自然言語処理 128－6
（1998. 11. 5）

係り受け文脈自由文法の強化法

D. トウシンバット　冨浦 洋一　日高 達
九州大学大学院システム情報科学研究科
〒812-81 福岡市東区箱崎 6-10-1
Tel: 092-642-3882
E-mail:{toushin,tom,hitaka}@lang.is.kyushu-u.ac.jp

あらまし　統語範疇をそれから導出される句の主辞の概念で細分化することにより，係り受け制約を生成規則として表現した文法 (係り受け文脈自由文法) が構成できる．しかし，詳細に係り受け制約を記述すると，生成規則の量が膨大となり，係り受け制約を表す生成規則を網羅することが困難となる．本論文では，係り受け文脈自由文法の強化法について述べる．主要なアイデアは，上位-下位関係を下位概念から上位概念に書き換える生成規則として文法に取り込むことであり，洩れた係り受け制約を含む文に対しても，意味的に近い係り受け制約を表す生成規則を用いて構文木を求めることができる．

キーワード　係り受け制約，文脈自由文法，シソーラス，構文木

How to Strengthen a Context Free Grammar Expressing Dependency Constraint

D.Toushinbatto , Yoichi TOMIURA and Toru HITAKA
Graduate School of Information Science and Electrical Engineering,
Kyushu University
Hakozaki 6-10-1, Higashi, Fukuoka City, Fukuoka 812-81, Japan
Tel: 0081-092-642-3882
E-mail: {toushin,tom,hitaka}@lang.is.kyushu-u.ac.jp

abstract　A Context Free Grammar expressing dependency constraints as production rules can be constructed, where a nonterminal is given by subdividing a syntactic category with the concerpt of the headword of the phrase derived from the syntactic category. But to describe dependency constraints in detail causes an explosion of the number of production rules, which makes it difficult to collect all production rules expressing dependency constraints. This paper proposes how to strengthen a Context Free Grammar expressing dependency constraints. The main idea is to take a superordinate-subordinate relation in the grammar as a production rule which rewrites a sub-concept into its super-concept, so that, also with a sentence including a dependency constraint which has not been collected, its syntactic tree can be given using the existed production rule expressing the semantically similar dependency constraint.

key words　Dependency Constraint, Context Free Grammar, Thesaurus , Syntactic Tree

1 はじめに

自然言語処理における問題点の一つとして，構文解析の際に一つの文に対応する構文木が一般に複数得られることである．しかし，一つの文に対してはただ一つの意味，すなわちただ一つの構文木が対応していると考えられているので，意味的に正しい構文木を選び出すことが重要である．そこで，統語範疇をそれから導出される句の主辞，あるいは主辞の概念で細分化し，係り受け制約を生成規則として取りこむことによって，係り受け制約を満足する構文木のみ導出する文法(係り受け文脈自由文法)が提案されている[1]．しかし，細かい意味レベルで係り受け制約を記述すると，係り受け制約を表す生成規則の数が膨大となり，文法からその一部が洩れてしまう．そのため，入力文によって解析不能の場合が生じる．

本稿では，新たにシソーラスの下位概念から上位概念へ書き換える生成規則を追加することにより，意味的に近い係り受け制約を表す生成規則を用いて，洩れた係り受け制約を含む文に対しても，構文木を求めることができる文法を提案する．構文木間に優先順位をつけるため，確率文法化[2]する．本稿で提案する文法の確率化の際のパラメタ推定法についても示す．

2 係り受け文脈自由文法

2.1 係り受け制約

統語範疇 Xから導出される句は，さらに Zと Y_1 , Y_2,..., Y_nから導出される句で構成されているとすると，

$$X \rightarrow Y_1 \ldots Z \ldots Y_n \qquad (1)$$

という生成規則で表される．Zから導出される句が，$Y_j(1 \leq j \leq n)$ から導出される句に修飾されている場合，Zから導出される句が Xから導出される句の全体の意味を代表していると考えられる．このとき Zから導出される句を Xから導出される句の head phrase と定義する．headword を head phrase の中心的意味を表す単語とし，headword の概念を head と定義する．以後，Xから導出される句の head を，単に Xの head と呼ぶ．

一般に修飾句はその被修飾句に対する修飾の種類を規定する情報を持っている．したがって，Y_jから導出される句には，Zに対する修飾の種類を規定する情報を持っている．これを Y_jから導出される句の function と定義する．以後，Y_jから導出される句の function を，単に Y_jの function と呼ぶ．

Y_jの head w_j,function f_j，Zの head が wであるとき，w_jは f_jを介して wに構造的係っている(構造的係り受け関係)と定義する．Y_jの句と Zの句の間に意味的に適格な修飾関係が成立しているとすると，Y_jの head w_j,function f_j及び Zの head が wには一定の意味的制約(係り受け制約)が成立している．構造的係り受け関係うち係り受け制約を満足してものを意味的に適格な係り受け関係と呼ぶ．

2.2 係り受け制約の取り込み

生成規則(1)において，Y_jからの導出と Zからの導出は独立に行なわれるため，Y_jの head と Zの head の間の係り受け関係が意味的に不適格なものまで導出されてしまう．そこで，非終端記号をそれから導出される句の head で細分化して，生成規則(1)を

$$X(h) \rightarrow Y_1(-h) \ldots Z(h) \ldots Y_n(-h) \qquad (2)$$

と

$$Y_j(-h) \rightarrow Y_j(h_j, f_j) \qquad (1 \leq j \leq n) \qquad (3)$$

という形に書き直す．但し，

$X(h)$: head が h である Xの句を導出する非終端記号

$Y_j(-h)$: head が h である句に係り得る Y_jの句を導出する非終端記号

$Y(h_j, f_j)$: head が h_j，function が f_jである Yの 句を導出する非終端記号

生成規則(3)は，h_jが f_jで hに係り得ること，すなわち係り受け制約を表す生成規則である．このような形に生成規則を直すことによって，Y_jと Zからの導出を係り受け制約を満足するものだけに制限できる．

2.3 係り受け制約を取り込むときの問題点

例えば，日本語の後置詞句 (PP) が動詞句 (VP) に係り得ることは，生成規則(4)と(5)で表される．

$$VP(h) \rightarrow PP(-h)VP(h) \qquad (4)$$

$$PP(-h) \rightarrow PP(h', f) \quad (5)$$

h, h'と fを単語の細かい概念レベルで，動詞と名詞の総数を考えたとき，生成規則 (5) の数は膨大となり，全てを収集することは困難となる．このような係り受け文脈自由文法では，(6) のような係り受け制約を表す生成規則の一部が文法から洩れてしまうという問題が生じる．

$$X(-h) \rightarrow X(h', f) \quad (6)$$

その解決法として，文献 1) では，head をシソーラスにおける比較的上位の概念とし，更にシソーラスの上位概念から下位概念へ書き換える生成規則を加えることで，係り受け制約の洩れを補う試みがなされている．この場合の文法の生成規則のパターンは次の通りとなる．

$$X(h) \rightarrow Y_1(-h)...X(h)...Y_n(-h) \quad (7)$$

$$Y(-h) \rightarrow Y(h', f) \quad (8)$$

$$Y(h, f) \rightarrow X(h)\ Z(f) \quad (9)$$

$$X(h) \rightarrow h \quad (10)$$

$$Z(f) \rightarrow f \quad (11)$$

$$h \rightarrow h_L \quad (12)$$

$$h \rightarrow w \quad (13)$$

但し，h_Lは hの下位概念の一つで，wは単語である．

例えば,「フォックステリアが歩く」に対し，図 2 に示したようなシソーラスで構文解析をするとき，生成規則 (14) が洩れたとしても，生成規則 (15)，(16) を用いて構文木を導出できる (図 3)．但し，[α] は単語αの概念である．

$$PP(-[歩く]) \rightarrow PP([フォックステリア],\ が) \quad (14)$$

$$PP(-[歩く]) \rightarrow PP([犬],\ が) \quad (15)$$

$$[犬] \rightarrow [フォックステリア] \quad (16)$$

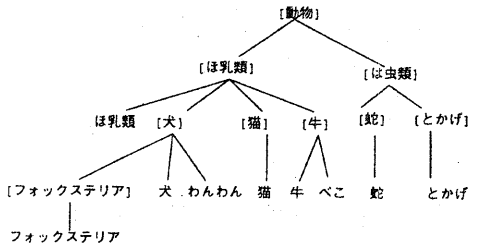

図 2:シソーラス

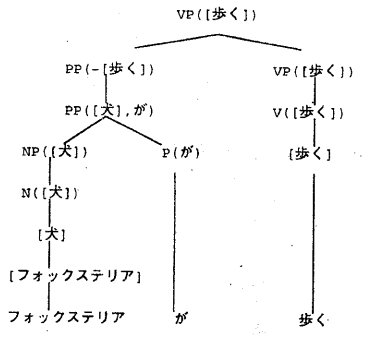

図 3:「フォックステリアが歩く」の構文木

しかし,「猫が歩く」対して構文解析をする場合，生成規則 (17) が洩れたとき，また [猫] の上位概念でも [歩く] と係るものもないときには，文献 1) の手法では構文木を得ることはできない．

$$PP(-[歩く]) \rightarrow PP([猫],\ が) \quad (17)$$

このような生成規則 (17) を文法から洩れたとする．

3 提案手法

3.1 生成規則の追加

本節では，係り受け文脈自由文法の生成規則の洩れに対する強化法を示す．係り受け制約に対して係る句の head と係られる句の head をそれぞれ意味的に近い概念に置き換えた場合，係り受け制約も満足しやすいと考えられる．例えば，[犬] と [歩く] の「が」での係り受け制約を，[犬] を意味的に近い [猫] に置き換えた場合，[猫] と [歩く] の「が」での係り受け関係は意味的に適格なものになると考えられる．[歩く] を [走る] に置き換えた場合も，同様である．

「猫が歩く」に対して，生成規則 (17) が洩れた場合には構文木を導出できないと前章で述べた．生成規則 (18) がある場合，それを用いて構文木を導出させるためには，生成規則 (19) を新たに追加する必要がある．生成規則 (18)，(19) を用いて導出された構文木は図 (4) のようになる．

$$PP(-[歩く]) \rightarrow PP([犬],\ が) \quad (18)$$

$$[犬] \rightarrow [ほ乳類] \quad (19)$$

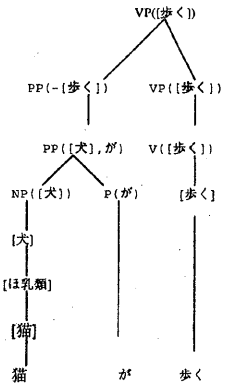

図 4:「猫が歩く」の構文構造

したがって，下位概念から上位概念への書き換える生成規則 (20) を，新たに文法 (7) − (13) に加えることにより，強化された係り受け文脈自由文法となる．

$$h \rightarrow h_H \qquad (20)$$

但し，h_H は h の上位概念の一つである．

3.2 提案手法の問題点

前節の「猫が歩く」に対して，生成規則 (17) が洩れた場合，生成規則 (18) と追加された生成規則 (19) を用いることで構文木を導出することが可能となった。しかし、生成規則 (21) がある場合，(22) と (23) を用いることで，図 5 に示すような構文木が導出される．

$$PP(-[歩く]) \rightarrow PP([とかげ],\ が) \qquad (21)$$

$$[とかげ] \rightarrow [は虫類] \qquad (22)$$

$$[は虫類] \rightarrow [動物] \qquad (23)$$

上位概念へ書き換える生成規則のために，複数の構文木が導出されることになる．

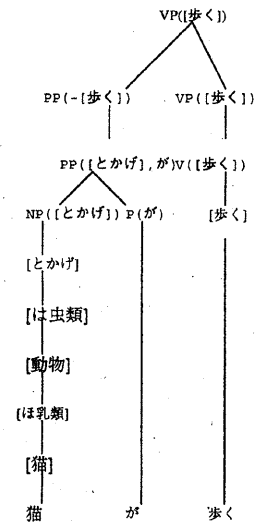

図 5:「猫が歩く」の構文構造

「コスモスが咲く丘を登る」に対して，[コスモス] は [咲く] に係るか，[登る] に係るかという曖昧さがある．[コスモス] の係り先に対して，従来の係り受け文脈自由文法では生成規則 (24) を用いて，係り先の正しい構文木のみに絞り込むことができる (図 6)．

$$PP(-[咲く]) \rightarrow PP([コスモス],\ が) \qquad (24)$$

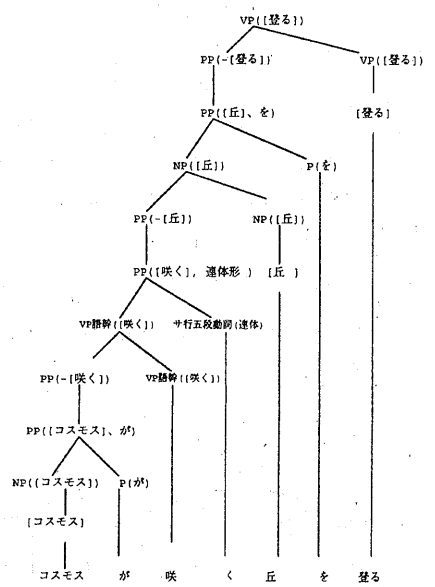

図 6:「コスモスが咲く丘を登る」の構文木

しかし，提案手法を用いることで，上位概念に書

き換える生成規則のため，図7で示した係り先の違う構文木も得られてしまう．

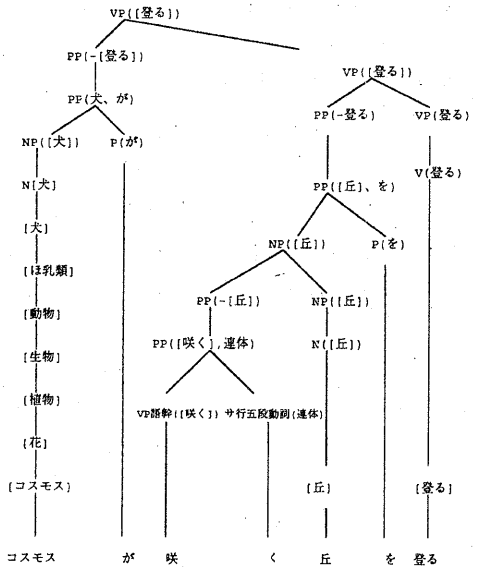

図7:「コスモスが咲く丘を登る」の係り先の違う構文木

「コスモスが咲く丘を登る」のように，従来のCFGと同様に，意味的に適格でない構文木の導出も許してしまう．

以上のことから，得られた構文木の間に優先順位を付け，構文木の絞り込みを行なう必要がある．「猫が歩く」の場合，[犬]と[とかげ]を比べれば，[犬]のほうが[猫]と意味的に近いため，図4で示した構文木を図5より高い優先順位を与える必要がある．「コスモスが咲く丘を登る」の場合，図6で示した構文木は図7より高い優先順位を与える必要がある．

4 パラメタの推定

構文木の間に優先順位を付け，絞り込みを図るため，本稿では確率文脈自由文法 (Probablistic Context Free Grammer:PCFG)[2]を用いることにする．

4.1 確率文脈自由文法

PCFGは次の5項組で定義される。

$$G = (\Sigma, V, P, S, p) \tag{25}$$

但し、

$$\begin{cases} \Sigma: & \text{終端記号の有限集合} \\ V: & \text{非終端記号の有限集合} \\ P: & \text{生成規則の有限集合} \\ S: & \text{開始記号 } (S \in V) \\ p: & \text{生成規則から区間 } (0,1] \text{ 上への写像} \end{cases}$$

$$X \to \alpha_i;\ X \in V,\ \alpha_i \in (\Sigma \cup V)^*,\ X \to \alpha_i \in P \quad (i = 1, 2, \cdots, I_X)$$

に対し、

$$\sum_{i=1}^{I_X} p(X \to \alpha_i) = 1 \tag{26}$$

が成り立つ．

サンプル構文木集合$T_1, T_2, \ldots T_N$が得られる確率(尤度)は

$$\prod_{k}^{N} Pr(T_k) = \prod_{k}^{N} \prod_{\delta \in P} p(\delta)^{n(T_k, \delta)} \tag{27}$$

で表される．ここで，$p(\delta)$は生成規則δの適用確率，$Pr(T)$は構文木Tの生起確率，$n(T, \delta)$は構文木T中に出現する規則δの回数である．

尤度(式(27))を最大にするパラメタ(適用確率)は式(28)で求められる．

$$p(X \to \alpha_i) = \frac{\sum_{k=1}^{N} n(T_k, X \to \alpha_i)}{\sum_{k=1}^{N} \sum_{i=1}^{I_X} n(T_k, X \to \alpha_i)} \tag{28}$$

PCFGでは，サンプルとして与えた構文木集合で多く用いられた規則を用いた構文木に生起確率を高く与える．サンプル中の構文木で用いられなかった生成規則のパラメタは0となってしまう．提案手法で導入した下位概念から上位概念へ書き換える生成規則は，サンプル中の構文木では用いられなかった．上位概念から下位概念へ書き換える生成規則の一部も，サンプル中の構文木では用いられない可能性がある．洩れた係り受け制約を表す生成規則を含む文に対して，提案手法では構文木を与えられるが，その構文木の生起確率が0となる．

4.2 推定法の改良

文献3)では，サンプルに出現しなかった生成規則のパラメタに微小な値を与える推定法が提案されている．この値はサンプル中の構文木で用いられる生成規則のパラメタより小さいものとしている．

ここで，h から下位概念への書き換える生成規則のなかで，サンプルの構文木に出現しなかった生成規則を

$$h \rightarrow h_i,\ (i = 1, 2, ..., m) \tag{29}$$

出現した生成規則を

$$h \rightarrow h_j,\ (j = m+1, m+2, ..., n) \tag{30}$$

とする．文献3)の手法を用いると，パラメタ推定値は以下の式で与えられる．

$$p(h \rightarrow h_H) = a \tag{31}$$

$$p(h \rightarrow h_i) = b,\ (i = 1, 2, ..., m) \tag{32}$$

$$p(h \rightarrow h_j) = (1 - a - mb)\frac{\sum_{k=1}^{N} n(T_k, h \rightarrow h_j)}{\sum_{k=1}^{N}\sum_{j=m+1}^{n} n(T_k, h \rightarrow h_j)} \tag{33}$$

$$(j = m+1, m+2, ..., n)$$

式(31),(32)の a と b の値を，サンプル中で一回だけ使われた生成規則のパラメタより小さく，また $a \leq b$ のように設定する．この推定法での場合を考えると，とくにシソーラス上の概念を辿る生成規則を用いた構文木の生起確率を，シソーラスの概念を辿る生成規則を用いなかった構文木の生起確率より低く与えることができる．また，シソーラスの上位概念へ書き換える生成規則を多く用いた構文木ほど，その生起確率が低くなるように設定することができる．

5 終りに

本稿では，従来の係り受け文脈自由文法の係り受け制約を表す生成規則の洩れに対する強化手法を提案した．

参考文献

[1] 田辺，冨浦，日高,“係り受け制約を取り込んだPCFGの評価”，九州大学大学院システム情報研究科報告第2巻第1号，pp.93-97(1997)

[2] 日高 “確率文法”，情報処理，Vol.36,No.2,PP169-176(1995)

[3] 和久田，D．トウシンバット，冨浦，日高，“制限されたパラメタ空間での最尤推定法”，電気関係学会九州支部連合大会論文集，pp.284(1998)