B6-4

論理文法におけるギャップの取扱いについて

On Handling Gaps in the Logic Grammar

徳永健伸，岩山真，奥村学，田中穂積
TOKUNAGA Takenobu, IWAYAMA Makoto, OKUMURA Manabu, TANAKA Hozumi

東京工業大学 工学部
Department of Computer Science
Tokyo Institute of Technology

本稿では，自然言語の移動現象を論理文法上で記述するための枠組を提案する．これまでにも，スラッシュ素性の考え方に基づいて，移動現象，特に左外置を扱う枠組はいくつか提案されているが，本稿では，さらにスラッシュ素性と双対な下位範疇化制約という概念を導入する．例として，いくつかの英語の移動現象がこの枠組で自然に記述できることを示す．また，Prolog上での実現についても述べる．

1 はじめに

これまでにも，PereiraのXG[1]やDahlのDG[2]など，論理文法上でギャップを扱う枠組がいくつか提案されている．このうち，文法の記述能力という点では，DGが非常に強力であるが，DGを効率的に実現する方法はまだ提案されていない．XGはDCG[3]を，英語などに現れる左外置現象を扱えるように拡張したもので，文法記述能力に関してはDGのサブセットと考えることができる[4]．また，XGをChomskyのGB理論に基づいて拡張した枠組としてRLGがある[5]．RLGでは，XGで導入された手法を右方移動も扱えるように拡張しているが，基本的な機構は同じである．我々もXGと同等の文法記述能力を持つ枠組XGSを提案し，上昇型構文解析器上に実現している[6,7]．

文法記述において，等位接続詞を含む構造(等位構造)は大きな問題となる．特に，等位接続される一方の構成素にギャップが含まれる場合は，XGやXGSのような枠組ではうまく扱えない．これまでの等位構造を扱う研究の流れの1つとして，'and'，'or'などの等位接続詞が文中に現れたら，通常の解析を中断し，等位構造の部分だけを特別な手続きによって解析する，という方法がある[8,9,10]．Dahlらの MSGは，この手法を論理文法上に実現したものである[11]．しかし，このような特別な手続き付加による方法では，文法開発者が解析器の動作を意識しなければならないし，手続きの部分を変更することも困難である．我々はあくまでも宣言的な記述によって等位構造を扱う方がよいと考えている．

本稿では，DCGに，スラッシュ記法と下位範疇化制約という2つの概念を導入することによって，これらのギャップを自然に記述する枠組を提案する．このうち，スラッシュ記法については，すでに，XGを始めいくつかの枠組で提案されているものと同じであるが，下位範疇化制約は，本稿で新たに提案するものであり，スラッシュ記法と組み合わせることによって強力な文法記述力を得ることができる．

以下，2節では，スラッシュ記法と下位範疇化制約について説明し，3節では，英語を例にとり，スラッシュ記法と下位範疇化制約を使った文法の記述例について説明する．4節では，DCGを拡張することによって，下位範疇化制約が容易に実現できることを示す．最後に，5節では，本稿のまとめと今後の研究課題について述べる．

2 スラッシュと下位範疇化制約

本節では，まず，スラッシュ記法を簡単に説明し，本稿で提案する下位範疇化制約について説明する．

2.1 スラッシュ記法

スラッシュ記法は，英語などに現れる左外置を記述するために，今野らのXGSで導入された記法である[6]．この記法はGPSG[12]のスラッシュ素性の考え方が基本になっている．たとえば，英語の関係詞節をXGSによって記述すると次のようになる．

```
np(_) → det(_), noun(N), rel_c(_)../np(N).      (1-a)
rel_c(_) → rel_pron(_), sentence(_).             (1-b)
```

ここで，記号"../"はスラッシュと呼ばれ，スラッシュの右側のカテゴリをスラッシュカテゴリという．一般に，カテゴリ"m../c"は，

> 「カテゴリ'm'を根とする解析木ができたときに，その根の下に，ギャップを直接構成素として支配するカテゴリ(スラッシュカテゴリ)'c'が1つ存在する」

ことを表している．したがって，上記の例の'rel_c(_)../np(N)'は，名詞句をギャップとして持つ関係節を表している．また，'noun'とスラッシュカテゴリ'np'の引数として同じ論理変数を与えることによって，ギャップをその先行詞に対応付けることが容易にできる．この記法の利点は，文法開発者がギャップが現れる位置を意識して文法規則を書かなくても，システムが解析中に自動的に痕跡の位置を発見することを保証している点である．この他にも，XGSはギャップの現れる有効範囲を制御するための記法なども用意している．また，このスラッシュ記法を用いて，英語の受動化，疑問文などの規則も記述できるが，詳細については[6,13]を参照して欲しい．

2.2 下位範疇化制約

下位範疇化制約は，スラッシュ記法と双対な概念である．スラッシュ記法が，「スラッシュカテゴリが親カテゴリの下で欠けていること」を要請するのに対し，下位範疇化制約は，「下位範疇化カテゴリが親カテゴリの下に必ず存在する」ことを要請する．例として，次の英語の文法規則を考えよう．

np(_) → det(_), noun(N), rel_c(_)@rel_pron(N). (2-a)
rel_c(_) → np(NP), sentence(_)//np(NP). (2-b)
rel_pron([ ]) → [which]. (2-c)

ここで，"//"はXGSのスラッシュと同じである．実現の都合上本稿では，この記号を使う．"@"は，下位範疇化制約を記述するために導入した記法で，"@"の右側を下位範疇化カテゴリという．一般に，カテゴリ"m@c"は，

「カテゴリ'm'を根とする解析木ができたときに，その根の下に，カテゴリ'c'が存在する」

ことを表している．規則(2-a)中の'rel_c(_)@rel_pron(N)'はカテゴリ'rel_c'がカテゴリ'rel_pron'を必ず支配していることを表している．この場合も，スラッシュ記法と同様に，論理変数によって情報を受け渡すことができる．

下位範疇化制約は，下位範疇化カテゴリとそれを支配するカテゴリの間で情報のやりとりをおこなう手段として使うことができる．具体的な使用例については次節で述べる．

3 記述例

本節では，英語を例にとり，いくつかの言語現象を我々の枠組で記述し，その有効性について述べる．

3.1 関係節

XGやXGSによる関係節の解析では，関係代名詞は関係節の存在を示す単なる指標としてしか扱われていない．たとえば，規則(1-a)では，先行詞'noun'と関係節が支配するギャップ'np'は論理変数'N'によって対応付けられている．しかし，規則(1-b)の関係代名詞'rel_pron'と，先行詞あるいはギャップとの対応はとれない．したがって，この規則では，次のような非文も受理してしまう．

*She is the girl whose I've been looking for.

論理文法では，情報のやりとりを論理変数を介しておこなうが，論理変数のスコープは同一節内に限られているため，先行詞，関係代名詞，ギャップを正しく対応付けるためには，次のように規則を展開しなければならない．

np(_) →
　det(_), noun(N),
　rel_pron(_), sentence(_)../np(N). (1-c)

しかし，この方法では最悪の場合，規則の数が'np'をヘッドに持つ規則数と'rel_c'をヘッドに持つ規則数の和から積に増えてしまう．また，図1に示す所有格の関係代名詞のように，関係代名詞を含む構造が外置される場合には，このように規則を展開するのは現実的ではない．[14]では，ギャップとして1度に複数の要素を指定することによって，所有格の関係代名詞を扱うことを提案しているが，正しい対応付けができないという点では，本質的な解決になっていない．

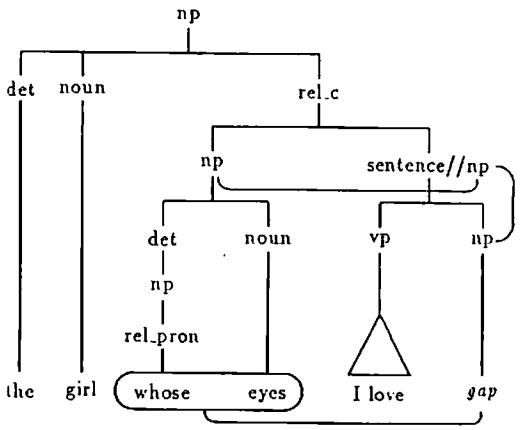

図1 誤った解析例

下位範疇化制約を用いれば，規則(2-a), (2-b), (2-c)および以下の規則で，すべての関係節を扱うことができる．

np(Rel) → rel_pron(Rel) (2-d)

図1に規則(1-a), (1-b)に基づく解析木，図2に規則(2-a)～(2-d)に基づく解析木を示す．太線は支配関係を，細線はカテゴリ間の対応関係を表す．図2では，先行詞，関係代名詞，ギャップの対応関係が正しくとれていることがわかる．

3.2 先導の規約

関係節を形成する時に，関係節中から接頭に移動する構成素が，単に先行詞と同一の名詞句だけでなく，その名詞句を支配する上位の名詞句が移動することができる．これを先導の規約という[15]．たとえば，次のような文である．

[$_{np_1}$ the book [$_{np_2}$ the covers of which]
　[$_{rel\_c}$ I designed *gap*]]

この例では，先行詞'book'と関係代名詞'which'の対応をとり，さらにその関係代名詞を支配している上位の名詞句'np_2'が関係節の中で欠けていると考える．先導の規約による移動は，規則(2-a)～(2-d)を使い，関係節とまったく同様に解析できる．

3.3 等位構造

'and'や'or'などの等位接続詞によって，任意の構成素が接続される構造を等位構造という．もちろん，等位接続される構成素の組合せには制約があるが[16]，ここでは，言語学的な詳細には立ち入らない．

もっとも単純な等位構造は，

Tom loves Mary and John loves Jane.

のように完全な構成素が等位接続詞によって接続される場合である[1]．これは，DCGの枠組でも容易に記述できる．しかし

[1]必ずしも同じカテゴリでなくてもよい．

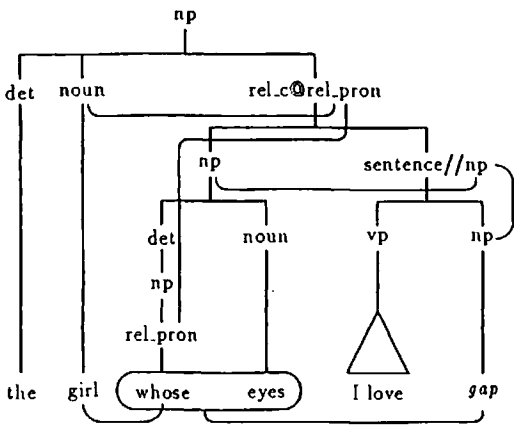

図2 正しい解析例

ながら，等位接続される構成素の一方がギャップを含む場合は，簡単ではない．たとえば，次の例文を考えよう．

I love but John hates the girl.

XGSで，この文を解析するため規則を書くと次のようになるであろう．

```
sentence(_) →
    sentence(_)../np([case:obj]), [but], sentence(_).
```

スラッシュカテゴリ'np'の引数として'[case:obj]'を渡している点に注意して欲しい．これによって，'sentence'の下で欠けている'np'は目的格の素性を持っていなければならないことを要請している．しかしながら，この規則では，等位接続される最初の'sentence'の目的語と次の'sentence'の目的語が同一であることを記述していない．XGSでは，この規則中で2番目の'sentence'が支配する構成素に関する情報を参照する手段がないため，このような記述は不可能である．

一方，下位範疇化制約を用いると，この対応を次のように記述することができる．

```
sentence(_) →
    sentence(_)//np([case:obj|NP]), [but],
    sentence(_)@np([case:obj|NP]).
```

この規則はボディの最初の'sentence'の下で欠けている'np'の持つ情報と，2番目の'sentence'が支配する'np'の情報が単一化可能であることを要請している．したがって，意味的にも最初の'sentence'と2番目の'sentence'の目的語が同一であることが正しく解析できる．

4 実現

本節では，2節で述べたスラッシュ記法と下位範疇化制約の実現について述べる．実現はDCGを拡張することによっておこなった．本稿で提案した文法記述形式を仮にDCG++と呼ぼう．DCG++で記述された文法規則は，変換系によりDCGに変換された後，Prologシステムに入力され，いくつかの補助述語と共にPrologインタプリタによりそのまま実行される．したがって，解析器はDCGと同様に下降型深さ優先解析器として動作する．

4.1 スラッシュ記法の実現

スラッシュ記法の実現は，[1,6]と同様な手法を用いているので簡単に説明する．詳細は[1,6]を参照して欲しい．ギャップの情報を引き渡すために，各文法規則のカテゴリに，スタックを表す2つの余分な引数を用意する．この2つの引数は，それぞれ，そのカテゴリを解析する前(入力スタック)と解析した後(出力スタック)のスタックの状態を保持している．Pereiraにならい，以下，これをXリストと呼ぶ．スラッシュカテゴリを含むカテゴリは，変換により，スラッシュカテゴリを入力スタックにプッシュしたカテゴリに変換される．スラッシュカテゴリをスタックからポップするのは，スラッシュカテゴリと同じカテゴリが完成した時であるから，文法規則中にスラッシュカテゴリとして出現するカテゴリについては，

cat(Args, [cat(Args)|X0], X0, S, S).

という規則を追加する[2]．これは，入力スタックの先頭に文法カテゴリ'cat'があればこれをスタックからポップし，残りのスタックを返すための規則である．また，スラッシュを含むカテゴリを解析した直後には，スタック中のスラッシュカテゴリをポップしたことを検査する述語'depth_check'を補強項として挿入する．

4.2 下位範疇化制約の実現

下位範疇化制約もスラッシュ記法と同様な手法によって実現できる．下位範疇化制約の情報を扱うために，Xリストと同様に，入力用，出力用の2本のリストを用意する．このリストを以下では，Aリストと呼ぶ．スラッシュカテゴリとギャップの対応付けは非交差でなければならないので，Xリストはスタックによって実現されているが，下位範疇化制約では，下位範疇化カテゴリと実際に見つかったカテゴリの対応付けが非交差である必要はないので，Aリストはスタックではなくリスト(正確には集合)であることに注意されたい．

以下の手順にしたがって規則の変換をおこなう．

- "@"を含むカテゴリの入力Aリストには下位範疇化カテゴリを追加する．
- 下位範疇化カテゴリを含むカテゴリの直後には，スラッシュ記法の場合と同様に述語'depth_check'を補強項として挿入する．これは，下位範疇化カテゴリが親カテゴリの下に存在することを確かめるためである．
- 下位範疇化カテゴリとして文法規則中に現れるカテゴリをヘッドに持つ文法規則の末尾には，述語'found'を補強項として挿入する．

規則(2-a), (2-b), (2-c)の変換結果を以下に示す．'A', 'X'で始まる変数はそれぞれ変換系によって付加された，Aリスト，Xリストである．

[2]ただし，ここでは，次に述べるAリストは省略してある．また，'S'は並分リストを表す．

```
np(NP, A0, A0, [np(NP)|X0], X0, S, S).          (2-a')
np(_, A0, A3, X0, X3) →
    det(_, A0, A1, X0, X1),
    noun(N, A1, A2, X1, X2),
    rel_c(_, [rel_pron(N)|A2], A3, X2, X3),
    { depth_check([rel_pron(N)|A2], A3) }.      (2-a")
rel_c(_, A0, A2, X0, X2) →
    np(NP, A0, A1, X0, X1),
    sentence(L, A1, A2, [np(NP)|X1], X2),
    { depth_check([np(NP)|X1], X2) }.           (2-b')
rel_pron([ ], A0, A1, X0, X0) →
    [which],
    { found(A0, A1, rel_pron([ ])) }.           (2-c')
```

述語'found'は，下位範疇化カテゴリの引数として指定された変数(上からの情報)と実際に解析によって完成したそのカテゴリの引数(下からの情報)を単一化する．ここで，下位範疇化カテゴリの引数として渡された変数に値がすでに束縛されていた場合，'found'による単一化の結果をどのようにして親カテゴリに返すかが問題となる．値を返すために別の変数を用意することも考えられるが，本稿の実現では，CIL[17]で導入された部分項を用いてこれを実現している．部分項は，ラベルと値の対の集合である．部分項の単一化は基本的にマージ操作なので，同じ変数を用いて値を返すことが可能になる．実際には，最後尾の要素が必ず変数であるようなPrologのリストを用いて部分項を実現している．述語'found'の定義を以下に示す．ここで，述語'unify_args'は，リスト中の各部分項について，単一化をおこなう述語である．

```
found([ ],[ ],_) :-!.
found([Tc|Ss],Ss,T) :-
    Tc =.. [Pc|Acs],
    T =.. [Pc|As],
unify_args(Acs,As), !.
found([Tc|Ss],[Tc|Ss],T) :-
    found(Ss,Ss,T).
```

この'found'の定義からわかるように，ある親カテゴリが支配する下位範疇化カテゴリは一意に決まるものと仮定している．これは，無駄な単一化による効率の低下を防ぐためである．

5 おわりに

本稿では，論理文法上でギャップを扱うために，下位範疇化制約という新しい概念を導入し，いくつかの英語の移動現象が自然に記述できることを示した．また，DCGを拡張することによって容易に実現できることも述べた．しかしながら，本稿で紹介した実現は，下降型深さ優先解析器なので，DCGと同様に左再帰規則が扱えないという大きな問題がある．また，BUP[4]やLangLAB[7]のように再計算を回避する機構も持っていない．今後，下位範疇化制約を上昇型解析器上に実現すること，および再計算を防ぐなどの効率化を図る必要がある．

参考文献

[1] F. Pereira. Extraposition grammar. *American Journal of Computational Linguistics*, 7(4):243–256, 1981.

[2] V. Dahl and P. Saint-Dizier. *Constrained Discontinuous Grammars : a linguistically motivated tool for processing language*. Technical Report, INRIA, 1986.

[3] F. Pereira and D. Warren. Definite clause grammar for language analysis– a survey of the formalism and a comparison with augmented transition networks. *Artificial Intelligence*, 13(3):231–278, 1980.

[4] Y. Matsumoto. *Natural Language Parsing Systems based on Logic Programming*. PhD thesis, Kyoto University, 1989.

[5] E. P. Stabular, Jr. Restricting logic grammars with government-binding theory. *Computational Linguistics*, 13(1-2):1–10, 1987.

[6] 今野聡，田中穂積．左外置を考慮したボトムアップ構文解析．コンピュータソフトウエア，3(2):115–125，1986．

[7] 徳永健伸，岩山真，田中穂積，上脇正．自然言語解析システム LangLAB．情報処理学会論文誌，29(7)，1988．

[8] W.A. Woods. Experimental parsing system for transition network grammar. In *Natural Language Processing*, Algorithmic Press, 1971.

[9] N. Sager. *Natural Language Information Processing: A Computer Grammar of English and Its Applications*. Addison-Wesley, 1981.

[10] 武舎広幸．LLパーザに基づく決定性構文解析．，日本ソフトウエア科学会第5回大会論文集，65–58ページ，1988．

[11] V. Dahl and M. C. McCord. Treating coordination in logic grammars. *American Journal of Computational Linguistics*, 9(2):69–91, 1983.

[12] G. Gazdar and A. F. Pullum. *Generalized Phrase Structure Grammar:A Theoretical Synopsis*. Indiana University Linguistics Club, 1982.

[13] 岩山真，徳永健伸，田中穂積．*LangLAB User's Manual*．東京工業大学 工学部 情報工学科 田中研究室，第1版，1989．

[14] 林達也．拡張CFGとその構文解析法YAPXについて．情報処理学会論文誌，29(5):480–487，1988．

[15] 大塚高信，中島文雄．新英語学辞典．研究社，1982．

[16] I. Sag, Gazdar G., T. Wasow, and S. Weisler. *Coordination and How to Distinguish Categories*. Technical Report CSLI-84-3, CSLI, 1984.

[17] K. Mukai and H. Yasukawa. Complex indeterminates in prolog and its application to discourse models. *New Generation Computing*, 3(4):441–466, 1985.