

# MSFA Lite の記述を基にした 文生成プログラムの動作原理と仕様

黒田 航

Modified 03/11, 02/08, 12, 14; Created on 02/07/2008

## 1 はじめに

この仕様書の第一の目的は MSFA Lite の記述を基にした文生成プログラム (frame-based sentence-generator: fbsgen) の仕様を解説することにある。第二の目的は、その説明を通じて、まだ完全に定まっていない MSFA Lite のタグづけの仕様の動機づけになる処理を解説することである。この点は実際にタグづけをする人の作業上の助けになるのはまちがいないと思う。

### 1.1 利用の効能

文生成マクロが導入されるまで、タグづけの結果の妥当性に対する明示的な評価がタグづけ作業者に明示的に教示としてフィードバックされる機会はなかった。だが、このプログラムの利用には次のような効能が期待できるので、それを使って自分のタグづけ結果を自己評価することが可能になった：

- (1) a. EVO 要素と GOV 要素の特定の妥当性の評価
- b. 指定されたフレーム名 (かそれを代表するものとして選ばれた述

語) の妥当性の評価

- c. フレーム要素の境界の妥当性の評価
- d. MSFA Lite 版での#要素 (e.g., 助詞) の指定の妥当性の評価

特に (1d) に効果が大きいと考えている.

## 2 文生成処理の仕様

### 2.1 動作原理

フレームを単位とした文生成処理に基本となるアルゴリズムは次の通り, かなり単純である:

- (2) 列  $j (=Cj)$  の  $[RiCj]$  から  $[Ri+1Cj]$  までの範囲が (色づけによって), フレーム  $f$  のフレーム要素  $f.e$  として認識されているとする時,
  - a.  $[RiCj]$  の指定  $x$  が “#” で始まる定項 (e.g., “#が”) ならば,  $x$  自体を,
  - b.  $x$  が “#” で始まらない変項 (e.g.,  $f.GOV, f.EVO$ ) ならば, その外部の参照値  $v$  を順番につなげて文を生成する.
- (3) フレーム  $f$  の記述に  $f.GOV$  か  $f.X=PRED$  (e.g.,  $f.EVO=PRED$ ) が現われていない場合,
  - a. Predicate for GOV 行に値  $p$  が指定されている場合には  $p$  を, Predicate for GOV 行に値が指定されていない場合にはフレームの名称  $f$  を最後の追加する.
  - b. 追加する  $f$  が “X の指定” か “X の評価” の場合には  $f$  の代わりに NULL を実現値とする (つまり, “X の指定” や “X の評価” はフレーム名としては通用するが述語化との対応づけ (=述語化) はなされない).

外部の参照値  $v$  の決定の仕方は §2.2 で説明する.

### 2.1.1

コードで書くと次の (4) の通り:

```
(4) s = "/"
    for each fe in fe_array:
        if fe begins_with "#":
            s = s + fe + "/"
        else:
            s = s + "/" + m_array[fe.index]
s = s + "/"
```

ただし, “+” は文字列の連結演算子で, `fe_array` は MSFA の  $C_j$  にある (フレームごとの) `fe` 列, `m_array` は 1 列目の形態素列とする (`fe_array` を与える前に要素の並べ替えを行っているなら, 後処理なしで  $C$  が実装済み).

## 2.2 変項の参照値の決定の手順

### 2.2.1 (# で始まる) 定項の処理

# で始まるフレーム要素  $e$  (これを「#要素」と呼ぶ) はすべて定項である. それは外部に実現値をもたないので, 形態素列を参照しない. ただし, #要素が複数のセルに渡る場合の挙動は (7d) で説明するように変則的である.

### 2.2.2 (# で始まらない) 変項の処理

# で始まらないフレーム要素  $e$  はすべて変項であり, 外部の値を参照する. それには二つのタイプがある.

- (5) a. タイプ 1:  $e$  がフレーム  $f$  の支配項  $f.GOV$  か  $f.X=GOV$  か  $f.X=PRED$  の場合

- b. タイプ 2: それ以外のすべての場合 (e.g,  $f$  者, 対象 [ $f$  の], 目的 [ $f$  の])

単純化して言うと, “.GOV”, “=GOV”, “=PRED” は変項の参照先を形態素列からフレーム名セル (= R3C $j$ ) か述語指定セル (= RiC $j$ , 通常は  $i=4, 5$ ) に転換する信号だと理解するのがよい.

これらは場合に応じて, 次の三つの仕方を実現値をもつ:

- (6) a. i. (3 行目にある) フレーム名  $f$  を実現値にもつ (タイプ 1 の場合).  
ii. Predicate for GOV の行にあるセルに指定されている (複合) 述語  $p$  を実現値にもつ (タイプ 1 の場合).  
b.  $e$  に対応する形態素列の文字列の範囲を実現値にもつ (タイプ 2 の場合).

(6 (a) i) と (6 (a) ii) は排他的, (6b) と (6 (a) i) か (6 (a) ii) は排他的である.

### 2.2.3 単一のフレーム要素が複数のセルに渡っている場合の処理

タイプ 1 とタイプ 2 で挙動が異なるので注意が必要である.

タイプ 2 の場合, フレーム要素  $e$  が複数のセルに渡る場合 (RC 表記では RiC $j$  から Ri+kC $j$  までのセルに渡る場合), その全体に対して形態素列の要素が対応づけられる.

タイプ 1 の  $f$ .GOV と  $f$ .X=PRED (e.g.,  $f$ .EVO=PRED  $\neq$   $f$ .EVO) と, #要素は次のようにおのおの別の仕方で行の要素とは対応づけられない:

- (7) a.  $f$ .GOV の実現値は, 3 行目に指定されたフレーム名の  $f$  になる (ただし, Predicate for GOV のセルに述語として別の値  $p$  が指定されている場合,  $p$  が実現値になる).  
b.  $f$ .X=GOV と  $f$ .X=PRED は  $f$ .GOV と同じように扱われる.

- c. # 要素は変項でなく定項なので、実現値はそれ自身である (そのため外部に値をもつことはない).
- d. X.SUP と X.EXT が複数のセルからなる時、先頭のセルのみが形態素列の値と対応づけられる (範囲の先頭だけが部分的に変項になる# 要素のように扱われる).

(7c) の扱いには次の §2.2.4 で説明する特徴がある.

#### 2.2.4 “# $X_1, \dots, X_n=Y_1, \dots, Y_m$ ” の指定

“# $X_1, X_2, \dots, X_n=Y_1, Y_2, \dots, Y_m$ ” という指定がある場合、生成された plain 文に現われるのはセパレータ “=” の左にある “# $X_1, X_2, \dots, X_n$ ” のみである (右側の要素 “ $Y_1, Y_2, \dots, Y_m$ ” を削除する処理を実装している).

### 3 注意点