生成文法における文法知識とは

上山, あゆみ
九州大学大学院人文科学研究院文学部門言語学 : 助教授 : 生成文法, 日本語統語論

https://doi.org/10.15017/3636

出版情報：文學研究. 104, pp.79-100, 2007-03-01. 九州大学大学院人文科学研究院
バージョン：
権利関係：

生成文法における文法知識とは*

上　山　あゆみ

1．はじめに

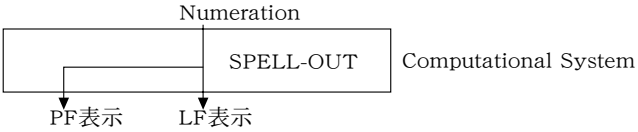
一般に、「○○語ができる」という場合、それはその言語の文法知識が習得されていて運用可能な状態であることを意味している。しかし、この場合の「文法知識」とは、具体的にどういう知識を指すと考えるべきだろうか。本小論の目的は、現代の生成文法の観点からみた場合に「文法知識」というものがどのように位置づけられるのか、関連する問題点を整理することである。

2．生成文法の観点から見た言語運用のモデル

2．1．Computational System

生成文法の中心にあるのは、Computational Systemと呼ばれる計算アルゴリズムである。これは、いくつかの単語の集合(Numeration)を入力とし、それらをつなげて1つの構成素にしたり(Merge)、語順を変えたり(Move)、組み合わせ可能かどうかをチェックしたり(Agree)、などの操作を行ったのちに、入力された単語がすべて組み込まれた構築物としての表示を2つ(PF表示とLF表示)出力するアルゴリズムである。このPF表示とLF表示は独立に作られるのではなく、Numerationから出発した作業の前半は共通で、SPELL-OUTと呼ばれるタイミングのあとで2つに分岐すると考えられている。

(1)



チョムスキーは、この計算システム（と同等の働きをする仕組み）が実際に私たちの脳の中で作動していると考えていることがうかがえる（cf. Chomsky（2000））。しかし、このシステムが言語運用の中でどのように働いているのかということについては、必ずしも具体的なモデルが提案されているわけではない。

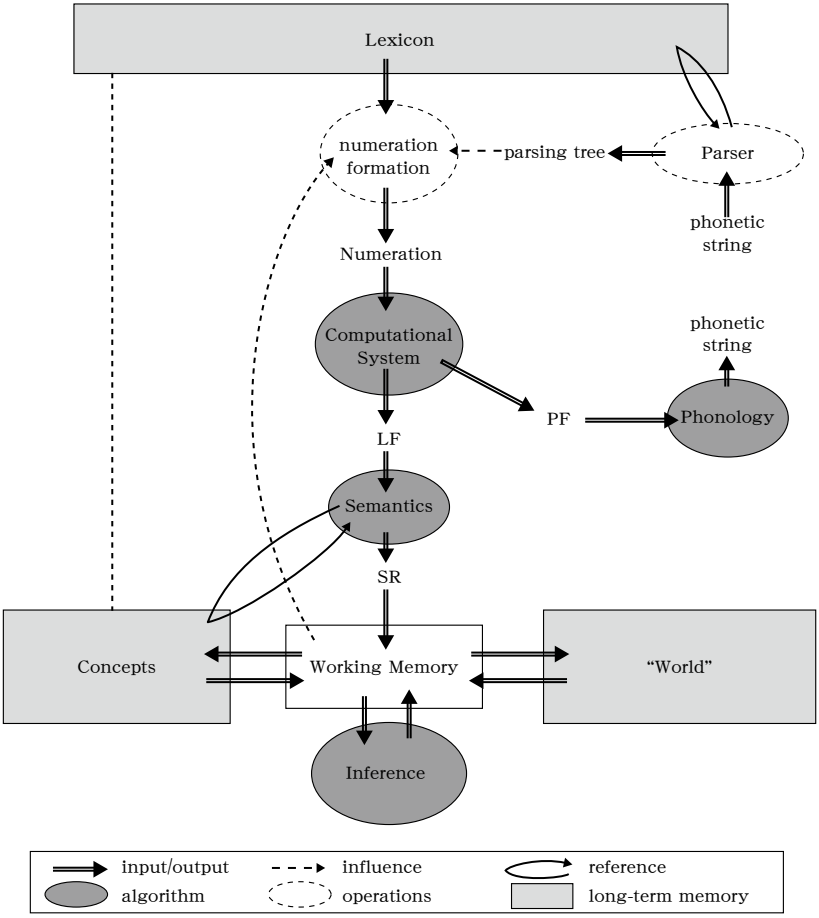
もちろん、言語運用全体を視野に入れると、アルゴリズムとしてモデル化することが困難な部分がたくさんある。だからこそ、チョムスキーは、説明対象を（言語運用全般ではなく）この計算システムに関わる部分に限ることによって、「科学としての言語学」という生成文法のプロジェクトを打ち立てた。しかし、実際の研究活動においてデータとなるのは、研究者自身の内省であれ、他人の反応の観察であれ、どちらにしても、それは言語運用の結果である。これがパラドキシカルな状況を作り出している。この問題は、完全に取り除くことは困難かもしれないが、現場の研究者としては、いろいろな実践的方策を工夫し、問題を軽減していかなければならない。

そのためには、まず、特に研究のデータとなる内省について言語運用の全体的なモデルを提示し、計算システムがどこにどのように関わっているのかということを明示しながら分析を進めていくことが必要である。言語現象の中には、明らかに計算システムの外の仕組みが大きく関わっていると思われる側面もある。しかし、だからといって、そういう現象が計算システムの追究に無関係であるということにはならない。内省データが言語運用の結果であり、「計算システムのみが関わっているデータ」というものがありえない以上、計算システムの中で起こっていることに、その外の仕組みも影響を与

えていることは当然である。

次の図１がここで想定している言語運用のモデルである。

図１



この言語運用のモデルには、(1) の計算システムが直接組み込まれている。生成文法の言語観は、「言いたい意味」が「文という音」に変換されたり、「文と

いう音」から「その文の意味」が伝わる、というような従来の言語観とは根本から異なっている。「文という音」のベースとなるPF表示と「文の意味」のベースとなるLF表示は、どちらも計算システムの出力であり、一方が他方の入力になっているわけではない。図１のモデルのポイントを簡単に説明しておく。

２．２．encoding専用のアルゴリズム

自発的な文の生成の場合には、何らかの方法でLexiconの中の語彙からNumerationが形成されると考えておく[1]。それに対して、例文判断のように、提示された音連鎖（phonetic strings）がある場合には、その音連鎖に単語認識や統語解析などの作業を行い、その結果がnumeration formationに影響を与えていると考えたい。ここでは便宜的に、その一連の操作の総体に対してParserという名称を与え、その出力の結果をまとめたものをparsing treeと呼んでおく。

言語の研究においては、文を聞いたり読んだりする場合、Numerationの形成を仮定しないモデルが想定されている場合も多い。つまり、統語解析器であるParserによって意味解釈が可能になる、という考え方である[2]。この考え方をつきつめると、文の生成に関わるメカニズムと、文の理解に関わるメカニズムは、まったく別のものであるということになる。確かに、文の生成と文の理解というのは、かなり異なった行為であるには違いないが、しかし、そこに関わる「文法の知識」は共通であると考えたい。しばしば、「Parserは文法を参照する」という仮定が言及される場合もあるが[3]、文法の主体が規則の集積であった標準理論（Standard Theory）の場合ならばともかく、Numerationを入力としてLFとPFを出力するという動的なアルゴリズムとしての計算システムを仮定する以上、Parserがそれを「参照する」ことは不可能であろう。つまり、文法の中核をアルゴリズムとしてとらえる限り、この計算システムそのものが、文の生成にも文の理解にも関わっていると仮定

せざるをえない。

parsing treeについては、今後、より具体的に論じていく必要があるが、現時点では、認識された各単語がどのような順番に並んでいるかということと、解析の結果得られた構造に関する情報とが表されているものを想定している。Numerationにおいては、音として具現される単語だけでなく、少なからぬ数の機能範疇や抽象的な素性が必要となる。これらがnumeration formationにおいて単に恣意的に選択されるのならば、聞いたとおりの文が出力されるようなNumerationを形成するのは不可能に近いかもしれない。しかし、parsing treeにおいて、どの単語がどこに係っているかという関係等が示されれば、Numerationにどのような要素が必要かということを、かなりの精度でしぼりこむことができるはずである。

parsing treeが計算システムの出力としての表示と等しいものである必要はまったくない。LF表示やPF表示を出力するのは計算システムの役割なのであるから、統語解析においては、Numerationを効率よく形成する役割が果たせれば十分である。そもそも、numeration formationというものは、自発的な文の生成の場合、Parserの力を借りることなく作業が完遂できるモジュールなのであるから、文を聞いたり読んだりした場合であっても、音連鎖から得られる情報に100%依拠していると考える必要はない。Parserの出力に基づいてもNumerationが一意に決定されないという状態は、むしろ普通であろう。足りない情報は、numeration formationが自由に補って、Numerationを形成するのである。

いったん、Numerationが計算システムに入力されれば、そのアルゴリズムに従ってPF表示とLF表示が得られる。そこで、出力されたPF表示から派生される音連鎖と、聞いたり読んだりした音連鎖の違いが気にならなければ、そのLF表示に基づいた解釈がその文の「意味」として知覚されると考えたい。つまり、私たちは、耳で聞いた文と同じ（だと思っている）文を作ることによって、元の文の内容をくみ取ろうとしているという仮説である。

伝えたい内容を「音」という形で運ぶということは、一種の暗号化と考えてもよい。この表現を用いて上の考え方を言い換えるならば、私たちの頭の中にある言語のシステムは、暗号化（encoding）はできるが、直接その暗号を解読（decoding）することはできず、同じような暗号を自分で作ってみることによって、発話者の意図を推測する方式になっていることになる。

2．3．意味解釈

LF表示とは、（PF表示と同様に）単語が構造化された表示である。図1では、このLF表示がSemanticsというモジュールの入力となっている。ここで想定しているSemanticsというアルゴリズムの働きは、次の通りである。

（2）Semanticsの働き：

i. LF表示の構造はそのままに、それぞれの単語を対応する概念に置き換える[4]。
ii. 姉妹関係になっている概念を合成する[5]。
iii. それらの概念の組み合わさった構築物が意味表示（SR）であり、これがいわゆる命題に相当する。

たいていの単語には、それに該当する概念がある。それらの概念は、図1の「Concepts」の中に蓄えられているとする。仮に、すべての概念に通し番号がついていると仮定しよう。しばしば、単語というものは、音韻素性と形式素性と意味素性の束である、という言い方がされるが、その通し番号がその場合の意味素性に相当すると考えておきたい。SRを形成するのはSemanticsの働きであるが、その構造はLF表示に基づいているので、SRの構造は間接的に計算システムが形成することになる。SRというものは概念からできた構築物であり、全体で、命題（もしくはある種の情報）を表す[6]。

これが、作業記憶領域（Working Memory）に入って、その人が長期記憶として蓄えている情報と合わせて「理解」されたのちに、長期記憶へと運ばれていく。

この「理解」という作業の中には、たとえば次の（3i,ii）などが含まれる[7]。

(3) i. その新しい情報を組み込むことによって、既存の情報を変更する必要があるかどうかを確認する。
ii. 新しい情報と既存の情報との関連性をなるべく打ち立てる。

このような作業が可能になるためには、新しく入ってきた命題に関連する命題が長期記憶から作業記憶の中に呼び起こされなければならない。しかし、何から何を「連想」するか、ということに対するアルゴリズムが立てられるとは考えにくい。この作業をつかさどっているのは、神経ネットワークの複雑な所業であると考えざるをえないであろう。ただし、予測不可能なのは、どういう命題が呼び起こされるかというところだけである。仮に、命題Pと命題Qが呼び起こされた状態を想定すれば、例えばその２つが矛盾の関係にあるかどうか、といったことに関する私たちの判断は、予測可能な現象である。そこで、そのような命題間の計算は、アルゴリズムにしたがって明示的に行われていると考えたい。図１のInferenceというモジュールは、いわゆる論理学的な推論規則が蓄えられている部署であり、ここで、矛盾のチェックや推論などが行われると仮定している[8]。

さて、図１では、長期記憶として、「Lexicon」「Concepts」「"World"」の３つの領域が描かれている。これらは、それぞれ要素が異なるデータベースである。

(4) Lexicon ... （厳密に言語としての）単語とその特性に関するデータベース。

(5) Concepts ... 命題を形成する部品となる概念のデータベース。ほとんどの場合、Lexicon中の単語と対応関係がある。
(6) "World" ... (その人が認識／想定している) 外の世界の (特定の) 出来事／状態を、命題の集積として表現しているデータベース[9]。

そのSRが外界で起きた出来事／状態を述べたものであるならば、その情報は ((3) の過程を経た後) 直接、"World" に登録される。しかし、私たちが文で表現することは、必ずしも外界における事象の記述だけとは限らない。"World" というものを (その人が認識／想定している) 外界の総体と位置づける以上、特定の出来事／状態を表現しない命題は、"World" には登録されない。その場合、その命題においてそれらの概念が組み合わされたということが、単にConceptsというデータベースのネットワークに吸収されるだけであると考えている[10]。Conceptsの中の概念どうしの「連想」関係の強度に影響を与えるが、"World" に書き込まれる情報とは異なるということである[11]。

3．生成文法における「文法」という概念

以上、図1のモデルに即して、言語運用とComputational Systemとの関わり合いを駆け足で概観した。その上で、生成文法における「文法」の位置づけについて述べておく[12]。

初期の生成文法では、「すべての文法的な文を、そして、それだけを生成する仕組み」を「文法」と呼び、その仕組みの追究というものを生成文法研究の目標であると位置づけた。当時の枠組における (特定言語の)「文法」とは、句構造規則と変形規則の総体を指すと言っていいだろう。これは、いわば、その言語の「構文」を形づくるものであるから、一般的な意味での「文法知識」という概念とほぼ一致する。

言語によって、その「文法」は異なるわけであるから、もちろん、それら

の句構造規則や変形規則は生得的ではなく、経験によって習得されるものである。これに対して、生得的な知識は普遍文法と呼ばれ、これが「文法知識」の発達の初期状態に相当すると考えられた。普遍文法は、句構造規則の一般型や変形に対する一般的制約を含んだものであるが、実際の規則は習得しなければならないものであるため、普遍文法だけでアルゴリズムができているわけではなかった。

しかし、その後、この考え方を押し進めていこうとしている間に、いろいろと概念的な不具合が指摘されるようになり、結果的に、句構造規則というとらえ方が全廃され、(1) のようなComputational Systemが提案された。現在の生成文法において中核の働きをしているのは、このComputational Systemであるから、このシステムを指して「文法」と呼ぶのは自然の成り行きかもしれない。しかし、Computational Systemと一般的な意味での「文法」との間には、かなり大きな乖離がある。

(7)（初期の生成文法における）句構造規則＋変形規則
- a．経験によって習得しなければならないもの
- b．言語によって異なっている
- c．どのような表示が文法的であるかが、比較的直接述べられている。

(8)（現代の生成文法における）Computational System
- a．生得的なもの
- b．言語普遍的
- c．それ自体は単なるアルゴリズムなので、結果的にどのような表示が出てくるのかは、実際に、そのアルゴリズムを動かしてみないとわからない。

そのかわりに、(7) のような働きをになうことになったのがLexiconであ

る。現代の生成文法では、いわゆる「構文的な知識」は、すべてLexiconの中の何らかの語彙の特性としてとらえていくことになる。

(9) いわゆる構文的知識は、すべて特定の語彙の特性としてみなすことにする。(＝その語彙の習得というものが、その語彙が形成する構文の習得に相当することになる。)

文という単位の主要部として、INFLやCOMPなど、必ずしも音として特定しにくい要素を仮定して話を進めるのも、このような全体の位置づけの変化に呼応するものである。

そこで問題になってしかるべきなのは、(9) が常に可能なのかということであろう。そこで、最も問題が大きそうなケースを例にとり、その分析を試みることによって、生成文法の研究の進め方の特徴を示したい。

4．ケーススタディ

4．1．reduplicationとプロソディ

ある構文に注目した場合、その構文に特徴的な語彙が出現するならば、もしくは、その構文の特徴が、その「中心的」な語彙の周りに集約されるならば、比較的、(9) の方針に沿った分析は考えやすい。これに対して、中心的な語彙というものが見当たらない場合には、(9) の方針の実現が難しいように見える。そこで、以下では、1つの語彙を2回繰り返すという形式 (reduplication) に注目してみたい。

同じ語彙を2回繰り返した場合、それらをどのようなプロソディで発音するかで、文全体の意味するところが大きく異なる。たとえば、(10) の場合、大きく分けて3種類のプロソディで読むことができる。

(10) 「受けた、受けた」

１つは、１回目を控えめに、２回目を少し高めに読む読み方である。この場合、文全体としては「良かった、良かった」というニュアンスを伝えることになる。これは、物語の読み聞かせなどの最後によく出てくる「めでたし、めでたし」という読み方とほぼ同じプロソディなので、以下では、**「めでたし」パターン**と呼ぶことにする。

また、それとは逆に、１回目を大げさに、２回目はそれを受けるように読む読み方も可能である。この場合には、ある種の感嘆文的な意味を持ち、「どれだけ受けたことか！」というニュアンスを伝える。以下では、これを**「滝つぼ」パターン**と呼ぶことにする。

また、上の２つと比べて、１回目と２回目をあまり音調を変えずに読むこともできる。この場合は、**無変化パターン**と呼んでおくことにする。

この３つのプロソディのパターンが常に可能であるわけではない。たとえば、(11) の場合には、「めでたし」パターンと無変化パターンは可能であるが、「滝つぼ」パターンは容認できない。

(11) 「なるほど、なるほど」

これは明らかに、「なるほど」という言葉に関して感嘆文的な意味合いを加えることが不可能であるからであろう。

以下、これら３つのプロソディにおけるreduplicationについて、現在わかっている範囲で少し記述をしておく。

４．２．「めでたし」パターン

「めでたし」パターンは、基本的に、確定した事象に対してプラスの評価を与える意味を持っている[13]。

(12) 「めでたし」パターン

ａ．めでたし、めでたし
ｂ．よかった、よかった
ｃ．感心、感心
ｄ．満足、満足
ｅ．満腹、満腹
ｆ．おみごと、おみごと
ｇ．完璧、完璧
ｈ．おたから、おたから
ｉ．ごりっぱ、ごりっぱ
ｊ．楽しみ、楽しみ

褒め言葉とは言えない表現の場合でも、このプロソディで発音すると、何らかのプラス評価のニュアンスを与えることが可能なことが多い。

(13)　ａ．はじまり、はじまり
　　ｂ．もつ鍋、もつ鍋
　　ｃ．残念、残念
　　ｄ．しゃーない、しゃーない[14]
　　ｅ．かまへん、かまへん[15]

そのため、どういう意味でプラス評価なのかがわかりにくい場合には、不自然に感じる。

(14)　ａ．*最悪、最悪
　　ｂ．#がいこつ、がいこつ

このプロソディは、定延 (2005) でいうところの「低い山、高い山」と関連

している可能性はあるが、「めでたし」パターンには、原則的に４モーラ＋４モーラのリズムを守ろうとする強い傾向があるという点で、一般的な「低い山、高い山」パターンとは一線を画している。

(15)　a．おしまい、おしまい
　　b．?おわり、おわり

(16)　a．たまげた、たまげた
　　b．*?びっくりした、びっくりした

４モーラでなくても、タイミングを合わせている例としては（17）や（18）があげられるが、それぞれ、４モーラ分の時間の長さが意識されていることがよくわかる。

(17)　しかたない、しかたない

(18)　a．サンキュ、サンキュ
　　b．えらい、えらい
　　c．上手、上手
　　d．たべた、たべた

また、単語が３回以上繰り返されている場合には、「めでたし」パターンは使えない。

(19)　「めでたし」パターンの場合：
　　a．*めでたし、めでたし、めでたし
　　b．*良い、良い、良い、良い

c．*ダメ、ダメ、ダメ、ダメ

4．3．「滝つぼ」パターン

「滝つぼ」パターンでは、1回目に、その単語のアクセントが非常に強調された形で発音され、2回目は大きく減衰する。したがって、平板なアクセントの表現の場合には、ピッチそのものは下がらないこともある[16]。

(20) a．[LHL] 受けた、受けた （＝どれだけ、受けたか！）
b．[HLL] たっかい、たかい （＝どれだけ、高いか！）
c．[HHHH] たいくつ、たいくつ （＝どれだけ、退屈か！）

このプロソディで繰り返しが起きた場合には、いわゆる感嘆文のような意味になる。

(21) a．おっきぃ、おっきい （＝どれだけ、大きいか！）
b．まっずい、まずい （＝どれだけ、まずいか！）
c．かっこええ、かっこええ （＝どれだけ、かっこいいか！）
d．静か、静か （＝どれだけ、静かか！）
e．器用、器用 （＝どれだけ、器用か！）

したがって、「どれだけ、〇〇か！」の形の感嘆文になれない表現はあらわれない。

(22) a．*なるほど、なるほど （*どれだけ、なるほどか！）
b．*12時、12時 （*どれだけ、12時か！）

逆に、形容詞でなくても、「どれだけ、〇〇か！」の形の感嘆文になれる表

現ならば許される。

(23) a．走った、走った （＝どれだけ、走ったことか！）
b．わろた、わろた （＝どれだけ、笑ったことか！）
c．(#)日本人、日本人 （＝どれだけ、日本人的か！）

このパターンでも、４モーラが基本になっている印象があるが、「めでたし」パターンよりも制限は緩い。

(24) a．食う、食う （＝どれだけ、食うことか！）
b．びっくりした、びっくりした （＝どれだけ、びっくりしたことか！）

しかし、３回以上の繰り返しは許されない。

(25) a．*食う、食う、食う
b．*たっかい、たかい、たかい

４．４．無変化パターン

無変化パターンの機能は多岐に及んでいるようにも見えるが、繰り返しが起こらなかった場合との意味の違いはほとんどないので、reduplication自体には意味的な機能は無いと考えることもできるだろう[17]。

(26) 注意の喚起：
a．信号、信号
b．ネコ、ネコ
c．きはった、きはった[18]

d．雨や、雨や
e．たおれる、たおれる
f．たいへん、たいへん
g．うしろ、うしろ
h．あぶない、あぶない

(27)　相手の動作を促す：
a．いこいこ
b．たべよたべよ
c．たべにいこ、たべにいこ
d．食べたい、食べたい
e．どしたん、どしたん
f．きいてーな、きいてーな

(28)　相手をなだめる返事として（通常、早口で切れ目なく）：
a．はい、はい
b．たいへん、たいへん
c．もちろん、もちろん

(29)　拒絶（通常、早口で切れ目なく）
a．ダメ、ダメ
b．いらん、いらん
c．知らん、知らん
d．あかん、あかん

(30)　ツッコミ（必ずしも早口でなくてよい）：
a．ない、ない

b．でけへん、でけへん
c．あらへん、あらへん
d．ちゃう、ちゃう
e．知らん、知らん

また、基本的に無変化パターンと思われる表現でも、あとのほうに向かって、ピッチや音量が大きくなるプロソディになることはある。

(31) 無変化パターン＋「クレッシェンド」効果：
a．ダメダメッ！
b．ダメダメダメッ！
c．ダメダメダメダメッ！
d．キタキタキタキタキターッ！
e．ほしいほしいほしいほしい！

これは、定延（2005）の「低い山、高い山」効果と同一視していいのではないかと考えている[19]。

(32) a．（イヤイヤ認めさせられているときの）
きれい、きれい、きれい（定延 2005: 90）
b．（面倒がっているときの）
行きます、行きます(定延 2005: 90)
c．うまくいきませんねえ （定延 2005: 91）
d．いやあーこの酒はうまいですなあー （定延 2005: 168）

場合によって、「めでたし」パターンと区別しにくいこともあるが、このパターンの場合は、(31) に示されているように、モーラ数や繰り返しの数に

特に制限がないのが特徴である。

4．5．考察

reduplicationのように、具体的な単語ではなく「繰り返している」という事実が機能を持つ場合には、そのような操作を指示する「語彙」を仮定せざるを得ない。

(33) reduplication
何らかの単語に適用して、その単語を繰り返すという操作を指示する抽象的語彙が存在している結果、引き起こされる現象

語彙というものは、原則的に、音韻素性・形式素性・意味素性の束と考えられているので、reduplicationのプロソディと意味が少なくとも3種類区別できるならば、3種類のreduplicating itemが存在するということになる。

(34) 「めでたし」パターンをうみだす抽象的語彙：
音韻的：「4モーラ（小）＋4モーラ（大）」を基本とし、項となる句（＝実際に繰り返される表現）のアクセントパターンに基づいて、全体のプロソディが決定される。
意味的：確定した事象に対して話者がプラス評価を持っていることを表すモダリティの一種

(35) 「滝つぼ」パターンをうみだす抽象的語彙：
音韻的：「4モーラ（大）＋4モーラ（小）」を基本とし、項となる句（＝実際に繰り返される表現）のアクセントパターンに基づいて、全体のプロソディが決定される。
意味的：話者が、項となる句が示す度合いが著しく大きいととらえて

いることを表すモダリティの一種

(36)　無変化パターンをうみだす抽象的語彙：
　　音韻的：項となる句を2回もしくはそれ以上、繰り返す
　　意味的：なし

これらの3つの「語彙」がLexiconの中に含まれていると想定すれば、上の現象は記述できることになる。

5．終わりに

言語習得に関する議論は、生成文法の初期から盛んに行われているため、生成文法の得意分野の1つであるかのように見なされていることも多い。しかし、生成文法系の言語習得理論の多くは、標準理論からGB理論にかけての時期の生成文法理論に基礎を置いており、それらは必ずしもMinimalist Programの考え方に沿っているとは言いがたい。Minimalist Programにおける言語観に従うならば、言語ごとの（いわゆる）「文法知識」は、「文法」部分にあるというよりは、Lexiconに集約されていると考えなければならず、言語の習得の問題は、語彙の習得の問題として考え直さなければならない。

この考え方に従えば、それぞれの「構文」に関する知識というものは、すべて、何らかの語彙の特性としてとらえていくことになり、その結果、たとえば(34) − (36)のような語彙を仮定しなければならない場合も出てくる。この小論では、単に1つの分析案として、これを提示しただけであるが、このような語彙を想定することの是非を検証し、論じていく必要がある。

生成文法の言語観を押し進めていくためには、Computational Systemの存在証明が最も核となる課題であるが、それと同時に、様々な構文がどのように記述できるのか、Lexiconに関する研究も同様に重要な課題であるに違いない。

参考文献

Chomsky, Noam (2000) *New Horizons in the Study of Language and Mind*, Cambridge University Press.
北川善久・上山あゆみ（2004)『生成文法の考え方』、研究社.
齊藤学（2006)『自然言語の証拠推量表現と知識管理』、博士論文、九州大学.
坂本勉（1998)「第１章　人間の言語情報処理」、『言語の科学11　言語科学と関連領域』、岩波書店、pp.1-55.
定延利之（2005)『ささやく恋人、りきむレポーター』、岩波書店.
田窪行則・金水敏（1996)「複数の心的領域による談話管理」、『認知科学』3-3, 日本認知科学会.
Takubo, Yukinori & Satoshi Kinsui (1997) "Discourse Management in terms of Mental Spaces," *Journal of Pragmatics*, Vol. 28, No.6. pp.741-758, Elsevier Science, Amsterdam.

＊本論文は、2006年10月６日に京都大学で行われた、Japanese/Korean Linguistics Conference 16のJapanese Pre-Conference Workshopにおいて発表された内容をまとめたものである。なお、本稿は、研究スーパースター支援プログラムの助成による研究成果の一部である。

1　この操作（numeration formation）については、厳密なアルゴリズムとしてのモデル化は不可能であると考えている。

2　Parserというものは音連鎖を入力とするものであるから、このモジュールにおいて意味解釈が可能になるということは、いわば、PFからLFに変換するメカニズムの構築が可能だと主張することになるが、筆者はcomputational systemの中の操作は可逆的なものではないと理解しているので、このような試みには無理があるのではないかと考えている。特に、LF移動を仮定する必要がある文の解釈などは、いったいParserがどのように対処しうるのか、きわめて難しい問題が提示されるであろう。

3　たとえば、坂本（1998：45－46）にその考え方が詳しく解説されている。

4　この論文では、詳しいことについては述べられないが、すべての単語がいわゆる「概念」と結びついているわけではない。たとえば、numerationの要素には、λ演算子に対応することになる要素もあると考えている。そういう場合も考慮にいれると、(2i) は「LFの構造はそのままに、それぞれの単語をSR要素に置き換える」と述べたほうが正確であるが、本文では、わかりやすさを考慮した述べ方にしてある。

5　各概念を組み合わせる典型的な方法としては、いわゆるfunction applicationが考えられるが、それが唯一の方法であるとは仮定していない。(2ii) において、合成（composition）という語は、もっとも広義のカバータームとして

用いている。

6　一般的には、Semanticsとはその文の真理条件を定める仕組みであるという位置づけが多いが、ここでは、その考え方はとっていない。

7　このあたりの仕組みについても、この論文では詳しく述べることができないが、基本的に齊藤 (2006) で展開されている知識モデルを想定している。齊藤 (2006) は、ラシイやヨウダの証拠推量の用法を、(3i, ii) の観点から考察したものである

8　ただし、このInferenceモジュールは、いわゆる「帰納的推論」を含まない。帰納的推論は、厳密にアルゴリズムで表現できるものではないと考えているからである。極端な場合、特定の１つの出来事や経験に基づいて一般的な命題を「推論」することも、私たちはしばしば行ってしまうものである。

(i)　傘を忘れたときに限って、雨が降る。

(ii)　フィンランド人は勤勉だ。

9　このデータベースには、一般に「知識 (knowledge)」と呼ばれるものだけでなく、「単なる信念 (belief)」と呼ばれるものも含まれている。knowledgeとbeliefを二値的に区別することはできないと考えているからである。そのかわりにここでは、齊藤 (2006) で提案されている通り、その情報の出処 (たとえば、直知によるものか、誰かから聞いたのか、もしくは、どれかの命題から自分が推量したのか、等) をそれぞれの命題にタグとして付しておくという形式を想定している。いわゆるknowledgeとしての命題Pと単なるbeliefとしての命題Pの違いは、この出処タグの違いとして相対的に表示されることになる。

10　作業記憶に何が呼び出されるかということがアルゴリズムでとらえられないことであると考えているのと同様、Conceptsの中の概念間のネットワークも非常に入り組んでおり、その関係の変化の仕方は、アルゴリズムで表現できるものではないと考えている。

11　注 6 でも述べたように、ここではSemanticsというモジュールを真理条件に関連づけていない。その理由の１つは、そもそも、真理条件が定めうる命題というのは、"World"に登録されるべき命題のみだと考えているからである。個人の頭の中の "World" に情報として蓄えられている「世界」と現実の「世界」との間に対応関係があってほしいというのは、確かに人間の傾向として認められるので、その働きをになっているモジュールも存在していると考えているが、それは言語というものの働きとは独立のものなので、ここでは議論の対象にしていない。

12　図１の全体像そのものは、筆者が想定しているものであるが、Computational SystemとLexiconの関わり方については、チョムスキーの生成文法の考え方そのものであり、以下の考察も特に「独自」のものではない。

13　ここで「確定した事象」とは、(i) すでに起こった出来事、(ii) 現存する状態、(iii) 予定が確定した出来事、のいずれかを指すこととする。

14　関西方言で「仕方ない、仕方ない」の意。

15　関西方言で「かまわない、かまわない」の意。

16　(20) に付してあるアクセントは、関西方言におけるアクセントである。

17　ただ、似たような意味だと思われる表現でも、この無変化パターンでの繰り返しが容認しにくい場合もあり、その区別の要因については、まだよくわからない。

(i)　a．*知らんわ、知らんわ (cf. (29c))
　　b．*知るか、知るか（cf. (30e)）
　　c．*んなあほな、んなあほな
　　d．*わかるか、わかるか

18　関西方言で、「来た」に対する敬意表現。

19　定延 (2005:91-94, 164-166) では、基本的にこのプロソディは、「話し手の悲観的なきもちと結びつく」とされている。これに対して、(i) のような例は、通常の意味では「悲観的なきもち」とは言いにくい。

(i)　a．ほんとに、がんばったね。
　　b．うまいこと、作ってあるねえ。

しかし、定延（2005: 167-168）では、日本の文化の中で、苦しみと感心が紙一重になっていると述べられているので、(i) の例も、その方向で説明されるのかもしれない。(31) の例も、あまり「悲観的なきもち」と結びつかないものであるが、おそらく、同じタイプのものであろうと判断した。