Lexique et morphologie

2 décembre 2019

—Kata Gábor
kata.gabor@inalco.fr—

Construction d'une morphologie computationnelle M

1. créer des automates pour les (classes de) radicaux
2. créer des automates pour les affixes (prééfixes, suffixes)
3. combiner les automates de radicaux avec les automates d'affixes (règles morphotactiques : *dans quel ordre les morphèmes se suivent, règles phonologiques et orthographiques à la frontière de morphèmes, ...*)

un **Transducteur à états finis (FST)** est constitué de :

— V, un vocabulaire ou alphabet d'entrée
— S, un vocabulaire ou alphabet de sortie
— Q, un nombre fini d'états dont
  — état initial (un ou plusieurs)
  — état final (un ou plusieurs)
— f, fonction de transition : donne pour chaque état q $\in$ Q et chaque élément du vocabulaire v $\in$ V, le ou les états que l'on peut atteindre en partant de q et en utilisant v : f(q,v) $\subset$ Q.
— h, fonction de sortie qui associe à un état q et une entrée v $\in$ V une sortie s $\in$ S

Opérations sur les FSTs :

— **inversion** : échanger les étiquettes, l'entrée devient la sortie et vice versa. Un FST analyseur peut être ainsi transformé en FST générateur.
— **union** de deux transducteurs $T_1 \cup T_2$: x [$T_1 \cup T_2$] y si x [T_1] y ou x [T_2] y (le transducteur $T_1 \cup T_2$ traduit x en y si T_1 traduit x en y ou T_2 traduit x en y).
— **composition** de T_1 et T_2 : applique T_1, puis applique T_2 à la sortie de T_1. x [T_1] y puis y [T_2] z : le vocabulaire de sortie de T_1 est le vocabulaire d'entrée de T_2.

Morphologie par FSTs

$$M \subseteq F \times C \tag{1}$$

où F est l'ensemble des formes et C est l'ensemble des descriptions (catégories).

1. le **FST** a un vocabulaire d'entrée et un vocabulaire de sortie
2. **FST** en tant que **traducteur** : de la langue L sur le vocabulaire F vers la langue L' sur le vocabulaire C
3. **FST** définit une langue de paires :

$$F \times C = \{(f, c) | f \in F \ \ et \ \ c \in C\} \tag{2}$$

4. **FST** peut être traduit en **FSA** qui *accepte* une langue :

$$V_{FSA} = \{(f, c) | f \in F \ \ et \ \ c \in C\} \tag{3}$$

accepte les paires $\{(f, c) \in M \ \}$

Implémentations

— KIMMO, PC-KIMMO : Koskenniemi 1983, Karttunen 1983
— pour la rapidité :
  — XFST : Xerox Finite State Tool
  — SFST : Stuttgart Finite State Tool (morphologie anglaise, allemande, turque, latine disponible)
— pour la facilité et l'interface graphique :
  — Unitex, https ://unitexgramlab.org/fr
  — NooJ, http ://www.nooj-association.org/