自然言語処理 117－10
（1997. 1. 20）

複数の接続表の制約のLR表への組み込み－LR表工学(2)

綾部寿樹, 徳永健伸, 田中穂積

東京工業大学大学院情報理工学研究科

概要

GLR法の応用として、隣接する終端記号間の接続関係という局所的制約をLR表に組み込むという研究がある。それにより形態素解析または音声認識と、統語解析を統合して扱うことが可能になる。

本論文では複数のレベルの接続制約を1つのLR表に組み込む方法を提案する。まずCFGの層という概念を導入し、CFGが層を持つ場合に2つの接続制約をLR表に組み込むためのアルゴリズムを提案する。さらに3つ以上の接続制約を組み込むことを述べる。

LR表に隣接音素間の接続制約と隣接形態素間の接続制約を組み込むことで、GLR法の枠組において音声認識から形態素解析、さらに統語解析までを統合して扱うことができることを述べる。

Integrating multiple Connection Constraints into an LR Table and the applications there of － LR Table Engineering(2)

Toshiki Ayabe,Takenobu Tokunaga,Hozumi Tanaka

Tokyo Institute of Technology

abstract

Integrating connection constraints into a GLR parser is one application of the GLR parsing algotithm. Through this, we can combine speech recognition (or morphological analysis) and syntactic analysis. However the method only allows the integration of one level of connection constraints (i.e. the terminal symbols of the CFG). this paper proposes a new method of integrating several levels of connection constraints into a GLR parser. In this way, speech recognition, morphological analysis, and syntactic analysis can be combined. Discussion is given to the advantages of our method and future problems.

1 はじめに

自然言語の統語解析のアルゴリズムとして、一般化LR(GLR)法がある。GLR法は、先読み語の持つ品詞情報を利用しつつ解析を進めるもので、経験的にもっとも効率の良いアルゴリズムであるとされている[1][2]。

GLR法は一般の文脈自由文法(CFG)の持つ制約をLR表の形式にコンパイルし、効率良く統語解析を進めるアルゴリズムである。しかしGLR法は、最近では統語解析のみならず、形態素解析及び音声認識にも応用されるようになってきている。形態素間の接続表の制約をLR表に組み込むことで、形態素解析と統語解析の統合が行なわれている[3][4]。異音間の接続表の制約をLR表に組み込むことで、音声認識システムにも応用されている[5][6]。

このような解析の方法は、人間がどのようにして自然言語や音声を解析しているかを考えた場合、非常に重要である。なぜならば、人間は多種多様な制約を同時に用いて、解析途中で発生する様々な曖昧性を早期に解消していると思われるからである。

ここで、形態素間の接続表や異音間の接続表といった、異なるレベルの複数の接続表の持つ制約をともにLR表に組み込むことが可能であれば、GLR法という1つの枠組で、複数の接続表による接続制約およびCFGによる文法的制約を同時に用いた解析が可能になるので、前述の観点から都合がよい。この方法を田中は制約統合型モデルと呼んでいる [7]。

本稿では、複数の接続表の制約をLR表に組み込む方法を説明する。2章では、制約統合型モデルの重要性を説明する。さらに、局所制約をCFGで記述した場合の問題点を指摘し、田中らの提案する1つの接続表の制約をLR表に組み込む方法の概略を説明する。3章では、2章の方法を応用して、複数の接続表の制約をLR表に組み込む方法を説明する。4章では本稿の手法の応用例を述べる。

2 接続表の制約のLR表への組み込み

2.1 制約統合型モデル

これまでの日本語の解析の多くは、解析の各レベルごとに異なる解析モジュールを用意して、それをカスケード型に結合して行なう方法が中心であった。すなわち、前段階の解析の出力を次の解析モジュールの入力として、順に行なっていく方法である。

例をあげると、分かち書きのなされていない日本語のテキストを統語解析する場合がある。カスケード型モデルにおいては、入力テキストをまず形態素解析モジュールで分かち書きし、分かち書きされたテキストを統語解析モジュールの入力とする。しかし、一般に形態素解析は接続表のみを利用するためその制約は十分でなく、多数の分かち書きされたテキストを生成してしまう。これは統語解析モジュールの負担を大きくすることにつながり、解析が非効率的になってしまうので好ましくない [7]。

一方、1章で述べたように、人間は自然言語を理解する際に多種多様な制約を同時に用いて、解析途中で生じる曖昧性をできるだけ早期に解消していると考えられる。すなわち、前記の例で言えば、分かち書きされていない日本語のテキストを解析する場合にも、人間ならば統語的な制約を満たさない単語の並びは早期に切り捨てるであろう、ということである。

カスケード型モデルにはこのような欠点があるため、可能なら、異なるレベルの制約を同時に使う制約統合型モデルへと移行していくことが必要であると考えられる。

2.2 CFGによる局所的制約の記述の限界

自然言語の統語解析に用いる文法的枠組として、CFGがある。一方、形態素解析では形態素間の接続可能性という局所的な制約がよく用いられる。この形態素間の接続可能性という制約をCFGで記述することは、原理的には可能であるがCFG規則の数が不必要に増え、CFG規則の体系が複雑化するという問題がある。CFG記述の立場から解決すべき問題は以下の3つである [8]。

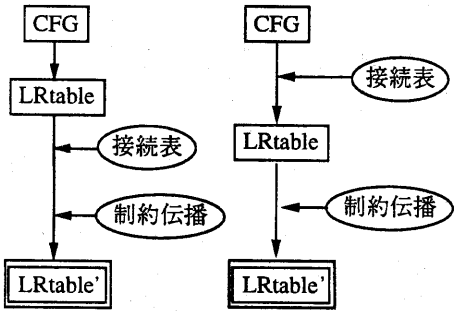
1. CFG記述者は、接続可能性を考慮した新たな非終端記号の導入を行なうことなしに、CFGの記述が可能であること
2. 接続制約の記述者は、CFG規則とは無関係に制約を記述可能であること
3. 局所的制約検査のタイミングは、なるべく早く行なうこと

これらを同時に解決する方法として、CFG規則とは独立に局所的制約を接続表という形で記述しておき、この制約をCFG規則から得られたLR表に組み込む方法を提案している [8]。

田中らは2つの方法を提案している。第一は、CFG規則からLR表を生成した後で接続表の制約を組み込む方法であり (図1(1))、第二はLR表を生成する際に接続表の制約を組み込む方法である (図1(2))。前者には既成のLR表生成アルゴリズムをそのまま用いることが出来るという長所があるが、規則数が数千の規模になると接続表の制約を組み込む前のLR表のサイズが巨大になり、実用に耐えられない。例えば、規則数約1000のCFGを用いて正準LR表を生成すると、その状態数は10000を越えてしまうことが報告されている [9]。また、正準LR表に比べ

て状態数が少なくてすむとされるLALR表を用いてさえ、規則数2655のCFGから表を作成した場合、約25MBにも達することも報告されている[9]。したがってここでは後者の方法を説明する。

なお本章以降ではギリシャ文字の$\alpha, \beta, \ldots$などは次のいずれかの記号列を表す：終端記号の列、非終端記号の列、終端記号と非終端記号の混在した列。



(1) (2)

図1:2つのアルゴリズム

2.3 接続表

終端記号の集合$V_t(=\{v_1, v_2, \ldots, v_n\})$の各要素間の接続可能性についての制約を$connect(v_i, v_j)$という関数を用いて表すことが出来る。すなわち以下のように定義する。

1. 記号v_i, v_jがこの順に接続可能なら、
   $connect(v_i, v_j) = 1$
2. 記号v_i, v_jがこの順に接続不可能なら、
   $connect(v_i, v_j) = 0$

ここで定義した関数*connect*は、終端記号の集合$V_t(=\{v_1, \ldots, v_n\})$に対して図2のような表にすることが出来る。

RIGHT

LEFT	v_1	v_2	$\cdots$	v_i	$\cdots$	v_j	$\cdots$	v_n
v_1								
v_2								
$\vdots$								
v_i			$\cdots$	1		1		
$\vdots$								
v_j			$\cdots$	0		1		
$\vdots$								
v_n								

図2:接続表

この表を接続表と呼ぶ。

2.4 制約を組み込んだLR表生成アルゴリズム

2.4.1 GOTOグラフ作成時のアイテム削除

(1)V $\rightarrow \gamma\ v_i$ (5)X $\rightarrow \alpha$ V
(2)V $\rightarrow \delta\ v_j$ (6)Y $\rightarrow$ W β
(3)W $\rightarrow v_i\ \zeta$ (7)Z $\rightarrow$ X Y
(4)W $\rightarrow v_j\ \eta$

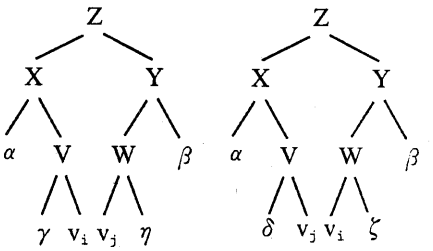


図3:文法

例えば図2の接続表を図3のCFGに組み込む場合を考える。このとき、図3に示すCFGから通常通りLR表をする過程で以下のアイテム

$$[V \rightarrow \cdot\gamma v_j; v_i]$$

が生成される。しかし、図1の終端記号の接続表からv_j, v_iのこの順での接続は許されていないので、このアイテムは生成されるべきではない。

このような、接続表の制約を利用し、LR表の作成時に不要なアイテムを削除する方法を次のように述べることができる。

[右矢印(→)の直右にドット記号を持つ
アイテムの生成手続き(closure生成手続き)]

与えられたCFGと接続表とを用いて、次の1,2の手続きにしたがってアイテムを生成する。ただし記号Xを根とする解析木の右分枝の先端に現れる記号の集合は関数Last(X)を実行して得られるものとする。また記号Xを根とする解析木の左分枝の先端に現れる記号の集合は関数First(X)を実行して得られるものとする。特に記号Xが終端記号の場合には、First(X)=Last(X)={X}である。

1. アイテム$[V \rightarrow \cdot\xi X; v_i]$は次の条件を満たす時のみ生成する。
   - Last(X)に属する要素とv_iの対のうち、この順に接続可能な対が存在する(図4参照)。

2. 記号 X をシフトして到達した状態において、アイテム $[V \rightarrow \cdot W\xi; v_i]$ は、条件1を満たしかつ次の条件を満たす時のみ生成する。

   - Last(X) に属する要素と First(W) に属する要素の対のうち、この順に接続可能な対が存在する (図5参照)。

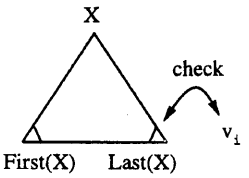


図 4:接続のチェック 1

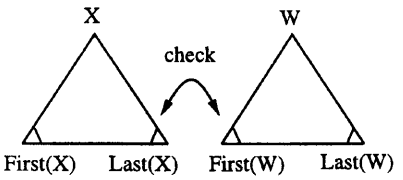


図 5:接続のチェック 2

接続表は一般にスパースであるため接続を禁止されているペアが多いので、上記の手続きにより closure 生成時のアイテム生成が相当に抑制される。アイテムの大量な削減は、LR 表の大量の状態数の削減に直結する [8]。

2.4.2 制約伝播

以上の方法により作成した LR 表中のアクションをさらに削減することができる。これを制約伝播と呼ぶ。

制約伝播によってアイテムが削除される場合は、以下の2通りにまとめられる。

1. LR 表中のあるアクション Act を実行した直後に実行すべきアクションが一つもない場合
2. LR 表中のあるアクション Act を実行する直前に実行すべきアクションが一つもない場合

場合1においては、Act の実行直後に明らかにエラーが生じる。よってこの Act は実行する意味がないので削除してよい。また場合2においては、この Act は永久に実行されない。よってこの Act を削除してよい。このようにして LR 表中のアクションを削除していく。

以上の操作をアイテムが削除できなくなるまで繰り返し行なうことにより、制約が伝播する。

本節のアルゴリズムにより使用記憶空間を大幅に削減し、最終的な LR 表を得る時間を一桁以上短縮できることが報告されている [5]。

3 複数の接続表の制約の LR 表への組み込み

前章において、終端記号の接続表の制約を LR 表に組み込む方法を説明した。この方法により、形態素解析と統語解析、もしくは音声認識と統語解析を統合することが可能であることが示されている [3][6]。

ここで、異なるレベルの複数の接続表の制約を LR 表に組み込むことが可能であれば、音声認識、形態素解析及び統語解析の3つの解析を GLR 法という一つの枠組に統合することが可能になる。

隣接形態素間の接続制約と隣接異音間の接続制約という2つの接続制約 (接続表) を LR 表に組み込む場合を考える。辞書項目を異音と音素の列として記述すると[1]、異音が終端記号になるので、異音間の接続表の制約は前章の方法で組み込むことが可能である。しかし、LR(1) アイテムの先読み記号が異音であるため、異音とはレベルの異なる形態素間の接続表を組み込もうとしても、組み込むことができない。

異なる2つのレベルの接続制約を組み込むためには、対象となる CFG に本章で説明する「文脈自由文法の層」が存在することが必要である。また、アイテムの先読み記号が常に終端記号であることが不都合になるので、新たなアイテムの型を定義することも必要である。

これらを踏まえて、本章では2つの接続表の制約を LR 表へ組み込む方法を説明する。またその方法が3以上の接続表の制約を LR 表へ組み込むことへ応用できることを述べる。

[1] $N \rightarrow \mathrm{a}\ k_1\ \mathrm{i}$ など。ここで k_1 は a と i で前後をはさまれた音素 k の異音である。語頭と語末の音素 a と i は、それぞれ前後の音素が不明なため異音化できない。

3.1 文脈自由文法の層

まずはじめに文脈自由文法の層という概念を説明する。層を持つ CFG というのは、CFG 規則の集合を、開始記号からあるレベル L2 の記号列（例えば形態素列）を導出する規則と、そのレベル L2 の記号列から終端記号の記号列（例えば異音列）を導出する規則とに重なりなく分けられる CFG のことである。図 6 に例を示す。

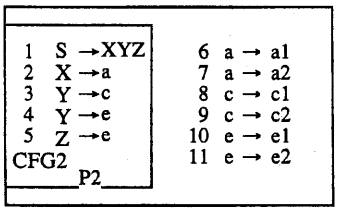


図 6:層を持つ CFG

例えば図 6 の CFG1 は、CFG2 という層を持っている。規則集合 P1 が、規則集合 P2 と P1−P2 の部分に重複なくわかれていて、P2 では開始記号から記号 a,c,e の記号列、P1−P2 では記号 a,c,e の記号列から a1,c1 などの記号列を導出する。

なお、これから先の説明において、CFG1 が CFG2 という層を持つものとし、CFG1 の終端記号の集合を Vt1、CFG2 の終端記号の集合を Vt2 とする。また、CFG1 規則の集合を P1、CFG2 規則の集合を P2 と呼ぶこととする。

3.2 LR(1) アイテムの拡張

GLR 法でアイテムを生成する場合、その先読み記号は終端記号すなわち Vt1 に属する記号であった。したがって、例えばアイテム $[V \rightarrow \cdot\xi X; v_i] : v_i \in$Vt1 の生成時に、$X$ と v_i が Vt2 のレベルで接続可能かどうかをチェックすることはできない。よってアイテムの定義を拡張する必要がある。

ここでは、先読み記号として Vt2 に属する記号をとるアイテムと、先読み記号として Vt1 に属する記号をとるアイテムを別々に定義するという方法をとる。前者を型 2 のアイテム、後者を型 1 のアイテムと呼ぶ。

[型 2 のアイテムの定義]

以下のようなアイテムを型 2 のアイテムとする。

$[X \rightarrow \alpha \cdot \beta; v]$

ただし $v \in$ Vt2, $X \rightarrow \alpha\beta \in$ P2

[型 1 のアイテムの定義]

以下のようなアイテムを型 1 のアイテムとする。

$[X \rightarrow \alpha \cdot \beta; v_i]$

ただし $v_i \in$ Vt1, $X \rightarrow \alpha\beta \in$ P1−P2

3.3 2 つの接続表の制約の LR 表への組み込み

CFG1 が CFG2 という層を持つ場合、2 つの接続表の制約を LR 表に組み込むことが出来る。その概略を説明する。

まず、CFG2 に対して、Vt2 に属する記号間の接続可能性の制約 (接続表 2) を組み込んだ GOTO グラフを作る。この GOTO グラフを構成するアイテムはすべて型 2 のアイテムである。そして次に P1−P2 の規則と Vt1 に属する記号間の接続表 (接続表 1) を用いて GOTO グラフを拡張する。この拡張時に追加されるアイテムはすべて型 1 のアイテムである。

(1)CFG2 における接続表 1 と接続表 2 の制約を同時に組み込んだ closure の生成

CFG2 における closure を生成し、GOTO グラフ G2 を作成する。ただし右辺の最左にドットがあるアイテムの生成には以下のアイテム生成手続き 1 を用いる。

[アイテム生成手続き 1]

与えられた CFG2 と接続表 1 と接続表 2 を用いて、次の 1,2 にしたがってアイテムを生成する。ただし記号 X を根とする解析木の右分枝の先端に現れる Vt2 に属する記号の集合は、関数 Last2(X) を実行して得られるものとする。また記号 X を根とする解析木の左分枝の先端に現れる Vt2 に属する記号の集合は、関数 First2(X) を実行して得られるものとする。特に記号 X が Vt2 に属する記号の場合には、First2(X)=Last2(X)={X} である。

1. アイテム $[V \rightarrow \cdot\xi X; v] : v \in$Vt2 の生成は以下の条件を満たす場合にのみ行なう。
   - Last2(X) に属する要素と v の対のうち、接続表 2 においてこの順に接続可能な対

が存在し、かつその対を u,v としたとき、Last(u) に属する要素と First(v) に属する要素の対のうち、接続表 1 においてこの順に接続可能な対が存在する。

2. 記号 X をシフトして到達した新しい状態におけるアイテム $[V \rightarrow \cdot W\xi; v] : v \in$Vt2 の生成は、条件 1 を満たしかつ以下の条件を満たす場合にのみ行なう。

   - Last2(X) に属する要素と First2(W) に属する要素の対のうち、接続表 2 においてこの順に接続可能な対が存在し、かつその対を x,w とした時、Last(x) に属する要素と First(w) に属する要素の対のうち、接続表 1 においてこの順に接続可能な対が存在する。

このアイテム生成法は、2 章の方法と基本は同じである。相違点は次の 2 つである。

1. 対象が CFG2 であるため、終端記号のかわりに Vt2 に属する記号を用いている。よって生成されるアイテムは型 2 のアイテムである

2. 接続表 2 の制約をかけると同時に接続表 1 による制約をもかけている

以上で GOTO グラフ G2 が作成される。

(2)P1–P2 の規則を用いた拡張

(1) で作成した GOTO グラフ G2 を型 1 のアイテムへ展開する。

G2 の各状態の、ドットのすぐ右に Vt2 に属する記号を持つアイテムを核として、P1–P2 に属する規則の closure を生成し、GOTO グラフ G1 を作成する。ただしアイテム生成手続きとしては以下のアイテム生成手続き 2 を用いる。

[アイテム生成手続き 2]

与えられた CFG と接続表 1 を用いて次の 1,2 を満たす場合のみアイテムを生成する。

1. アイテム $[V \rightarrow \cdot\xi X; v_i] : v_i \in$Vt1 の生成は次の条件を満たす場合にのみ行なう。

   - Last(X) に属する要素と v_i の対のうち、この順に接続可能な対が存在する。

2. 記号 X をシフトして到達した状態におけるアイテム $[V \rightarrow \cdot W\xi; v_i] : v_i \in$Vt1 の生成は、条件 1 を満たしかつ次の条件を満たす場合にのみ行なう。

   - Last(X) に属する要素と First(W) に属する要素の対のうち、この順に接続可能な対が存在する。

ここで得られた GOTO グラフ G1 には、接続表 1 と接続表 2 の 2 つの接続制約が組み込まれている。

(3)LR 表の生成

G1 から、LR 表を以下の点に留意しつつ作成する。

1. Vt2 に属する記号のシフトは GOTO である。

2. レデュース用のアイテムの先読み記号が Vt2 に属する記号 w の場合、そのアイテムの規則の右辺最右の記号を X とすると、First(w) に属する記号のうち、接続表 1 において Last(X) に属する記号のいずれかと接続可能な記号すべてを先読み記号としたレデュースアクションを作成する。

(4) 制約伝播

(3) で得られた LR 表に制約伝播を施す。これは 2 章の方法と同一である。

以上のようにして 2 つの接続制約を組み込んだ LR 表の作成が可能であるが、同様なことを繰り返すことにより、3 以上の接続表の制約を LR 表に組み込むことが可能になる。

4 応用例

前章では、複数の接続表の制約を組み込んだ LR 表を作成する方法を説明した。本章ではその方法を応用することで可能になると思われる応用例について述べる。

4.1 HMM-LR への応用

音声認識に GLR 法を用いる HMM-LR においては、異音間の接続表を組み込んだ LR 表を作成している。具体的には図 7 に示す構造を持つ [9]。

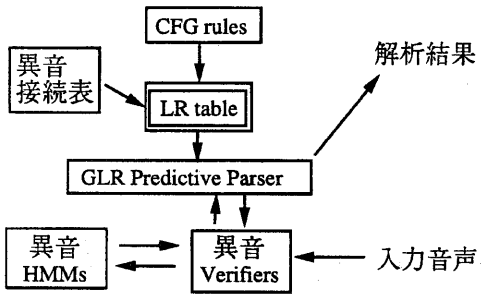


図 7:HMM-LR システム

HMM-LR では、2 章の方法を用いることで終端記号、すなわち異音の接続表を組み込んだ LR 表を作成し、LR 表の状態における表中のアクションのある先読み記号を予測異音としている。接続表を組み込むことによって予測異音を絞りこみ、音声認識をスムーズに行なうことを可能にしている [10][11]。

しかし、3 章の方法を用いることで、異音間の接続表のみならず、さらに形態素間の接続表を組み込んだ LR 表を作成することができる。3 章の方法を用いて形態素、異音両方の接続表を組み込んだ HMM-LR システムの構造は、図 8 に示す構造になる。

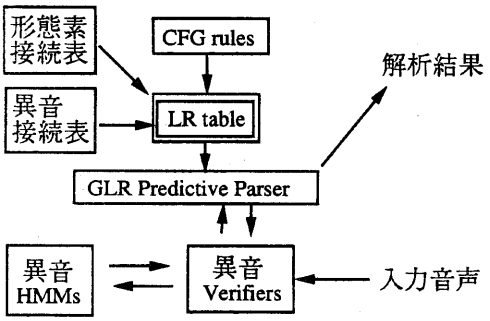


図 8:modified-HMM-LR システム

この場合、形態素レベルの接続制約が組み込まれているので、統語的または異音列的には可能であっても形態素列的には不可能な並びを受理しないような LR 表を生成することが期待できる。そのことは予測異音の特定にも大いに役立つと考えられるため、音声認識の精度向上にも役立つであろう。

また、2,3 章において接続表の値は 2 値 {0,1} であるが、李は接続表の値として bigram 確率をふって確率 LR に応用する方法を提案している [9]。この方法も接続表を複数組み込む場合についても応用可能であると考えられる。

現在我々は、このような方向での応用を検討している。

5 おわりに

本稿では複数の接続表の制約の LR 表への組み込みが可能である事を示した。この方法によって、4 章で述べたように形態素間の接続表、異音間の接続表を両方組み込み、音声認識、形態素解析、統語解析をすべて統合することができる。2 章の方法にさらに制約を加えた方法になるので、2 章の方法よりもさらに LR 表のサイズを小さくし、本当に必要なアクションだけを残した LR 表を作成することが可能であると考えられる。

われわれは、すでにこの方法による LR 表生成プログラムをワークステーション上に実装することに成功している（使用言語は C である）。

今後なすべきことは、大規模な CFG に対して実際に接続表を 2 つ組み込んだ LR 表を作成し、本稿で述べたことを裏付けることであろう[2]。

謝辞

この論文を書くにあたり、プログラムのテストのために多くの文法を提供して下さったランゲージウェアの衛藤さんに大いに感謝致します。また昨年度東京工業大学大学院博士課程を卒業された李さんには研究に関する助言など数多く賜わりました。この場を借りて深く感謝の意を表します。

参考文献

[1] A.V. Aho, S. Ravi, and J.D. Ullman. *Compilers, Principle, Techniques, and Tools.* Addision Wesely, 1986.

[2] M Tomita. *Generalized LR Parsing.* Kluwer Academic Publishers, 1991.

[2]正確には文法規則数 2000 程度の CFG についての実験は行ない、動作を確認したが、その際には、異音の列を導出する CFG に対して、非終端記号のレベルにあたる形態素の接続表のみを組み込んだ実験しか行なっていない。このこと自体も新しい試みではあるが、2 つの接続表を同時に組み込む利点を実証したことにはなっていない。

[3] 植木正裕, 徳永健伸, 田中穂積. ＥＤＲ辞書を用いて形態素解析と統語解析を行なうシステム. EDR 電子化辞書利用シンポジウム論文集, pp. 33–39, 1995.

[4] H. Tanaka, T. Tokunaga, and M. Aizawa. Integration of morphological and syntactic analysis based on lr parsing algorithm. *Journal of Natural Language Processing*, Vol. 2, No. 2, pp. 59–74, 1995.

[5] Suresh K.G. Li H. and Tanaka H. Incorporation of connection constraints into generation process of allophone-base lr table. 情報処理学会第 50 回全国大会講演論文集, 1995.

[6] 田中穂積, 李輝, 徳永健伸. Incorporation of phoneme-context-dependence in lr table through constraint propagation method. 人工知能学会第 8 回言語・音声理解と対話処理研究会, pp. 15–22, 6 1994.

[7] 田中穂積. パージングー制約統合型モデルの提案ー. 人工知能学会誌 vol.11 No.4, pp. 507–513, 1996.

[8] 田中穂積, 李輝, 徳永健伸. 自然言語解析の新しい方法－ＬＲ表工学の提案 (1). 人工知能学会, 1995.

[9] Li.H. Integrating connection constraints into a glr parser and its applications in a continuous speech recongnition system. TR96-0003, 1996.

[10] 北研二, 川端豪, 斉藤博昭. HMM 音韻認識と拡張 LR 構文解析法を用いた連続音声認識. 情報処理学会論文誌, Vol. 31, No. 3, pp. 472 – 480, 3 1990.

[11] 伊藤克亘, 速水悟, 田中穂積. 音素文脈依存モデルと高速な探索手法を用いた連続音声認識. 電子情報通信学会論文誌, Vol. J75-D-II, No. 6, pp. 1023–1030, 6 1992.