人工知能と対話技法 8－1
（1978. 11. 17）

意味抽出における概念階層の利用と手続き付加

田中穂積，内田ユリ子
（電子技術総合研究所・推論機構研究室）

§1 はじめに

我々が日常，言語を理解するために用いている知識の体系は複雑であり未知の事柄が多い。しかしそれは，少なくとも乱雑なものではないだろう。知識（概念）の体系には，一定の秩序もしくは構造があるに相違ない。

このような概念構造の典型的なもの（の1つ）として，概念を階層状に構成したもの，すなわち概念階層（Conceptual Hierarchy）構造が，心理学者，言語学者によって考えられてきた。たとえば心理学者の Collins & Loftus［8］は，概念階層の心理的実在性を強く主張している。Bever & Rosenbaum［5］の部分全体関係に関する階層構造の考察も興味深い。Miller & Johnson－Laird［2］は，locative inclusion, part-whole (hasa), class inclusion (isa) 関係からなる3種の概念階層構造を提案し，それが，我々の言語理解と密接に関連していることを，英語について説明している。一方，言語学者も，古くから意味分類を行なう研究（意味分野の研究）で，概念階層の考え方が有用であることを論じている（Lyons［1］，Nida［3］，Palmer［4］）。我が国では，栗原［15］等による意味辞書作成の試みが注目される。

最近の言語理解システムを目指す試みにおいて，知識表現（knowledge representation）が大きな比重を占めてきている。そこでは，概念を階層状に表現することの利点が論じられている。彼等により，概念階層が用いられるのは，上述した心理学者，言語学者からの知見の他に，つぎの2つの理由があると思われる。

第1に，概念階層により，上位概念のもつ性質を下位に遺伝（伝播）させることができる。これにより，幾つかの概念に共通した事柄を，上位の概念に記述しておくだけで済むので，概念の追加修正が容易になると共に，記憶容量も節約される。第2に，本稿で考察するような概念階層では，上位と下位の概念間に推移律が成立しているから，概念階層をたどることにより，簡単な推論を行なうことができる。Carbonell はそれをフルに活用し，南米の地理に関する質問応答システム SCHOLAR を作成した［7］。

しかしながら，文の意味構造を抽出する部分のプログラムに関していえばこれまで概念階層の考え方を十分活用しているとはいえない。本稿では，日本語文の意味構造を抽出するプログラム EXPLUS［13］について説明するが，そこでは2種類の概念階層を区別し，使い分けている。文の意味構造を抽出するプログラムでは，このような区別が必要になることを主張することが，本稿の1つの目的である。

本稿で提案する意味表現用言語 SRL では，下位から上位概念のもつ性質の一部を除去したり，下位から上位概念に至る経路を制御できる。このような考え方の重要性は，Raphael［16］によって既に指摘されている。最近では Reiter［9］，Shapiro［11］が再びそれを論じている。第2章で説明する意味表現用言語 SRL を用いた記述法は，それらに対する一つの解を与える。

第3章では，日本語の ~~格助詞［13］~~

≪「ソノ」が，概念階層構造と深く関係していることを明らかにしてみたい。英語でも，~~「ノノ」~~「ソノ」に対応する~~「of」~~「the」について，これと全く同様な議論が可能であると考えている。第3章では特に，SRLを用いた概念階層をベースにすれば，「ソノ」の機能の一部が，より明確化し，新しい興味ある知見が得られることを示す。

第4章では，意味構造を抽出する時に，EXPLUSが，概念構造をどのように利用しているかを説明する。'isa'に関する概念階層を利用して意味構造を抽出することは，定理証明の問題を解決することと同等であることが示される(§4.1)。また，手続き付加とプロダクション・システムの考え方が有効であることを示す。局所レベルでの手続き付加については§4.2に，大域レベルの手続き付加については§2.4の終りで簡単に説明する。

§2 意味表現用言語SRL

SRL (Semantic Representation Language)の一部は，既に田中[13]で概略説明されているが，本章では，新たに付加された事柄も含めて，SRL全体についてのやや形式的な説明を行なう。SRLは，Bobrow & Winograd[6]の提案した知識表現用言語KRLを参考にして設計されている。今後の使用経験により，仕様が変ることは十分ありうる。したがって，本章での説明は，現時点でのまとめであるということに注意されたい。

§2.1 ユニット記述

SRLによる意味表現の基本は，ユニット記述である。ユニット記述は図2.1のようなスロットの集合から構成された1つのフレーム[6,14]である。図中，スロット<unit-desig>から

```
( <unit_desig>   unit
        <part-of>
        <self>
        <sem-feature>
        <slot>  ....<slot>
        <infer>  ....<infer>  )
```

図2.1 ユニット記述

<sem-feature>までは，特別なスロットで省略不可能である。またスロット<part-of>から<sem-feature>までは，概念階層に関する特殊なスロットであり，§2.2.1，§2.2.2で詳しく説明される。<slot>以降は，必要に応じて付加されるスロットで，文の意味構造を抽出する時に使われる(§4.1)。

<unit-desig>のBNF記法による定義は以下のようである。

```
<unit-desig>
    ::= <unit name>
                ((<unique name> . <unit name>))
<unit name>::= KAO | RENZU | ....
<unique name>::= N000001 | "L1" | ....
```

<unique name>は，LISPのGENSYM関数で生成された番号(N000001)か，「レンズL1」のように文中で与えられた固有名詞("L1")である。

SRLによる意味記述例を以下に示す。これはEXPLUSで実際に用いられているものである。

```
((N000001 . KAO) unit
    (part-of (a KARADA))
    (self (a BUTTAI))
    (of %PART)
    (-NO %ANIMAL =SUPERP))
```

図2.2 SRLによるユニット記述例(1)

§2.2 概念階層の種類とその表現形式

§1で述べたように，我々の概念の体系には，数種の階層があると考えられている。 KRLでは，概念階層を，ただ1つのスロット<self>で表現しようとしているために限界がある。SRLでは，part-whole（'hasa'）関係と，class-inclusion（'isa'）関係についての概念階層を表現するために，<part-of>スロットと<self>スロットが設けられている。 第3章と4章で明らかにするように，これら2種類の概念階層は，意味構造抽出に大きな役割を果たす。

§2.2.1 <part-of>スロット

<part-of>スロットにより，part-whole（'hasa'）関係の概念階層を表現することができる。 Miller & Johnson-Laird は，それを partonymy とよんでいる。 partonymy の例を図2.3に示す。 たとえば，「顔」は「目」，「鼻」，「口」等の上位部分である。 また「目」は，「瞳（黒目）」と「白目」の上位部分である。

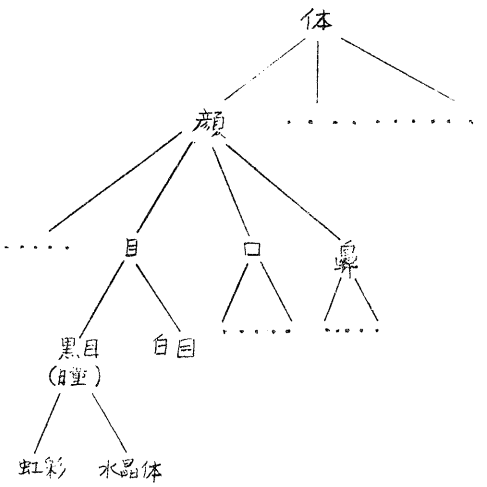


図2.3 partonymy の例

'hasa' に関する推移律により，顔は白目，黒目の上位部分であることがいえる。 <part-of>スロットのBNFによる定義を以下に与える。

<part-of> ::= <category>
| (part-of <h-description>$_1$, ... <h-description>$_m$)
<h-description> ::= <prototype description>
| <selector description>
<prototype description> ::= (a <unit name>)
<selector description> ::=
(a <unit name>$_1$ of a <unit name>$_2$)
<category> ::= basic | specialization | ...

<category>, <prototype description> はKRLの形式と同じである。 <selector description> については§2.3で詳しく説明される。

§2.2.2 <self>スロット

<self>スロットにより，class inclusion（'isa'）関係の概念階層を表現することができる。 Lyons は，それを hyponymy とよんでいる。 hyponymy の例を図2.4に示す。

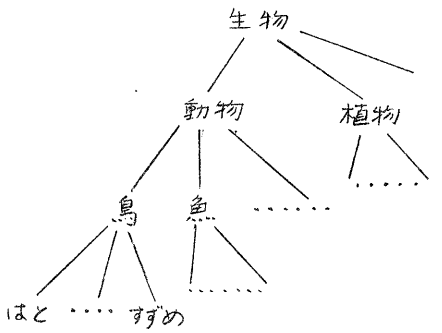


図2.4 hyponymy の例

hyponymy では，<self>スロットを介して上位概念のもつ性質が下位へ伝播する。 たとえば，「鳥」と「鳩」に関する（SRLによる）つぎの記述により，「鳩」は飛ぶことがいえる。

```
(TORI   unit   specialization
        (self (a  DOUBUTU))
        (hasp  =  TOBU) ·····)

(HATO   unit   specialization
        (self (a  TORI)) ·····)
```

図2.5　SRLによるユニット記述例(2)

推移律に関してつぎのことを注意しておく。(A isa B)&(B isa C)なら(A isa C)が成立する。hyponymyに関しては，isaを論理記号⟶(implies)でおきかえることができる。partonymyでも，(A hasa B)&(B hasa C)なら(A hasa C)が成立する。しかし，hasaを⟶でおきかえることはできない。

以下にBNFによる<self>スロットの定義を与える。

```
<self>  ::= (self <i-description>1
                  ···· <i-description>n)
<i-description> ::= <prototype description>
    | <perspective> | <without description>
<perspective> ::= (a <unit name>
    with (<slot name>1 = <value>1)
     ···· (<slot name>i = <value>i))
<without description> ::=
    (a <unit name>
       without <slot specifier>1,
            ···· <slot specifier>j)
```

<perspective>はKRLのものとほぼ同じである。たとえば
(a DENKYU with (KOUDO = (1cd)))
は，<perspective>の例である。
<without description>は§2.3で詳しく説明される。

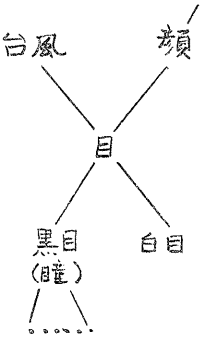
§2.3　概念階層表現の精密化

我々の概念の体系は，前節に述べたような整然とした概念階層により組織されているのだろうか。残念ながら，概念の世界はもっと複雑である。例外が無数にある。しかし，例外があるからといって，概念階層の考え方を捨てる必要はない。本節では，むしろ，例外を例外として認め，それを，これまで述べてきた概念階層へ自然な形式で付加して，概念階層を精密化する手法について論じる。(このような観点は，従来の知識表現の研究で見過されてきたことのように思えるので，特に重要性を強調したい。)すなわち，<selector description>(§2.2.1)と<without description>(§2.2.2)である。

§2.3.1 <selector description>

<part-of>スロットのなかで用いられる<selector description>により，下位から上位部分に至るpartonymy上での経路を制御することができる。

図2.3の「目」を考えてみる。その時「目」をもつものは，「顔」だけではないことに気付いただろうか。「台風」も「目」をもっている(これは英語でもいえる)。したがって，図2.3の「目」に関係する部分のpartonymyを右図のように訂正すべきである。



そうすると，hasa関係についての推移律を用いて，「台風」が「黒目」や「白目」をもつという奇妙なことになってしまう。これを避けるために，SRLでは，<selector description>を用いて，「黒目」や「白目」は「台風」ではなく「顔」の一部であることを宣言する。すなわち，

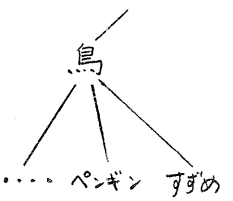
```
(KUROME  unit
     (part-of (a ME of a KAO))
     ·····)
```

のようにする。

§2.3.2 <without description>

<self>スロットのなかで用いられる<without description>により，上位概念のもつ性質の一部が，下位に遺伝することを阻止できる。

図2.4の「鳥」を考えてみる．図2.5で，「鳥」は飛ぶことが表現され，その性質は，下位の「鳩」に遺伝する．多くの場合，これは合理的である。しかし，筆者の知る限り，「だ鳥」と「ペンギン」は飛ぶことができない。にもかかわらず「鳥」であるから，hyponymyは右図のようになる。このままでは，上位概念の「鳥」がもつ飛ぶという性質は「ペンギン」や「だ鳥」にも遺伝してしまう。



それを避けるために，SRLでは，<without description>を用いる．すなわち

```
(PENGIN  unit  specialization
        (self (a TORI without
                (hasp = TOBU )))
        . . . . . )
```

のようにする。without以下に列記する<slot specifier>（今の場合は(hasp = TOBU)）は，それを含むユニットの上位概念（今の場合，TORI）のなかの特定のスロットを指定できるだけの詳細さがあれば良い。今の場合には，(a TORI without TOBU)と書いても良い。

以上で，概念構造を精密するためにSRLで用いる記法を述べた。つぎの注意をしておく。本節の冒頭で述べたように，<selector description>と<without description>は，例外的事実の表現形式である。これは，否定事実の表現と深い係り合いがある。

また，概念階層において，同一レベルの概念は，相互にdisjointである．この事実は，概念階層を利用する場合にdefaultとして仮定されているのであるが，§4に述べるように，スロットの意味記述を行なう場合に考慮しておかなければならない事柄である(§4.1)。

§2.4 その他のスロット

これまで説明したスロット以外のもの（図2.1参照）につき，概略説明する。

まずスロット<sem-feature>には，言語学者が選択制限規則によく用いる意味素性（的なもの）を記述する．EXPLUSでは，意味素性をも階層構造化しているが，そこでは，<part-of>や<self>のように，<selector description>や<without description>を用いたきめ細かな意味記述は不可能である。すなわち粗い意味記述で，高速に意味的検査を行なうためのものである。これについての詳しい説明は省略する．比較的詳しい説明は，田中[13]を参照してほしい。

<sem-feature> ::= (sf <s-marker>, ... <s-marker>$_m$)
<s-marker> ::= %LIVING | %NONLIVING | . . . .

つぎに<slot>には，それを含むユニット自体の意味的性質が記述される。これは意味構造を抽出する場合の基本になるものであり，<slot>の構成要素を含めて§4で詳しく説明される。

<slot> ::= (<slot name> <constriction> <action>, . . . <action>$_n$)

<action>は，手続き付加の典型である。<constriction>にも手続き付加が可能である．前者はwhen-filled methodに，後者はto-fill method[6,14]（FRL[10]ではrequire methodとよんでいる）に対応している。

<infer>は，文の意味解釈が一通り終り，大域的立場から適当な推論を加

える場合に用いるmethodである。

<infer> ::= (infer <LISP関数>)

この部分を用いて，「花子の車は赤い」という文から「花子は車を持っている」ことを推論することができる。この部分は，使用者に開放されており任意の仕事を行なうことができる。

§3. 概念階層と「ソノ」

§2では，SRLを用いて2種類の概念階層を表現できることを述べた。本章では，それらをベースにすれば，「ソノ」の機能の一部がより明確になることを示してみたい。これは，1つのメモリ・モデル上での「ソノ」の機能の説明にもなっている。本章の議論は，英語の「the」に対しても適用可能であると考えている。

以下では

{ ……X……。
{ ……ソノY……。

という文脈に限定して「ソノ」の機能を考察してみることにする。それゆえ，「ソノY」に関する後方照応は考えないことにする。

最も使用頻度の高い用法は，i)の場合である。

i) XとYの場合（X = ソノY）；

例1) { すずめがいる。
{ ソノすずめの足は……。

この場合は一見単純そうに見えるが困難な問題を含んでいる。XとYの外延的同一性の判定をどのようにしたら良いか分からないからである。XとYの名前が同じであっても，別々の個体を指すことがありうる（harmlack ヒツジ）。すなわち，内包的同一性は必ずしも外延的同一性を意味しない（逆も成立つ）のである。この問題の完全な解決は，今後の研究の進展に待たねばならないが，これは，以下の全ての場合について言えることである。EXPLUSではその部分的解決がはかられているがアルゴリズムの説明は別稿に譲りたい。

つぎに，2種類の概念階層に関する「ソノY」について考察する。

ii) YがXの（hyponymy上での）上位概念（X = ソノY）；

例2) { すずめがいる。
{ ソノ鳥は全国に分布している。

これをSRLの表現で図示すれば図3.1のようである。

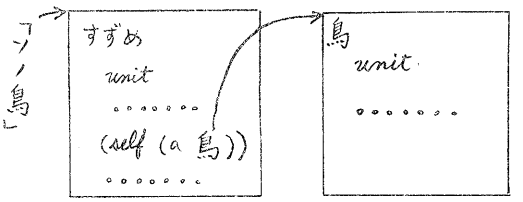


図3.1

iii) YがXの（partonymy上での）上位部分；

例3) { 寒さのため旅人は思わず襟を立てた。
{ ソノ外套は……。

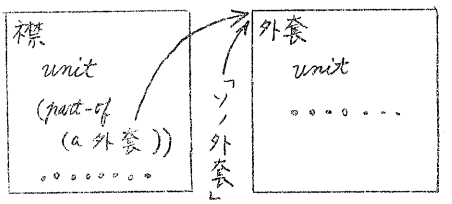


図3.2

iv) YがXの（partonymy上での）下位部分（ソノ = Xの：代行指示）；

例4) { 自転車を買った。
{ ソノ サドルは……

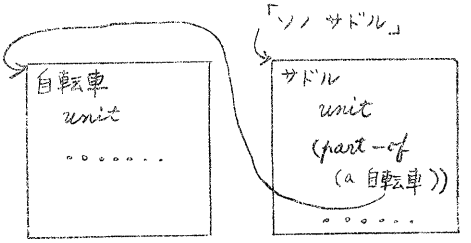


図3.3

組み合せ上から，　Yが Xの（hyponymy 上での）下位概念の場合があると推測される。　しかし，その例文を発見することがむずかしい*。　筆者の1人は，その問題も含めて，現在具体的な言語資料にあたり，より精密な分析を進めている。　機会を見て報告したいと考えている。

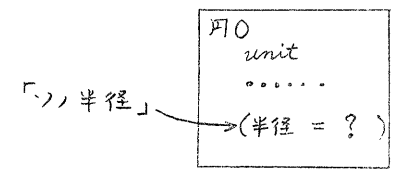
さらに，図3·1と図3·2 & 図3·3を比較すれば，ii)では，「ソノY」はXを指すが，iii)，iv)では，「ソノY」がY自身を指しているのは興味深い。

以上の2つの観察は，hyponymy と partonymy の差が，具体的な言語現象として表出しているものと解釈されよう。　そして，それが　具体的なメモリ・モデルを通じて観察されたことは興味深い。

以上の他に，つぎのような用法もある．

v) YがXの属性（ソノ＝Xの）；

例5) {円Oがある．
　　{ソノ半径は10cmである。



つぎの用法は，概念階層と直接関係せず，ソノとXが関係する。

vi) (ソノ＝Xの)の意味に解釈されるが，Xが Yの基準点をあらわす；

例6) {凸レンズを置く。
　　{ソノ左に .....

以上の例は，XとYとが，ソノや概念階層を介して何らかの関連をもつものであった．　以下の例は，そのような直接的な関連のないものである。

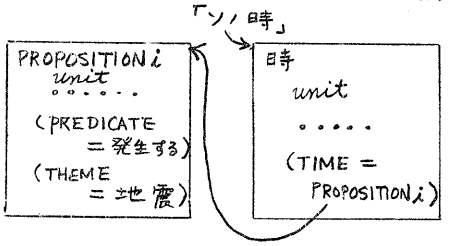
vii) 前文により，何らかの新しい実体が生成され，Yがそれを指す[12,17]。

例7) {自転車を売った。
　　{ソノ金で本を買った。

viii) 前文が行為もしくは出来事をあらわし，ソノがそれを指す；

例8) {地震が発生した。
　　{ソノ時私は京都にいた。

これは，EXPLUSでは，次のように解釈される．（ソノ＝PROPOSITIONi）



§4　意味構造の抽出と概念階層および手続き付加

意味構造を抽出する時にも，概念階層が頻繁に利用される。　EXPLUSでは，ユニット間干渉という考え方を用いて意味構造が抽出される[13]。

2つのユニットの，係る側をFILLER，受ける側をORIGINとよぶことにする。　ユニット間干渉の基本は，FILLERをORIGIN中のいずれかのスロットに埋め込むことである。　その時，概念階層が利用される。それには，

i) FILLERを満たしうるスロットが発見されるまで，<self>スロット(hyponymy)を介し，ORIGINの上位概念をつぎつぎにたぐる。

ii) あるスロットがFILLERを満たしうるかどうかの検査にpartonymyとhyponymyが利用される。

i)は，hyponymyの性質(§2·2·2)から自明のこととして説明を省略。　ii)について，すこし考察してみよう。そのために，§2·4で詳しい説明を保留しておいた<slot>の各要素について説明する。　正確には，<slot>が満たされているかどうかによって<slot>の記述形式が異なる。

<slot> ::= <satisfied slot>
　　| <unsatisfied slot>

---

* {レンズを買った。
　{ソノ凸レンズの焦点距離は .....
は，やや不自然であるという人がいる。

<satisfied slot> ::= (<slot name> = <value>)
(<slot name> ? <value>)
<unsatisfied slot> ::=
(<slot name> <constriction>
<action>$_1$ ・・・・ <action>$_n$)

<satisfied slot>については例をあげるにとどめる; (KOUDO = (1 cd)); (REL-PL = (SORE-WA (N000001 • HAKO) NO-MAE))

§4-1 <unsatisfied slot>

辞書項目に対するユニット記述中には，満たされていない<slot>が多数存在する。その記述は，前述のii)のベースとなるものである。EXPLUSは日本語を対象にしているため，<unsatisfied slot>の記述には言語の特殊性が反映される。

<slot name> ::= -WO | (@OR -GA -WA) | ・・・・

<slot name>は，FILLERに後続しうる格助詞を書く。上例では「が」，「は」のいずれもが後続しうることを@ORを用いて記述している。

<constriction> ::= <atomic> | <compound>
<atomic> ::= <s-marker>
| <prototype description>
| (@TO-TEST <LISP function>)
| (@NOT <atomic>)
<compound> ::= (@OR <compound>$_1$ ・・・ <compound>$_m$)
| (<compound>$_1$ ・・・ <compound>$_n$)
| <atomic>

<compound>では，AND記号(@AND)は省略される。また NOT記号(@NOT)は，<atomic>に直接付随したものだけが許されることに注意。@TO-TESTと<action>は，手続き付加に関係している。これは次節で詳しく説明することとし，ここでは，スロットを満たす条件について述べる。(なお，図2・2のスロット(-NO %ANIMAL =SUPERP)は<unsatisfied slot>の例である。)

FILLERがスロットを満たしうるかどうかの検査は，つぎのようにする。

i) <slot name>を用いてFILLERに後続しうる格助詞の妥当性を確認する。
ii) FILLERの<prototype description>を(a F)とした時つぎの論理式

(a F) ⟶ <constriction> (1)

が成立するかどうかを調べる。

(1)の検査には，hyponymyに関する概念階層が用いられる。<prototype description>には，述語が対応している。たとえば(a 鳥)に対して鳥(x)。以下では，<prototype description>の代りに，述語論理の表現を用いる*。(1)は

∀x(F(x) ⟶ <constriction>) (2)

のように書ける。(2)は，たとえば図2・4のhyponymyを考慮して解かれることになる。図2・4のhyponymyから§2・2・2に述べたように，

$$\begin{cases} \forall x\,(\text{鳩}(x) \longrightarrow \text{鳥}(x)) \\ \forall x\,(\text{すずめ}(x) \longrightarrow \text{鳥}(x)) \\ \quad \cdots\cdots\cdots\cdots \\ \forall x\,(\text{鳥}(x) \longrightarrow \text{動物}(x)) \\ \quad \cdots\cdots\cdots\cdots \end{cases} \quad (3)$$

が成立する。

図2・4は，さらに，同レベルの概念間にdisjoint性が成立しているから，

$$\begin{cases} \forall x\,(\text{鳥}(x) \longrightarrow \sim\text{魚}(x)) \\ \forall x\,(\text{鳥}(x) \longrightarrow \sim\text{哺乳類}(x)) \\ \quad \cdots\cdots\cdots\cdots\cdots\cdots \\ \forall x\,(\text{鳩}(x) \longrightarrow \sim\text{すずめ}(x)) \end{cases} \quad (4)$$

(2)は，(3)，(4)と共に解かれることになる。すなわち(2)は，定理証明の問題に帰着される。

しかし，SRLでは，述語論理を用いた定理証明を行なう代りに，hyponymyをたどることで(2)を証明する。(3)はhyponymyを上位方向にたどり，左辺から右辺のノードに到達可能を意味している。一方(4)は，左辺のノードから上位にたどり，右辺の否定を除いたノードに到達不可能なことを意

---

* 意味素性<S-marker>についても述語論理表現が用いられるものとするが，本節の説明ではそれを省略する。厳密には，FILLERの<S-marker>の集合をSF(x)とすれば(2)は ∀x(SF(x) & F(x) ⟶ <construction>)である。

味している。 いずれも *hyponymy* を上位方向にたどることで解決することができる。 図2.4に関していえば，「鳥」を上方にたどることにより，「魚」に到達できないことが理解できるだろう。 このことが $\forall x$ (鳥(x) → ∼魚(x)) の意味である．一方，「鳥」を上方にたどることにより，「動物」に到達することができる。 これが $\forall x$ (鳥(x) → 動物(x)) の意味である。

以上のように，*hyponymy* を上方にたどることにより，(2) が解かれるのであるが，(2)の右辺の <construction> 部には，否定 (@NOT) があらわれることがある。 前述したように，否定記号は <atomic> に直接付随しているので，この場合も，*hyponymy* 上をたどることで解決される。

「私の目」という文では，*partonymy* が利用されるが，(2)を用いない。単に「私」と「目」が *partonymy* 上で，どのような上下関係にあるかが検査される。

§4.2 手続き付加 [6,10,14]

前節に述べた @TO_TEST と <action> は SRL で用いられる手続き付加の例である。 @TO_TEST の引数 (<LISP function>) は，FILLER がそれを含むスロットを満たすかどうか検査される時に評価され，その結果の真偽は，前節の式 (2) の成立に関係する。 これは *to-fill method* である。 FILLER に関する，より精密な条件検査に用いられる。

一方 <action> は，FILLER がそのスロットを満たした時に起動されるもので，これは *when-filled method* である。EXPLUS では，これを用いて意味構造の抽出過程を制御したり，より深い意味的情報を抽出したりする [13]。

```
<action> ::= <LISP function>
           | <production rule name>
           |(<body of production rule>)
```

EXPLUS では，<action> としてプロダクション・ルールが用いられる。 プロダクション・ルールは手続きというより，規則であるが，その起動によりスロットを書き変えたり (RPL型)，付加したり (ADD型)，除去したり (REM型) する。 プロダクション・ルールは，

```
<production rule> ::=
   (<production rule name>
      <body of production rule>)
```

プロダクション・ルールの例を示す。

```
(=SUPERP (LOCUS = ***)
         (*** (@SUPP ORIGIN ....)))
```

名称は =SUPERP で，(LOCUS = ***) は書き変えパターン，(*** (@SUPP …)) は，書き変えパターン中の *** を，関数 (@SUPP …) の評価結果で書き変えることを意味している。 (*** (@SUPP ....)) は，プロダクション・ルールが起動された時の環境を用いて書き変えスロットパターンを組み立てるものである。それは <environment> とよばれ，変数とS式のペアのリストである。

```
<body of production rule>
   ::= <slot>
       (<variable>1 <S-expr>1)
       . . . . . .
       (<variable>n <S-expr>m)
```

RPL型と ADD型の場合には，意味抽出が進むにつれて何回もプロダクション・ルールが起動されることがある。プロダクション・ルールの起動によりスロットが変化することは，スロットが FILLER の意味記述を行なっていることから (§4.1)，新しい FILLER が予測されることと等価である。このように，EXPLUS では，プロダクション・ルールにより，意味的予測を行ないながら，意味構造抽出過程を制御している。

プロダクション・ルールの起動により，「半径が10cmの棒」や「壁から10cmの棒」 「10cmの棒」という文から意味構造が抽出される様子を図4.1に示す。*

* EXPLUSの意味構造抽出過程を(説明を容易にするために)簡略化して示す。

「棒」のユニット記述中のスロット ⓐと「10cm」が⊗で干渉し，i)は ii) のように変形する。 ⓐはプロダクション・ルール ?NAGASA，ADD1:，LISP関数 (APPLY ‥‥) により，それぞれ，ⓑ，ⓒ，ⓓのように変化する。単に「10cmの棒」なら，iii)の中に (NAGASA ? (10cm)) が残り，後で「?」は「=」に変えられる。

もし，「10cm」の左に「壁から」があると，スロット ⓓは iii)のⓔのように書き変えられる。一方「10cm」の左に「半径が」があるとスロット ⓒ は iv) のスロットⓕのように書き変えられる。

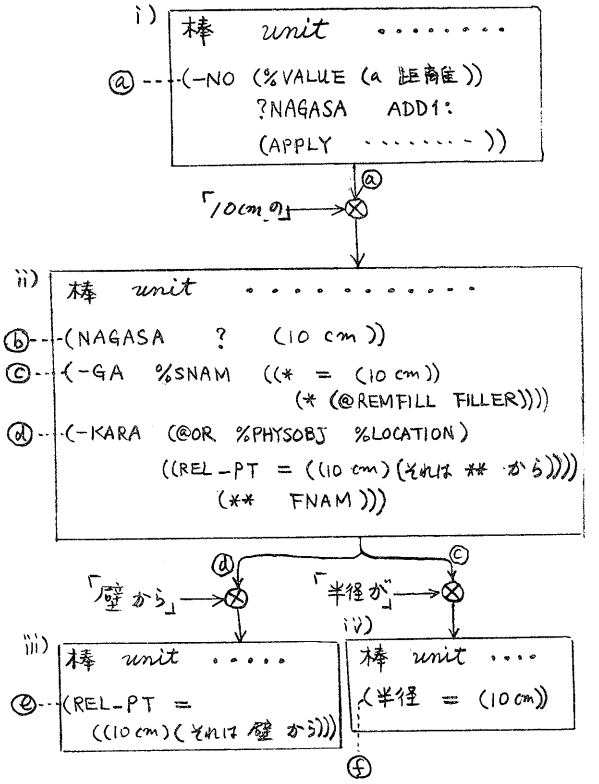


図 4.1 プロダクション・ルールの起動と意味構造の抽出

§5 おわりに

本文中でもふれておいたが，困難な問題は多数ある。§3のi)でさえ，完全な解はない。本稿では紙面の関係から十分に説明できなかったが，2種類の概念階層が，我々の言語理解に大切な役割を果していることを明らかにし，その動作しうるモデルを作成した。今後もその延長上の仕事を進める積である。

謝辞 電総研の渕 一博推論機構研究室長，古川康一主任研究官からは，本稿の考え方に対して貴重なコメントをいただいた。富山大学の寺津典子講師からは，言語学者の立場から，筆者の <selector description> の考え方が Bever 等の考え方に近いことを指摘していただいた。以上の諸氏に深謝する。真摯な言語学者であり，筆者の友人であった都立大 原田信一助教授は，本稿を構想している段階で急逝された。生前，筆者が氏から受けた影響の大きさは計り知れない。本稿の内容中，<without description> が言語学者のいう semantic distinguisher に近いことなどは氏の指摘による。ここに氏の生前の助言に対し深謝し，慎んで哀悼の意を表す。

参考文献

[16] Raphael: "SIR: A Computer ‥‥", in Minsky (ed): "Semantic Information Processing" The MIT Press, 1968.

[17] Charniak & Wilks: "Computational Semantics", North-Holland, 1976.

[1] Lyons: "Introduction to Theoretical Linguistics", Cambridge Univ. Press, 1968

[2] Miller & Johnson-Laird: "Language and Perception", Havard Univ. Press, 1976

[3] Nida: "Componential Analysis of Meaning", Mouton & Co. N.V., 1975

[4] Palmer: "Semantics", Cambridge Univ. Press, 1976.

[5] Bever & Rosenbaum: "Some Lexical Structures and Their Empirical Validity", in Steinberg et al (Eds.): "Semantics", Cambridge Univ. Press, 1971.

[6] Bobrow & Winograd: "An Overview of KRL", Cognitive Science, 1, 1, 1977.

[7] Carbonell: "AI in CAI ‥‥", IEEE, MMS-11, 4, Dec. 1970.

[8] Collins & Loftus: "A spreading-Activation Theory of Semantic Processing", Psychological Review, 82, 6, 1975.

[9] Reiter: "On Reasoning by Default", TINLAP-2, ACM, 1978

[10] Roberts & Goldstein: "The FRL Primer", MIT AI Memo 408, 1977.

[11] Shapiro: "Path-Based and Node-Based Inference in Semantic Networks", TINLAP-2, ACM, 1978

[12] 長尾，辻井："意味および文脈を考慮した日本語文の処理"，情報処理，1，1976

[13] 田中(穂)："日本語の意味構造を抽出するシステム EXPLUS について"，通学論誌，61-D，8，1978

[14] Winograd: "Frame Representations and The Declarative-Procedural Controversy", in Bobrow & Collins (Eds.): "Representation and Understanding", Academic Press, 1975
(淵 監訳："人工知能の基礎"，近代科学社，1978)

[15] 栗原(俊)："自然言語の機械処理"，情報処理，14，4，1973.