AmpEngine: sharp and make_apply for morphemes

Fyodor Sizov <prodotiscus at github.com>

January 2018

1 Sharp (universal:token → universal:morpheme)

- UNIVERSAL:TOKEN => XYYM
- Set of container elements from *universal:morpheme* which have at least one *universal:char* (char sublayer) in common and have appropriate *LinkSentence* patterns:

$$\left.\begin{array}{rcc} 1_s)\ \text{UNIVERSAL:MORPHEME/SUFFIX}_{spec} & => & \text{YM} \\ props : B, C & & \\ 2_s)\ \text{UNIVERSAL:MORPHEME/SUFFIX}_{spec} & => & \text{M} \\ props : D, E & & \\ 3_s)\ \text{UNIVERSAL:MORPHEME/SUFFIX}_{spec} & => & \text{Y} \\ props : A, F & & \end{array}\right]$$

- List of stems found with the *resource handler*:

$$\left.\begin{array}{rcc} 1_r)\ \text{UNIVERSAL:MORPHEME/ROOT}_{spec} & => & \text{X} \\ props : \varnothing & & \\ 2_r)\ \text{UNIVERSAL:MORPHEME/ROOT}_{spec} & => & \text{XY} \\ props : \varnothing & & \end{array}\right]$$

- Log → ALL_POSITIONS{ELEMENT} := {0, 1, 2, 3}:

x	y	y	m
0	**1**	**2**	**3**

- Log → POS_PROHIB{ELEMENT} := {}
- Log → POS_PROHIB_IN_SPEC{ELEMENT} := {}
- $$\left.\begin{array}{c} 1_r)\ \text{ROOT} = 1_r = \text{X} => Log.\text{POS\_PROHIB}\{element\}\{1_r\} \leftarrow \{0\} \\ 2_r)\ \text{ROOT} = 2_r = \text{XY} => Log.\text{POS\_PROHIB}\{element\}\{2_r\} \leftarrow \{0,1\} \end{array}\right]$$
- # returns TRUE if at least one of Log.POS_PROHIB{element}{N_r} is not equal to Log.ALL_POSITIONS{element}

- n_r (ROOT) set is an example *spec* set which can be extracted from the parent layer without having to use container data, but via resource handler, i. e.

2 *make_apply* (universal:token → universal:morpheme)

- SPEC := suffixes, so we use the n_s set
- non-SPEC elements of *universal:morpheme* (found) are: ROOT (n_r set); *(PREFIX not found)*
- *INSIDE_SPEC* := n_s set => $[1_s \| 2_s \| 3_s]$ => $[ym \| m \| y]$
- *OUT_OF_SPEC* := n_r set => $[1_r \| 2_r]$ => $[x \| xy]$
- for each in OUT_OF_SPEC (n_r):
  - 1_r)
    1. Exclude positions from Log.POS_PROHIB{element}{1_r} => {0}:

x	y	y	m
0	**1**	**2**	**3**

=>

y	y	m
1	**2**	**3**

    2. permutations: for each in INSIDE_SPEC:
       * 1_s)
       * yym = ... + **ym** (1_s)
       * Log.POS_PROHIB{element}{1_r}{1_s} ← {2, 3}

y	y	m
1	**2**	**3**

=>

y
1

       * 2_s)
       * yym = (**y** + ...) OR (... + **y** + ...) OR (**y** + **y** + ...)

         Log.POS_PROHIB{element}{1_r}{2_s} ← {1}

y	y	m
1	**2**	**3**

=>

y	m
2	**3**

         OR
         Log.POS_PROHIB{element}{1_r}{2_s} ← {2}

y	y	m
1	**2**	**3**

=>

y	m
1	**3**

         OR
         Log.POS_PROHIB{element}{1_r}{2_s} ← {1, 2}

y	y	m
1	**2**	**3**

=>

m
3

       * 3_s)

* yym = ... + **m**
* Log.POS_PROHIB{element}{1_r}{3_s} ← {1, 2}

y	y	m
1	**2**	**3**

=>

y	y
1	**2**

– 2_r)

1. Exclude positions from Log.POS_PROHIB{element}{2_r} => {0, 1}:

x	y	y	m
0	**1**	**2**	**3**

=>

y	m
2	**3**

2. permutations: for each in INSIDE_SPEC:
   * 1_s)
   * ym = **ym** (1_s)
   * Log.POS_PROHIB{element}{1_r}{1_s} ← {2, 3}

y	m
2	**3**

=> ∅

   * 2_s)
   * ym = ... + **m**

   Log.POS_PROHIB{element}{1_r}{2_s} ← {3}

y	m
2	**3**

=>

y
2

   * 3_s)
   * ym = **y** + ...
   * Log.POS_PROHIB{element}{1_r}{3_s} ← {1, 2}

y	m
2	**3**

=>

m
3