase starts from a content word and ends before the next content word. In learning, phrases are recognized automatically by discrimination between content and function words.

An algorithm for SCFG learning using a bracketed corpus that has been subjected to syntactic parsing beforehand was proposed in Ref. 5. Since a bracketed corpus



(a) Both 'K-SCFG2' and 'PK-SCFG' (b) 'K-SCFG2' only

Fig. 3. Examples of derivations in the proposed method (PK-SCFG) and the previous method (K-SCFG2) [8].

Table 1. Relationships between SCFGs

			Word level		Phrase level	
Dependency constraints	not used	Item	SCFG		P-SCFG	
		Rules	$\alpha \to \beta\gamma$		(*interphrase*)	$\alpha \to \beta\gamma$
			$\alpha \to w$		(*intraphase*)	$\alpha \to w_c$
						$\alpha \to \beta w_f$
		Computational complexity	$O(N^3L^3)$		$O(N^3M^3 + N^2MI)$	
	used	Item	K-SCFG	K-SCFG2	PK-SCFG	
		Rules	$\alpha \to \beta\alpha$	$\alpha \to \beta\alpha$	(*interphrase*)	$\alpha \to \beta\alpha$
			$\alpha \to w$	$\alpha \to w_c$	(*intraphrase*)	$\alpha \to w_c$
				$\alpha \to \beta w_f$		$\alpha \to \beta w_f$
		Computational complexity	$O(N^2L^3)$		$O(N^2M^3 + N^2MI)$	

N: number of nonterminal symbols

L: number of words in one sentence

M: number of phrases in one sentence ($\approx \frac{1}{2} L$)

I: number of words in one phrase

implies text partitioning, the syntax becomes more definite, and the computational load decreases. On the other hand, in the method proposed in this study, phrases are determined automatically in the course of SCFG learning; thus, bracketing can be performed automatically in learning, and there is no need for preparation of a bracketed corpus.

In addition, the proposed PK-SCFG uses normal grammar inside a phrase, and dependency grammar among phrases; on the other hand, in the method reported in Ref. 5, SCFG is applied both within and between phrases. Therefore, even though Ref. 5 employs corpus bracketing by phrase definitions as proposed here, the computational complexity would be even higher than in PK-SCFG.

In the case of phrase partitioning, the computational complexity of the method of Ref. 5 is $O(N^3M^3 + N^3MI^3)$ considering intraphrase calculation; on the other hand, in the proposed PK-SCFG, the complexity is $O(N^2M^2 + N^2MI)$. Here N, M, and I denote, respectively, the number of nonterminal symbols, the number of phrases, and the length of one phrase (number of words). The relationships between the various SCFGs are summarized in Table 1.

4. Probability Estimation Algorithm for Phrase-Level Dependency SCFG

The following notations will be used in the explanation of the algorithm.

M: number of phrases in one sentence
w_{mc}: content word in the m-th phrase
$w_{mf,i}$: i-th function word in the m-th phrase
K_m: number of function words in the m-th phrase

One sentence is composed of M phrases:

$$S \to P_1 \dots P_m \dots P_M$$

A phrase is defined as a *word sequence of one content word and some (including zero) function words*:

$$P_m = w_{mc} w_{mf,1} w_{mf,2} \dots w_{mf,Km}$$

The production probabilities of the productions expressed by $\alpha \to \beta\alpha$, $\alpha \to w_c$, and $\alpha \to \beta w_f$ are, respectively, $a(\alpha \to \beta\alpha)$, $b(\alpha \to wc)$, and $c(\alpha \to \beta w_f)$.

Below we explain the probability estimation algorithm for phrase-level dependency SCFG. The flowchart is shown in Fig. 4.

(1) Initial setting

Initially, the probabilities $a(\alpha \to \beta\alpha)$ are set equal, and the probabilities $b(\alpha \to w_c)$, $c(\alpha \to \beta w_f)$ are set randomly.

In addition, the initial setting method using part-of-speech texts or part-of-speech + high-frequency word texts [8] may also be used.

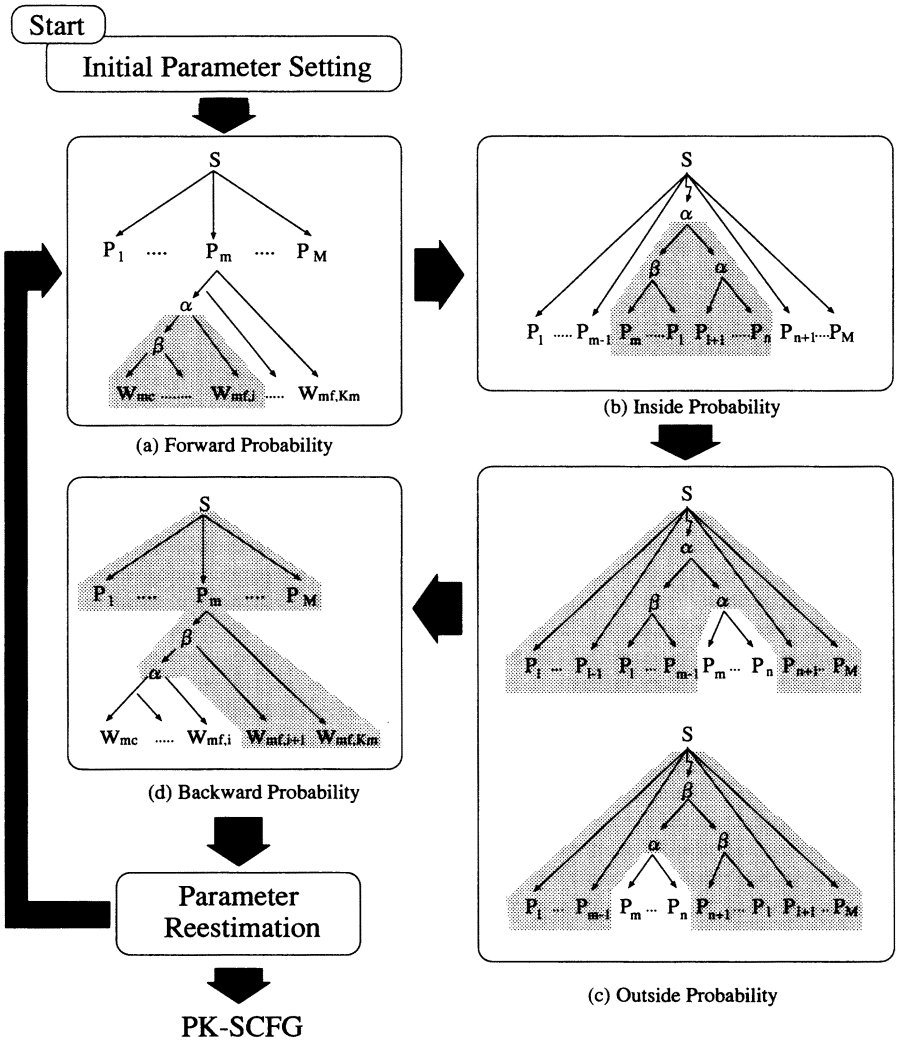


Fig. 4. Estimation algorithm for phrase-level dependency grammars.

(2) Calculation of forward probability inside phrase

The probability that the word sequence $w_{mc}w_{mf,1}\ldots w_{mf,i}$ of the m-th phrase is generated from non-terminal symbol α is calculated in the following way as the forward probability [Fig. 4(a)]:

$$h(m,i,\alpha) = P(\alpha \rightarrow w_{mc}w_{mf,1}\ldots w_{mf,i}) = \begin{cases} b(\alpha \rightarrow w_{mc}) & \text{if } i=0 \\ \sum_{\beta} h(m,i-1,\beta)c(\alpha \rightarrow \beta w_{mf,i}) & \text{if } i>0 \end{cases} \quad (6)$$

(3) Calculation of inside probability among phrases

Using the forward probability, the inside probability is calculated in the following way [Fig. 4(b)]:

$$e(m,n|\alpha) = P(\alpha \rightarrow P_m\ldots P_n) = \begin{cases} h(m,K_m,\alpha) & \text{if } m=n \\ \sum_{l=m}^{n-1}\sum_{\beta} a(\alpha \rightarrow \beta\alpha)e(m,l|\beta)e(l+1,n|\alpha) & \text{if } m<n \end{cases} \quad (7)$$

(4) Calculation of outside probability among phrases

Using the inside probability, the outside probability is calculated as follows [Fig. 4(c)]:

$$
\begin{aligned}
&f(m,n|\alpha) \\
&= P(S \rightarrow P_1 \ldots P_{m-1} \alpha P_{n+1} \ldots P_M) \\
&= \sum_{l=1}^{m-1} \sum_{\beta} a(\alpha \rightarrow \beta\alpha) e(l, m-1|\beta) f(l,n|\alpha) \\
&\quad + \sum_{l=n+1}^{M} \sum_{\beta} a(\beta \rightarrow \alpha\beta) e(n+1, l|\beta) f(m,l|\beta)
\end{aligned} \tag{8}
$$

(5) Calculation of backward probability inside phrase

Using the outside probability, the backward probability is calculated in the following way [Fig. 4(d)]:

$$
\begin{aligned}
&r(m,i,\alpha) \\
&= P(S \rightarrow P_1 \ldots P_{m-1} \alpha w_{mf,i+1} \ldots \\
&\qquad w_{mf,K_m} P_{m+1} \ldots P_M) \\
&= \begin{cases} f(m,m,\alpha) & \text{if } i = K_m \\ \sum_{\beta} c(\beta \rightarrow \alpha w_{mf,i+1}) r(m, i+1, \beta) & \\ & \text{if } i < K_m \end{cases}
\end{aligned} \tag{9}
$$

(6) Parameter reestimation

Using probabilities (2) to (5), the parameters are modified according to the following reestimation rules:

$$
\begin{aligned}
&\hat{a}(\alpha \rightarrow \beta\alpha) \\
&= \frac{\sum_{m=1}^{M-1} \sum_{n=m+1}^{M} \sum_{l=m}^{n-1} g(m,l,n;\alpha \rightarrow \beta\alpha)}{e(1,M|S)}
\end{aligned} \tag{10}
$$

$$
\begin{aligned}
&\hat{b}(\alpha \rightarrow w_c) \\
&= \frac{\sum_{m=1;w_{mc}=w_c}^{M} b(\alpha \rightarrow w_c) r(m,0,\alpha)}{e(1,M|S)}
\end{aligned} \tag{11}
$$

$$
\begin{aligned}
&\hat{c}(\alpha \rightarrow \beta w_f) \\
&= \frac{\sum_{m=1}^{M} \sum_{i=1;w_{mf,i}=w_f}^{K_m} h(m,i-1,\beta) c(\alpha \rightarrow \beta w_f) r(m,i,\alpha)}{e(1,M|S)}
\end{aligned} \tag{12}
$$

Here,

$$
\begin{aligned}
&g(m,l,n;\alpha \rightarrow \beta\alpha) \\
&= e(m,l|\beta) e(l+1,n|\alpha) a(\alpha \rightarrow \beta\alpha) f(m,n|\alpha)
\end{aligned} \tag{13}
$$

(7) Procedures (2) to (6) are repeated until convergence occurs.

5. SCFG Evaluation Experiments

5.1. Experimental conditions

Five SCFG variations were developed using the EDR corpus shown in Table 1, and were evaluated in terms of perplexity. The experimental conditions are presented in Table 2.

The initial setting was done in terms of equal probabilities and random numbers. Models with 5, 10, 15, 20, 25, and 30 learning iterations were prepared for each of five SCFGs.

5.2. Experimental results and discussion

The perplexity and learning time for 100 test sentences are shown in Figs. 5 and 6, respectively. In the diagrams, SCFG stands for word-level SCFG, P-SCFG for phrase-level SCFG, K-SCFG for dependency SCFG without distinction between content and function words, K-SCFG2 for dependency SCFG with distinction between content and function words [8], and PK-SCFG for phrase-level dependency SCFG (as proposed in this study).

As is evident from Fig. 5, phrase-level dependency SCFG (PK-SCFG) achieves the same degree of perplexity as the dependency SCFG in Ref. 8 (K-SCFG2): that is, the introduction of phrase units has little effect on model performance. Compared to word-level SCFG (SCFG, K-SCFG), phrase-level SCFG (P-SCFG, PK-SCFG) has faster convergence, and hence less time is required for building the models. The dependency SCFG proposed in Ref. 8 (K-SCFG2) is a word-level SCFG, but distinguishing between content and function words improves learning convergence nearly to the level of PK-SCFG.

On the other hand, as suggested by Fig. 6, the processing time per learning iteration is reduced by a factor of 20 by the application of dependency as proposed in Ref. 8;

Table 2. Experimental conditions of SCFG evaluation

Number of nonterminal symbols	20	
Vocabulary size (number of terminal symbols)	3032 words (occurred in learning text two or more times)	
Content words	Nouns, verbs, adjectives, adverbs, adjectival verbs, conjunctions, adnominals, prefixes, interjections, numbers, symbols	
Function words	Particles, auxiliary verbs, postpositions, suffixes	
Unknown words	Content and function words	
Texts used	EDR corpus	
Number of sentences	Learning text	Evaluation text
Number of words	2000	100
	53,910	2782
Proportion of unknown words	10.3%	22.0%

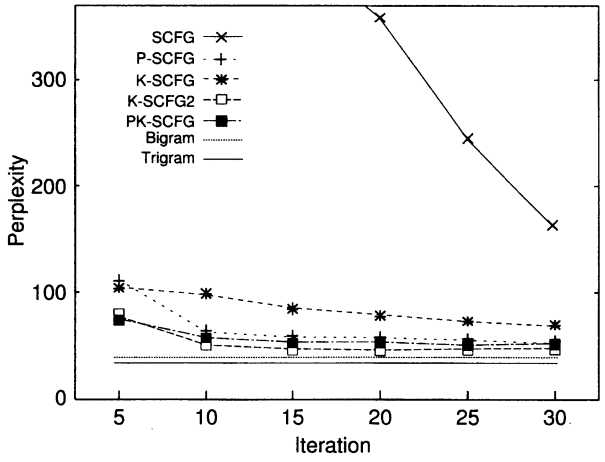


Fig. 5. Perplexity of various types of SCFGs.

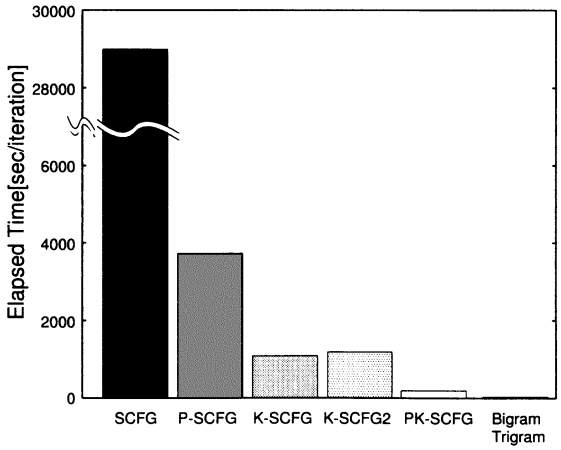


Fig. 6. Processing time required for SCFG parameter estimation.

the phrase-level grammar proposed here provides a further reduction by a factor of 160.

The above results prove that the phrase-level dependency SCFG (PK-SCFG) achieves considerable time saving while maintaining the performance of the dependency SCFG as proposed in Ref. 8.

6. Evaluation of Phrase-Level Dependency SCFG by Experiments with LVCSR

6.1. Recognition task

The recognition task involved speaker-independent continuous speech recognition of newspaper articles read aloud. The vocabulary included 5000 most frequent words (morphemes) in articles from the newspaper *Mainichi Shimbun* from January to September 1994.

6.2. Speech samples and speech analysis

15,732 sentences read by 102 males from the ASJ newspaper article read-speech corpus and phoneme-balanced sentences were taken as acoustic model learning samples. The evaluation samples consisted of 100 sentences from the same corpus read by 10 different males, 10 texts each. The 100 sentences used for evaluation were selected from articles published in the newspaper *Mainichi Shimbun* from October through December 1994 that did not include unknown words with a vocabulary of 5000 words.

The conditions of the speech analysis are given in Table 3.

Table 3. Conditions of speech analysis

Sampling frequency	16 kHz
Quantization	16 bit
Frame length	32 ms
Frame period	8 ms
Analysis window	Hamming window
High emphasis	$1 - z^{-1}$
Feature vector	1st to 12th order LPC cepstrum coefficients and logarithm powers as well as their 1st and 2nd order regression coefficients (total of 39 dimensions)
Normalization	Cepstral mean normalization for every utterance

6.3. Acoustic model

HM-Net based on state clustering [9] was employed as the acoustic model; 34 phonemes plus silence were used as phonemic categories. The initial HM-Net included 102 (34 × 3) states, constructed from phoneme context-independent HMMs, 3 states for each phoneme. Partitioning into 2000 states was performed, resulting in a total of 2003 states including three silent models added after structure design. The output probability distribution for every state was restructured in 16 mixtures.

6.4. Language models

A total of 46,301 sentences containing 3 or more words, with a vocabulary of 5000 words, were selected from articles published in *Mainichi Shimbun* in January through September 1994. Bigram, trigram, and phrase-level dependency SCFGs (PK-SCFG, with 100 and 120 nonterminal symbols) were built as language models. In PK-SCFG learning, the initial setting was performed as proposed in Ref. 8 using frequent words. The most frequent words were selected from the learning corpus, and the other words were replaced by their part-of-speech. The texts thus obtained were used to learn the production probabilities for the initial PK-SCFG productions $a'(\alpha \to \beta\alpha)$, $b'(\alpha \to w_c)$, $c'(\alpha \to \beta w_f)$, and initial setting was performed in the following way:

$$a(\alpha \to \beta\alpha) = a'(\alpha \to \beta\alpha) \tag{14}$$

$$b(\alpha \to w_c) = \begin{cases} b'(\alpha \to w_c) & \text{if } w_c \in W_h \\ \sum_{h_c} P(w_c|h_c) b'(\alpha \to h_c) & \text{else} \end{cases} \tag{15}$$

$$c(\alpha \to \beta w_f) = \begin{cases} c'(\alpha \to \beta w_f) & \text{if } w_f \in W_h \\ \sum_{h_f} P(w_f|h_f) c'(\alpha \to \beta h_f) & \text{else} \end{cases} \tag{16}$$

Here W_h is the set of frequent words, h_c and h_f are, respectively, part-of-speech of w_c and w_f, and $P(w_c|h_c)$ and $P(w_f|h_f)$ are part-of-speech unigrams. After that, the PK-SCFG was retrained using the conventional corpus of word sequences for final parameter setting.

This algorithm for initial setting achieved faster convergence and better perplexity than when the conventional corpus was used from the very beginning.

6.5. N-best rescoring

In the first pass, a time-synchronous beam search was performed using word bigrams, and word graphs were generated [1]. Then the top 100 candidate sentences found using the word graphs were scored in the following way:

$$\begin{aligned} &Score(W|O) \\ &= w_{pkscfg} \log P_{pkscfg}(W) + w_{ng} \log P_{ng}(W) \\ &\quad + acoust(O|W) + len(W) \times penalty \end{aligned} \tag{17}$$

Here $P_{pkscfg}(W)$ is the linguistic likelihood in terms of PK-SCFG, $P_{ng}(W)$ is the linguistic likelihood in terms of n-grams, w_{pkscfg} and w_{ng} are the respective weights, $acoust(O|W)$ is the acoustic likelihood, $len(W)$ is the length of the candidate (number of words), and *penalty* is the insertion penalty. The above formula provides a score when PK-SCFG and n-gram are used in combination; when each is used alone, a zero weight is assigned to the other model. The texts with the highest scores were selected as the recognition results, and the language models were inferred by the word error rate.

The word error rate was calculated as

$$WER = \frac{S + I + D}{L} \times 100 \quad [\%] \tag{18}$$

Here S, I, and D are, respectively, the numbers of substitution errors, insertion errors, and deletion errors, and L is the number of words in the correct sentence.

6.6. Experimental results and discussion

The perplexity of the correct sentences (utterances of 100 evaluation sentences) for all language models is shown in Fig. 7. Figure 8 presents the word error rates in LVCSR. The word error rates for all models are minimal values obtained by using optimal values of the model weights and the insertion penalty in rescoring as shown in Table 4.

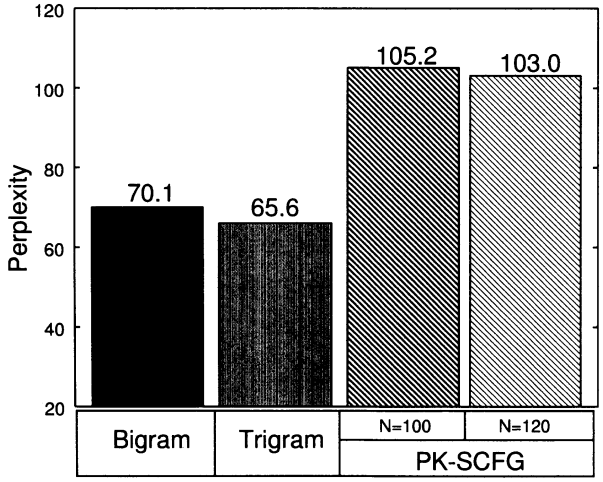


Fig. 7. Perplexity for evaluation sentences.

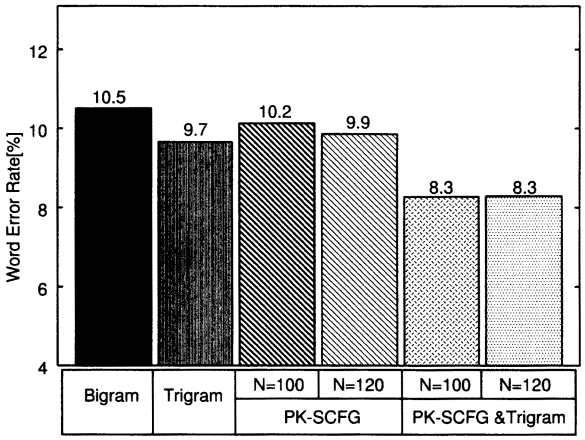


Fig. 8. Word error rate in speech recognition.

Table 4. Optimum weights of linguistic score and insertion penalty

	w_{ng}	w_{pkscfg}	penalty
Bigram	16	0	−20
Trigram	17	0	−20
PK-SCFG 100	0	19	−16
PK-SCFG 120	0	18	−20
Trigram+PK-SCFG 100	14	10	0
Trigram+PK-SCFG 120	6	16	−12

In terms of the perplexity of the evaluation sentences, the bigram and trigram models obtained lower values than PK-SCFG. On the other hand, in evaluation by speech recognition, PK-SCFG had an error rate intermediate between those of trigram and bigram; however, trigram may be outperformed by using more nonterminal symbols. In addition, the combined use of PK-SCFG and trigram gave a 14% error reduction compared to trigram alone.

The perplexity is an index expressing the extent to which the range of candidates can be narrowed; as suggested by the results in Fig. 7, PK-SCFG is outperformed by an n-gram model describing the concatenation constraints on individual words. However, in the case of a large search space, a great effect is achieved in the first pass; in the second pass, when considerable numbers of candidates have been screened out, selection of correct word strings (assigning the highest scores) becomes even more important than further screening. As is evident from Fig. 8, PK-SCFG has a lower error rate than bigram in the second pass; thus, PK-SCFG is better at correct selection. When PK-SCFG and trigram are used in combination, their respective strengths reinforce each other and their weaknesses are compensated.

Now consider the recognition errors for the trigram model and PK-SCFG, and the possibility of improvement by the combined use of both models.

In the recognition results obtained for the trigram model, few unnatural word sequences appear locally; however, there is considerable substitution and deletion of particles. As a result, unnatural phrases occur often, even though local word sequences look plausible. That is because the n-gram model, by its nature, cannot implement constraints over the sentence as a whole.

The following are typical examples of trigram errors (recognized correctly by PK-SCFG).

Example 1: particle substitution
Correct sentence: *gaikokujin senshu no katsuyaku mo medatta*
Trigram: *gaikokujin senshu no katsuyaku ga medatta*
PK-SCFG: no errors

Example 2: particle omission
Correct sentence: *hoka ni mo raibaru ga ooi*
Trigram: *hoka ni mo raibaru ___ ooi*
PK-SCFG: no errors

Example 3: unnatural phrase
Correct sentence: *sore o jieitai ga yatte ii wake ga nai*
Trigram: *sore o jieitai ga atte ii wake ga nai*
PK-SCFG: no errors

On the other hand, the recognition results for PK-SCFG (120 nonterminal symbols) are quite different from those of the trigram model; namely, quite a few unnatural word sequences appear. This indicates weaker local constraints than the trigram model. However, PK-SCFG achieves correct particle insertion as well as correction of unnatural phrases due to sentence-level constraints. Particle insertion errors were very few and no particle insertion in unnatural positions was found.

Here are typical examples of PK-SCFG errors (recognized correctly by trigram)

Example 4: unnatural word junction (1)
Correct sentence: *tanjoubi ka wa fumei*
Trigram: no errors
PK-SCFG: *tanjoubi ka wa fumei*[*]

Example 5: unnatural word junction (2)
Correct sentence: *shinsei doori nennai ninka no houshin*
Trigram: no errors
PK-SCFG: *shinsei doori nennai ninka no houshin*[†]

Example 6: particle insertion
Correct sentence: *sekiyu gyoukai de wa sukunai kisei kanwa suishin ha*
Trigram: no errors
PK-SCFG: *sekiyu gyoukai de wa sukunai kisei kanwa suishin ha wa*

In the combined use of PK-SCFG and trigram, the recognition accuracy was improved in most cases. In particular, errors were corrected in all of the above cases except for Example 6. Thus, the two models complement each other to improve recognition performance.

7. Conclusions

This study has dealt with the development of language models based on stochastic context-free grammar, and their evaluation in speech recognition experiments. The phrase-level dependency SCFG (PK-SCFG) proposed in this study achieves a considerable reduction of the processing time required for learning, and makes possible the estimation of SCFG parameters using large corpora, which was hitherto difficult. In speech recognition experiments with a large vocabulary of 5000 words, the proposed PK-SCFG used in combination with the trigram model achieved better performance than the trigram model alone.

In the future, more detailed analysis of speech recognition using SCFG will be necessary; in addition, experiments should be carried out with even larger vocabularies and bodies of learning data.

Acknowledgments. We express our gratitude to Dr. T. Hori (now at NTT Communication Science Labs) for providing the decoder, and to Professor S. Furui of Tokyo Institute of Technology for his kind cooperation.

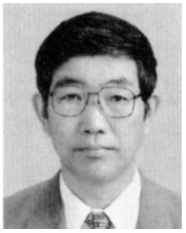
[*]Tr. note: *ka* (interrogative particle) replaced by homonymous *ka* (-ize).
[†]Tr. note: *shinsei* (application) replaced by homonymous *shinsei* (neoplastic).

REFERENCES

1. Hori T, Oka N, Katoh M, Ito A, Kohda M. A study on a phoneme-graph-based hypothesis restriction for large vocabulary continuous speech recognition. Trans IPS Japan 1999;40:1365–1373.
2. Kawahara T, Lee A, Kobayashi T, Takeda K, Minematsu N, Ito K, Ito A, Yamamoto M, Yamada A, Utsuro T, Shikano K. Evaluation of Japanese Dictation ToolKit 1997 version. IPSJ SIG Notes, 98-SLP-21-10, 1998.
3. Lari K, Young SJ. The estimation of stochastic context free grammars using the Inside–Outside algorithm. Computer Speech and Language 1990;4:35–56.
4. Lari K, Young SJ. Application of stochastic context free grammars using the Inside–Outside algorithm. Computer Speech and Language 1991;5:237–257.
5. Pereira F, Schabes Y. Inside–outside reestimation from partially bracketed corpora. Proc ACL-92, p 128–135.
6. Sanchez J-A, Benedi J-M. Learning of stochastic context-free grammars by means of estimation algorithms. Proc Eurospeech'99, Vol. 4, p 1799–1802.
7. Yoshida S. Syntax analysis of Japanese sentence based on Kakariuke relation between two bunsetsu. IEICE Trans Inf Syst 1972;J55-D:238–244. (in Japanese)
8. Yaginuma M, Katoh M, Ito A, Kohda M. A study of language modeling using stochastic context free grammar with dependency grammar. Tech Rep IEICE 1997;NLC97-33:33–40.
9. Hori T, Katoh M, Ito A, Kohda M. A study on a state clustering-based topology design method for HM-Nets. IEICE Trans Inf Syst 1998;J81-D-II:2239–2248. (in Japanese)

AUTHORS (from left to right)



Chiori Hori received her B.E. and M.E. degrees in electrical and information engineering from Yamagata University in 1994 and 1997 and joined the Faculty of Literature and Social Sciences. She enrolled in the Ph.D. course in the Graduate School of Information Science and Engineering at Tokyo Institute of Technology in 1999 and received her Ph.D. degree in 2002. Her research interests are speech information processing, including speech recognition, speech summarization, and speech understanding. She is now a member of NTT Communication Science Laboratories, Nippon Telegraph and Telephone Corporation. She is a member of IEEE and the Acoustical Society of Japan.

Masaharu Katoh received his B.E. and M.E. degrees in electrical and information engineering from Yamagata University in 1991 and 1993 and joined the Faculty of Electrical and Information Engineering. His research interests include speech recognition, speaker recognition, speech coding, and language modeling. He is a member of the Acoustical Society of Japan.

Akinori Ito received his B.E. degree in communication engineering from Tohoku University in 1986 and M.E. and D.Eng. degrees in information engineering in 1988 and 1991 and joined the Research Center for Applied Information Sciences. He became an associate professor at Yamagata University in 1995. He was a visiting researcher at Boston University in 1998. He is now an associate professor in the Graduate School of Tohoku University, and has been engaged in research on spoken language processing, especially language modeling. He is a member of the Acoustical Society of Japan, the Information Processing Society of Japan, and the International Speech Communications Association.

Masaki Kohda received his B.E., M.E., and D.Eng. degrees from Nagoya University in 1965, 1967, and 1979. From 1967 to 1987, he was engaged in research on speech processing at the Electrical Communication Laboratories of Nippon Telegraph and Telephone Corporation. He is currently a professor in the Department of Informatics of Yamagata University. His research interests include speech recognition, speaker recognition, speech synthesis, speech coding, and language modeling. He is a member of the Information Processing Society of Japan, the Acoustical Society of Japan, the Japanese Society for Artificial Intelligence, and the Association for Natural Language Processing.