係り受け文脈自由文法とその日本語への適用

田辺利文[†] 冨浦洋一[†] 日高達[†]

自然言語処理における構文解析では，一般に入力文に対応する構文構造は複数存在し，その中から意味的に正しい構文構造を選択することが重要である．意味的に正しい構文構造を選択するための解決策として係り受け制約を用いる方法が考えられる．本論文では文脈自由文法の生成規則として係り受け制約を記述する文法（係り受け文脈自由文法）を提案する．この文法は非終端記号をそれから導出される句の head（その句の主要な意味を担う概念）とその句の function（係りの種類を規定する情報）で細分化して係り受け制約を表現したものである．さらにこの文法を具体的に日本語に適用した場合の日本語係り受け文脈自由文法の構成法について述べる．

Context Free Grammar Expressing Dependency Constraint and Its Application to Japanese Language

Toshifumi Tanabe,[†] Yoichi Tomiura[†] and Toru Hitaka[†]

In Natural Language Processing, there are lots of syntactic trees corresponding to an input sentence. It is important how to choose the correct one among these syntactic trees, and in this case, it is effective to use Dependency Constraint. This paper presents a Context Free Grammar expressing Dependency Constraint, whose set of nonterminals is given by subdividing syntactic categories according to their **head**s and **function**s. A head of a phrase is the main concept of the phrase, and a function of a phrase is the information prescribing a dependency type of the phrase. Furthermore, this paper shows how to construct a concrete Context Free Grammar expressing Dependency Constraint to Japanese language.

1. はじめに

自然言語処理における構文解析では，一般に入力文に対応する構文構造が複数存在し，それらからどのようにして最適な構文構造を選択するかが問題点の１つである．構文構造の中には意味的に不適格なものも含まれるため，意味処理による意味的に適格な構文構造の絞りこみが重要になる．実用的な意味処理の構文解析への導入として係り受け制約を用いる方法がある．

現在，構文解析には効率的な構文解析法の存在などから文脈自由文法が広く用いられている．従来，文脈自由文法では，入力文がどのような統語構造をしているかを求めるために，どの統語範疇がどういった順序で結び付いて新たな統語範疇を形成するかという統語制約をもとにして文法を作成していた．そのような文法では，統語的には合っているが意味的に不適格であるものの解析をも許してしまうことになる．このような文脈自由文法を用いた構文解析に係り受け制約を適用する方法としては，構文解析途中の部分木の構文構造をもとに，逐次係り受け関係を抽出し，それらが意味的に適格な係り受け関係にあるかどうかを調べ，意味的に適格でない構文構造を構文解析の途中で排除する手法があった．

しかし，従来手法には確率化した場合における整合性の欠如という問題点があった．つまり，大量の言語データを反映させて構文構造の絞りこみの質を上げる手法として確率文法が用いられているが，この場合には確率文法による統語制約の満足度（確率）と係り受け制約の満足度を組み合わせて構文構造の順位つけを行う場合，確率文法による構文構造の確率が高いが係り受け制約をあまり満足していないものと，構文構造の確率は低いが係り受け制約を満足している場合，どちらの構文構造を優先するかが自明ではなかった．

この問題点を解決するには，係り受け制約を文脈自由文法の生成規則として記述する方法が考えられる[11]．確率文脈自由文法の生成規則は係り受け制約を表す生成規則と統語制約を表す生成規則に分けられ，しかも，この文法による構文木の確率は構文木を構成するすべての生成規則の適用確率の積で表されるので，この値

† 九州大学大学院システム情報科学研究科
Graduate School of Information Science and Electrical Engineering, Kyushu University

のみで統語制約および係り受け制約の満足度を考慮した構文構造の絞りこみができるようになる．

係り受け制約を記述することができ，確率化が容易な文法として，TAG（木接合文法）などが考えられたが[1]，強力な構文解析法はまだ開発されておらず，機械処理上での重大な問題点であった．文脈自由文法はTAGに比べて強力な構文解析法が存在し，しかも，係り受け制約を文脈自由文法の生成規則に記述することが不可能であるとは証明されていなかった．

そこで本論文では，係り受け制約を文脈自由文法の生成規則として表現した文法（係り受け文脈自由文法）と，この枠組みを日本語に適用した場合の日本語係り受け文脈自由文法の構成法について述べる．

2. 係り受け制約と文脈自由文法への組み込み

2.1 係り受け制約

句 γ は，句 α と，α を修飾するいくつかの句 $\beta_1 \cdots \beta_l$ から構成されるものとすると，句 α は句 γ 全体の意味を代表する句である．ここで句 α を句 γ の head phrase と定義する．

句が他の句を修飾するときには一般に修飾する方の句の中にその修飾の種類を規定する情報があり，これを句の function と定義する．function としては日本語文では助詞や文節末の活用語の活用形が，英語では前置詞や位置情報☆などがあげられる．

たとえば，“frog in the box” は，“frog” と “in the box” から構成され，“in the box” が “frog” を修飾している．したがって “frog in the box” の head phrase は “frog” であり，“in the box” の function は “in” である．また「リンゴを食べる」は,「リンゴを」と「食べる」から構成され,「リンゴを」が「食べる」を修飾している．したがって「リンゴを食べる」の head phrase は「食べる」であり,「リンゴを」の function は係りの種類を規定している「を」である．

X を root node に持つ部分木において，X の head phrase が α である場合

- α が終端記号のとき，α
- α が非終端記号のとき，α を root node とする部分木の head word

を X の head word と定義する．head word は，その句の意味を代表する語になる．また，

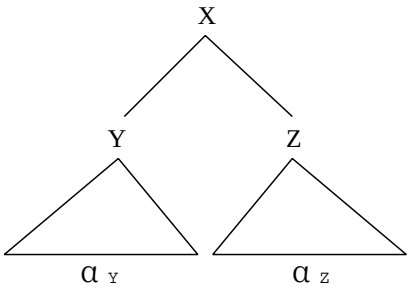

図 1 文脈自由文法による構文木
Fig. 1 A syntactic tree based on Context Free Grammar.

$$X \longrightarrow Y_1 \quad \cdots \quad Y_{i-1} \quad Z \quad Y_i \quad \cdots \quad Y_l \quad (1)$$

を頂点からの書き換えに適用した X を頂点とする構文木において，$1 \leq j \leq l$ なる j に対して，Y_j の head word が w_j, Y_j の function が f_j，Z の head word が w であるとき，w_j は f_j を介して w に構造的に係っている（構造的な係り受け関係にある）と定義する．

図 1 の文脈自由文法の構文木において，句 α_Y と句 α_Z の間に意味的に適格な修飾関係（α_Y が修飾句，α_Z が被修飾句つまり head phrase）が成立しているとすると，Y の head word と Y の function および Z の head word にはある一定の意味的な制約（係り受け制約）が成立している．構造的な係り受け関係のうち，修飾句の head word，function，被修飾句の head word が係り受け制約を満足しているものを，意味的に適格な係り受け関係と呼ぶ．

2.2 係り受け制約の文脈自由文法への組み込み

従来の構文解析に用いられてきた文脈自由文法の非終端記号は，名詞句，動詞句といった統語範疇に設定されていた．このような文法の生成規則，たとえば，日本語における後置詞句（名詞句に格助詞が接続した句）PP と動詞句 VP が結び付いてまた動詞句となることを表現する生成規則

$$VP \longrightarrow PP \quad VP$$

では，右辺の PP と VP からの導出は独立に行われるので，このままでは PP の head word と PP の function と右辺の VP の head word に対する係り受け制約を生成規則の形で表現できず，意味的に不適格なものを導出する可能性がある．これを防ぐためには，それぞれの統語範疇からの導出が他方の統語範疇からの導出に制限を加えるような機構を生成規則に設けるとよい．これは係り受け制約を生成規則の形で表現することを意味する．

そのために，従来用いられてきた統語範疇をその句

☆ 英語における生成規則の中には，非終端記号の並び方で名詞句の格を規定するものがある．たとえば，動詞句の前に位置する名詞句は主格になり，生成規則，$S \rightarrow NP \quad VP$ の NP は主格になる．

の head word の概念（本論文ではこれを head[☆1]と呼ぶ）と function で細分化したものを非終端記号とする[☆2]．まず，細分化された各非終端記号が以下のような意味を持つものと定義する．

$X(h,f)$　head が h であり，function が f である統語範疇 X の句を導出する非終端記号

$X(-h)$　head が h である句に係りうる統語範疇 X の句を導出する非終端記号

$X(h)$　head が h である統語範疇 X の句を導出する非終端記号

$X(f)$　function が f である統語範疇 X の句を導出する非終端記号

従来の統語範疇は，head のみを取りうるもの，function のみを取りうるもの，head および function の両方を取りうるもの，の 3 種類に分類される．つまり統語範疇 X が NP（名詞句）のときは $NP(\alpha)$ の α は head であり，P（前置詞）のときは $P(\alpha)$ の α は function であり，PP（前置詞句）のときは，$PP(\alpha,\beta)$，$PP(-\gamma)$ の α，γ は head，β は function である[☆3]．

したがって，生成規則 (1) のようないくつかの句が 1 つの句を修飾して大きな句を構成する生成規則に代えて，統語範疇 X の head になりうるすべての h について

$$X(h) \longrightarrow Y_1(-h) \cdots Z(h) \cdots Y_l(-h) \quad (2)$$

なる形態の規則を用意し，$Y_j(-h)$（$1 \le j \le l$）に対し，統語範疇 Y_j の head，function になりうる $\langle h', f\rangle$ のうち，h に係りうる（$\langle h', f, h\rangle$ が意味的に適格である）すべての $\langle h', f\rangle$ に対して，

$$Y_j(-h) \longrightarrow Y_j(h', f) \quad (3)$$

なる形態の規則を用意する[☆4]．生成規則 (3) は，h' が f を介して h に係りうる，つまり係り受け制約を表している．また，日本語文法における

$$PP \longrightarrow NP \quad P$$

（P は格助詞を function として導出するような非終端記号）のような修飾句となりうる句を構成する生成規則

$$X \longrightarrow Y \quad Z$$

に代えて，統語範疇 X の head になりうるすべての h および function になりうるすべての f に対して，

$$X(h,f) \longrightarrow Y(h) \quad Z(f) \quad (5)$$

なる規則を用意する．

【例 1】以下のような従来の文法

$NP \longrightarrow Adj \quad NP$
$NP \longrightarrow$ 車
$NP \longrightarrow$ 桃
$Adj \longrightarrow$ 速い
$Adj \longrightarrow$ 甘い

では「速い車」や「甘い桃」の他に「速い桃」や「甘い車」も導出してしまうが，以下のような本論文で提案する形態の文法

$NP(\overline{車}) \rightarrow Adj(-\overline{車}) \quad NP(\overline{車})$
$NP(\overline{車}) \rightarrow$ 車
$Adj(-\overline{車}) \rightarrow Adj(\overline{速い},\overline{連体})$
$Adj(\overline{速い},\overline{連体}) \rightarrow Adj$語幹$(\overline{速い})\ Adj$語尾$(\overline{連体})$
Adj語幹$(\overline{速い}) \rightarrow$ 速
Adj語尾$(\overline{連体}) \rightarrow$ い
$NP(\overline{桃}) \rightarrow Adj(-\overline{桃}) \quad NP(\overline{桃})$
$NP(\overline{桃}) \rightarrow$ 桃
$Adj(-\overline{桃}) \rightarrow Adj(\overline{甘い},\overline{連体})$
$Adj(\overline{甘い},\overline{連体}) \rightarrow Adj$語幹$(\overline{甘い})\ Adj$語尾$(\overline{連体})$
Adj語幹$(\overline{甘い}) \rightarrow$ 甘
Adj語尾$(\overline{連体}) \rightarrow$ い

☆1 head word に対応する品詞が複数ある場合，また head word が多義語である場合には，それぞれ別の head となる．

☆2 非終端記号を細分化することで文法が精密になるのは自明である．ここでの問題は，係り受け制約を，非終端記号を細分化することで CFG の生成規則として表現できるか，またその場合，どのように細分化すればよいかということであり，本論文ではこれに対する 1 つの解を与えている．

☆3 言語学では PP の主辞はその前置詞とするのが一般的であるが，係り受け制約を表現するためには，前置詞の目的語の名詞句の head も必要であり，本論文では名詞句の head を PP の head にしており，本来の主辞である前置詞を PP の function にしている．

☆4 生成規則 (2)，(3) の代わりに，

$$X(h) \rightarrow Y_1(h_1,f_1)\ Y_2(h_2,f_2) \cdots Z(h) \cdots Y_l(h_l,f_l) \quad (4)$$

でもよさそうであるが，この生成規則で表現される制約は多項の共起制約であり，文法を確率化した場合に推定される生成規則の適用確率の信頼性が低下する可能性がある．係り受け制約を二項の共起制約として表現する方法として生成規則 (4) の代わりに，

$$\begin{aligned}
X(h) &\longrightarrow Y_1(h_1,f_1) \quad Z_1(h)\\
Z_1(h) &\longrightarrow Y_2(h_2,f_2) \quad Z_2(h)\\
&\vdots\\
Z_{i-1}(h) &\longrightarrow Y_i(h_i,f_i) \quad Z_i(h)\\
Z_i(h) &\longrightarrow Z(h) \quad Z_{i+1}(h)\\
Z_{i+1}(h) &\longrightarrow Y_{i+1}(h_{i+1},f_{i+1}) \quad Z_{i+2}(h)\\
&\vdots\\
Z_l(h) &\longrightarrow Y_l(h_l,f_l)
\end{aligned}$$

（ただし $Z_1(h) \cdots Z_l(h)$ は，生成規則 (4) を含む文法で使われていない非終端記号）とすることも考えられるが，従来の統語範疇による生成規則から chomsky 標準形への変換を必要とするうえ，生成規則の数も大きくなってしまう．それらの解決法として，生成規則 (4) の非終端記号 $Y_i(h_i,f_i)$ の代わりに，$Y_i(h_i,f_i)$ を導出する非終端記号を別に設け，その非終端記号から $Y_i(h_i,f_i)$ を導出する生成規則を設けた．これらが生成規則 (2)，(3) である．

では,「速い車」や「甘い桃」のように意味的に適格なものだけを導出する☆. ■

【例 2】以下のような文法

$$VIP(\overline{eat}) \rightarrow VT(\overline{eat})\ NP_{obj}(-\overline{eat}) \quad (6)$$
$$NP_{obj}(-\overline{eat}) \rightarrow NP_{obj}(\overline{lunch})$$
$$VIP(\overline{eat}) \rightarrow VIP(\overline{eat})\ PP(-\overline{eat})$$
$$PP(-\overline{eat}) \rightarrow PP(\overline{with}, \overline{Ken}) \quad (7)$$
$$VT(\overline{eat}) \rightarrow eat$$
$$NP_{obj}(\overline{lunch}) \rightarrow NP_{obj}(\overline{lunch})\ PP(-\overline{lunch})$$
$$PP(-\overline{lunch}) \rightarrow PP(\overline{with}, \overline{tea}) \quad (8)$$
$$PP(\overline{with}, \overline{tea}) \rightarrow P(\overline{with})\ NP_{obj}(\overline{tea})$$
$$PP(\overline{with}, \overline{Ken}) \rightarrow P(\overline{with})\ NP_{obj}(\overline{Ken})$$
$$NP_{obj}(\overline{lunch}) \rightarrow lunch$$
$$NP_{obj}(\overline{tea}) \rightarrow tea$$
$$NP_{obj}(\overline{Ken}) \rightarrow Ken$$
$$P(\overline{with}) \rightarrow with$$

において，$VIP(\overline{eat})$ からは少なくとも “eat lunch with Ken” および “eat lunch with tea” を導出する☆☆. “eat lunch with Ken” では “with Ken” が “eat” に係ることを生成規則 (7) で，また “eat lunch with tea” では “with tea” が “lunch” に係ることを生成規則 (8) で表現している．そのためこの文法では，“with Ken” が “lunch” に係る文や “with tea” が “eat” に係る文を導出しない. ■

このように係り受け制約を文脈自由文法の生成規則として表現することで，構文解析中で係り受け制約を働かせることができる．

係り受け制約を表現できる文法として DCG（確定節文法）[2]，LFG（語彙機能文法）[3)~5)]，TAG があるが，DCG，LFG は確率文法化が難しく，一方 TAG は確率文法化でき[6),7)] さらに係り受けに交差を含む文の解析ができるが処理時間がかかるという問題があった．文脈自由文法は確率化が容易であり，処理時間も比較的かからないため，文脈自由文法の生成規則として係り受け制約を表現することの意義は大きい．

白井ら[10] は PCFG の語彙化（すなわち，head, function による非終端記号の細分化）によって，構文的な統計情報と語彙的な統計情報を 1 つのモデルに組み込むことの問題点を指摘しているが，我々の手法では，統語制約を表す生成規則 (2)，(5) と，係り受け制約を表す生成規則 (3) は分離されているため，この問題を回避できている☆☆☆．しかしながら，本論文で提案する文法を確率化した場合，推定すべき適用確率の数が多くなるのは事実で，冨浦ら[12)] や古海ら[13)] で述べている解決案を考慮中である．

なお，今回提案した文法では，以下を取り扱わない．

- 非交差性を満たさない文：係り受けの重要な性質として係り受けの非交差性がある．しかしそれを満たさない言語が少数であるが存在し，日本語においてもそのような文が見受けられる．文脈自由文法では係り受けに非交差性を満たさない文の構文解析はできない．しかしそのような文は非常に少ないものとして，本論文では取り扱わない．
- 並立句を含む文：並立句の場合は並立関係にある各々の句の head のうち，いずれかを句全体の中心的意味を担う語とすることは一般的にできない．ただし，n 個の名詞句で並立句を構成している場合には $NP(\langle h_1, h_2, \cdots, h_n\rangle)$ のように head の組に関して統語範疇を細分化することで係り受け制約を記述することができる．しかし，n は並立句によってまちまちであり，すべての場合について記述するといたずらに非終端記号が増えてしまうため，本論文では取り扱わない．
- 受身，使役の助動詞：これらが文中にある場合とない場合とでは文全体の格関係が大きく異なり，単独で取り扱うことはできない．これらは動詞の直後に来るので動詞とこれらをまとめて 1 つの動詞として扱うことが考えられる．

2.3 実　　験

「N_1 の N_2 の N_3」（=「名詞の名詞の名詞」）は日本語での曖昧さを持つ代表的な句であり，かつ従来曖昧さの解消が難しいとされていた.「N_1 の N_2 の N_3」における係り受けは，N_1 が N_2 に係るか，または N_3 に係るかの 2 種類のあいまいさが考えられる．このような名詞句を 2 章の係り受け文脈自由文法を確率化した文法で構文解析し，N_1 が，N_2 と N_3 のどちらに係るかを判定させる予備的な実験を行うことで係り受け文脈自由文法の有効性を確認する．

☆ head と function を，語と混同しないように，これから，例に出てくる生成規則中ではオーバラインをもって記述することにする．

☆☆ 英語などの場合で function が位置情報であるときには，function は生成規則に陽に現れない．英語では動詞句の後方にある名詞句は目的語になる．例の生成規則 (6) は，名詞句が，目的格（という function）で他動詞に係っていることを表している．目的格である名詞句を導出するという意味で，名詞句を目的格（obj）で細分化して NP_{obj} としている．

☆☆☆ ただし，生成規則 (2) での適用確率が h に依存しないように，つまり

$$p(X(h) \longrightarrow Y_1(-h)\ \cdots\ Z(h)\ \cdots\ Y_l(-h)) = p(X(h') \longrightarrow Y_1(-h')\ \cdots\ Z(h')\ \cdots\ Y_l(-h'))$$

（ただし，ここでは生成規則 X の適用確率を $p(X)$ としている）とする必要がある．

2.3.1 実験方法

EDR コーパス[8]から，名詞が「の」で連結された名詞句と，個々の名詞の概念（語義）およびその係り受けを抽出する．たとえば「谷の激流を身もだえしてサケが上る.」に対する，形態素データ，構文木データから，名詞句「谷の激流」，この名詞句における「谷」の概念記号が 3cec8a,「激流」の概念記号が 3cf2cf であること，および「谷」が「激流」に係ることが抽出できる．このようにして，コーパスの中の「名詞の名詞」☆を抽出する．

作成する生成規則のパターンは次のとおりである．

$$\begin{aligned} S &\longrightarrow NP(h) \\ NP(h) &\longrightarrow PP(-h) \quad NP(h) \\ PP(-h) &\longrightarrow PP(h', \overline{の}) \\ PP(h, \overline{の}) &\longrightarrow NP(h) \quad P(\overline{の}) \\ NP(h) &\longrightarrow w \\ P(\overline{の}) &\longrightarrow の \end{aligned} \tag{9}$$

ただし，S は開始記号，h，h' は概念記号，w は単語を表す．function は「$\overline{の}$」であり，生成規則 (9) では h は w の概念であることを表している☆☆．

確率文脈自由文法において，N 個の標本の構文木を $T_1, T_2, \cdots, T_N$，生成規則 $X \to \alpha$ が構文木 T の導出に適用された回数を $n(T, X \to \alpha)$，非終端記号 X を左辺に持つ生成規則の数を I_X とし，標本の採集が互いに独立に行われたと仮定すると，$X \to \alpha_i$ の適用確率 $p(X \to \alpha_i)$ の推定値 $\hat{p}(X \to \alpha_i)$ は次のようになる[9]．

$$\hat{p}(X \to \alpha_i) = \frac{\sum_{k=1}^{N} n(T_k, X \to \alpha_i)}{\sum_{k=1}^{N} \sum_{j=1}^{I_X} n(T_k, X \to \alpha_j)} \tag{10}$$

EDR コーパスから抽出した「名詞の名詞」を用いて標本を作成し，これをもとにして確率係り受け文脈自由文法を作成する．

EDR コーパスからテスト文「名詞の名詞の名詞」の概念およびその係り受けを抽出する．そして前述の方法で作成した確率係り受け文脈自由文法を用いてテスト文を構文解析し，係り先を推定する．「N_1 の N_2 の N_3」（＝「名詞の名詞の名詞」）において推定される係り受けには，N_1 が N_2 に係る場合と N_1 が N_3 に係る場合の 2 種類が考えられる．それぞれの場合の構文木の確率を算出し，確率の大きい方を構文解析における係り受け判定とする．これが EDR コーパスで示されるテスト文の係り受けと一致していれば正解として，全テスト文に対する正解の割合を求める．

実験は標本作成に使われたテスト文と標本作成に使われていないテスト文のそれぞれに対して行った．

表 1 「N_1 の N_2 の N_3」における N_1 の係り先判定手法の正解率
Table 1 The accuracy rate of the method deciding a governer of 'N_1' in "N_1 'no' N_2 'no' N_3".

標本作成に使われたテスト文	標本作成に使われていないテスト文
98.0%	82.4%

2.3.2 実験結果

標本中の「名詞の名詞」の数は 20000 個，標本作成に使われたテスト文および標本作成に使われていないテスト文はともに 500 個であった．

構文木が作成された割合は，標本作成に使われたテスト文では 100%，標本作成に使われていないテスト文では 13.6%であった．構文木が作成されたテスト文の中で，標本作成に使われたテスト文および標本作成に使われていないテスト文に対する構文解析結果が正しい係り受けと判定された割合は**表 1** のとおりであった．

2.3.3 考　　察

単語の係り先に曖昧さがあるときは，「単語から一番近い位置にある単語に係りやすい」というヒューリスティクスを用いることもできる．コーパス中にある「N_1 の N_2 の N_3」の個数は 8623 個であり，このヒューリスティクスによると N_1 が N_2 に係る方が N_1 が N_3 に係るより可能性が高いはずで，実際 N_1 が N_2 に係る方が 6230 個で全体の 72.25%を占めた．したがって，係り受け解析をするときに係り先に曖昧さがある文では，係りうる単語の中で一番近い単語に無条件に係るものとしてもある程度の結果は期待できる．今回の実験では，表 1 の標本作成に使われたテスト文の結果を見ると，98.0%という正解率を得ている．また，標本作成に使われていないテスト文も，構文木が作成された割合は 13.6%と低かったが，標本の数を増やせば構文木が作成される割合は 100%に近付き，さらに，解析されたもののうち正しい係り受けと

☆ 3 個以上の名詞がそれぞれ「の」で連結されているような文「N_1 の N_2 の $\cdots$ の N_l」においては，N_i（$1 \le i < l$）とそれが係る N_j（$1 < j \le l$）に対して「N_i の N_j」の組を抽出する．

☆☆ head を単語レベルの概念とすると，(3) の形の生成規則の数が膨大になり，適用確率の推定に十分な量の標本が得られないため，実際の実験ではシソーラスを利用して，その root ノードから数えて 7 段目の概念を用いて (9) の形の生成規則を記述した．なお，本論文は係り受け文脈自由文法の枠組みを述べることを本質としているため，具体的な head の選定に関することは今後の課題とする．

判定される割合（解析可能文中における正解率）も，一致性を満足するパラメータ推定（最尤推定法）を用いているので，少なくとも，表 1 に示す標本作成に使われていない文をテスト文とした場合の解析可能文中における正解率（82.4%）程度以上になる．したがって，実質的な正解率（入力文に対して正しい係り受けと判定される割合）

実質的な正解率

$$= \frac{\text{係り受けが正解であった文の数}}{\text{テスト文の数}}$$

= 解析可能率 × 解析可能文中における正解率

は，標本の量が十分に大きいときには，少なくとも 82.0%程度になることが期待できる．

3. 日本語文法への適用

係り受け文脈自由文法を日本語に適用する方法について述べる．2 章で係り受け文脈自由文法について述べたが，それを実現するためには生成規則 (5) における $Z(f)$ の function が適切に選定される必要がある．

日本語は文節がいくつか並んで文を構成している．文節は 1 個の自立語に 0 個以上の付属語が後接したものであり，function は一般に付属語列中に含まれている．日本語における係り受け文脈自由文法を構成するには，文節の付属語列中で何が function になるかが決定される機構が必要になる．

この章では，まず文節内の語の並びを規定する文節構造規則について述べ，次にどのような並びの付属語列のときに何が function になるかについて述べ，係り受け文脈自由文法を日本語に適用した場合の生成規則の構成法を示す．

3.1 文節構造規則

単語列 $w_0, w_1, \cdots, w_m$（ただし m は 0 以上）が文節であるためには，

- w_0 は自立語である
- w_k と w_{k+1} は連接可能である（ただし $0 \leq k < m$）
- w_m が文節末尾になりうる

が成立する必要がある．w_k と w_{k+1} の連接可能性は w_k の品詞と活用形，および w_{k+1} の品詞によって一意に決定される．また w_k が文節末尾になる可能性は w_k の品詞と活用形により一意に決定される．ただし，1 つの単語ごとに 1 つの品詞を設定しているものとする．また，辞書における記述量を考慮し，用言を語幹と活用語尾の 2 つに分けて考え，形式上，活用語尾は付属語として扱う．語幹と活用語尾の連接可能性は，両者の品詞活用型が一致しているか否かで一意に決定される．以降，品詞活用型も単に品詞と呼ぶ．

文節を導出する非終端記号を B，品詞 F の付属語が先頭である付属語列を導出する非終端記号を F，F'，自立語（用言の場合はその語幹）を b，付属語を w として，文節内における文法を正規文法で表現すると

$$B \longrightarrow b \quad F \tag{11}$$
$$B \longrightarrow b \tag{12}$$
$$F \longrightarrow w \quad F' \tag{13}$$
$$F \longrightarrow w \tag{14}$$

と表現できる．ただし，生成規則 (11) において，b と品詞 F の付属語が連接可能であり，生成規則 (12) において，b は文節末尾になることができ，生成規則 (13) において，w の品詞は F で，w と品詞 F' の付属語が連接可能であり，生成規則 (14) において，w の品詞は F で，w は文節末尾になることができなければならない．

3.2 function の決定

本論文では，係りの種類には，

(1) 「私が走る」のような格関係，
(2) 「スポーツするし，勉強もする」のような接続関係☆．
(3) 「青い海」,「楽しかったこと」,「この本」のような連体修飾関係，
(4) 「ゆっくり歩く」,「おいしく食べる」のような連用修飾関係

があるものと仮定している．

係りの種類が格関係の場合の function について説明する．格助詞は係りの種類（格関係）を規定している．また副助詞☆☆は本来,「限定」「程度」などの意味を付加するもので，係りの種類を規定しないが，格助詞がない場合には係りの種類を規定し，function になりうる．したがって，1 つの格助詞で付属語列を構成しているときはその格助詞が function になり，1 つの副助詞で付属語列を構成しているときはその副助詞が function になり，副助詞と格助詞で付属語列を構成しているときにはその格助詞が function になる．複数の格助詞が付属語列にある文では，係りの種類を規定するのは後方の格助詞であり，それが function になる．たとえば「東京にがいい」の場合には格助詞「が」が function になる．また格助詞がなく複数の副助詞が付属語列にある文では，係りの種類を規定するのは後方の副助詞であり，それが function になる．たとえば「お菓子ばかりなど食べる」の場合には副助詞「など」が function になる．

☆ 接続関係には，並立，順接，逆接などがあるが，並立は，2 章で説明したような現象が生じるので本論文では扱わない．
☆☆ 係助詞もこれに属するものとする．

係りの種類が接続関係の場合は，文節末尾にくる接続助詞が function になる．たとえば「勉強してから遊ぶ」では「から」が function になる．

活用語は文節末尾にあるときとそうでないときとで，その活用形の持つ役割が異なる．助動詞や自立語の活用語尾が文節末尾以外にあるときには，その活用形は次の語に対応して決まる．つまり活用形は語の連接条件のみに関係する．しかし助動詞や自立語の活用語尾が文節末尾にあるときは，その活用形は文節内の自立語が係る語の品詞を決める働きをする．すなわち活用形が連体形であれば体言に係り，連用形であれば用言に係る．係りの種類 (3) および (4) の,「青い」や「楽しかった」や「おいしく」のような活用語をともなう修飾については，活用語の活用形をもって，係りの種類を規定するものとして扱うことにする．したがって，助動詞や自立語の活用語尾が文節末尾にあるときには，その活用形が function になる．

3.3 文節文法の組み込み

基本的には，生成規則 (11)〜(14) を function で（生成規則 (11)，(12) に関してはさらに head で）細分化することにより，文節文法を組み込んだ係り受け文脈自由文法を実現することができる．これは係り受け文脈自由文法を日本語に適用したものである．

生成規則 (5) においては，右辺の $Y(h)$ から導出される末尾の語と $Z(f)$ から導出される付属語列とで文節を構成することになる．また日本語では，head のみを持つ統語範疇の句の末尾の語が自立語（またはその語幹）であり，句の主辞となっている．句の head h が単語レベルの概念であるとすると，h によりその句の末尾の語の品詞（品詞活用型）が一意に決まる．したがって，文節文法の生成規則 (11) の表す自立語と付属語列の連接関係の制約を (5) の形の生成規則として記述することができる．先頭の付属語の品詞が F，function が f であるような付属語列を導出する非終端記号を $F(f)$ で表すと，生成規則 (5) は

$$X(h,f) \longrightarrow Y(h) \quad F(f) \tag{15}$$

となる．生成規則 (5) と生成規則 (15) は，$Z(f)$ と $F(f)$ の違いだけのように見えるが，$Z(f)$ は function が f であり統語範疇が Z である句を導出する非終端記号であるのに対し，$F(f)$ は連接関係の制約を反映させるために $Z(f)$ を拡張したものとしている．当然，概念 h の単語と品詞 F の単語は連接可能でなければならない．ただし，用言は語幹と活用語尾に分け，活用語尾は付属語として扱っていることに注意すると，概念 h の単語が用言の場合，F は h の単語の品詞活用型であり，$F(f)$ から導出される先頭の語は概念 h の単語の活用語尾である．また，$Y(h)$ からの単語の導出は

$$Y(h) \longrightarrow w \tag{16}$$

である．ただし，概念 h の単語が体言のとき，w は概念 h の単語のつづりで，概念 h の単語が用言のとき，w は概念 h の単語の語幹のつづりである．

【例 3】「走る」を導出する生成規則は次のようになる．

$VP(\overline{\text{走る}},\overline{\text{連体}}) \rightarrow VP\text{語幹}(\overline{\text{走る}})$ ラ行五段動詞$(\overline{\text{連体}})$
$VP\text{語幹}\ (\overline{\text{走る}}) \rightarrow$ 走
ラ行五段動詞 $(\overline{\text{連体}}) \rightarrow$ る ■

生成規則 (12) は，b 単独で文節になるもので，連体詞や副詞がこれに相当する．したがって，生成規則 (12) を head，function で細分化して

$$X(h,f) \longrightarrow w \tag{17}$$

を得る．ただし，w は概念 h の単語のつづりで，概念 h の単語が副詞ならば f は「連用」，概念 h の単語が連体詞ならば f は「連体」である．

【例 4】「ゆっくり」を導出する生成規則は次のようになる．

$Adv(\overline{\text{ゆっくり}},\overline{\text{連用}}) \longrightarrow$ ゆっくり ■

生成規則 (13) の F と F' を function で細分化する．F' の function（F' から導出される付属語列の function）が，格助詞あるいは副助詞（係りの種類の (1) に相当）の場合と接続助詞あるいは活用形（係りの種類 (2)〜(4) に相当）の場合に分けて次のようになる．

- F' の function が格助詞あるいは副助詞の場合．
  $F(f) \rightarrow w \quad F'(f)$ (ただし $f \geq function(w)$)(18)
  $F(f_w) \rightarrow w \quad F'(f)$（その他） (19)
  ここで，$function(w)$ は w の function,「$\geq$」は全順序で，$p_1 \in$ 格助詞，$p_2 \in$ 副助詞，$p_3 \in$ 接続助詞，$infl \in$ 活用語尾，のとき
  $$p_1 \geq p_2 \geq p_3 = infl$$
  である．また，右辺の w は単語であり，生成規則 (19) における f_w は右辺の w に対応した function である．
- F' の function が接続助詞あるいは活用形の場合．
  $$F(f) \longrightarrow w \quad F'(f) \tag{20}$$

生成規則 (18)，(19)，(20) により，3.2 節で述べたように function を求めることができる．また，生成規則 (18)，(19)，(20) において，当然，w の品詞は F で，w と品詞 F' の単語は連接可能でなければならない．

付属語列の末尾の単語を導出する生成規則は生成規則 (14) の F を function で細分化して次のようになる．

$$F(f_w) \longrightarrow w \quad (21)$$
$$F(f) \longrightarrow w \quad (22)$$

生成規則 (21) は単語 w が助詞であるときの生成規則であり f_w は w に対応する function，生成規則 (22) は単語 w が助動詞であるときの生成規則であり f はその活用形である．

【例 5】「彼にだけ」を導出する生成規則は次のようになる．

$$PP(\overline{\text{彼}},\overline{\text{に}}) \longrightarrow NP(\overline{\text{彼}}) \quad \text{格助詞に } (\overline{\text{に}})$$
$$NP(\overline{\text{彼}}) \longrightarrow \text{彼}$$
$$\text{格助詞に } (\overline{\text{に}}) \longrightarrow \text{に} \quad \text{副助詞だけ } (\overline{\text{だけ}})$$
$$\text{副助詞だけ } (\overline{\text{だけ}}) \longrightarrow \text{だけ}$$

■

【例 6】「置いた」を導出する生成規則は次のようになる．

$$VP(\overline{\text{置く}},\overline{\text{連体}}) \rightarrow VP\text{語幹}(\overline{\text{置く}})\ \text{カ行五段動詞}(\overline{\text{連体}})$$
$$VP\text{語幹}(\overline{\text{置く}}) \rightarrow \text{置}$$
$$\text{カ行五段動詞}(\overline{\text{連体}}) \rightarrow \text{い} \quad \text{完了助動詞た}(\overline{\text{連体}})$$
$$\text{完了助動詞た}(\overline{\text{連体}}) \rightarrow \text{た}$$

■

3.2 節では，助詞の場合の function は助詞自身としていた．しかし副助詞が function の場合，それは格助詞の代用であるから，たとえば，生成規則 (15) の形式の

$$PP(h,\overline{\text{だけ}}) \longrightarrow NP(h) \quad F(\overline{\text{だけ}})$$

に対して，

$$PP(h,\overline{\text{が}}) \longrightarrow NP(h) \quad F(\overline{\text{だけ}})$$
$$PP(h,\overline{\text{を}}) \longrightarrow NP(h) \quad F(\overline{\text{だけ}})$$
$$\vdots$$

とすることにより，格関係を示す function を格助詞だけにすることができる．

3 章で述べる構成法は，係り受け解析に有効なモデルを構築するための良い方法といえる．なぜならば，何が function になるかという規則性がすでに分かっているならば最初からシステムに取り込んだ方が精度が上がるのは自明である．一方，function になる規則性を考慮に入れない状態で，機械的学習だけで，この規則性つまり格助詞は副助詞に優先される，function へのなりやすさが同じである助詞が並んでいるときには末尾の助詞が優先される，などの性質を学習させるためには相当の学習データ数が必要となり，現在それほどの学習データを収集することは不可能である．

4. 関 連 研 究

関連した研究について比較検討する．

本論文に示す手法の特徴は，次に示す 2 点である．

- 従来の統語範疇による生成規則，すなわち *NP* や *VP* のような統語範疇の並びに関する規則は保存したままで，二項の共起制約としての係り受け制約を生成規則として表現している．
- 日本語において function を決定する機構を考察し，これを生成規則として表現して，係り受け文脈自由文法を日本語文法に適用した具体的な文法を提案している．

生成規則として係り受け制約を表現することの利点は，確率化が容易であること（生成規則に確率を与えることで，従来から研究されてきた，PCFG という確率モデルになることが保証される），それを確率化した PCFG に対しては一致性を満足するパラメータ推定法（最尤推定法）が存在するということである．一致性を満足するパラメータ推定法とは，大雑把にいえば，学習データ量が十分に大きければ，信頼性のある確率パラメータ値（生成規則の適用確率）が推定される推定法である．また，係り受け制約に関しては，係る句の head と function および係られる句の head の間の二項の共起制約に限定している．これは，以下の 2 つの理由による．

(1) ほとんどの場合，係り受け制約として二項の共起制約を用いるだけで十分であり，二項の共起制約では扱えない「車がガソリンを食う」のような特殊な文は少ないと考えられる．
(2) 近年大規模なコーパスができてはいるが，信頼性のある確率パラメータを推定するのには依然として学習データが少ない状況にあり，二項の共起制約を扱ったモデルは，信頼性のあるパラメータを推定するための学習データ量が多項の共起制約を扱ったモデルより少なくて済む．

そして，日本語においては付属語列の function は末尾の語（活用語尾も含む）である場合が多いが，実際には末尾以外の語も function になり，function を決める機構を生成規則として表現したことの意義は大きい．

以上の点を考慮して，語彙化により，共起制約を表現した類似研究 14)，15)，16) で提案しているモデルとの比較を行う．

Hogenhout ら[14] のモデルを，本論文で示すタイプの生成規則で表すと，

$$X(h) \longrightarrow Y_1(h_1)\ Y_2(h_2) \cdots Y(h) \cdots Y_n(h_n) \quad (23)$$
$$Y_i(h_i) \longrightarrow h_i$$

となる．生成規則 (23) は，h_1，h_2，$\cdots$，h_n すべてが，h に係っており，h_ℓ と h の間の共起制約だけでなく

h_ℓ と h_m（ただし $1 \leq \ell \leq n$，$1 \leq m \leq n$，$\ell \neq m$）の間にも共起制約があることを示し，係り受け制約を多項の共起制約（この場合には n 項）として捉えている．

Charniak[15] の提案するモデルは，二項の共起制約としての係り受け制約を用いている点では本論文で示している手法と同じであるが，直接語彙化した生成規則に確率を与えているわけではない．PCFG とは異なる機構で語彙化された構文木の確率を求める機構を与えているが，構文木の確率を求める機構の妥当性，信頼性については言及されていない．

一方，Collins[16] の提案するモデルは本論文で示す手法と同じように直接語彙化した生成規則に確率を与えている．生成規則は，

$$X(h) \rightarrow L_m(l_m) \cdots L_1(l_1)\ Y(h)\ R_1(r_1) \cdots R_n(r_n) \tag{24}$$

であり，このままでは多項の共起制約となるため，生成規則 (24) の適用確率 p を，

$$p = p_H(Y|X,h) \cdot \prod_{i=1}^{m+1} p_L(L_i(l_i)|X,h,Y) \cdot \prod_{i=1}^{n+1} p_R(R_i(r_i)|X,h,Y) \tag{25}$$

と，二項の共起制約として近似している．ただし，

$$L_{m+1}(l_{m+1}) = STOP, R_{n+1}(r_{n+1}) = STOP$$

である．このモデルには，以下のような 2 つの問題がある．

(1) 左辺を $X(h)$ とする生成規則が以下の 2 つを含む場合（ただし，右辺を代表する head phrase は Y）

$$X(h) \longrightarrow A(a) \quad B(b) \quad Y(h) \tag{26}$$
$$X(h) \longrightarrow B(b) \quad A(a) \quad Y(h) \tag{27}$$

(25) の右辺に表れる条件付き確率を以下のように推定したとする．

$$p_H(Y|X,h) = q_1$$
$$p_L(A(a)|X,h,Y) = q_2$$
$$p_L(B(b)|X,h,Y) = q_3$$
$$p_L(STOP|X,h,Y) = q_4$$
$$p_R(STOP|X,h,Y) = q_5$$

すると，生成規則 (26) と (27) に与えられる適用確率は，

$$p(X(h) \rightarrow A(a)\ B(b)\ Y(h)) = q_1 q_2 q_3 q_4 q_5$$
$$p(X(h) \rightarrow B(b)\ A(a)\ Y(h)) = q_1 q_2 q_3 q_4 q_5$$

となり，生成規則 (26) と (27) の適用確率の推定値が必ず一致してしまう．つまり，head phrase の左の句の順序，右の句の順序が無視されることになる．

(2) 左辺を $X(h)$ とする生成規則が以下の 2 つしかない場合（ただし，右辺を代表する head phrase は Y）

$$X(h) \longrightarrow Y(h) \tag{28}$$
$$X(h) \longrightarrow A(a) \quad Y(h) \tag{29}$$

(25) の右辺に表れる条件付き確率を以下のように推定したとする．

$$p_H(Y|X,h) = 1 \tag{30}$$
$$p_L(A(a)|X,h,Y) = \alpha$$
$$p_L(STOP|X,h,Y) = 1-\alpha \tag{31}$$
$$p_R(STOP|X,h,Y) = 1 \tag{32}$$

条件付き確率の定義より，(30)，(31)，(32) の条件付き確率はそれぞれ 1，$1-\alpha$，1 となる．すると，生成規則 (28) と (29) に与えられる適用確率は，

$$p(X(h) \rightarrow Y(h)) = 1-\alpha \tag{33}$$
$$p(X(h) \rightarrow A(a)\ Y(h)) = \alpha(1-\alpha) \tag{34}$$

となる．α の値が何であっても，生成規則 (28) と (29) の適用確率の和は，$1-\alpha+\alpha(1-\alpha) = 1-\alpha^2 < 1$ となってしまう．生成規則 (28) と (29) の適用確率の和が 1 ではないため，(25) の近似によるモデルは，確率モデルとはなっていない．

5. おわりに

文脈自由文法の非終端記号をそれから導出される句の有限個の概念（head）および function により細分化することで，係り受け制約を組み込んだ文脈自由文法（係り受け文脈自由文法）の構成法を提案した．それを確率化した係り受け文脈自由文法を用いて，名詞句「N_1 の N_2 の N_3」において function は「$\overline{\text{の}}$」を対象にして実験を行った結果，有効性を確認した．

さらに文節文法の非終端記号を head および function で細分化することにより係り受け制約文脈自由文法を日本語文法に適用する方法について述べた．

文脈自由文法に対する効率的なパーザ（構文解析器）のアルゴリズムとしては，Earley 法や Chart 法が知られているが，係り受け文脈自由文法に対してこれらのアルゴリズムを用いて構文解析を行う場合には解析時間が問題になる．そこで，Earley 法を拡張して，生成規則数の多い大規模文法に適した効率的な構文解析アルゴリズムを考案しており，報告する予定である．

参 考 文 献

1) Aravind, K.J. and Schabes, Y.: Tree Adjoining Grammars and Lexicalized Grammars, Nivat, M. and Podellski, A. (Ed.), *Tree Automata and Languages*, Elsevier Science (1992).
2) Pereira, F.C.N. and Warren, D.H.D.: Definite clause grammars for Language analysis-a survey of the formalism and a comparison with augmented transition networks, *Artifical Intelligence*, 13, pp.231–278 (1980).
3) Bresnan, J. (Ed.): *The Mental Representation of Grammatical Relations*, MIT Press, Cambridge, Masachusetts (1982).
4) Sells, P.: Lectures on Contemporary Syntactic Theories, CSLI Lecture Note, No.3, CSLI Stanford University (1985).
5) Winograd, T.: *Language as a Cognitive Process*, vol.1-Syntax, Addison-Wesley (1983).
6) Schabes, Y.: Stochastic Tree-Adjoining Grammars, *Proc. COLING*, 1 (1992).
7) Resnik, P.: Probabilistic Tree-Adjoining Grammar as a Framework for Statictical Natural Language Processing, *Proc. COLING*, 1 (1992).
8) 日本電子化辞書研究所：EDR 電子化辞書仕様説明書 (1995).
9) 日高 達：確率文法，情報処理，Vol.36, No.2, pp.169–176 (1995).
10) 白井清昭，乾健太郎，徳永健伸，田中穂積：統計的構文解析における構文的統計情報と語彙的統計情報の統合について，言語処理学会学会誌「自然言語処理」，Vol.5, No.3, pp.85–106 (1998).
11) 田辺利文，冨浦洋一，日高 達：係り受け関係の記述能力を持つ PCFG，平成 6 年度電気関係学会九州支部連合会大会講演論文集，p.685 (Sep. 1994).
12) 冨浦洋一，日高 達：スパースな学習データにおける PCFG の確率パラメタの推定法，電子情報通信学会技術研究報告（言語理解とコミュニケーション），pp.39–46 (Jul. 1998).
13) 古海真吉，D. トウシンバット，冨浦洋一，日高達：係り受け制約を表現するスパースデータに頑強な確率文脈自由文法の構成法，平成 9 年度電気関係学会九州支部連合会大会講演論文集，p.281 (Oct. 1997).
14) Hogenhout, W.R. and Matsumoto, Y.: Experiments with Using Semantical Categories in Parsing Systems，言語処理学会年次大会 (1996).
15) Charniak, E.: *Statistical parsing with a context-free grammar and word statistics*, AAAI (1997).
16) Collins, M.: *Three Generative, Lexicalised Models for Statistical Parsing*, ACL (1997).

(平成 10 年 10 月 12 日受付)
(平成 11 年 11 月 4 日採録)

田辺 利文（学生会員）

昭和 45 年生．平成 5 年九州大学工学部情報工学科卒業．平成 7 年同大学院工学研究科電子工学専攻修士課程修了．現在同大学院システム情報科学研究科知能システム学専攻博士後期課程在学中．工学修士．平成 7 年度情報処理学会九州支部論文奨励賞受賞．自然言語処理，言語学に興味を持つ．

冨浦 洋一（正会員）

昭和 36 年生．昭和 59 年九州大学工学部電子工学科卒業．昭和 61 年同大学院工学研究科電子工学専攻修士課程修了．平成元年同大学院工学研究科電子工学専攻博士後期課程単位取得退学．同年九州大学工学部助手，平成 7 年同助教授，現在同大学院システム情報科学研究科助教授．工学博士．平成 3 年度情報処理学会研究賞受賞．自然言語処理，計算言語学，人工知能に関する研究に従事．人工知能学会会員．

日高 達（正会員）

昭和 14 年生．昭和 40 年九州大学工学部電子工学科卒業．昭和 42 年同大学院工学研究科電子工学専攻修士課程修了．昭和 44 年同大学院工学研究科電子工学専攻博士後期課程中退．同年九州大学工学部助手，昭和 48 年同講師，昭和 55 年同助教授，昭和 63 年同教授，現在同大学院システム情報科学研究科教授．工学博士．形式言語の方程式論，自然言語処理，手書き文字認識の研究に従事．電子情報通信学会，人工知能学会会員．