Théorie des Langages

Elana Courtines

2022-09-13

Emmanuel Rio - emmanuel.rio@univ-tlse3.fr

1 Rappels

Écrire sous forme d'expression régulière :
- $\{a, b\}^2 \Leftrightarrow a.a + a.b + b.a + b.b$
- $L = aL + b \Leftrightarrow a^*b$ (Lemme d'Arden)
- $\lambda^n \Leftrightarrow \lambda$
- $a.(a + b) \Leftrightarrow aa + ab$

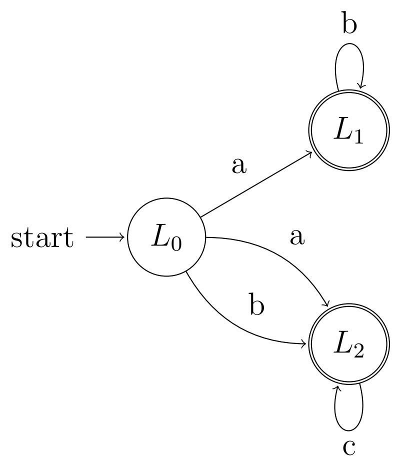
2 Analyse Lexicale

Exercice 2 :

Expression Régulière de départ :
$a.b^* + (a + b).c^*$

Ceci est un exemple d'une résolution menant à un automate non déterministe **à ne pas faire** :
$L_0 = a.L_1 + a.L_2 + b.L_2$
$L_1 = b^* = b.L_1 + \lambda$
$L_2 = c^* = c.L_2 + \lambda$

D'où l'automate :



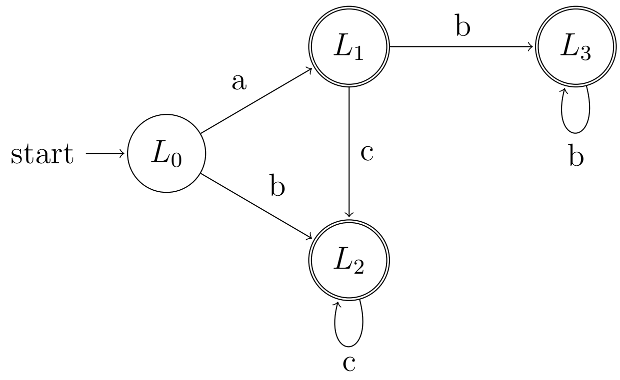
Ceci est une version plus correcte :
$L_0 = a.b* + (a+b).c*$
$L_0 = a.b* + a.c* + b.c*$
$L_0 = a.(b* + c*) + b.c*$
$L_0 = a.L_1 + b.L_2$

$L_1 = b* + c*$
$L_1 = b.b* + \lambda + c.c* + \lambda$
$L_1 = b.L3 + c.L2 + \lambda$

$L_2 = c* = c.L_2 + \lambda$

$L_3 = b* = b.L_3 + \lambda$

D'où l'automate :

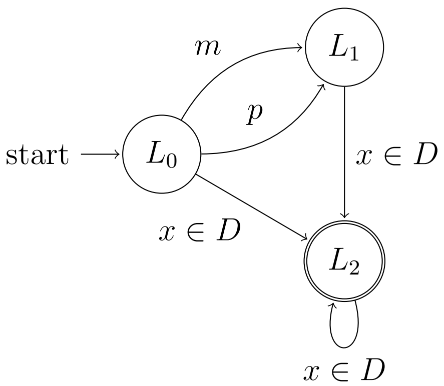


Exercice 3 :
Lexique :
- p = positif
- m = négatif
- D = ensemble des Digits
- $x \in D$ = un digit

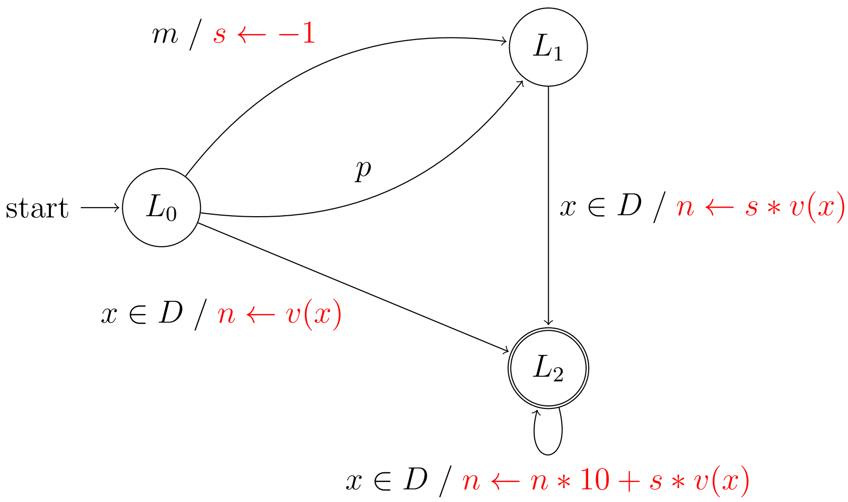
Une Expression régulière pour représenter les entiers signés serait :
$$L_0 = (p + n + \lambda).D.D^* \\ = p.D.D^* + n.D.D^* + D.D^* \\ = p.L_1 + n.L_1 + D.L_2$$

$$L_1 = D.D^* = D.L_2$$

$$L_2 = D^* = D.D^* + \lambda \\ = D.L_2 + \lambda$$



Ajoutons maintenant les actions :
- *let* $n \leftarrow 0$ (entier)
- *let* $s \leftarrow 1$ (signe)



3 Analyse Syntaxique

Exercice 1 :

Soit la grammaire G_1 définie comme suit :
(0) - $S' \rightarrow S\ \$^k$
(1) - $S \rightarrow a\ S\ b\ S$
(1) - $S \rightarrow \lambda$

Avant même de vérifier si une grammaire est $LL(1)$, on peut essayer de regarder si elle est $LL(0)$.

Rappel :
Une grammaire $LL(0)$ est une grammaire pour laquelle il n'y a aucun choix sur les dérivations, par exemple :
- $S' \rightarrow S\$$
- $S \rightarrow aA$
- $A \rightarrow bB$
- $B \rightarrow \lambda$
Formera obligatoirement et uniquement le mot $\{ab\$\}$.

Pour en revenir sur G_1, il y a deux règles de productions pour S donc celle-ci n'est pas $LL(0)$.

G_1 est-elle $LL(1)$? Pour cela, on teste les 1-lookahead de toutes les règles de production afin de déterminer si il y a des intersections non vides.

1-lookahead($S' \rightarrow S\$$)
$= first_1(S\$\ \cancel{follow_1(S')})$
$follow_1$ est inutile ici car on a la garantie de tomber sur \$.
$= first_1(aSbS\$\ \cancel{follow_1(S')})\ \cup\ first_1(\lambda\$\ \cancel{follow_1(S')})$
$= \{a\} \cup \{\$\}$
$= \{a, \$\}$

1-lookahead($S \rightarrow aSbS$)
$= first_1(aSbS\ \ \cancel{follow_1(S)})$
$= first_1(aSbS)$
$= \{a\} = I_1$

1-lookahead($S \rightarrow \lambda$)
$= first_1(\lambda\ follow_1(S))$
$= follow_1(S)$
$= first_1(\$\ \cancel{follow_1(S')})$ ← ce qui suit S dans la partie droite de la règle (0)
$\cup\ first_1(bS\ \cancel{follow_1(S)})$ ← ce qui suit le premier S dans la partie droite de règle (1)
$\cup\ first_1(follow_1(S))$ ← ce qui suit le deuxième S dans la partie droite de règle (1)
$= \{\$\} \cup \{b\} \cup \cancel{follow_1(S')}$ ← point fixe, on peut donc l'ignorer.
$= \{\$, b\} = I_2$

Comme $I_1 \cap I_2 = \varnothing$, on peut affirmer que G_1 est $LL(1)$.

Pour en extraire la table, il suffit de suivre ces étapes :

1. Ajouter une colone pour chaque terminal :

a	b	$

2. Ajouter une ligne pour chaque non-terminal :

	a	b	$
S'			
S			

3. Pour chaque $1 - lookahead$, vérifier quel(s) terminal(aux) est(sont) obtenu(s), et ajouter la règle correspondante dans la case sur la ligne du non-terminal associé pour tous ces terminaux.

- 1–$lookahead(S' \to S\$) = \{a, \$\}$, il faut donc ajouter la règle (0) pour les terminaux a et $ sur la ligne du non-terminal S' :

	a	b	$
S'	S$(0)	err	S$(0)
S			

- 1–$lookahead(S \to aSbS) = \{a\}$, il faut donc ajouter la règle (1) pour le terminal a sur la ligne du non-terminal S :

	a	b	$
S'	S$(0)	err	S$(0)
S	aSbS(1)		

- 1–$lookahead(S \to \lambda) = \{b, \$\}$, il faut donc ajouter la règle (2) pour les terminaux b et $ sur la ligne du non-terminal S :

	a	b	$
S'	S$(0)	err	S$(0)
S	aSbS(1)	λ(2)	λ(2)