格フレーム解析を統合した日本語係り受け解析

松尾義博　　白井諭
NTTコミュニケーション科学研究所
{yosihiro, shirai}@cslab.kecl.ntt.co.jp

1 はじめに

日本語の構文解析では、文法的な制約を主に用いて文節間の依存関係を解析する係り受け解析が一般に行われている。これは、実用的な文法を記述しようとすると、省略や語順の入れ替わりを無視できず、英語のように句構造文法で記述することが困難であるためである。しかしながら、文法的な制約のみで正しい構造を解析できることは望めず、係り受け解析では複数の構造候補を出力し、後段の意味解析で意味的に適切な構造を選択する手法が一般に用いられる。意味解析の主たる目的は、語句の語義の決定と、語句の意味的機能(深層格)の決定であるが、この手法では、正しい文構造の選択という機能も持っていると考えられる。

一方、日本語の動詞句の意味構造を記述した辞書としては、格フレーム辞書が挙げられる[2]。格フレームでは動詞の各語義について、取り得る格要素が意味属性や語表記を用いて記述されている。格フレームは動詞の語義を決定するための意味辞書ではあるが、正しい文構造を選択する機能を考えると、格フレームこそが日本語の動詞に関する語彙化された文法規則と捉えることもできる。

しかし、格フレームを直接用いて構文解析することもやはり困難であることから、前記のように、一旦係り受け解析を行う手法が取られている。格フレーム解析で係り受け構造を選択する場合は、係り受け構造全体を覆うのに最も適した格フレームの組合せを求め、それを最適構造としている。しかし文法的に可能性のあるすべての候補を一旦出力することは、その候補総数が組合せ的に増加するために困難であり、枝狩りが必要である。したがって、係り受け解析が出力する候補には正解が含まれていない可能性があり、解析失敗の原因となり得る。

そこで、本稿では、日本語の格フレームを係り受け解析の文法規則に変換することにより、構文解析と意味解析を一度に行なう手法について報告する。本手法では、文法的制約のみを先行適用することによる正解候補の脱落という問題は生じず、また、係り受け解析と同時に動詞の語義と格要素の深層格も決定されており、効率的な解析が可能となる。

2 解析アルゴリズム

格フレームを係り受け解析に直接適用するための基本的な考え方は、格フレームを二文節関係だけの文法規則に分割することである(図1)。それぞれの文法規則の右辺には二つの文節だけが記述されており、文法規則の組合せ適用によって、全体の格フレームが構成される。分割された文法規則の継りを保持するために、「文法規則ID(GID)」と「接続ID(CID)」の二つの識別子を導入する。GIDとCIDについては後述する。

2.1 文法規則

2.1.1 係受解析のためのデフォルト文法

本手法では2種類の文法規則を用いる。ひとつは、係受解析のデフォルトとして機能する簡単なCFG規則である(図2)。極めて単純化して記述すると、日本語は下記の規則を持っている。

- 各文節は自立部と付属部から成る。
- 各文節は以下のどれかの修飾機能を持つ:(1) 名詞句の修飾 (2) 動詞句の修飾 (3) 文末。
- 各文節は後方の1つの文節を修飾する。

上記規則に従って、形態素解析は各文節を2つのシンボル(品詞)列と解析するように作成する。1つめのシンボルは自立部の種類(V または N)で、2つめのシンボルは修飾機能の種類(V-V, V-N, N-V, N-N)である[1]。後者は、"修飾元の自立部の種類-修飾先の自立部の種類"を意味する。ただし、文末は修飾機能を持たず1つめのシンボルのみである。また、形態素解析が単一の品詞を決定することができなかった場合、複数の品詞候補が出力される。例えば、「彼と東京に行く」は以下のように解析される。

- 彼/N と/N-V/N-N 東京/N に/N-V 行く/V

図2にデフォルト文法規則を示す。それぞれの規則の右辺には最大2つの文節のみが含まれており、格要素が省略された文や文節順が入れ換わった文を受理できる。

2.1.2 格フレームから文法規則への変換

もうひとつの文法規則は、格フレームから変換して作られる。格フレームは下記のように記述できる。

$$Pat_i = (Pred_i, Case_{i1}, ..., Case_{in})$$

[1] ここでは、副詞は動詞(V)と動詞修飾(V-V)の並びとして扱い、形容詞は動詞(V)と動詞か名詞への修飾(V-VまたはV-N)の並びとして扱う。形容詞の修飾の種類はその活用形で決定される。

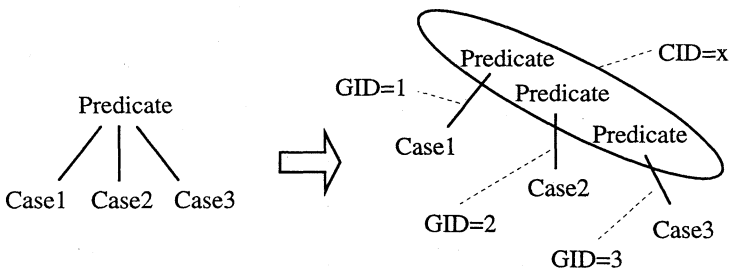

図 1: 格フレームから文法規則への変換

$$
\begin{aligned}
\mathrm{S} &\rightarrow \mathrm{VP} \\
\mathrm{VP} &\rightarrow \mathrm{NP\ N\text{-}V\ VP} \\
\mathrm{VP} &\rightarrow \mathrm{VP\ V\text{-}V\ VP} \\
\mathrm{VP} &\rightarrow \mathrm{V} \\
\mathrm{NP} &\rightarrow \mathrm{NP\ N\text{-}N\ NP} \\
\mathrm{NP} &\rightarrow \mathrm{VP\ V\text{-}N\ NP} \\
\mathrm{NP} &\rightarrow \mathrm{N}
\end{aligned}
$$

図 2: デフォルト規則

$$Case_{ij} = (Noun_{ij}, Mark_{ij})$$

ここで、$Pred_i$ は述語が満たすべき制約を意味し、$Case_{ij}$ はそれぞれの格要素が満たすべき制約を意味している。$Case_{ij}$ のうち $Noun_{ij}$ は自立語部が満たす制約で、$Mark_{ij}$ は付属部が満たす制約である[2]。

それぞれの格フレーム Pat_i を以下の文法規則に変換する。

$$
\begin{aligned}
\mathrm{VP} \rightarrow\ & \mathrm{NP}(Noun_{i1})\ \mathrm{N\text{-}V}(Mark_{i1}) \\
& \mathrm{VP}(Pred_i,[\mathrm{CID}\ i]),\ \mathrm{GID} = (i,1) \\
& \vdots \\
\mathrm{VP} \rightarrow\ & \mathrm{NP}(Noun_{in})\ \mathrm{N\text{-}V}(Mark_{in}) \\
& \mathrm{VP}(Pred_i,[\mathrm{CID}\ i]),\ \mathrm{GID} = (i,n)
\end{aligned}
$$

さらに埋め込み文を解析するために、主部が NP である文法規則も合わせて生成する。

$$
\begin{aligned}
\mathrm{NP} \rightarrow\ & \mathrm{VP}(Pred_i,[\mathrm{CID}\ i])\ \mathrm{V\text{-}N}(\phi) \\
& \mathrm{NP}(Noun_{i1}),\ \mathrm{GID} = (i,1) \\
& \vdots \\
\mathrm{NP} \rightarrow\ & \mathrm{VP}(Pred_i,[\mathrm{CID}\ i])\ \mathrm{V\text{-}N}(\phi) \\
& \mathrm{NP}(Noun_{in}),\ \mathrm{GID} = (i,n)
\end{aligned}
$$

[2]ALT-J/E [3] の格フレーム辞書では、$Pred_i$ は単語表記であり、$Noun_{ij}$ は意味属性か表記、$Mark_{ij}$ は助詞の表記である。

表 1: 不活性弧のデータ構造

$<(beg,end), grammar, CID, GID\_list, head>$	
beg, end	この弧の範囲
$grammar$	この弧を生成した文法規則
CID	接続 ID。もし未定義ならば ϕ
GID_list	文法の主部を構成する不活性弧の GID のセット
$head$	この弧の主辞

表 2: 活性弧のデータ構造

$\ll (beg,end), grammar \gg$	
beg, end	この弧の範囲
$grammar$	文法規則

それぞれの文法規則には文法規則 ID "GID= (i,j)" が付与され、右辺シンボルのうち VP には接続 ID "[CID i]" が付与される。

2.1.3 文法規則 ID

格フレームから変換された文法規則は、同じ述語に対して 2 度適用することはできない。多重適用を避けるために、それぞれの文法規則にユニークな GID を与える。パーサーは新たな構造を生成する時に、既に適用された文法規則を用いないように GID を確認しながら構造を生成していく。

さらに、埋め込み文のための規則、NP→ ··· は同じ格要素から変換された VP→ ··· と同時に適用されてはならない。同一の格要素から変換される文法規則には同じ GID を与えることにより、同時適用を避ける。

一方、自由格のためのデフォルト文法規則は GID を持っておらず、この規則は複数回適用することができる。

2.1.4 接続 ID

分割された文法規則の同一性を保つために、文法規則の右辺シンボルのうち、格フレーム木構造の連結点である述語部には CID を与える。この時、同じ述語から変換されたシンボルには同一の CID を与える。CID を持つ文法規則は、同じ CID を持った構造にだけ適用され、別々の格フレームがある述語に二重に適用されることを防ぐ。

デフォルト文法規則には CID を与えない。したがって、これらはどの述語にも接続できる自由格として機能する。

2.2 解析アルゴリズム

解析アルゴリズムは基本的にはボトムアップチャート法である。弧のデータ構造を表 1 と表 2 に示す。一般的なチャート法と比較すると、それぞれの不活性弧は CID と GID_list を追加情報として持っている。さらに、格フレームから変換された文法規則は語彙に関する制約を持っているので、$head$ を不活性弧中に保持する。ただし、ここで用いる文法では主部は常に最後のシンボルであり、また、解析途中で語中の属性値が書き換わることはないので、$head$ は $end-1$ と end の間の語に常に等しい。アルゴリズムを図 3 に示す。新しい弧を生成する時にパーサーは以下の 2 点を満たすかどうかを調べることにより、格フレームの不適切な組合せを避けながら解析を進める。

- もし、新しい弧の構成要素である不活性弧が CID を持っている場合、CID は文法規則に記載された CID と同じでなくてはならない。
- もし、新たな弧を生成する文法規則が GID を持っている場合、その GID は構成要素の不活性弧の GID_list に含まれていてはならない。

3 効率と精度

3.1 計算量と所要メモリ

本解析の計算量と所要メモリは本質的にチャート法と同等である。解析中にはパーサーは、範囲と非終端記号、CID、GID_list、$head$ がすべて同じ不活性弧については一つだけを保持する。このことは、CID、GID_list のバリエーションは非終端記号のバリエーションと同等であることを意味している[3]。CID の数は文法の大きさ $|G|$ に比例する。また、ある格フレームにおける GID_list のバリエーションの数は、m をその格フレームの格要素数とすると、2^m である。したがって、入力文の長さを n とすると、所要メモリは $O(n^2 2^m |G|)$ であり、計算量は $O(n^3 2^m |G|^2)$ となる。なお、格要素の数 m は一般に 2 ～ 3 程度の小さな数である。

表 3 は下記の試験例文を Sparc Station 20 で解析した時の解析時間と所要メモリである。

- A が B を投げる。

[3] $head$ は常に最後の文節であるので範囲からユニークに決まる。

1. 節点 i と $i+1$ の間の語 W_i (品詞 $= X$) に対して、不活性弧

$$< (i, i+1), X \rightarrow W_i, \phi, (), W_i >$$

を生成する。

2. 不活性弧

$$C =< (j, i+1), X \rightarrow \cdots, a, (g_1 \cdots g_l), h >$$

に対して、

(a) 文法規則 $Y \rightarrow X$ と C が [A] の条件を満たすならば、不活性弧

$$< (j, i+1), Y \rightarrow X, u, g', h >$$

を生成する。ただし、u と g' は [B] で計算される。

(b) 文法規則 $Y \rightarrow X X_1 \cdots X_m$ と C が [A] の条件を満たすならば、活性弧

$$\ll (j, i+1), Y \rightarrow X \circ X_1 \cdots X_m \gg$$

を生成する。

(c) 活性弧

$$\ll (k, j), Y \rightarrow X_1 \cdots \circ X \cdots X_m \gg$$

の文法規則と C が [A] の条件を満たすならば、活性弧

$$\ll (k, i+1), Y \rightarrow X_1 \cdots X \circ \cdots X_m \gg$$

を生成する。

(d) 活性弧

$$\ll (k, j), Y \rightarrow X_1 \cdots X_m \circ X \gg$$

の文法規則と C が [A] の条件を満たすならば、不活性弧

$$< (k, i+1), Y \rightarrow X_1 \cdots X_m X, u, g', h >$$

を生成する。ただし、u と g' は [B] で計算される。

[A] 不活性弧 $< reg, grammar, a, (g_1 \cdots g_l), h >$ と文法規則 $Y \rightarrow \cdots$, GID $= g$ 内の右辺シンボル $X(x)$ の比較は、以下の通り：

1. 制約 x が [CID b] を含むならば、b は a と同じか、または、a は ϕ であること。
2. もし $g \neq \phi$ ならば、g は $(g_1 \cdots g_l)$ に含まれていないこと。
3. h は [CID b] 以外の制約 x を満たすこと。

[B] 条件 [A] の不活性弧と文法規則から新しい CID c と GID_list g' を生成する規則は以下の通り：

1. [CID c] は [CID a] と [CID b] の単一化の結果。
2. $g' = (g_1 \cdots g_l) \cup (g)$。

図 3: 解析アルゴリズム

表 3: 解析時間と所要メモリ

述語数	語数	解析時間 (sec)	活性弧数	不活性弧数	候補総数
1	5	0.022	115	29	11
2	11	0.030	381	98	220
3	17	0.053	815	206	8069
4	23	0.100	1417	353	364034
5	29	0.181	2187	539	18290244
6	35	0.301	3125	764	$\sim 10^9$
7	41	0.475	4231	1028	
8	47	0.710	5505	1331	
9	53	1.013	6947	1673	
10	59	1.399	8557	2054	

- A が B を投げ、C が D を投げる。
- A が B を投げ、C が D を投げ、E が F を投げる。
- ...

それぞれの名詞 (A, B, ...) は格フレーム辞書中のすべての制約を満たす。格フレーム辞書には、NTT の日英機械翻訳実験システム ALT-J/E [3] の結合価パタン辞書 (約 14,000 パタン収録) を用いた。同辞書中に「投げる」は 4 つの格フレームが記載されている。結果は、計算量がほぼ $O(n^3)$ であり、所要メモリが $O(n^2)$ であることを示している。

3.2 解析精度

解析精度を調べるために、本手法によるパーサーを新聞記事に適用し、格要素の係り先が正しい述語である割合を測定した。格フレーム辞書は前実験と同様に ALT-J/E の辞書である。格フレーム辞書は名詞–動詞の係り受けに関しての辞書であるので、その他の関係を多数含む実際の文章に本手法を適用するために、いくつかのヒューリスティクスをパーサーに導入した。

- VP と VP の接続に関しては、[1] で提案された優先順位[4]を用いる。
- 「A と B」の場合は、A と B が同一の意味カテゴリを持っている場合は、その弧には追加スコアを与え、違う意味カテゴリの場合には、減点する。
- 格フレーム辞書から作られた文法規則はデフォルト規則よりも優先される。
- 以上で順位づけできなかった場合には、近い係り先のものを優先する。

これらのヒューリスティクスは、不活性弧のデータ構造に *score* を追加することにより、不活性弧のスコアとして実現する。パーサーは解析途中で最高点のもののみを保持する。

実験に用いたのは日経産業新聞の記事から抽出した 200 文である。うち 100 文は格フレーム辞書を作成する時に例文として参照した文であり、残りの 100 文は参照していない文である。文は平均 46 文字 (8.7 文節) であった。

結果を表 4 に示す。それぞれ 217 と 233 の文節が格要素と解析され、既知の文については 93.1%の係り先が正解であり、未知の文については 86.7%の係り先が正解であった。

表 4: 新聞記事での精度

コーパス	格要素数	正解数
既知	217	202(93.1%)
未知	233	202(86.7%)

4 おわりに

本稿では、格フレームから変換した CFG 規則を用いて、日本語係り受け解析を行う手法について報告した。本手法では、係り受け解析と同時に格フレームによる意味解析が終了しており、効率的な日本語処理プログラムの構築が可能である。

今後、本手法を用いた機械翻訳システムの構築を進める予定である。

参考文献

[1] 白井、池原、横尾、木村. 階層的認識構造に着目した日本語従属節間の係り受け解析の方法とその精度. **情報処理学会論文誌**, 36(10), 1995.

[2] 池原他. **構文体系**, volume 5 of **日本語語彙大系**. 岩波書店, 1997.

[3] S Ikehara, S Shirai, A Yokoo, and H Nakaiwa. Toward an MT system without pre-editing — effects of new methods in **ALT-J/E** —. In *Proceedings of MT Summit III*, 1991.

[4] 日本語の動詞を 52 に分類し、それぞれの分類に優先順位を与える。この優先順位を用いて、VP と VP の接続の尤度を決定する。