Notes du module de grammaires lexicalisées

Hugo Mougard

16 septembre 2013

Table des matières

0.1 Grammaires lexicalisées

Une grammaire lexicalisée est une grammaire dont toutes les règles contiennent au moins un terminal.

0.2 Forme normale de Greibach

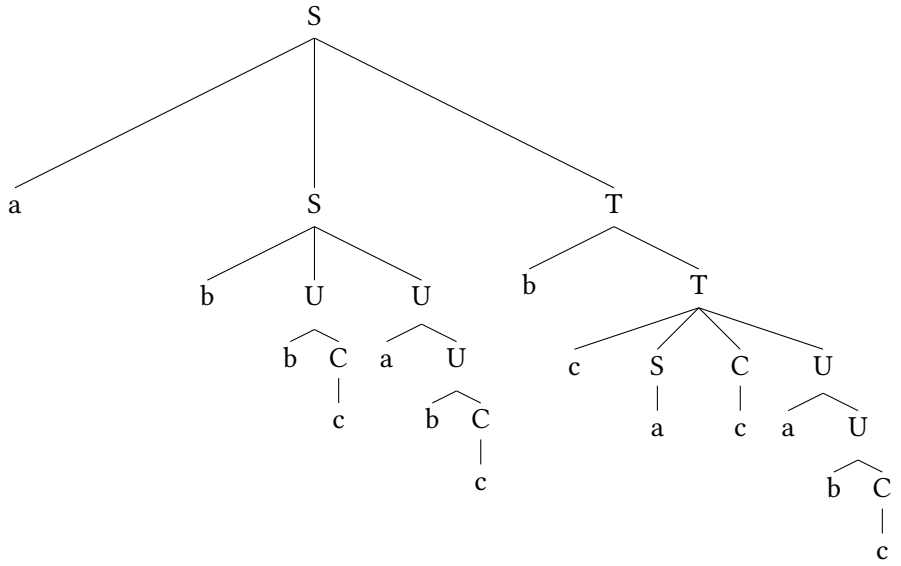
Concerne les langages propres (sans ε) dont les grammaires (propres elles aussi, ou réduites) sont sous la forme

$$V \longrightarrow XV^*$$

où X est terminal et V non-terminal.
Par exemple, voici une grammaire lexicale :

$S \longrightarrow aST \mid bUU \mid a$
$T \longrightarrow bT \mid cSCU \mid cTU$
$U \longrightarrow aU \mid bC$
$C \longrightarrow c$

Et l'arbre de parsing de la phrase abbcabcbcacabc en utilisant cette grammaire :



On peut aussi « inverser » la grammaire pour la centrer sur les terminaux :

a ⟶ S
aST ⟶ S
aU ⟶ U
bUU ⟶ S
bT ⟶ T
bC ⟶ U
cSCU ⟶ T
cTU ⟶ T
c ⟶ C

Traduction en automate à piles :

En utilisant la grammaire inversée ci-dessus, on peut produire le tableau ci-dessous :

	S	T	U	C
a	ε		U	
	ST			
b	UU	T	C	
c	SCU			ε
	TU			

Le tableau se lit, par exemple pour la case (a, S) : on peut dépiler S si on lit a et qu'on empile ε ou T puis S.

Exemple d'exécution :

bande : abb, pile S
⟶ lit a
⟶ bande : bb, dépile S, empile ST, pile ST
⟶ lit b
⟶ bande : b, dépile S, empile UU, pile UUT
⟶ lit b
⟶ bande : #, dépile U, empile C, pile CUT

Transformation en forme normale de Greibach :

Tout d'abord, il faut noter quels sont les problèmes :

— les règles produisant ε ne sont pas tolérées (on vise une grammaire propre)

— les règles récursives à gauche ne sont pas tolérées

Voici maintenant l'algorithme pour transformer les règles récursives à gauche en règles récursives à droite :

Pour tout V_i non terminal, on introduit V_i' puis, on transforme tout règle de la forme :

$$V_j \longrightarrow V_i m_1 \mid V_i m_2 \mid \ldots \mid V_i m_p \mid w_1 \mid w_2 \mid \ldots \mid w_q$$

en

$$V_j \longrightarrow w_1 V_i' \mid w_2 V_i' \mid \ldots \mid w_q V_i'$$

et

$$V_i' \longrightarrow m_1 V_i' \mid m_2 V_i' \mid \ldots \mid m_p V_i' \mid m_1 \mid m_2 \mid \ldots \mid m_p$$

Exemple :

A ⟶ BC
B ⟶ AB | a
C ⟶ AC | b