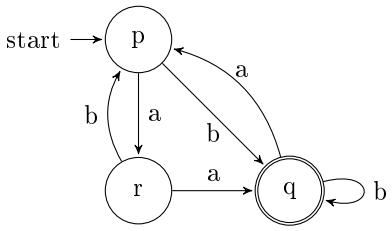
ENSIIE - Cours Langages et Systèmes Formels 1

TD 5

Frédéric Blanqui

Exercice 17 Calculez une expression régulière pour le langage reconnu par l'automate suivant:



Solution. Les langages associés à chaque état vérifient les équations suivantes:

$$\begin{cases} P &= aR + bQ \\ Q &= aP + bQ + \varepsilon \\ R &= aQ + bP \end{cases}$$

En appliquant le lemme d'Arden sur la 2e équation ($\varepsilon \notin b$), on obtient $Q = b^*(aP + \varepsilon) = b^*aP + b^*$. Ainsi, $R = aQ + bP = (ab^*a + b)P + ab^*$. Donc, $P = aR + bQ = XP + Y$ où $X = a^2b^*a + ab + b^+a$ et $Y = a^2b^* + b^+$. En appliquant à nouveau le lemme d'Arden ($\varepsilon \notin X$), on obtient $P = X^*Y$. ■

Exercice 20 Etant donné un mot $u = a_1 \dots a_n$ de longueur n, soit $u^R = a_n \dots a_1$ le mot miroir de u. Etant donné un langage L, soit $L^R = \{l^R \mid l \in L\}$ le langage miroir de L fait de tous les mots miroir de L.

(a) Montrer que, si L et M sont deux langages sur Σ, alors $(LM)^R = M^R L^R$, où $LM = \{lm \in \Sigma^* \mid l \in L, m \in M\}$.

(b) Montrer que, si L est un langage, alors $(L^*)^R = (L^R)^*$.

(c) Définir une fonction permettant de transformer une expression régulière e décrivant un langage L en une expression régulière e^R décrivant le langage L^R. Montrer que le langage décrit par e^R est bien égal à L^R.

(d) En déduire que L est régulier si et seulement si L^R est régulier.

Solution.

(a) Montrons que $(LM)^R \subseteq M^R L^R$. Soit $u \in (LM)^R$. Alors, $u = (lm)^R$ avec $l \in L$ et $m \in M$. Comme $(lm)^R = m^R l^R$, $u \in M^R L^R$. Montrons maintenant que $M^R L^R \subseteq (LM)^R$. Soit $u \in M^R L^R$. Alors, $u = m^R l^R$ avec $m \in M$ et $l \in L$. Comme $m^R l^R = (lm)^R$, $u \in (LM)^R$.

(b) Montrons que $(L^*)^R \subseteq (L^R)^*$. Soit $u \in (L^*)^R$. Alors, $u = (u_1 \dots u_n)^R$ avec $u_i \in L$. Comme $(u_1 \dots u_n)^R = u_n^R \dots u_1^R$, $u \in (L^R)^*$. Montrons maintenant que $(L^R)^* \subseteq (L^*)^R$. Soit $u \in (L^R)^*$. Alors, $u = u_1^R \dots u_n^R$ avec $u_i \in L$. Comme $u_1^R \dots u_n^R = (u_n \dots u_1)^R$, $u \in (L^*)^R$.

(c) L'ensemble des expressions régulières sur Σ est le plus petit ensemble $\mathcal{E}$ tel que:

- $\emptyset \cup \Sigma \subseteq \mathcal{E}$,
- si $e_1, e_2 \in \mathcal{E}$ alors $e_1 e_2, e_1 + e_2 \in \mathcal{E}$,
- si $e \in \mathcal{E}$ alors $e^* \in \mathcal{E}$.

On définit $e \mapsto e^R$ par récurrence sur e ainsi:

- $\emptyset^R = \emptyset$,
- si $a \in \Sigma$ alors $a^R = a$,
- $(e_1 e_2)^R = e_2^R e_1^R$
- $(e_1 + e_2)^R = e_1^R + e_2^R$
- $(e^*)^R = (e^R)^*$

Soit $L(e)$ le langage décrit par e. Montrons que $L(e^R) = L(e)^R$ par récurrence sur e:

- $L(\emptyset^R) = L(\emptyset) = \emptyset = L(\emptyset)^R$.
- $L(a^R) = L(a) = L(a)^R$.
- $L((e_1 e_2)^R) = L(e_2^R e_1^R) = L(e_2^R) L(e_1^R)$. Par hypothèse de récurrence, $L(e_i^R) = L(e_i)^R$. Donc $L((e_1 e_2)^R) = L(e_2)^R L(e_1)^R = (L(e_1) L(e_2))^R = L(e_1 e_2)^R$.
- $L((e^*)^R) = L((e^R)^*) = L(e^R)^*$. Par hypothèse de récurrence, $L(e^R) = L(e)^R$. Donc, $L((e^*)^R) = (L(e)^R)^* = (L(e)^*)^R = L(e^*)^R$.

(d) Un langage L est régulier si et seulement s'il existe une expression régulière e telle que $L = L(e)$. Ainsi, si L est régulier, alors $L = L(e)$ où e est une expression régulière. Donc, $L^R = L(e^R)$ est régulier. Réciproquement, si L^R est régulier, alors $(L^R)^R = L$ est régulier. ■

Exercice 53 Donnez un algorithme prenant en entrée une expression régulière e sur un alphabet Σ et retournant OUI si $L(e) = \Sigma^*$ et NON sinon, où $L(e)$ est le langage décrit par e.

Solution. A chaque construction d'expression régulière, on peut associer un automate avec ε-transitions avec un état initial unique et état final unique. Ainsi, en procédant récursivement, on peut construire un automate avec ε-transitions pour e. Ensuite, on élimine les ε-transitions, on déterminise, on

complète et on minimise. Si l'automate obtenu est l'automate avec un seul état initial et final q et $q \xrightarrow{a} q$ pour tout $a \in \Sigma$ comme transitions, alors $L(e) = \Sigma^*$ et on retourne OUI. Sinon, $L(e) \neq \Sigma^*$ et on retourne NON.

Remarque: on peut éviter la minimisation en observant qu'un automate déterministe complet accepte tous les mots si et seulement si tous les états accessibles sont finaux. ■

Exercice 52 Dans les expressions arithmétiques en notation préfixée (ou polonaise) on met le symbole de l'opération avant les deux opérandes et on ne met pas de parenthèses. Par exemple, au lieu de $2 + 3$ on écrit $+23$, au lieu de $(1 + 2) \times (3 + 4)$ on écrit $\times + 12 + 34$.

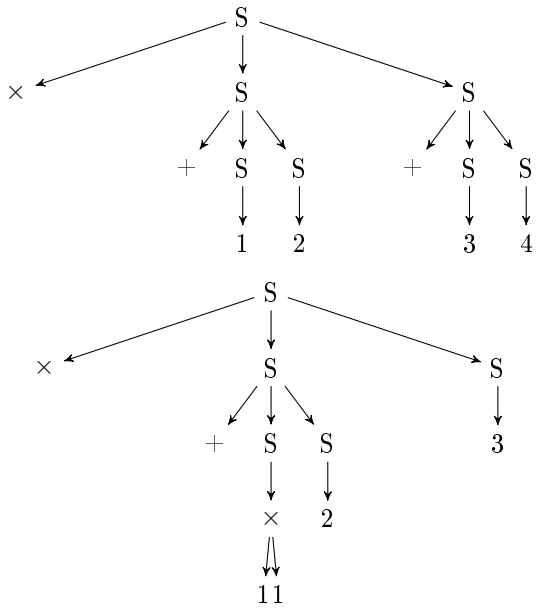
(a) Définir une grammaire algébrique pour les expressions aithmétiques en notation polonaise.

(b) Trouver les arbres de dérivation pour $\times + 12 + 34$ et $\times + \times 1123$.

Solution.

(a) $S \to 0 \mid \ldots \mid 9 \mid +SS \mid \times SS$

(b)



■

Exercice 58 Soit G la grammaire sur $\Sigma = \{(, a,)\}$ dont le non-terminal de départ est L et les règles sont:
$L \to (S)|a$
$S \to LS|\varepsilon$.
Donnez la table de transition permettant de reconnaître $L(G)$ par une machine à pile. Ce language est-il $LL(k)$?

Solution. Pour chaque règle $X \to r$, on calcule $Pre_1(rSuiv_1(X))$:

- $L \to (S)$. $Pre_1((S)Suiv_1(L)) = \{(\}$.
- $L \to a$. $Pre_1(aSuiv_1(L)) = \{a\}$.
- $S \to LS$. $Pre_1(LSSuiv_1(S)) = Pre_1(L) = \{(, a\}$.
- $S \to \varepsilon$. $Pre_1(Suiv_1(S)) = \{)\}$.

On obtient ainsi la table:

	(	a	)	ε
L	(S)	a		
S	LS	LS	ε	

Les cases vides sont des cases d'erreur.
Comme il n'y a qu'une règle par case, la grammaire est $LL(1)$. ■

Exercice 18 Soit $G = (\Sigma, \mathcal{N}, \mathcal{R}, I)$ la grammaire telle que:

- $\Sigma = \{\mathsf{if}, \mathsf{then}, \mathsf{else}, \mathsf{instr}, \mathsf{cond}\}$,
- $\mathcal{N} = \{I\}$,
- $\mathcal{R} = \{I \to \mathsf{instr} \mid \mathsf{if\ cond\ then}\ I \mid \mathsf{if\ cond\ then}\ I\ \mathsf{else}\ I\}$.

Questions:

1. G est-elle $LL(k)$ pour un certain k?
2. Donnez une grammaire $LL(1)$ G' équivalente à G.
3. Donnez la table de transition permettant de reconnaître $L(G')$ à l'aide d'une machine à pile.

Solution. Soit $a = \mathsf{instr}$, $b = \mathsf{if\ cond\ then}$ and $c = \mathsf{else}$. G est équivalente à $\{I \to a \mid bI \mid bIcI\}$.

1. Le problème vient du fait que G n'est pas factorisée à gauche. Pour tout k, nous avons $I \to^k b^k I$. Ainsi, $I \to bI \to^k b^{k+1} I$ et $I \to bIcI \to^k b^{k+1} IcI$. Donc, G n'est $LL(k)$ pour aucun k.

2. Soit alors $G' = (\Sigma, \{I, J\}, \mathcal{R}', I)$ où $\mathcal{R}' = \{I \to a \mid bIJ, J \to \varepsilon \mid cI\}$.

   - $L(G) \subseteq L(G')$. Car $\to_G \subseteq \to_{G'}^+$. En effet, $I \to_{G'} a$, $I \to_{G'} bIJ \to_{G'} bI$ et $I \to_{G'} bIJ \to_{G'} bIcI$.
   - $L(G') \subseteq L(G)$. Montrons que, pour tout k et tout $u \in \Sigma^*$, si $I \to_{G'}^k u$, alors $I \to_G^* u$, par induction sur k. On procède par cas sur la première règle:
     - $I \to_{G'} a$. Alors $u = a$ et $I \to_G u$.
     - $I \to_{G'} bIJ$. Alors $u = bvw$ avec $I \to_{G'}^l v$, $J \to_{G'}^m w$ et $l + m < k$. Par hypothèse d'induction, $I \to_G^* v$ et $I \to_G^* w$. Si la première règle utilisée pour réduire J en w est $J \to \varepsilon$, alors $w = \varepsilon$ et $I \to_G bI \to_G^* bv = u$. Sinon, $J \to_{G'} cI$ et $w = cx$ avec $I \to_{G'}^{m-1} x$. Par hypothèse d"induction, $I \to_G^* x$. Donc, $I \to_G bIcI \to_G^* bvcx = u$.

3. Pour chaque règle $X \to r$, on calcule $Pre_1(rSuiv_1(X))$:

   - $Pre_1(aSuiv_1(I)) = \{\mathsf{instr}\}$
   - $Pre_1(bIJSuiv_1(I)) = \{\mathsf{if}\}$
   - $Pre_1(\varepsilon Suiv_1(J)) = Pre_1(\widehat{Suiv}(J)) = Pre_1\{\varepsilon\} = \{\varepsilon\}$
   - $Pre_1(cISuiv_1(J)) = \{\mathsf{else}\}$

   On obtient ainsi la table de transition suivante:

	instr	if	cond	then	else	ε
I	instr	bIJ				
J					else$I$	ε

   Les cases vides sont des cases d'erreur. ■

Exercice 13 Montrez d'un langage $LL(0)$ a au plus un mot.

Solution. Soit L un langage engendré par une grammaire $LL(0)$. Supposons que u et v soient deux mots distincts de L. Alors, u et v ont deux dérivations distinctes à partir de S. C'est-à-dire, il existe deux règles distinctes $X \to r_1$ et $X \to r_2$ telles que $S \to^* \alpha X \beta$, $\alpha r_1 \beta \to^* u$ et $\alpha r_2 \beta \to^* v$. Comme la grammaire est $LL(0)$ et $u|_\varepsilon = \varepsilon = v|_\varepsilon$, $r_1 = r_2$. Contradiction. ■

Exercice 14 Montrez que tout langage régulier est $LL(1)$.

Solution. Soit L un langage régulier. Soit A un automate déterministe engendrant L. Soit alors G la grammaire linéaire à droite associée à A, c'est-à-dire avec des règles de la forme $X \to \varepsilon$ ou $X \to aY$. Alors, toute dérivation d'un mot de L est de la forme $S \to a_1 Y_1 \to a_1 a_2 Y_2 \to \ldots a_1 \ldots a_n Y_n \to a_1 \ldots a_n$. Autrement dit, si $S \to_g^* uXv$ alors $v = \varepsilon$. Supposons maintenant que $X \to \varepsilon$ et $X \to aY$ soient deux règles de G. Alors $uX \to u$ et $uX \to uaY \to_g^* uaw$ et

$\varepsilon|_1 \neq (aw)|_1 = a$. Supposons maintenant que $X \to aY$ et $X \to bZ$ soient deux règles de G. Nous avons $a \neq b$ car A est déterministe (dans ce cas, on ne peut pas avoir $X \xrightarrow{a} Y$ et $X \xrightarrow{a} Z$). Alors $uX \to uaY \to_g^* uay$ et $uX \to ubZ \to_g^* ubz$ et $(ay)|_1 = a \neq (bz)|_1 = b$. Donc, L est $LL(1)$. ■

Exercice 56 Montrez que la grammaire suivante est LL(3) mais pas LL(2).
$S \to aAaB \mid bAbB$
$A \to a \mid ab$
$B \to aB \mid a$