7C−1

単一化に基づく生成の効率向上

上田良寛
ATR自動翻訳電話研究所

1. はじめに

双方向文法を用いる生成のメカニズムとして、タイプ付素性構造主導型生成を提案した[1]。これは、タイプ付素性構造を用いた制約条件の宣言的な記述により、文法規則の適用可能性をチェックし、これによってトップダウン生成システムを制御するものである。

このような単一化を基礎メカニズムとして用いているシステムでは、単一化それ自体が計算コストのかかるプロセスであるため、計算時間のほとんどが単一化によって消費される。このため、単一化のアルゴリズムを改良する、または、単一化の適用回数を減少させることにより、生成システムの効率を向上させることが出来る。ここでは、後者の方法、すなわち、単一化の適用回数を減らすことによる効率向上を、文法および生成メカニズムの両面から試みた。

はじめに、生成メカニズム自身の問題を考察する。次に、ここで用いる文法のフォーマリズムを簡単に説明し、この文法を生成に適用する際の問題点の考察を行う。第4章ではそれらに対する解決策を検討し、第5章はその効果を実験によって示す。

2. メカニズムの問題点

2.1 単一化による制約条件のチェック

タイプ付素性構造主導型生成では、ルールの適用可能性の制約条件は、CFG規則に付与された単一化の指示のなかに埋め込まれている。例えば、次の規則では最後の行が制約条件になっている。

```
(defrule vp =hc*=> (dyadic xp xp)
  (<!m !head> == <!h-dtr !head>)  ;Head FP
  (<!m !content> == <!h-dtr !content>)  ;Semantic FP
  (<!h-dtr !subcat first> == <!c-dtr-1 !synsem>)
  (<!h-dtr !subcat rest> == <!m !subcat>) ;Subcat FP
  (<!m !content reln> == [dyadic]))    ;生成時の制約
```

このように制約を融合させることの利点は、統一的な記述ができることと、特別な解釈メカニズムを導入する必要がないことである。

しかし、単一化は、成功か失敗かによって制約条件をチェックすることと、単一化された結果の素性構造を用いる(生成では親句構造から子句構造へ素性を伝播させる)ことの2つの目的で用いられる。結果として得られる素性構造がほぼ出来上がった段階で、制約条件の部分により単一化が失敗すると、それまでの過程が無駄になる。

2.2 素性構造中の選言による効率の低下

素性構造中の選言は、辞書の表層形の複数の候補があり、それらからの選択が、文の他の構成要素が決定されるまでなされない場合に用いられていた。例えば、動詞の表層形は主語の単複によって決定される。選言を導入すれば、主語が決定した段階で、自動的に表層形が決定される。

しかし、選言を含む素性構造の単一化は、一般にコストのかかるプロセスである。選言を保持しながら単一化を進めることは、効率を低下させる原因になる。

3. 文法を生成に適用する際の問題点

ここで用いる文法は、HPSGの新しい解析方法[2]に基づいている。この解析では、Head feature principle, Semantic feature principle などの基本的な概念は同一であるが、全体の一貫性を保ちながら多くの言語現象が説明できるよう拡張されている。

新しいHPSGでは、素性構造中で集合を扱い、単一化でその和をとっている。例えば、形容詞による修飾は、形容詞と名詞の制約の和(UNION)をとって、次のような意味表現を形成する。ここで中括弧({...})は集合を表す。

$$\begin{bmatrix} PARAM & [1] \\ RESTR & \left\{ \begin{bmatrix} RELN & book \\ INST & [1] \end{bmatrix} \begin{bmatrix} RELN & red \\ ARG & [1] \end{bmatrix} \right\} \end{bmatrix}$$

また、Quantifier Storage (Cooper Storage)という概念が導入され、quantifierは意味素性と別に記述されるようになった(QSTORE素性)。これも集合で表現され、それを伝播させるQuantifier Inheritance Principleは、子のQSTORE素性の和を親に与えている。

これらを生成に適用した場合には、集合の要素をそれぞれの構成要素(子句構造)に適切に分配させなければならない。Shieber[3]は、集合をリストで表現し、分配の際にshufflingという操作を行っているが、これは非決定性を増加させることになり、効率の低下をもたらす。

4. 効率向上の検討

4.1 制約条件チェックの独立

2.1で述べたように、単一化には制約条件のチェックと素性の伝播という2つの意味が受け持たせられている。制約条件のチェックの部分を先に行い、それに生き残ったものに対してだけ、素性を伝播させるための部分を単一化するようにすると効率を向上させることができると考えられる。

また、この制約条件のチェックを行った結果は、一般的には不要であるため、単一化ではな

Efficiency Improvement of Unification-based Generation
Yoshihiro Ueda
ATR Interpreting Telephony Research Laboratories

く、単一化の可否のチェックをするだけのプロセスですますこともできる。

文法規則中では、制約条件の部分は、その他の部分と明示的に分離して記述する。制約条件は生成だけで用いられる情報であり、文法をコンパイルする際にはこれを解析の文法から除くことが可能になる。これにより、解析の際の単一化に余分な負荷がかかることがなくなる。

4.2 表層形選択の遅延

先に述べたように、選言を含む素性構造を生成プロセスのあいだ保持していることは効率の低下をもたらす。ここでは、選言を付加するかわりに辞書エントリへのidを付加するにとどめ、最終的に句構造全体が完成してから選言部分と単一化し、必要な表層構造を得られるようにする。

4.3 文法の非決定性の排除

素性構造中の集合とその和の計算は、単一化の形式化にはなじまない。また、単一化を拡張して、関数呼び出しができるようにすることも考えられるが、解析と生成の双方向で文法を扱うことができなくなる。

ここでは、集合のかわりに差分リストを用いる。差分リストをそのまま用いると表層の構成要素順序が意味表現に反映されることになる。逆に言えば、生成システムの入力素性構造表現が、生成結果を規定してう。例えば、形容詞や副詞の適切な順序を生成システムがコントロールできなくなり、好ましくない。しかし、可能な順序の組み合せを選言として表現しておき、文法の制約によってそれにフィルタをかけて適切な語順で生成することも可能である。

差分リストを含めリストの欠点は、先頭から順にたどることしかできないことである。形容詞の生成の場合のように差分リストの先頭要素から順に生成されるので問題がない。しかし、QSTOREのようにそうでない場合には、先頭要素が決定されるまで、それ以外の子構成要素の内容は決定されない(図1における子句構造2)。

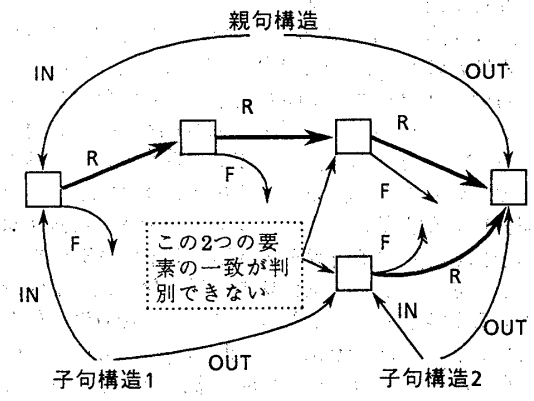

図1 差分リストからの要素の供給

これを解決する方法として、

(1) 差分リストの単一化の改良

(2) チェック関数の導入

の2つの方法を考察する。

差分リスト単一化の改良としては、差分リストを双方向にすることが考えらる。子構成要素が末尾にあれば適切に意味表現が供給できることになるが、実際の生成においてはQSTOREのリストの途中の要素から生成される可能性もあり、この方法でも不十分である。

次に手続きによるチェックを導入することを検討する。このQuantifier Storageによる問題に対する手続きは、意味素性に対応するquantifier要素を決定するようなものが必要になる。

5. 実験結果

生成する例文として、次の文を考える。

My boss attends every big conference.

まず、制約条件のチェック方法による違いを見る。ここでは、Sun Common Lisp / SPARCstation 1+で実験を行っている。

(1)制約条件を融合した単一化の場合	4.15 sec
(2)制約条件のみ独立に単一化した場合	3.56 sec
(3)単一化の可否でチェックした場合	4.10 sec

わずかではあるが、制約条件を独立させる効果がわかる。単一化の可否でチェックした場合の効果はここでは現れていない。

次にQSTOREによる非決定性の排除の実験結果を、3種の制約条件チェックそれぞれについて示す。

双方向差分リスト	(1)2.34	(2)2.26	(3)2.17
チェック手続	2.45	1.77	1.55

となり、効果が分かる。また、ここでは単一化の可否でチェックした場合(3)の効果も現れている。

表層形選択の遅延による効果を見るために次の例文を考える。

This is the conference office.

この例では、文の主語に相当する意味表現が*speaker*というラベルだけで与えられており、主動詞の決定時には人称、数が不明になっており、am/are/isのいづれかが決定できない。

表層形選択の遅延を導入すると、5.42 sec→2.41 secへと高速化された。

6. おわりに

以上、先に提案したタイプ付素性構造主導型生成の効率化について述べた。今後はこのような高速化手法を用いながら、システムおよび文法辞書の拡張を図っていきたい。

参考文献

[1] 上田、小暮:「タイプ付素性構造による生成過程の制御」、情報処理学会第40回全国大会、1E-2、1990.

[2] Pollard, C. and I. A. Sag: "An Information-Based Syntax and Semantics, Volume 2, Topics in Binding and Control", CSLI Lecture Notes, 1991 (to appear).

[3] Shieber, S. M., et al.: "Semantic-Head-Driven Generation", Computational Linguistics, Vol. 16, No. 1, 1990.