TR-I-0169

音声認識における文法活用の有効性

Efficiency of Regular Grammar for Speech Recognition

坂野俊哉　　森元逞

Toshiya Sakano　Tsuyoshi Morimoto

1990.8

概要

ＡＴＲでは，構文解析処理を積極的に取込んだ音声認識（HMM－ＬＲ）を行っている．将来，確率的な構文解析に拡張することで処理の効率化を図りたいが，確率文法の本格的な適用に備えて，現在使用されている認識用文法の妥当性を調査した．その結果，文法の効率が無視できないものであり，その効率化の必要性を認識した．それと同時に，文節文法が確率的木構造によって十分に表現できることが分った．

ＡＴＲ自動翻訳電話研究所

ＡＴＲ Interpreting Telephony Research Laboratories

もくじ

1　音韻探索としての音声認識

音声認識を簡単に述べると，予め定義されている音声シンボル集合から，発声された音声に対応するシンボルを探索することである．その探索結果は確率的である．即ち，音響パターン集合 $\{A_i\}_i$，音声シンボル集合 $\{W_j\}_j$ にとすると，発声された音声Aに対して，

$$W_0 \quad = \quad arg\max_{w} P_r(W|A)$$

を A に対応する音声シンボルと確率的に評価される．この場合，音声シンボル集合 $\{W_j\}$ が発声音声Aに対する全探索空間である．

音声シンボルは音節列，文字列，単語列に相当するが，この探索空間の大きさは，音声シンボルの平均長を l，音声シンボルを列として構成する要素数を n とすれば，

$$|\{W_j\}_j| \quad = \quad n^l$$

であり，一般にこの大きさは非常に大きなものとなる．例えば，$l = 6.9$，$n = 50$ の場合，約 1.6338×10^{11} 個の音声シンボルを含む探索空間を持つことになる．

2　探索空間限定としての文法利用

音声認識における探索空間を次のようなレベルに分類する．（図1参照）

（S）音声領域

（L）言語領域

（G）文法領域

（T）タスク領域

音声領域Sは音韻シンボル系列の集合であり，先程の音声シンボル集合に相当する．HMM（Hidden Markov Model）のように，音韻連鎖をn重マルコフ過程でモデル化している場合，人間が通常発声しない音韻連鎖もモデルに含むことになるので，このモデルによる音声領域は，人間が発声可能な音韻シンボル系列の集合による音声領域よりも一般には大きくなる．

言語領域Lは言語的に意味を持つ音声領域Sの部分集合である．つまり，

$$L \quad = \quad \{l \in S | \; l \; is \; linguistic\} \; \subset \; S$$

として定義される．

文法領域Gは一つの言語文法により決定される音声領域Gの部分集合として規定される．言語文法記述のいかんによっては言語的に意味を為さない場合もあり，一般にはこの領域は言語領域の部分集合とならない．

タスク領域は，処理対象とする音声領域の部分集合である．例えば，連続数字音声認識では数字列に対応する音韻系列集合がそのタスク領域である．

音声認識は音声領域内での音韻シンボル系列の探索であり，その探索範囲は非常に大きいと述べた．もし，タスク領域が明示されていれば，探索範囲をタスク領域に限定することが出来る．その場合，探索に要する時間は明らかに短縮され，また探索誤り率も小さくなると予想される．タスク領域は言語的に有意であるから，そのタスク領域を規定する言語文法を定義し，その文法による文法領域を新たな探索空間とすることが考えられる．つまり，音声認識において言語文法の利用は探索空間の限定という意味で効果的ということが出来る．

しかし，タスクが連続数字音声認識のように，明示的に規定できるとは限らず，音声認識におけるタスクの設定は一般の自然言語処理と同等な問題を生じることになる．

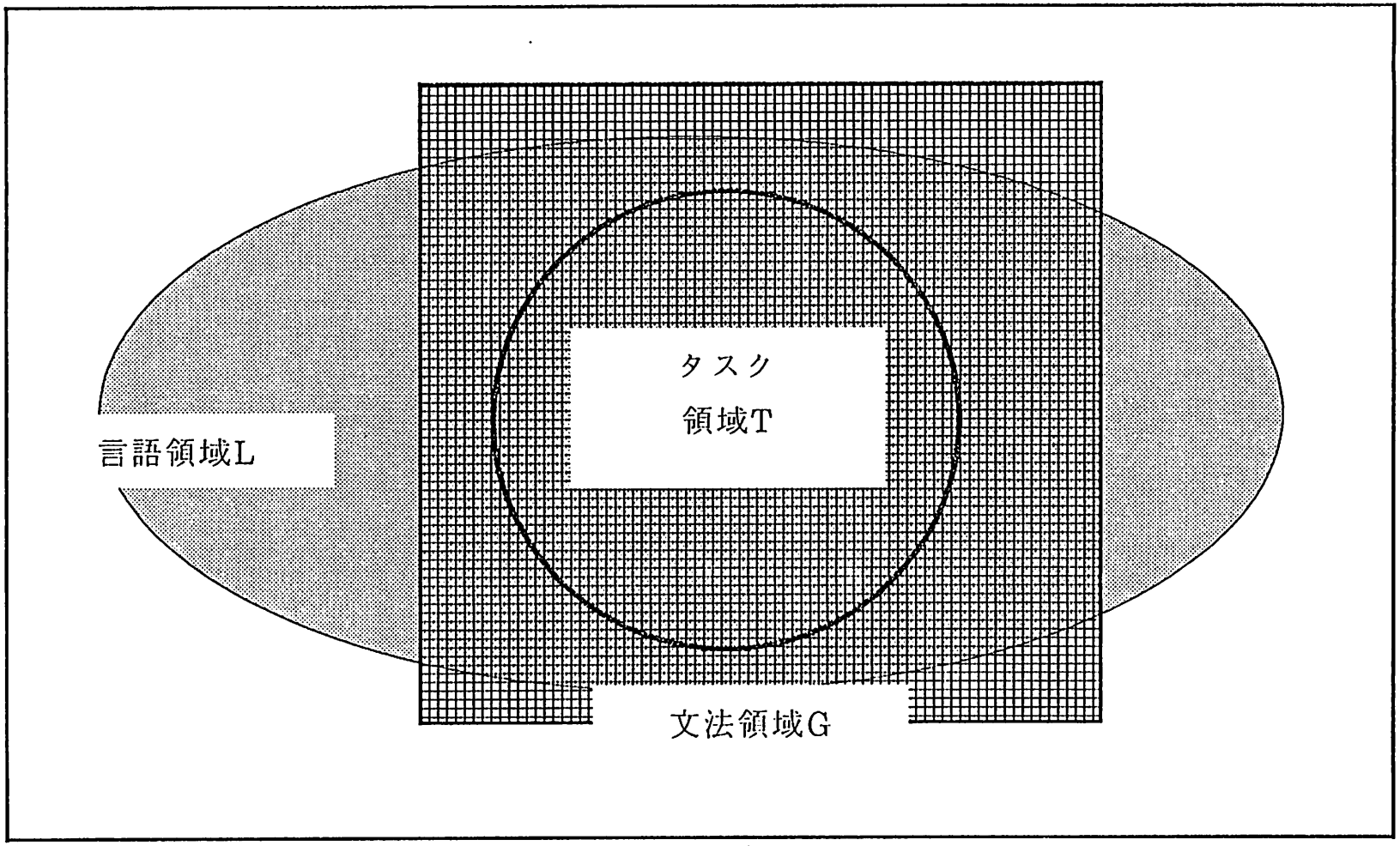

図1　音声領域S

3 文法記述能力の整理

タスクを表現するための文法記述法に関して整理する.

1. 再帰構造を含む記述法

$G = (V_N, V_T, P, S)$

V_N : 変数の集合

V_T : 終端記号の集合

P : 生成規則の集合

S : 開始記号

を文法とする.

(a) **文脈依存** (context sensitive grammar)

生成規則

$p: \quad \alpha \rightarrow \beta$

に対して, 条件 $|\alpha| \leq |\beta|$ を満たす.

($|\alpha|$ は列 α の長さを表す.)

(b) **文脈自由** (context free grammar)

生成規則

$p: \quad \alpha \rightarrow \beta$

に対して, $|\alpha|$ は 1 個の変数, $|\beta|$ は 空列 ϵ 以外の任意の列.

(c) **正規** (regular grammar)

生成規則は,

$p: \quad A \rightarrow aB$

又は,

$p: \quad A \rightarrow a$

である. (A, B は変数, a は終端記号)

2. 再帰構造を含まない記述法

(a) 木構造

(b) 列挙法

タスクを表現する文法を記述するには，対象タスクの大きさ・複雑さによって適当な文法記述を選択する必要がある．例えば，数字の選択のような小さなタスクでは列挙法で記述し，音声認識による駅の券売機では木構造による文法記述でも充分である．また，ＡＴＲでは現在「国際会議に関する問い合せ」を自動翻訳電話の対象タスクとして研究を進めているが，このタスクでは正規文法あるいは文脈自由文法による記述が必要である．いずれにせよ，タスクに適した文法を選択しなければならない．

次に，文法の効率に関して述べる．

4　文法の効率

対象タスク領域Tを記述する文法gによる文法領域をGとする．この時，TのGに対する濃度の割合によって文法gの効率が定義できる．例えば，TとGの濃度がいずれも有限個ならば，文法効率は，

$$E_g \quad = \quad \frac{|T|}{|G|} \quad (\leq 1)$$

（$|T|, |G|$ は各々 T, G の個数を表す．）
と定義することができる．この定義では，E_g が1に近いほど効率的だといえる．

先程述べた文法の効果は，対象とするタスクに対する文法の効率に相関するのは明らかであろう．では，どのような文法記述をどのように用いるとその文法効率に影響がでるのであろうか．今回，実際に音声認識システム（HMM-LR）で使用されている文法（認識用文法）の効率を測定を行った．以下では，使用データ，認識用文法の概観，効率測定法，測定結果を順次述べる．

5　言語データ構文解析実験結果

対象タスクはＡＴＲで取り扱っている「国際会議に関する問い合せ」であるが，その明示的な定義は困難である．現在，ＡＴＲでは上記タスクを想定したキーボード会話シミュレーションによるコーパス（言語データベース）の作成を行っている．今回の測定ではこの言語データベースを対象タスク領域とみなすことにした．使用したデータ数は195会話（29,656文節）である．図2は言語データを今回の解析に適した形式に変換したものの1部である．データは予め形態素解析されており，各形態素には図3のような品詞・活用形を示す番号が与えられている．また，文節単位で区切り記号を挿入している．文節の定義は曖昧であるので形態素解析の結果を利用して，「自立語1個を中心に文節は構成される」と考え，

接頭辞＋自立語＋接尾辞＋付属語

を基本ルールとしている．

使用する文法は音声認識用の文法（以下認識用文法と呼ぶ）を用いている．現段階では，音声認識は文節を単位として行っており，認識用文法も文節内文法を用いている．従って，上記言語データベースも文節を単位としてタスク領域としている．但し，言語データベースでの文節と認識用文法の文節とではその定義は必ずしも一致していない．

図4が認識用文法の文法記述形式例である．基本的には文脈自由文法で記述しているが，現在のバージョンでは再帰構造は含まれていない．また，例えば文法規則中の”Ｐ”のように文法規則間で共有されている文法カテゴリ（変数）が存在するが，便宜上図5のように共有されている文法カテゴリを複製して相異なる文法カテゴリとして取り扱うことにする．その為，文法は完全な木構造として表現することができる（図5では，文法をＡＮＤ－ＯＲ木で表現している）．この意味から，測定する認識用文法は本質的には再帰のない正規文法である．

実際に使用している認識用文法の1部を図6に示す．各行は左辺から右辺への書き換え規則を表している．ｘを記号列とすると，この規則の中の＜ｘｘｘ＞は変数，（ｘ　ｘ　ｘ）は終端記号としての音素列である．また，図の右側に文法に対応する木構造を表現しており，例えば，文法カテゴリ $<bunsetu>$ は $<np>$ または $<wh-np>$ に書き換えられ，$<np>$ は $<n>+<p>$ に書き換えられると読み，以下，右方に伸びるほど規則が深くなる．認識用文法の規則数は1,239個，終端記号（音素列）は654個である．

```
そう8  9
です12 2
----------
かいぎ 4  9
に   15 9
----------
もうしこみ 32  1
たい12  3
の   34  9
です12  2
が   14  9
----------
```

図2　言語データ

1: 形容詞
4: 普通名詞
5: サ変名詞
6: 代名詞
7: 数詞
8: 副詞
9: 連体詞
10: 接続詞
11: 感動詞
12: 助動詞
13: 副助詞
14: 接続助詞
15: 格助詞
16: 終助詞
17: 接尾語
18: 接頭語
19: 補助動詞
31: 形容名詞
32: 本動詞
33: 間投詞
34: 準体助詞
35: 並列助詞
36: 係助詞

0: 未然形
1: 連用形
2: 終止形
3: 連体形
4: 仮定形
5: 命令形
6: 語幹形
9: 活用なし

図3　言語データベース文法体系

$$
\begin{aligned}
A &\leftarrow B_1 \\
A &\leftarrow B_2 \\
B_1 &\leftarrow C_1 + P \\
B_2 &\leftarrow C_2 + P \\
P &\leftarrow p_1 \\
P &\leftarrow p_2
\end{aligned}
$$

図4　文法記述例

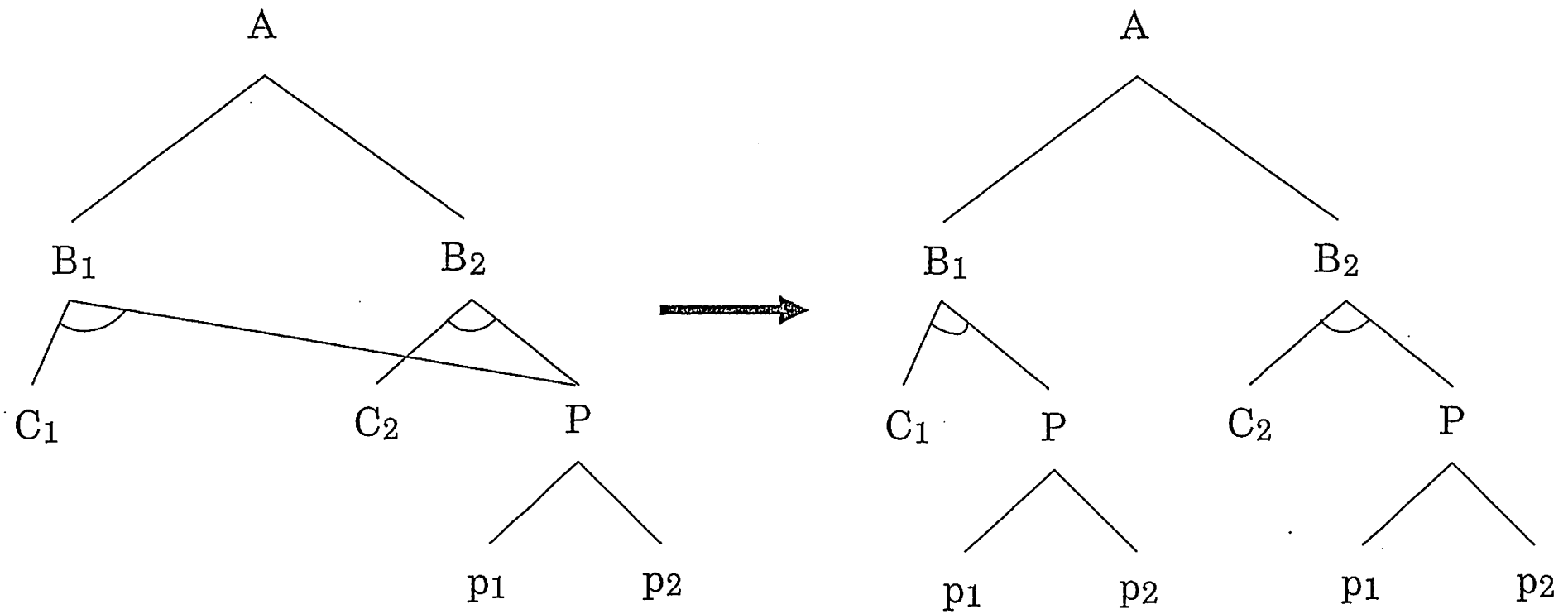

図5　文法カテゴリの非共有化

```
(<bunsetu> ←→ (<np>))
(<bunsetu> ←→ (<wh-np>))
(<np> ←→ (<n> <p>))
(<wh-np> ←→ (<wh-pro> <p>))
(<wh-pro> ←→ (i ts u))
(<wh-pro> ←→ (n a n i))
(<wh-pro> ←→ (n a =))
(<wh-pro> ←→ (d a r e))
(<wh-pro> ←→ (i k u r a))
(<p> ←→ (<p-kaku2> <p-k>)
(<p-kaku2> ←→ (n i))
(<p-kaku2> ←→ (t o))
(<p-kaku2> ←→ (d e))
(<p-kaku2> ←→ (e))
(<p-kaku2> ←→ (w a))
(<p-k> ←→ (w a))
(<p-k> ←→ (d e m o))

<bunsetu>    <np>        <n>
                         <p>      <p-kaku2>
                                  <p-k>
             ------------
             <wh-np>     <wh-pro>
                         <p>      <p-kaku2>
                                  <p-k>
```

図6　認識用文法(一部)

次に，認識文法の効率測定を説明する．処理概要はおよそ次の通りである．

(*a*) 文節単位の言語データを木構造表現された認識用文法で構文解析を行う．

(*b*) 構文解析が成功したら，解析で用いられた生成規則をトレースする．例えば，図7の左側の木構造ｇを認識用文法として記号列ａｘを解析すると，文法カテゴリ列Ａ－Ｂ－Ｄ－ａおよびＡ－Ｂ－Ｅ－ｘがトレースされ，その際，各文法カテゴリを表すノード内のカウンタが1つずつ増える．（ノードD，Ｅ，ａ，ｘは1つカウントされ，ノードA，Ｂは2つカウントされる．）

(*c*) 総ての言語データの解析が終えると，木構造ｇのうちトレースされている枝だけで構成された木構造ｔを求める．

(*d*) 木構造ｇで表現された文法から生成される文節数を求めその集合をG，同様に木構造ｔで表現された文法（効率化文法）から生成される文節の集合をＴとして，$|T|/|G|$ を認識用文法の効率とする．

この測定では，認識文法を表現する木構造ｇのうち言語データベースを構文解析できる最小部分木ｔを求め，ｔから生成される文節集合をタスク領域とみなしている．（ａ）の構文解析において，言語データベースの形態素解析で用いられている文法と認識用文法とが異なるので，文法間の対応表（図8はその1部）を参照しながら解析を進めている．この対応表の左端の数字は言語データベースで用いられている品詞コードおよび活用形コードであり，対応する認識用文法内の文法カテゴリの一覧が後に続く．

今回の文法効率測定では，言語データを終端記号レベルで完全に構文解析することは必要としておらず，文法規則の適用連鎖を多量のデータで把握することを目的としているので，構文解析の途中で言語データ中の単語が認識用文法の終端記号として存在しない場合でも，無矛盾で構文解析できていれば，そのデータが認識用文法で受理されたとしている．その際，複数の部分解析木が存在するが，最も深いレベルを持つ部分木を解析結果として出力している．また，解析の効率を図るため認識文法の名詞の終端記号（音素列）は総て省き，名詞に相当する最上位の文法カテゴリをその終端としている．

図9は構文解析例であり，文節毎に入力データとその解析結果を出力している．解析結果中の文法カテゴリの前の数字は，文法カテゴリで示される文法規則がそれまでに適用された回数を表している．文法カテゴリは文法木構造内では総て異なるノードとして扱われているので，構文解析後の書き換え規則連鎖はこの木構造に完全に保存される．

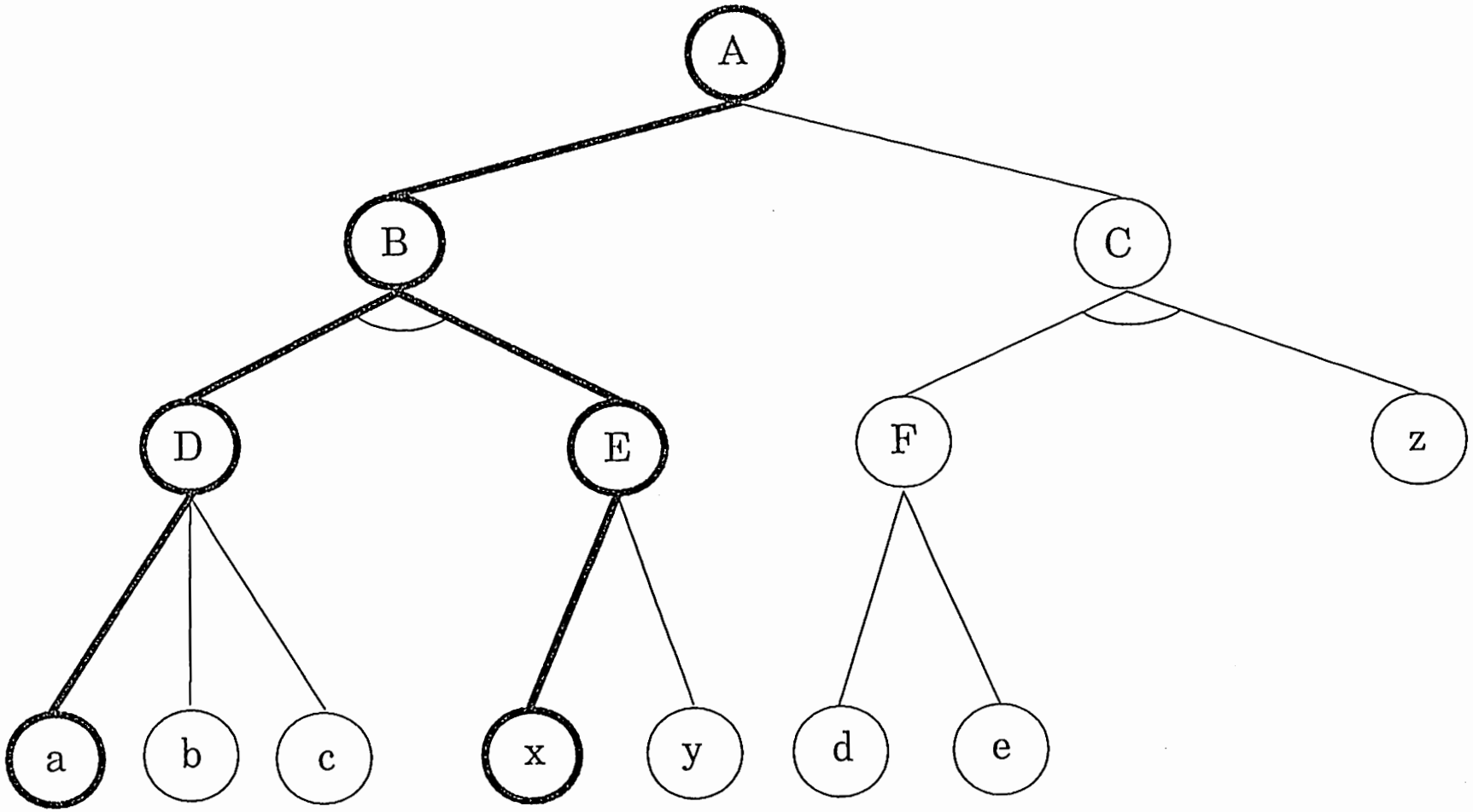

図7　書き換え規則のトレース

12	0	<v-cop-mizen> <masu-mizen1> <masu-mizen2> <caus-seru-mizen> <caus-saseru-mizen> <polt-aux-mizen> <negt-mizen> <optt-mizen> <evid-mizen> <flex-kdo-mizen2>
12	1	<meta-renyo> <v-cop-renyo2> <cop-renyo2> <masu-renyo> <caus-seru-renyo> <caus-saseru-renyo> <deac-reru-renyo> <deac-rareru-renyo> <desu-renyo> <negt-renyo1> <negt-renyo2> <polt-aux-renyo> <evid-renyo1> <evid-renyo2> <evid-renyo-coord> <optt-renyo1> <optt-renyo2> <flex-kdo-renyo1> <flex-kdo-renyo2> <flex-kdo-renyo3> <flex-p-kdo-renyo2> <negt-renyo>
12	2	<v-cop-rentai> <masu-rentai> <caus-seru-rentai> <caus-saseru-rentai> <deac-reru-rentai> <deac-rareru-rentai> <optt-rentai>

図8　文法カテゴリ対応表(一部)

```
もうしこみ  32    1
たい    12    3
の    34    9
です    12    2
が    14    9


[1]: (5)<bunsetu> --- <vaux-s> --- <vaux> --- <v-5dan-i>
                                              <optt-rentai> --- tai
                                              <polt-n-rentai> --- <p-j> --- no
                                                                  <polt-aux-rentai> --- desu
                                  <sp-rentai> --- nga

[2]: (5)<bunsetu> --- <vaux-s> --- <vaux> --- <v-renyo1>
                                              <optt-rentai> --- tai
                                              <polt-n-rentai> --- <p-j> --- no
                                                                  <polt-aux-rentai> --- desu
                                  <sp-rentai> --- nga


[Result]
(6)<bunsetu> ---(1) <vaux-s> ---(1) <vaux> ---(1) <v-5dan-i>
                                                   <optt-rentai> ---(1) tai
                                                   <polt-n-rentai> ---(1) <p-j> ---(1) no
                                                                          <polt-aux-rentai> ---(1) desu
                                  <sp-rentai> ---(1) nga
```

図9　構文解析例

図10に言語データを総て構文解析した後の文法木構造の状態を示す．各文法カテゴリの前の数字は，そのカテゴリが示す書き換え規則が上位のカテゴリによって適用される確率を示している．例えば，文法カテゴリ＜np＞が＜n＞に書き換えられる確率は0.1531,＜n＞＋＜p＞に書き換えられる確率は0.8409である．この確率は構文解析時にストックされていた文法規則適用確率を基に計算された値である．この結果の中で，例えば，文法規則適用連鎖，

＜b u n s e t u＞→＜n p＞→＜n＞＋＜p＞

↓

＜p－k a k u 2＞＋＜p－k＞

と，

＜b u n s e t u＞→＜w h－n p＞→＜w h－p r o＞＋＜p＞

↓

＜p－k a k u 2＞＋＜p－k＞

において，＜$p-kaku2$＞＋＜$p-k$＞が適用される確率は前者では0.0356，後者は0.0と異なっている．

この解析結果より，認識用文法および効率化文法から生成される文節数を計算した結果が表1である．この表では，文節を構成する単語数毎に生成される文節数の分布を示している．認識用文法から生成される文節数は296,504，効率化文法のそれは1,709であり，これより認識用文法の効率は，

$$\frac{1,709}{296,504} = 0.0058$$

となる．

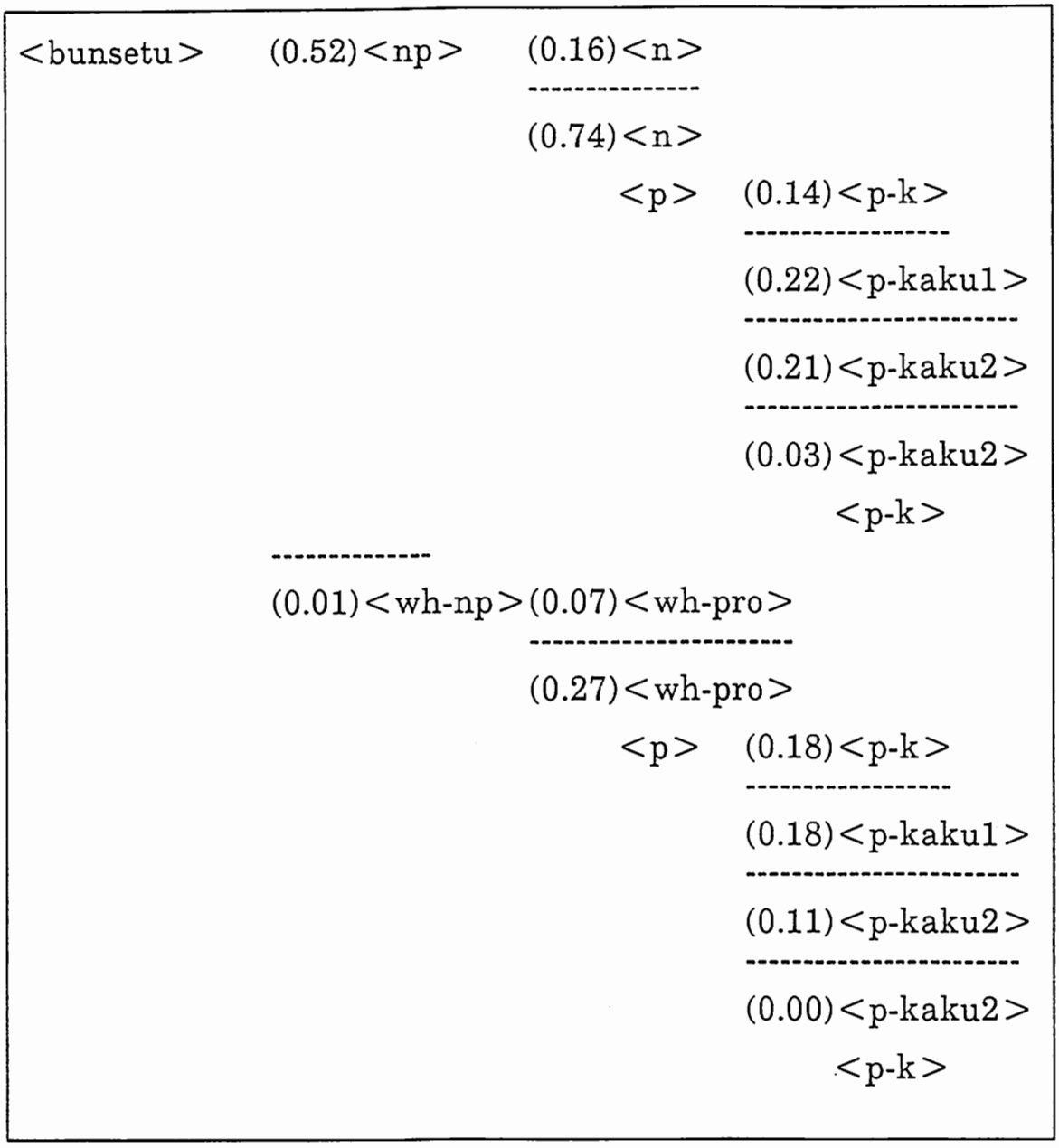

図10　構文解析後の文法木構造

形態素数	認識用文法	効率化文法
1	32	16
2	838	209
3	5,671	384
4	13,003	305
5	35,184	295
6	61,142	232
7	59,494	156
8	63,973	80
9	37,695	28
10	13,432	4
11	5,116	0
12	924	0
合　　計	296,504	1,709

表1　文法により生成される文節数

6　文法効率に関わる要因

文法効率は文法の構成法により決まる．文法効率が低下する要因としては次のようなものが挙げられる．

1. 文法カテゴリが構造間で共有されている．
    - 共有カテゴリ内に未使用語を含む．
    - カテゴリ連接（ａｎｄ結合）のうち不要連接が存在する．
    - 不要な書き換え規則連鎖の存在．
2. 文法規則を等確率で適用する．

7　効率化文法による認識実験

2節では，文法の音声認識への活用が探索空間の限定という意味で効果的であると述べたが，認識用文法に対する効率化文法もまた同様な意味で効率的であると考えられる．これを1節で紹介したHMM－ＬＲを用いて簡単に検証した．

認識用文法に対する効率測定の際に得られた構文解析木から無駄な文法カテゴリやその連鎖を手作業で取り除いた最適化文法を構成した．表2には，この最適化文法によって特定話者音声認識を行った結果と，元の認識用文法による結果を併せて示す．これから認識率・認識失敗数と共に良くなっていることが確認される．

ランク	認識用文法		効率化文法	
	度数	累積	度数	累積
1	81.93%	81.93%	87.95%	87.95%
2	8.43%	90.36%	4.82%	92.77%
3	4.82%	95.18%	2.41%	95.18%
4	0.00%	95.18%	1.20%	96.39%
5	0.00%	95.18%	0.00%	96.39%

表2　音声認識結果

8　音声認識における確率化文法の可能性

前節までで，対象タスクに対する文法の記述による効果・効率を考慮する必要性と，実際に音声認識システムで用いられている音声認識用文法の効率を測定し，その効率が無視できない場合があると述べた.

次に音声認識と言語文法との関わり合いで重要なのが規則適用確率と音声認識確率との融合である.

確率的構文解析に関しては，それが構文解析のあいまいさの除去に有効であると報告されている.

文法駆動による音声認識法では，その構文解析のあいまいさが認識のあいまいさ（雑音）につながるので，確率的構文解析が音声認識にも有効であろうと想像される.

以下では，確率化文法の音声認識への可能性を述べる.

方法１　文法の効率化

この方法は，規則適用確率を基にして，より最適な文法を構成するものである．これは，規則適用確率を符号の構造に変換できるという意味から，情報理論での符号化問題に帰着できる.

例えば，前節での認識用文法に対する効率化文法がこれに相当する．また，ＩＢＭのＢａｈｌらは，言語データを学習することにより情報量（エントロピー）が最小となるような２進木言語モデルを構成して，それによる音韻予測を行っている.

文法の効率化による方法の問題点としては，符号化された文法の適用が飽くまで離散的であることである．つまり，この符号化は規則適用確率の近似的な実現に過ぎない.

方法２　評価関数の設定

規則適用確率 P_G と音声認識確率 P_S を変数として持つ評価関数 $F(P_G, P_S)$ を設定し，新たな認識尤度とする方法である.

確率化文法を用いない音声認識法では，

$$F(P_G, P_S) \quad = \quad P_S$$

を評価関数としていた.

この方法では，評価関数を適当に選ぶことにより，方法1に比べてよりsensitiveな確率文法の適用が可能となる．

評価関数としてたとえば，

$$F(P_G, P_S) \quad = \quad w \times P_G + (1-w) \times P_S \quad (0 \leq w \leq 1)$$

のような重み付けによる関数が考えられるが，音声認識確率特性を損わないような関数を選択する必要がある．

因みに、規則の適用連鎖をTrigramに限定した文法の効率を測定したところ、木構造の文法効率とほぼ同等の結果を得た。このことは、文節内の文法に限っていえば、規則適用連鎖のTrigramによって求めた確率化文法で十分であることを示している。