

# 計算機上で動作する日本語 HPSG 文法の構築\*

橋本 力 (chashi@sils.shoin.ac.jp)  
 神戸松蔭女子学院大学 大学院

## 1 はじめに

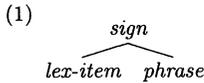
理論言語学と自然言語処理はかつて互いに刺激しあい発展してきたが両者の溝は近年ますます深まっている。本研究では2つのアプローチの橋渡しとなる計算機上で動作する言語学的に妥当な日本語文法を HPSG に基づき構築する。実装にはスタンフォード大学の文法開発環境 LKB を用いる。現在までに基本的な文の統語と意味を扱える文法を主辞素性原理、意味的継承原理等の HPSG の原理、制約を明示的に抽象化して実装した。本稿ではこの文法の詳細を報告し本研究が理論言語学と自然言語処理に与える影響を考察する。

## 2 日本語 HPSG 文法

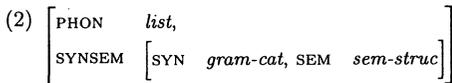
本稿で想定している日本語文法について述べる。HPSG の詳細は [5] を参照。

### 2.1 sign の階層

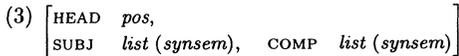
タイプ階層上部は (1) の通り。



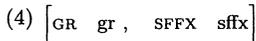
sign では (2) の素性が導入される。



gram-cat では (3) の素性が導入される。



HEAD には品詞に相当するタイプ pos が割り当てられる。品詞は名詞 *n*、動詞 *v*、後置詞 *p*、形容詞 *adj*、副詞 *adv* を想定している。adj と adv のスーパータイプとして品詞 *m* を設定している。全ての品詞タイプには可能な修飾先を示す素性 MOD が設定される。n には MOD の他に以下のような素性が割り当てられる。

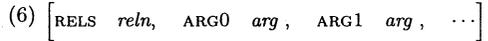


GR はその名詞の文法関係を表し、SFFX は格を表す。意味論的形式化は実装の簡易さを考慮したものとなっている。INDEX は動詞であれば event を、名詞であれば object を指示する (5)。

\* 文法は <http://sils.shoin.ac.jp/~chashi/talks/6> から入手可能



REXTER は意味の具体的内容に相当する predication (6) を値とする。RELS は関係を表し ARG 群は項を表す。



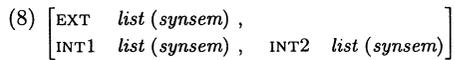
predication には RELS の項の数により arg1-predication、arg1-2-predication、arg1-2-3-predication があり、項をそれぞれ 1 個、2 個、3 個取る。predication には必ず ARG0 が存在し INDEX 値と同一視される。意味論的枠組みは Minimal Recursion Semantics (MRS) に依拠しており、詳細は [2] を参照。

### 2.2 lex-item の階層

lex-item は lexeme と word から成る (7)。

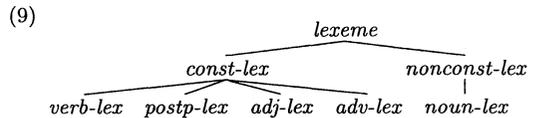


lex-item では項構造を表す素性 ARG-ST が導入される。ARG-ST は外項 (EXT) と内項 (INT) の観点から構造化されている (8)。

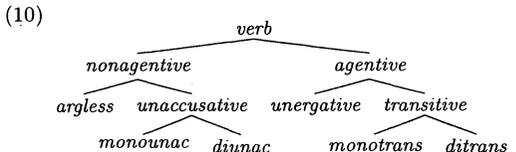


#### 2.2.1 lexeme の階層

lexeme 階層は品詞の観点から分類されている (9)。



動詞の階層 動詞の階層は [3] に基づいており、(10) のように、項構造の観点から分類されている。

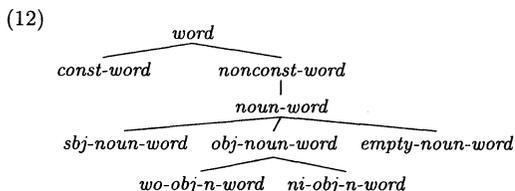


項具現化原理 項具現化原理 (ARP) は *lexeme* で適用される。外項の有無により 2 つに分けられる (11)。

- (11) a. 
$$\left[ \begin{array}{l} \text{SUBJ } \boxed{1}, \text{ COMP } \boxed{2}, \\ \text{ARG-ST } \left[ \text{EXT } \langle \rangle, \text{ INT1 } \boxed{1}, \text{ INT2 } \boxed{2} \right] \end{array} \right]$$
 b. 
$$\left[ \begin{array}{l} \text{SUBJ } \boxed{1}, \text{ COMP } \boxed{2} \oplus \boxed{3}, \\ \text{ARG-ST } \left[ \text{EXT } \boxed{1}, \text{ INT1 } \boxed{2}, \text{ INT2 } \boxed{3} \right] \end{array} \right]$$

### 2.2.2 word の階層

*lexeme* は何らかの語彙規則の適用を受けて *word* となり文中に現れる。*word* の階層 (12) と制約 (13) を示す。

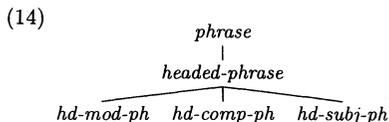


- (13) 
$$\left[ \begin{array}{l} \text{SYN } \boxed{1}, \text{ SEM } \boxed{2}, \text{ ARG-ST } \boxed{3}, \\ \text{STEM } \left[ \begin{array}{l} \textit{lexeme} \\ \text{SYN } \boxed{1}, \text{ SEM } \boxed{2}, \text{ ARG-ST } \boxed{3} \end{array} \right] \end{array} \right]$$

(13) により対応する *word* と *lexeme* の間で SYN 値 SEM 値 ARG-ST 値が等しくなる。*const-lex* に属する *lexeme* とそれに対応する *word* のマッピングは全て定常語彙規則 *const-irule* により行われる。*const-irule* はカテゴリを *lexeme* から *word* に変えるだけで他は何もしない。*nonconst-lex* に属する唯一の *lexeme* である名詞とそれに対応する *word* のマッピングを担う語彙規則は主格名詞語彙規則 *ga-noun-irule*、対格名詞語彙規則 *wo-noun-irule*、与格名詞語彙規則 *ni-noun-irule*、無接辞名詞語彙規則 *empty-noun-irule* の 4 つに分けられる。これらは音形変化 (接辞付加) にのみ関わり、それぞれ「が」「を」「に」「 $\phi$ 」を付加する。文法関係と格はマッピング先のタイプ階層により決定される。

### 2.3 phrase の階層

*phrase* は *hd-subj-ph*、*hd-comp-ph*、*hd-mod-ph* の 3 つであり階層は (14) のようになる。



*phrase* の段階で素性 DTRS を導入する (15)。

- (15) 
$$\left[ \text{H } \textit{list}, \text{ NH1 } \textit{list}, \text{ NH2 } \textit{list} \right]$$

H は主辞、NH1、NH2 は非主辞を表す。*phrase* には意味的構成性原理が適用される (16)。

- (16) 
$$\left[ \begin{array}{l} \text{RELS } \boxed{1} \oplus \boxed{2} \oplus \boxed{3} \\ \text{DTRS } \left[ \begin{array}{l} \text{H} | \text{RELS } \boxed{1}, \\ \text{NH1} | \text{RELS } \boxed{2}, \text{ NH2} | \text{RELS } \boxed{3} \end{array} \right] \end{array} \right]$$

*headed-phrase* には主辞素性原理 (17) 意味的継承原理 (18)、そしてデフォルト制約である結合価原理 (19) が適用される。

- (17) 
$$\left[ \text{SYN} | \text{HEAD } \boxed{1}, \text{ DTRS } \left[ \text{H} | \text{SYN} | \text{HEAD } \boxed{1} \right] \right]$$

- (18) 
$$\left[ \text{SEM} | \text{INDEX } \boxed{1}, \text{ DTRS } \left[ \text{H} | \text{SEM} | \text{INDEX } \boxed{1} \right] \right]$$

- (19) 
$$\left[ \begin{array}{l} \text{SYN } \left[ \text{SUBJ } / \boxed{1}, \text{ COMP } / \boxed{2} \right], \\ \text{DTRS } \left[ \text{H} | \text{SYN } \left[ \text{SUBJ } / \boxed{1}, \text{ COMP } / \boxed{2} \right] \right] \end{array} \right]$$

*hd-subj-ph*、*hd-comp-ph*、*hd-mod-ph*、開始記号はそれぞれ (20)(21)(22)(23) のようになる。

- (20) 
$$\left[ \text{SYN} | \text{SUBJ } \langle \rangle, \text{ H} | \text{SYN} | \text{SUBJ } \boxed{1}, \text{ NH1 } \boxed{1} \right]$$

- (21) 
$$\left[ \begin{array}{l} \text{SYN} | \text{COMP } \langle \rangle, \\ \text{H} | \text{SYN} | \text{COMP } \langle \boxed{1}, \boxed{2} \rangle, \text{ NH1 } \boxed{1}, \text{ NH2 } \boxed{2} \end{array} \right]$$

- (22) 
$$\left[ \text{H } \boxed{1}, \text{ NH1} | \text{SYN} | \text{HEAD} | \text{MOD } \boxed{1} \right]$$

- (23) 
$$\left[ \begin{array}{l} \textit{start} \\ \text{SYN } \left[ \text{HEAD } v, \text{ SUBJ } \langle \rangle, \text{ COMP } \langle \rangle \right] \end{array} \right]$$

## 3 日本語文法の実装

### 3.1 文法開発環境 LKB

LKB はスタンフォード大学 CSLI で LinGO プロジェクトの一環として継続的にメンテナンスされている文法開発用ソフトウェアである。LKB はタイプ付き素性構造の形式に従った文法が入力されると、その文法に従って構文解析、文生成を行う [1]。

### 3.2 実装用日本語文法

実装の際には LKB による制限のため文法の形式化をいくつか変更した。

*sign* の素性 *sign* で導入される素性を変更した (24)。

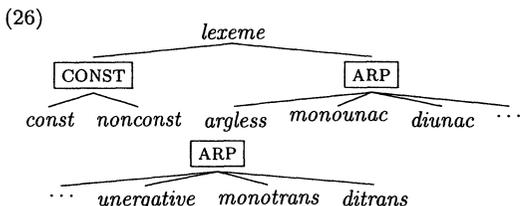
- (24) 
$$\left[ \begin{array}{l} \text{ORTH } \textit{*dlist*}, \text{ SYN } \textit{gram-cat}, \\ \text{SEM } \textit{sem-struc}, \text{ ARGS } \textit{*list*} \end{array} \right]$$

ORTH は PHON に対応する。ARGS は構成素を表す素性であり、*phrase* なら daughter (ARGS から DTRS にマッピングされる) に、*word* なら STEM に対応する。

項具現化原理 LKBでは(11)のような振舞いをそのまま実装することはできない。外項の有無だけでなく内項の有無、項を全く取らない場合をそれぞれ別のARPにより扱わなくてはならない。本文法では(10)に基づきARPを(25)のように6つに分けて規定した。

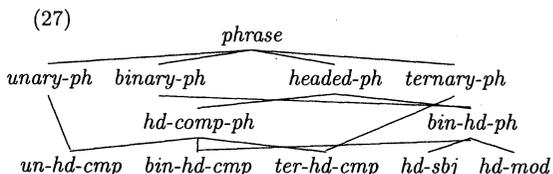
- (25)
- |                   |  |     |                            |        |  |
|-------------------|--|-----|----------------------------|--------|--|
| <i>argless</i>    | <table border="1"> <tr> <td>SYN</td> <td>[SUBJ *null*, COMP *null*]</td> </tr> <tr> <td>ARG-ST</td> <td>[EXT *null*, INT1 *null*, INT2 *null*]</td> </tr> </table> | SYN | [SUBJ *null*, COMP *null*] | ARG-ST | [EXT *null*, INT1 *null*, INT2 *null*] |
| SYN               | [SUBJ *null*, COMP *null*]   |     |                            |        |  |
| ARG-ST            | [EXT *null*, INT1 *null*, INT2 *null*]   |     |                            |        |  |
| <i>monounac</i>   | <table border="1"> <tr> <td>SYN</td> <td>[SUBJ ①, COMP *null*]</td> </tr> <tr> <td>ARG-ST</td> <td>[EXT *null*, INT1 ①, INT2 *null*]</td> </tr> </table>           | SYN | [SUBJ ①, COMP *null*]      | ARG-ST | [EXT *null*, INT1 ①, INT2 *null*]      |
| SYN               | [SUBJ ①, COMP *null*]  |     |                            |        |  |
| ARG-ST            | [EXT *null*, INT1 ①, INT2 *null*]  |     |                            |        |  |
| <i>diunac</i>     | <table border="1"> <tr> <td>SYN</td> <td>[SUBJ ①, COMP ②]</td> </tr> <tr> <td>ARG-ST</td> <td>[EXT *null*, INT1 ①, INT2 ②]</td> </tr> </table>                     | SYN | [SUBJ ①, COMP ②]           | ARG-ST | [EXT *null*, INT1 ①, INT2 ②]           |
| SYN               | [SUBJ ①, COMP ②]   |     |                            |        |  |
| ARG-ST            | [EXT *null*, INT1 ①, INT2 ②]   |     |                            |        |  |
| <i>unergative</i> | <table border="1"> <tr> <td>SYN</td> <td>[SUBJ ①, COMP *null*]</td> </tr> <tr> <td>ARG-ST</td> <td>[EXT ①, INT1 *null*, INT2 *null*]</td> </tr> </table>           | SYN | [SUBJ ①, COMP *null*]      | ARG-ST | [EXT ①, INT1 *null*, INT2 *null*]      |
| SYN               | [SUBJ ①, COMP *null*]  |     |                            |        |  |
| ARG-ST            | [EXT ①, INT1 *null*, INT2 *null*]  |     |                            |        |  |
| <i>monotrans</i>  | <table border="1"> <tr> <td>SYN</td> <td>[SUBJ ①, COMP ②]</td> </tr> <tr> <td>ARG-ST</td> <td>[EXT ①, INT1 ②, INT2 *null*]</td> </tr> </table>                     | SYN | [SUBJ ①, COMP ②]           | ARG-ST | [EXT ①, INT1 ②, INT2 *null*]           |
| SYN               | [SUBJ ①, COMP ②]   |     |                            |        |  |
| ARG-ST            | [EXT ①, INT1 ②, INT2 *null*]   |     |                            |        |  |
| <i>ditrans</i>    | <table border="1"> <tr> <td>SYN</td> <td>[SUBJ ①, COMP &lt;②, ③&gt;]</td> </tr> <tr> <td>ARG-ST</td> <td>[EXT ①, INT1 ②, INT2 ③]</td> </tr> </table>               | SYN | [SUBJ ①, COMP <②, ③>]      | ARG-ST | [EXT ①, INT1 ②, INT2 ③]                |
| SYN               | [SUBJ ①, COMP <②, ③>]  |     |                            |        |  |
| ARG-ST            | [EXT ①, INT1 ②, INT2 ③]  |     |                            |        |  |

lexemeの階層 実装用文法のlexemeの階層は(25)のARPによる分類と屈折変化のある無しによる分類に従う。よってlexemeの最上部は(26)のようになる。



ARP群は動詞以外の品詞にも汎用的に適用される。具体的には名詞、副詞、形容詞には*argless-arp*が、後置詞には*monotrans-arp*が適用される。

phraseの階層 LKB上に実装する際はdaughterが1つの場合(*unary-phrase*)、daughterが2つの場合(*binary-phrase*)、daughterが3つの場合(*ternary-phrase*)を別々に規定しなくてはならない(27)。



意味的構成性原理も(28)のように3つに分れる。

- (28)
- |                |  |      |  |
|----------------|--|------|--|
| <i>unary</i>   | <table border="1"> <tr> <td>RELS</td> <td>①, DTRS [H   RELS ①]</td> </tr> </table>                                     | RELS | ①, DTRS [H   RELS ①]                                     |
| RELS           | ①, DTRS [H   RELS ①]   |      |  |
| <i>binary</i>  | <table border="1"> <tr> <td>RELS</td> <td>① ⊕ ②, DTRS [H   RELS ①, NH1   RELS ②]</td> </tr> </table>                   | RELS | ① ⊕ ②, DTRS [H   RELS ①, NH1   RELS ②]                   |
| RELS           | ① ⊕ ②, DTRS [H   RELS ①, NH1   RELS ②]   |      |  |
| <i>ternary</i> | <table border="1"> <tr> <td>RELS</td> <td>① ⊕ ② ⊕ ③, DTRS [H   RELS ①, NH1   RELS ②, NH2   RELS ③]</td> </tr> </table> | RELS | ① ⊕ ② ⊕ ③, DTRS [H   RELS ①, NH1   RELS ②, NH2   RELS ③] |
| RELS           | ① ⊕ ② ⊕ ③, DTRS [H   RELS ①, NH1   RELS ②, NH2   RELS ③]   |      |  |

head-comp-phraseも3つに分かれる(29)。

- (29)
- |                |   |      |   |      |   |
|----------------|---|------|---|------|---|
| <i>unary</i>   | <table border="1"> <tr> <td>SUBJ</td> <td>①</td> </tr> <tr> <td>DTRS</td> <td>[H [SUBJ ①, COMP *null*]]</td> </tr> </table>                     | SUBJ | ① | DTRS | [H [SUBJ ①, COMP *null*]]               |
| SUBJ           | ①   |      |   |      |   |
| DTRS           | [H [SUBJ ①, COMP *null*]]   |      |   |      |   |
| <i>binary</i>  | <table border="1"> <tr> <td>SUBJ</td> <td>①</td> </tr> <tr> <td>DTRS</td> <td>[H [SUBJ ①, COMP ②], NH1 ②]</td> </tr> </table>                   | SUBJ | ① | DTRS | [H [SUBJ ①, COMP ②], NH1 ②]             |
| SUBJ           | ①   |      |   |      |   |
| DTRS           | [H [SUBJ ①, COMP ②], NH1 ②]   |      |   |      |   |
| <i>ternary</i> | <table border="1"> <tr> <td>SUBJ</td> <td>①</td> </tr> <tr> <td>DTRS</td> <td>[H [SUBJ ①, COMP &lt;②, ③&gt;], NH1 ②, NH2 ③]</td> </tr> </table> | SUBJ | ① | DTRS | [H [SUBJ ①, COMP <②, ③>], NH1 ②, NH2 ③] |
| SUBJ           | ①   |      |   |      |   |
| DTRS           | [H [SUBJ ①, COMP <②, ③>], NH1 ②, NH2 ③]   |      |   |      |   |

デフォルトの扱い (29)では本来なら結合価原理より扱われるべきmotherとhead daughterのSUBJの同一性がいちいち述べられている。これはLKBがデフォルトを扱えないためである。このことは*hd-subj-phrase*(30)と*hd-mod-phrase*(31)にもあてはまる。

- (30)
- |      |                                  |
|------|----------------------------------|
| SYN  | [SUBJ *null*, COMP *null*]       |
| DTRS | [NH1 ①, H [SUBJ ①, COMP *null*]] |

