タイプ付き素性構造主導型生成†

上田 良寛††*

素性構造を入力とする生成システムにおいて，タイプ付き素性構造を導入することにより，CFG 規則のトップダウンの適用を，宣言的な記述により制御する方法について述べる．この生成システムで用いる文法は，解析と共有できる双方向文法となっており，解析と生成の文法の一貫性を保つのに効果がある．また，この生成システムでは，句構造の複製を減らすために，選言を含む素性構造を導入している．このシステムは対話文翻訳システムの生成部として開発したものであり，その文法は，電話会話における話者の意図を適切に解析し，生成文中に再現するよう設計されている．

1. はじめに

対話においては，発話に含まれる話者の意図（発語内的力）は，その命題内容と同様に重要な働きをもっている．このため，対話文の翻訳システムでは，発話意図を正しく解析し，生成文中で再現することが必要になる[11]．このような対話文翻訳システムの生成部は，適切な発語内的力をもつ文を生成する必要があるため，文法・辞書が複雑になる傾向がある．この負担を軽減するため，それに用いる文法は，宣言的に記述できることが望ましい．この目的に適した文法として，HPSG[14),15)] などの単一化文法がある．特に，解析と文法・辞書を共有することのできる双方向文法ならば，文法の一貫性を保つのに効果がある[2)]．

このような宣言的に記述された文法を，単純にトップダウンに適用するような生成システムでは，同じ規則が何度も適用され，停止に至らない場合がある[17)]．また，最終的に書き換えの失敗に至る無駄な規則が適用され，処理効率を下げる可能性がある．

これまでに開発されてきた双方向文法を用いた自然言語生成システムでは，このような問題を避けるため，文法中で句構造の展開の順序を指定することによって制御する方法などが採用されてきた[5)]．しかし，単一化文法の宣言的記述ができるという利点を損なうことのないように，宣言的な記述により制御できることが望ましい．

このため，ここでは，文法規則に付加された制約条件によって，適用する文法規則を選択し，生成プロセスの制御を行うことができるようにする機構を導入する．この制約条件を統一的に記述するために，タイプ付き素性構造[1)]を導入した．本稿では，このタイプ付き素性構造によって文法適用の制御を行う方法について述べる．

また，規則適用の際に複数の規則の候補が生じた場合，その数だけ派生木（素性構造で表現された句構造）が複製される．素性構造に選言を含めることにより，派生木全体を複製することを避け，生成プロセスの効率向上を図る方法を示す．

本稿では，このような機構をもつ，双方向性文法を用いる日英翻訳システムの英文生成部と，そのための英文法について述べる．まず次章でルール適用制御の必要性を，3章でその方法を述べる．4章では，これをタイプ付き素性構造の導入によって解決する方法を示す．5章では選言を含む素性構造の導入による効果を示す．6章では本システムで採用しているHPSG[14),15)] に基づいた文法について述べ，生成例を示す．7章には他の研究との関連について述べる．

2. ルール適用制御の必要性

2.1 停止性の問題

この生成システムのメカニズムは，基本的には，CFG 規則をスタートシンボルからトップダウンに適用することにより句構造を組み立てるものである．これと同時に，規則に付加された素性構造の指定に従って親ノードの素性構造を子ノードに伝播させる．

Shieber ら[17)]は，このようなトップダウン生成システムには停止性の問題があることを指摘している．この例として次の規則(1)を考える（ここで用いる文法規則の記法に関しては**図1**を参照されたい）．

$$S \Rightarrow (NP\ VP) \qquad (1)$$

† Typed-Feature-Structure-Directed Generation by YOSHIHIRO UEDA (ATR Interpreting Telephony Research Laboratories).

†† (株) ATR 自動翻訳電話研究所
* 現在 富士ゼロックス(株)システム技術研究所
System Technology Research Lab., Fuji Xerox Co., Ltd.

```
<!m !content>
    == <!h-dtr !content>
<!h-dtr !subj>
    == <!c-dtr-1 !synsem>
```

この規則では，右辺の NP（名詞句）に対するノード（!c-dtr-1 で与えられる）には，素性構造による制約が親ノードからは何も与えられていない．このため，このノードは変数のようにふるまい，NP から始まるすべての規則が適用可能で，NP というカテゴリに属するどのようなもの（句，節，語）にも書き換え可能になる．NP に前置詞句や関係詞節が付加されたものも，やはり NP となるため，再帰的に無限に書き換えが行われることになる．これは，句構造のどのノードから派生させていくかという問題であるということができる．

左再帰文法規則も停止性の問題がある．次に示す規則（2）は動詞（右辺の VP）の SUBCAT リスト（下位範疇化フレームのリスト）を，ある要素 XP で埋める規則である．

```
VP =HC*=> (VP XP)    (2)
<!m !content>
    == <!h-dtr !content>
<!h-dtr !subcat first>
    == <!c-dtr-1 synsem>
<!m !subcat>
    == <!h-dtr !subcat rest>
```

解析の場合，この規則の適用は，SUBCAT リストの要素を 1 個減らすことになるので，SUBCAT リストが尽きた時点で，それ以上この規則が適用されることはない．また，解析がボトムアップに行われる場合には，入力文字列が有限長であることも，規則の無限回の適用を制限している．一方，トップダウン生成においては，規則（2）を VP ノードの SUBCAT リストに要素を 1 個増やす方向に適用するため，これらの制約がかからず，無制限に適用可能になる．

適切な時点でこの左再帰規則を適用させることをやめ，他の規則（この場合は次に示す動詞（V）への書き換え規則（3））を適用させることが必要になる．

```
VP == (V)    (3)
```

この問題は，したがって，句構造中のあるノード（この場合は VP）に対して，どの規則を適用させるか

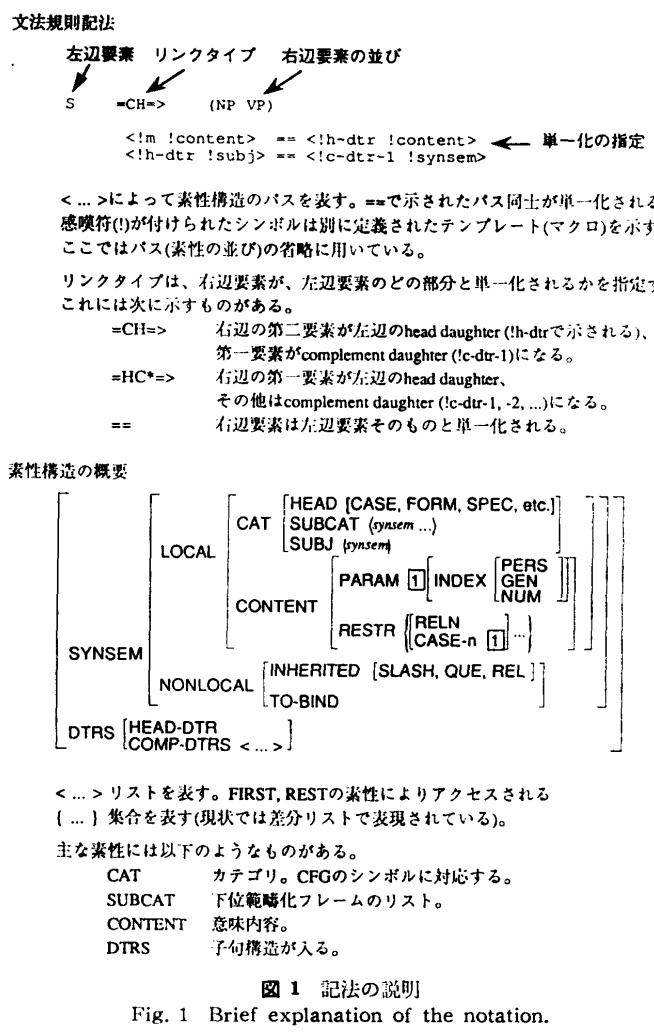

図 1 記法の説明
Fig. 1 Brief explanation of the notation.

という問題であるということができる．

2.2 適切な文法規則の選択

無限に規則が適用される場合でなくとも，文法規則の適切な選択が必要になる場合がある．適切な選択を行って，最終的に失敗につながる句構造の派生を避ければ，効率の向上を図ることができる．

例えば，次の 2 つの文法規則（4），（5）を考える．（4）は形容詞（A）で，（5）は関係詞節（SREL）で，それぞれ名詞（UNDETNOM）を修飾する規則である．

```
UNDETNOM =CH=> (A UNDETNOM)
    !schema-6 !sem-fp-2    (4)
UNDETNOM =HC*=>
    (UNDETNOM SREL)    (5)
    !schema-6 !sem-fp-2
```

ここで，テンプレート !schema-6 と !sem-fp-2 は次のように定義されている．

```
(deffstemp schema-6 ( )
  (<!adj-dtr !mod>
    == <!h-dtr !synsem>))
(deffstemp sem-fp-2 ( )
  (<!m !content>
    == <!adj-dtr !content>))
```

この中で !adj-dtr は !c-dtr-1 と同様に定義されており，規則(4)では A，規則(5)では SREL が指されることが，それぞれのリンクタイプ (=CH=> と =HC*=>) によって指定されている．

例として，これらの文法規則で，名詞句 "the big blue book that I have" を解析して得られる素性構造を考える (**図 2**)．規則(4)，(5)の修飾詞は被修飾名詞の統語意味記述を mod 素性で受け取る．それぞれの修飾詞では，図中に示される parameter-restrictions 構造 (PARAM 素性と RESTR 素性で表現されている) の restriction のひとつを追加する記述がなされており，これが親に渡される．

どちらの規則も同じ構造をもった意味記述を与える．生成の場合，この素性構造からそれぞれ適切な文法規則を適用しなければならない．適切な規則が選択できなければ，間違った方向に派生を続けていき，結果的に途中で行き詰まることになる．このような間違った方向への派生は効率に大きな影響を及ぼす．

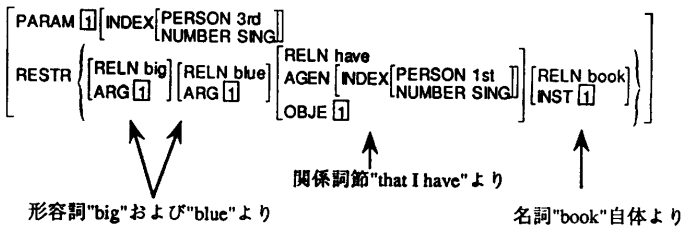

図 2 素性構造の例
Fig. 2 A feature structure example.

3. 文法規則の適用の制御の方法

3.1 既存のシステムにおける解決方法

これまでの双方向システムでは，これらの問題をどのように扱ってきたのだろうか．

Jacobs の PHRED[8] では入力される意味記述から直接すべての構文パターンが得られ，また，その構成要素に意味記述が与えられるようにしている．例えば，動詞 "remove" に対するパターンは，以下のように示される (一部省略)．

```
PATTERN:     ; 構文パターン
  <agent><root=remove><physob>
    <<word=from><container>>
CONCEPT:     ; 意味記述
  (state-change
    (object ?rem-object)
    (state-name location)
    (from (inside-of (object ?cont)))
    (to ...))
  PROPERTIES:
    rem-object=(value 3)
    cont=(value 5)
```

ここで，CONCEPT で表される意味記述と PATTERN で表される構文パターンの記述は，PROPERTIES で示される記述によって結び付けられている (左辺は CONCEPT の要素，右辺の数字は PATTERN 中に出現する要素の番号)．この PROPERTIES の記述によって，各構文要素に意味記述が適切に配分されている．これにより，意味記述の欠如によって生じる第一の停止性問題は回避される．また，構文パターンが一度に与えられるということは，左再帰規則が必要でないことになり，第二の停止性問題も起こらない．

しかし，この方法では，文法規則として一般化されるべき文法の性質を，それぞれの辞書項目に与えていることになる．これでは，一般性の高い言語現象，例えば前置詞の目的語がギャップになる場合などの記述が困難になる．

一方，CRITTER[5] では，生成用に右辺の要素の派生順序を記述する方法を採用した．規則(1)に対応する CRITTER での記述は次のようになる．

```
s(S)  ->
    np(NP),                  #(3),
    vp(VP),                  #(2),
    {combine1 (NP, VP, S)}, #(1).
```

右辺の各要素の右に付けられた番号が生成時に評価される順序を示す．combine1 の項は規則(1)では素性構造の指定部分に対応する．この素性を右辺要素に伝播させる部分を，#(1) によって最初に行い，その後 VP, NP の順に派生するよう指定していることになる．

この生成専用情報を含んだ文法規則から，文法のコンパイル時に生成用文法と解析用文法の 2 つの文法が得られる (double compilation)．この生成用の適用順

序を適切に与えることにより，意味記述の欠如による第一の停止性問題も，左再帰規則による第二の停止性問題も回避することができる*.

しかし，この方法では，文法記述者に生成の適用順序を常に意識しておくことを強いることになる.

3.2 派生を行うノードの選択

2.1 節で示した最初の停止性問題は，意味記述のないノードを派生させた場合の問題であった．これは，句構造中のどのノードから派生を行うかという問題である.

ここでは，この問題を，派生可能なノード(句)が満たすべき制約条件を与えることによって解決する．この条件は，そのノードに意味素性（ここでは，〈synsem local content〉で示されるパスで指示される素性）が与えられていることである．ノードは，意味素性が与えられるまで，派生が行われず待ち状態になる．規則(2)の NP は，VP の派生が主動詞まで達して，その動詞の辞書記述が subj 素性を与えることによって初めて意味素性が与えられ，派生可能となる．ここで示した条件は，Wedekind の "connected" と同様のものとみなすことができる[18].

3.3 適用する規則の選択

2.1 節で示した左再帰規則による停止性問題も，2.2 節の適切な規則の適用と同じ問題に帰着できる．前者の場合は，先に示した2つの規則(2)，(3)のうちから，その時点での適切な規則を，どうやって選択するかという問題になる.

```
VP =HC*=>  (VP XP)          (2)
    ...
VP == (V)                   (3)
    ...
```

この選択は，素性構造から得られる情報に基づいて行わねばならない．このときに用いることのできる情報としては，その時点での SUBCAT リストの長さと，意味素性中の relation (reln 素性で与えられる)に対して与えられるべき argument の数ということになる．この argument の数はそれぞれの reln 素性の値によって決定される.

次に，2.2節で示した UNDETNOM に対する2つの修飾規則の選択に利用できるのは，規則によって追加される restriction の reln 素性であろう．reln 素性の値がタイプ分けしてあれば，そのタイプを規則適用の制約条件に用いることができる.

いずれの場合にも reln 素性の値によって適用される規則が決定される．reln 素性が適切にタイプ分けされていれば，そのタイプにより規則を選択することができる．次章では，タイプ付き素性構造を規則選択に利用する方法を考察する.

4. タイプ付き素性構造の導入

素性の値にタイプを与えることは，通常の素性構造では，特別な解釈機構を導入しなければ達成することができない．しかし，Ait-Kaci によるタイプ付き素性構造[1]を導入し，素性をタイプ階層で表現すれば，特別な解釈機構なしに統一的に扱うことができる.

このタイプは，単純な分類でなく，タイプ階層（束）を形成する．図3にタイプ階層の一部を示す．タイプ階層において，上位タイプ (supertype) は下位タイプ (subtype) の和 (OR 結合) を表し，下位タイプは上位タイプの積 (AND 結合) を表す．タイプ階層の積極的な利用を 4.2 節および 4.3 節で考察する.

4.1 文法規則の選択

タイプ付き素性構造を用いた制約条件を付加して，規則(4)，(5)は次のように記述することができる.

```
UNDETNOM =CH=>  (A UNDETNOM)
    !schema-6 !sem-fp-2                    (4')
    (<!m !content restr first reln>
     == [a])
UNDETNOM =HC*=>
            (UNDETNOM SREL)                (5')
    !schema-6 !sem-fp-2
```

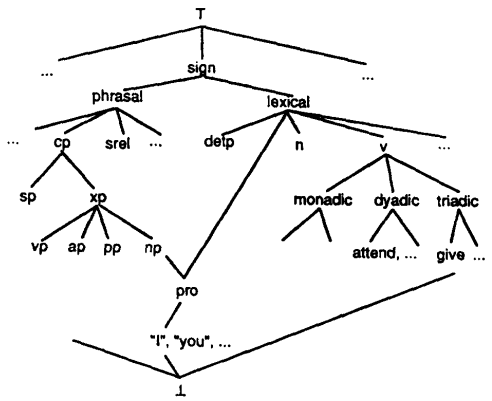

図 3 タイプ階層の一部
Fig. 3 Part of the type hierarchy.

* CRITTER では規則(2)に対応する左再帰規則は用いられていない．左再帰規則の例として，名詞句に対して前置詞句を付加する規則が出されている．この規則は，ここでの規則(5)の SREL を PP に交換したものと考えてよい.

(<!m !content restr first reln>
== [*v*])

ここで規則(4')の下線部は，<!m !content restr first reln>で示される素性(restrの第一要素のreln素性)がタイプ*a*と単一化されることを示す．この素性の値がタイプ階層の中でタイプ*a*の下にある場合に，この単一化が成功し，この文法規則は適用可能である．

タイプ付き素性構造の単一化を導入することで，このような制約条件を，統一的に，かつ宣言的に記述することができる．

4.2 カテゴリとの関連づけ

タイプ階層の利用方法のひとつに，カテゴリ(ノンターミナルシンボル)との関連づけがある．カテゴリはタイプを用いて表現され，階層化がなされている．例えば，規則(2)で現れる下位範疇化フレームに入る要素XPは，図3に示したようにNP, VP, PP, APを一般化したものとなっている．次に示す規則群により，これらの要素は，規則(2)中でXPとして扱われることになる．

XP == (VP) (6)
XP == (NP)
XP == (PP)
XP == (SP)

実際に下位範疇化フレームに入る要素は，個々の動詞で規定される．"will"の辞書項目中では，下位範疇化フレームの要素が1個だけで，そこには動詞句が入るということが，次のように記述されている．

(deflex "will" dyadic
(<!subcat first !cat> == [*VP*])
(<!subcat rest> == [*list-end*]))
...)

"will"が規則(2)と単一化されるときに，規則中のXPは，"will"のSUBCATフレームの記述により，タイプ*VP*と単一化され，VPとなる．その後の派生はVPからなされる(**図4**)．

このような未指定カテゴリを扱うものとして，解析における例ではあるが，D-PATR[9]がある．D-PATRでは，ダミーカテゴリとして，特別なシンボルX, Yなどを CFG に導入し，CFG 規則中でこのようなシンボルにあたったときに，このシンボルではなくcat素性で与えられる値を用いるようにしている．未指定カテゴリをタイプ階層の中に組み入れることにより，このような特別な機構を用いなくとも，無理なく対処することができる．

4.3 停止性問題への対処

規則(2)の停止性の問題は，SUBCAT リストの長さと，意味素性中の reln 素性で制約を加えることによって解決できる．動詞タイプを，それがとるargumentの数に応じてさらに3つのサブタイプ(Monadic, Dyadic, Triadic)に分類し，次の制約を規則(2)に加える．

(:or ((<!head-dtr !subcat rest>
== [*list-end*])
(<!content reln>
== (:or [*dyadic*] [*triadic*])))
((<!head-dtr !subcat rest rest>
== [*list-end*])
(<!content reln> == [*triadic*])))

これの制約により，*dyadic* タイプの動詞には規則(2)は1回，*triadic* タイプの動詞には2回適用されることになる．タイプが階層化されていることにより，このような制約を記述することが可能になった．

ここでは，タイプ付き素性構造を導入することにより適切な文法規則を選択する方法を示した．ここでは，意味素性(semantics)を制約条件に利用する例だけを挙げたが，意味論だけでなく，統語論(syntax)を表す素性，運用論(pragmatics)を表す素性なども同様に制約条件に用いることができる．また，これらを組み合わせて制約条件を記述することができる．文法

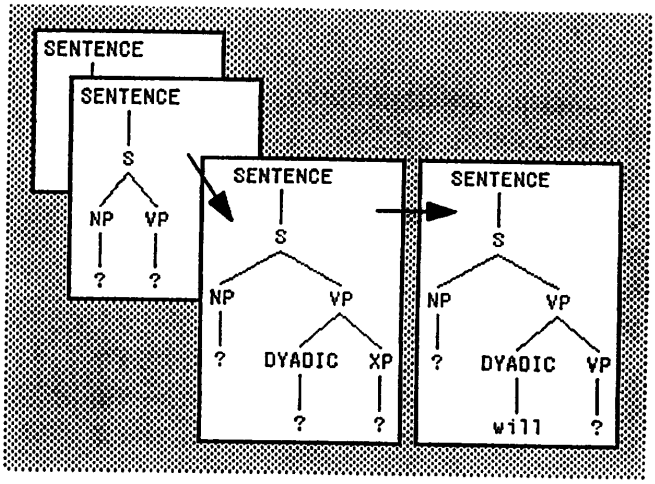

図4 動詞のSUBCATフレームによるComplement要素の特殊化
Fig. 4 Complement specialized by the SUBCAT frame of a verb.

と生成例の章で，意味論以外の素性を用いた例について説明する．

5. 選言を含む素性構造の導入

派生を行う際に適用可能な規則が複数あると，それぞれを適用してみることになる．もしそれらの規則の差異が小さいならば，規則適用の際に行われる単一化のほとんどの部分は，規則ごとに繰り返されていることになる．さらに複数の規則が適用に成功した場合には，派生木が複数できることになり，その後の規則適用は，これらの派生木すべてに対して行わねばならない．これは処理効率を大きく落とす原因となる．

このように規則の大部分が同じで一部だけが異なっているものの代表例として，ターミナルへの書き換え規則，すなわち，辞書記述がある．動詞の辞書記述では，意味や下位範疇化フレームが同一で，時制や主語の数・人称との一致による表層形の違いだけが異なることになる．通常は，動詞に達する際は時制は決定していることが多く，また主語もその動詞の下位範疇化フレームによって決定されるので，表層形はひとつに絞られることになる．この場合，共通部分を何度も単一化することだけが無駄になることになる．

一方，主語の意味素性の数，人数が，動詞が決定されても与えられず，主語がターミナルへ達してはじめて決定される場合がある．複数の派生木のそれぞれに規則を適用しなければならなくなる場合である．この文法においては，主語として次のような話者を表現するラベル表記で指定される場合にこのようなことが起こり得る．

[label *speaker*]

話者は通常 “I”（第一人称，単数の代名詞）で表現されるが，電話会話においては，“this is the conference office” のように “this” を生成しなければならない場合がある．主語が “I” と “this” のいずれかに決定されるまで，主動詞が決定できないことになる．

このような規則の小さな違いをひとつの規則にまとめ，単一化の無駄を少なくするために，素性構造中に選言を導入した．動詞の表層形の違いは，lexical unit と呼ぶ，まとめられた辞書項目に記載される．“be” 動詞の場合の例を示す．

```
(deflex-unit |be-Unit| dyadic
  (:or
    (!finite-form !present-tense
      (:or
        ((<word> == "am") !1sg-subj-agr)
        ((<word> == "are")
         (:or (!2sg-subj-agr)
              (!pl-subj-agr)))
        ((<word> == "is")
         !3sg-subj-agr)))
    ……)          ;過去形，過去分詞など
```

ここで選言は :or によって表現されている．また，アグリーメントは，テンプレートを用いて次のように表現されている．

```
(deffstemplate !3sg-subj-agr
  (<!subj-1 local content param index
       pers> == 3rd)
  (<!subj-1 local content param index
       num> == sing))
```

主語に対して人称 (pers)，数 (num) を含む意味素性が与えられると同時に，この3つの表層形の候補のうちから1つが決定される．多くの場合，主語の意味素性は動詞の subcat フレームにより与えられるので，動詞の lexical unit が選択された段階で，主語の意味素性が決定され，動詞の表層形が同時に決定されることになる．

主語が話者の場合は，主語が完全に決定するまで選言部分が残ることになる．これは，複数の派生木を保持し，規則をそれぞれに適用する場合に比較すると，大きく効率を向上させることになる．

選言を含む素性構造の単一化には Kasper の方法[10]を用いているが，このアルゴリズムに限らず，一般に選言を含む素性構造の単一化は処理時間のかかるプロセスである[4]．主語が話者の場合のように，素性構造中に選言を残したままにすることは，選言のない素性構造がひとつだけある場合と比較して，処理効率が悪い．このような場合には，Shieber らが採用している表層選択の遅延[17]を採用することも考えられる．

6. 文法と生成例

この生成システムは，音声言語日英翻訳システム SL-TRANS[13] 用に開発された単一化文法に基づく解析システム[11]に，3-5 章で述べた生成メカニズムを追加したものである．このシステムを用いれば，既述したテンプレート，マルチリンクタイプの導入により，HPSG のモジュラリティを損なわない文法記述ができる．

ここで用いる文法は，HPSG の新しい解析方法[15]

を基に作成した．また，Borsley による改良[3]を取り入れている．これは，主語の特殊性を適切に扱うため，主語を subcat 要素から独立させたものである．

さらに，この文法は，会話文で重要な働きをもつ発話意図（発語内的力）を扱うよう拡張されている．例えば，行為拘束型の発語内的力タイプ PROMISE は，助動詞 "will" の辞書項目中に次のように記述される．

```
(deflex-unit |will-Unit| dyadic
  (:or
   ((<!content reln> == [*promise*])
    (<!content obje>
        == <!subcat-1 local content>)
    (<!subcat-1 local content reln>
        == [*action*])  ;; 主動詞は行為型
    !1st-subj-agr)  ;; 主語は1人称（肯定文）
   …  ;;「未来」の意味の記述
```

発語内的力のうちのいくつかは，辞書項目ではなく，文法規則中に記載されている．例えば，発語内的力 REQUEST（行為指導型）は，命令文の規則に記述される．また，"Would you send me a registration form?" のように，疑問文の形式でも REQUEST を表現することができるため，疑問文規則中にも記述されている．

生成例を付録に示す．この素性構造 (fs-1) は発語内的力 REQUEST をもつもので，この生成結果として3つの文が生成されている．これらの文はいずれも同じ発語内的力を表現しているが，話者の意図が全く同じというわけではない．ここで異なっているのは，これらの文に表現される丁寧さの度合であろう．運用論的情報として丁寧さのレベルを示す素性（<prag params politeness> で示される）を導入し，対応する文法規則中でこれを扱うようにする．先の素性構造に丁寧さのレベルを最高に設定したもの (fs-1-1) を与えると，この中でもっとも丁寧な表現だけが得られる．

このように，双方向文法生成システムは過生成の傾向がある．過生成された結果の差異を検討することにより，文法の詳細化（特に話者の意図の違いに関して）に寄与することができる．

ここで示したように，このシステムでは意味論だけではなく，運用論などの情報も規則の選択のための制約条件に用いることができ，文法規則の記述における自由度が大きい．

7. 他の研究との関連

解析と生成で文法・辞書を共有する双方向システムとして，PHRED[8] と CRITTER[5] を挙げた．これらは，文法に制限があったり[8]，文法記述者が生成の順序を指定しなければならない[5]などの欠点をもっていることは述べた．

文法・辞書だけでなく，基本メカニズムも共有するものとして，Shieber の研究[16]や Emele らの生成の研究[6),7]がある．Shieber の生成メカニズムは，彼自身も述べている[17]とおり，適用できる文法に制限がある．この制限は，"semantic monotonicity" と呼ばれ，左辺の各々のノンターミナルの意味構造は，右辺の意味構造中に含まれていなければならないというものである．これは，イディオムが記述できないことを意味し，制限が大きすぎる．

われわれのシステムは，タイプ付き素性構造のタイプ階層と単一化という静的な側面だけを利用しているのに対して，Emele らは，解析生成の共通メカニズム[6),7]として，Ait-Kaci のタイプ書き換え機構が利用できることを示した．ただし，このシステムで想定している生成の入力は構文を強く意識したものとなっており，われわれが想定している機械翻訳の用途には適切でない．

Shieber らは先のメカニズム[16]の欠点を克服した生成メカニズム (semantic-head-driven generation) を発表している[17]．これは，トップダウン生成の停止性問題を解決することを目的としたものである．このシステムでは，意味構造から文法規則の semantic-head となるものの辞書項目を得，その後，semantic-head 以外の要素の派生を行う．早い時点で辞書情報が利用できることにより，SUBCAT 要素の数（およびその内容）が決定され，停止性問題は生じない．

彼らのシステムは，1) 論理式を入力表現としており，複数の種類の情報（運用論的情報など）を一貫性のある形式で表現しづらい，また，2) そのための文法の拡張も容易でない，3) 単一化に選言を含むことができないため，文法規則中のちょっとした条件の違いのためにも複数個の規則を用意せねばならず，文法の複雑さと処理効率の低下を招くなど，われわれの目的にあう文法を開発するのには不適切である．しかし，基本的なメカニズムは効率のよいものであるため，われわれの生成システムの特徴である複数の種類の情報を用いて生成の制御を行えることと組み合わせ

れば，高い効果を得られるものと考えられる．

8. おわりに

本稿では，双方向文法を用いる生成のプロセスを，宣言的な記述で制御する方法について述べた．Aït-Kaci によるタイプ付き素性構造[1]を用いれば，単純な素性の分類でなく，タイプ階層を有効に活用した記述が可能になる．また，タイプ階層を CFG のシンボルと結び付けた動的な利用も可能になった．

ここで示した文法は，まだ十分でない．今後，電話会話における種々の発話意図を扱うため，さらに拡張する必要がある．

また，この文法では，例に示したように表層発話と発語内的力の関係を直接記述している．これは，文法が複雑化するおそれがある．久米らは，発語内的力を表層に近いものと抽象的なものの2つのレベルに分け，抽象発語内的力からプランニングを通じて表層発語内的力を決定する方法を提案した[12]．この方法との結合も今後の課題である．

謝辞 有益な助言・議論をとおしてこの研究を助けてくださった ATR 自動翻訳電話研究所言語処理研究室の飯田主幹研究員（現在同研究室室長）をはじめとする研究員諸氏，および現在 NTT 基礎研究所の小暮主任研究員に感謝の意を表明します．また，論文執筆に当って適切なアドバイスをいただいた査読者の方々に感謝いたします．

参考文献

1) Aït-Kaci, H.: An Algebraic Semantics Approach to the Effective Resolution of Type Equations, *Theoretical Computer Science 45*, pp. 293-351, North-Holland (1986).
2) Appelt, D. E.: Bidirectional Grammars and the Design of Natural Language Generation Systems, *Theoretical Issues in Natural Language Processing-3*, pp. 185-191, Las Cruses (1987).
3) Borsley, R. D.: Subjects and Complements in HPSG, Report No. CSLI-87-107, CSLI (1987).
4) Carter, D.: Efficient Disjunctive Unification for Bottom-Up Parsing, *Coling-90*, Vol. 3, pp. 70-75, Helsinki (1990).
5) Dymetman, D. and Isabelle, P.: Reversible Logic Grammars for Machine Translation, *2nd International Conference on Theoretical and Methodological Issues in Machine Translation*, Pittsburgh (1988).
6) Emele, M. C.: A Typed Feature Structure Unification-based Approach to Generation, 電子情報通信学会言語処理とコミュニケーション研究会, NLC-88-32 (1988).
7) Emele, M. C. and Zajac, R.: Typed Unification Grammars, *Coling-90*, Vol. 3, pp. 293-298, Helsinki (1990).
8) Jacobs, P. S.: PHRED: A Generator for Natural Language Interfaces, *Computational Linguistics*, Vol. 11, No. 4, pp. 219-242, MIT Press (1985).
9) Karttunen, L.: D-PATR: A Development Environment for Unification-Based Grammars, Report No. CSLI-86-61, CSLI (1986).
10) Kasper, R. T.: A Unification Method for Disjunctive Feature Descriptions, *25th Annual Meeting of the Association for Computational Linguistics*, pp. 235-242, Stanford (1987).
11) Kogure, K.: A Method of Analyzing Japanese Speech Act Types, *2nd International Conference on Theoretical and Methodological Issues in Machine Translation*, Pittsburgh (1988).
12) Kume, M., Sato, G. K. and Yoshimoto, K.: A Descriptive Framework for Translating Speaker's Meaning, *4th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 264-271, Manchester (1989).
13) 森本ほか：音声言語日英翻訳実験システム (SL-TRANS) の概要，第 39 回情報処理学会全国大会論文集，4 G-4 (1989).
14) Pollard, C. and Sag, I. A.: *An Information-Based Syntax and Semantics, Volume 1, Fundamentals, CSLI Lecture Notes Number 13*, CSLI (1987).
15) Pollard, C. and Sag, I. A.: *An Information-Based Syntax and Semantics, Volume 2, Topics in Binding and Control, CSLI Lecture Notes*, CSLI (to appear).
16) Shieber, S. M.: A Uniform Architecture for Parsing and Generation, *Coling-88*, pp. 614-619, Budapest (1988).
17) Shieber, S. M., van Noord, G., Moore, R. C. and Pereira, F. C. N.: Semantic-Head-Driven Generation, *Computational Linguistics*, Vol. 16, No. 1, pp. 30-42, MIT Press (1990).
18) Wedekind, J.: Generation as Structure Driven Derivation, *Coling-88*, pp. 732-737, Budapest (1988).

```
> (pprint-fs fs-1)        ;;「会議を参加することを要請する」意味記述

[SENTENCE[SYNSEM [[LOCAL [LOCAL[CONTENT [CIRC[RELN [*REQUEST*]]
                                          [AGEN ?X07[[LABEL  *SPEAKER*]]]
                                          [RECP ?X06[[LABEL  *HEARER*]]]
                                          [OBJE [CIRC[RELN [*ATTEND-2*]]
                                                     [AGEN ?X06]
                                                     [SLOC #|*|#

[REF-OBJ[PARAM ?X04[NPRO[INDEX [INDEX[PERS  3RD]       ;;"the conference"の意味記述
                                      [NUM [SING]]]]
                        [DET [THE]]
                        [HUMAN  -]]]
        [RESTR (:DLIST        [[RELN [*CONFERENCE-1*]]
                               [INST ?X04]]
                      |?X05| )]]
                                                              #|*|#]]]]]]]]]]
         [PRAG [[HEARER ?X06]
                [SPEAKER ?X07]]]]
7

> (gen3 fs-1)
("could you attend the conference" "would you attend the conference" "attend the
conference")
            ;; fs-1からは3つの生成結果を得た。

>(setq fs-1-1  (unify-fs fs-1 (make-fs-from-exprs '((<prag params politeness> ==
4)))))
SENTENCE[(SYNSEM PRAG)]
            ;; fs-1にpolitenessを最高レベルに設定したものをfs-1-1とした。

> (gen3 fs-1-1)
("could you attend the conference")
            ;; fs-1-1からの生成結果は一つに絞られた。
```

付録 生成結果
Appendix Generation result.

(平成3年3月29日受付)
(平成3年11月5日採録)

上田 良寛(正会員)

1957年生. 1980年京都大学工学部電気工学第二学科卒業. 1982年同大学院修士課程修了. 同年富士ゼロックス(株)に入社し, 自然言語処理等の研究に従事. 1988年から1991年まで(株)ATR自動翻訳電話研究所に出向し, 対話文自動翻訳における英文生成の研究を担当した. ACL会員.