université PARIS DIDEROT PARIS 7

UNIVERSITÉ PARIS - DIDEROT - PARIS 7

▼

VÉRIFICATION DES APTITUDES ET DES CONNAISSANCES

DATE : .....

UV : .....

Après avoir rempli l'en-tête, rabattre et coller le coin noirci

Nom : .....

Prénom : .....

N° d'étudiant : .....

N° de place : .....

N.B. - Il est interdit aux candidats, sous peine d'exclusion, de signer leur composition ou d'y apporter un signe distinctif quelconque.

Correcteurs			
Nom : Appréciations :	NOTE	Nom : Appréciations :	NOTE

Note définitive :

Exercice 1

1) Voir poly/cours. → voir annexe

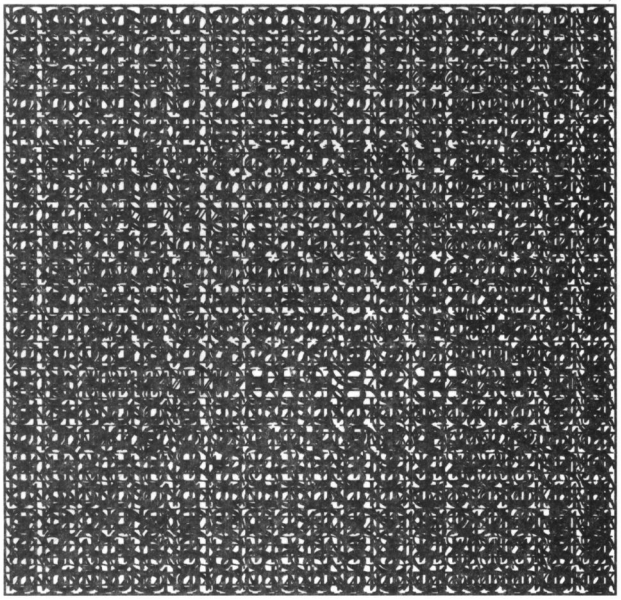
2) $P = \begin{cases} S \to XY \\ X \to aXb \mid \varepsilon \\ Y \to cY \mid c \end{cases}$ $\quad G = \langle \{a, b, c\}, \{S, X, Y\}, S, P \rangle$

3) a) En donnant une grammaire algébrique reconnaissant L.
b) En donnant un automate à pile reconnaissant L.

4) Le lemme de pompage algébrique.
Voir énoncé dans le poly/cours → Soit L un langage algébrique,
$\exists k \in \mathbb{N}, \forall w \in L$, si $|w| > k$
alors $\exists u, x, y, z, v$ tq $w = uxyzv$, $|x| + |y| \geq 1$, $|xyz| < k$
et $\forall n \in \mathbb{N}$ $ux^n y z^n v \in L$

▲

Ne rien inscrire ici



NE RIEN ÉCRIRE ICI

Exercice 2 :

5) Voir dans le cours. (Une grammaire dt tt les règles sont de la forme $A \to BC$ ou $A \to a$ $(A, B, C \in V, a \in \Sigma)$ ou $S \to \varepsilon$ si S est l'axiome et est inaccessible)

6) ——— → voir annexe

7) $G = \langle \{a, b\}, \{S, A, B, C, Y_1, Y_2, Y_3, X_a, X_b\}, S, P \rangle$

$$P = \begin{cases} S \to A Y_1 \\ Y_1 \to B Y_2 \\ Y_2 \to C X_b \\ A \to a \mid X_a A \\ B \to A X_a \\ C \to X_b Y_3 \\ Y_3 \to B X_a \\ X_a \to a \\ X_b \to b \end{cases}$$

8) $G = \langle \{a, b\}, \{S, A, A', B, B'\}, S, P \rangle$

$$P = \begin{cases} S \to AB \\ A \to aA' \mid BA' \\ A' \to BbA' \mid BA' \mid \varepsilon \\ B \to bB' \mid aA'SB' \\ B' \to A'SB' \mid \varepsilon \end{cases}$$

(Il faut utiliser la méthode de dérécursivisation générale, qui fait notamment intervenir la formule de dérécursivisation directe)

Exercice 3 :

9) a) Le langage reconnu est $\{a^n b^n c \mid n \geq 0\}$

b) C'est une grammaire algébrique.
(La nature de la grammaire vient de la forme de ses règles.)

10) a) Premier(S) = $\{a, c\}$

—— X = $\{a, \varepsilon\}$

—— Y = $\{c\}$

Suivant(S) = $\{\$\}$

—— X = $\{b, c\}$

—— Y = $\{\$\}$

Table LL(1) :

	a	b	c	$
S →	XY		XY	
X →	aXb	ε	ε	
Y →			c	

b) La grammaire est LL(1) car il y a au plus une règle par case dans la table LL(1).

c) Une grammaire LL(1) est très pratique pour faire du parsing descendant car il suffit de regarder la première lettre pas encore analysée pour savoir quelle règle appliquer.
L'arbre d'exploration ne contient, si l'on utilise cette information, qu'une seule branche.

11) a) On part de (S, ω) pour arriver à $(\emptyset, \emptyset)$.

b) —— $(\emptyset, \omega)$ —— $(S, \emptyset)$.

12) a) $(S, aaabbbc) \xrightarrow{S \to XY} (XY, aaabbbc) \xrightarrow{X \to aXb} (aXbY, aaabbbc)$
$\xrightarrow{X \to aXb} (aXbbY, aabbbc) \xrightarrow{X \to aXb} (aXbbbY, abbbc) \xrightarrow{X \to \varepsilon} (bbbY, bbc)$
$\xrightarrow{Y \to c} (\varepsilon, \emptyset)$
$= (\emptyset, \emptyset)$

β) $(\emptyset, aaa\beta\beta c) \xrightarrow{S} (a, aa\beta\beta c) \xrightarrow{S} (aa, a\beta\beta c)$
$\xrightarrow{S} (aaa, \beta\beta c) \xrightarrow{R(X\to\varepsilon)} (aaaX, \beta\beta c) \xrightarrow{S} (aaaX\beta, \beta c)$
$\xrightarrow{R(X\to aX\beta)} (aaX, \beta\beta c) \xrightarrow{S} (aaX\beta, \beta c) \xrightarrow{R(X\to aX\beta)} (aX, \beta c)$
$\xrightarrow{S} (aX\beta, c) \xrightarrow{R(X\to a\beta)} (X, c) \xrightarrow{S} (X\varsigma, \emptyset) \xrightarrow{R(X\to c)} (X\gamma, \emptyset)$
$\xrightarrow{R(S\to X\gamma)} (S, \emptyset)$

Exercice 4 :

Voir poly / cours

Une grammaire est propre si - elle est ε-libre
- elle ne contient pas de symbole inutile
- elle ne contient pas de production singulière.

En gros : pour $G = \langle \Sigma, V, S, P \rangle$
- on commence par rendre la grammaire ε-libre :
  on calcule (méthode de point fixe) $\mathcal{A} = \{A \in V / A \xrightarrow{*} \varepsilon\}$
  on part d'un ensemble de règles vide P'
  $\forall r \in P$ on ajoute dans P' toutes les possibilités de règle construite à partir de r en remplaçant ou non dans la partie droite les $A \in \mathcal{A}$ par ε.
  Si $S \to \varepsilon \in P$, on crée un nouveau non-terminal S', axiome de notre nouvelle grammaire, et on ajoute $S' \to S$ et $S' \to \varepsilon$ à P'
- on trouve via une méthode de point fixe tous les symboles ~~inutil~~ inutiles, on les supprime de la nouvelle grammaire.
- On se débarrasse des productions singulières :
  on calcule $\mathcal{B} = \{(A, B) \in V^2 / A \xrightarrow{*} B\}$
  on supprime toutes les productions singulières de P'
  $\forall (A, B) \in \mathcal{B}, \forall B \to \alpha \in P'$ on ajoute $A \to \alpha$ à P'

université PARIS DIDEROT PARIS 7

rappel n° de place :

Algorithme de mise en FNC :

Pour $G = \langle \Sigma, V, S, P \rangle$ une grammaire algébrique propre.

On copie V : $V' \leftarrow V$

On crée un ensemble de règles vide : $P' \leftarrow \emptyset$

- $\forall a \in \Sigma$: Soit X_a un nouveau non-terminal ($X_a \notin V'$),
  $V' \leftarrow V' \cup \{X_a\}$ (on ajoute X_a à V')
  $P' \leftarrow P' \cup \{X_a \to a\}$ (on ajoute $X_a \to a$ à P')
- $\forall r \in P$: si $r = S \to \varepsilon$
  ou $\exists A \in V, a \in \Sigma$, $r = A \to a$:
  $P' \leftarrow P' \cup \{r\}$

  Sinon :

  on constitue la règle r' construit à partir de r en remplaçant tout terminal a par le non-terminal associé X_a

  On écrit $r' = A \to B_1 B_2 \dots B_k$ $(A, B_1, \dots, B_k \in V')$

  Si $k = 2$: $P' \leftarrow P' \cup \{r'\}$

  Sinon : $V' \leftarrow V' \cup \{Y_1, Y_2, \dots, Y_{k-2}\}$ (nouveaux non-terminaux)

  $P' \leftarrow P' \cup \{A \to B_1 Y_1,\ Y_1 \to B_2 Y_2, \dots, Y_{k-3} \to B_{k-2} Y_{k-2},\ Y_{k-2} \to B_{k-1} B_k\}$

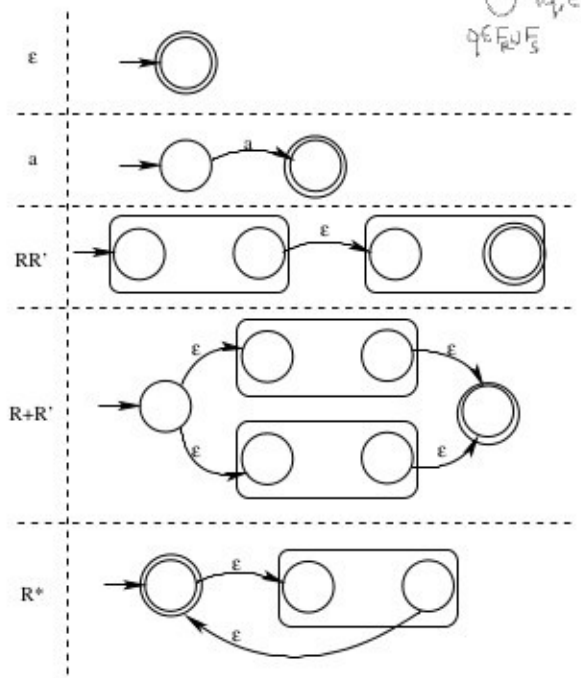
On renvoie $G' = \langle \Sigma, V', S, P' \rangle$

Traduction récursive d'une expression rationnelle en un automate

1. Au mot vide ε, on associe l'automate $\langle X, \{q_0\}, \{q_0\}, \{q_0\}, \emptyset\rangle$
2. À l'expression rationnelle x ($x \in X$), on associe l'automate
$$\langle X, \{q_0, q_1\}, \{q_0\}, \{q_1\}, \{(q_0, x, q_1)\}\rangle$$
3. Soit R une expression rationnelle, associée à l'automate $\langle X, Q_R, I_R, F_R, \delta_R\rangle$; à R^*, on associe l'automate $\langle X, Q_R \cup \{Q_0\}, \{Q_0\}, \{Q_0\}, \delta'_R\rangle$[5], où $\delta'_R = \delta_R \cup \bigcup_{q\in I_R}(Q_0, \varepsilon, q) \cup \bigcup_{q\in F_R}(q, \varepsilon, Q_0)$
4. Soient R et S deux expressions rationnelles auxquelles ont été associés respectivement $\langle X, Q_R, I_R, F_R, \delta_R\rangle$ et $\langle X, Q_S, I_S, F_S, \delta_S\rangle$, dont on suppose que tous les états sont distincts ($Q_S \cap Q_R = \emptyset$).

   (a) À RS on associe l'automate
$$\left\langle X, Q_R \cup Q_S, I_R, F_S, \delta_R \cup \delta_S \cup \bigcup_{q\in F_R}\bigcup_{q'\in I_S}(q, \varepsilon, q')\right\rangle$$

   (b) À $R|S$ on associe l'automate
$$\left\langle X, Q_R \cup Q_S \cup \{Q_0, Q_F\}, Q_0, Q_F, \delta_R \cup \delta_S \cup \bigcup_{q\in I_R\cup I_S}(Q_0, \varepsilon, q) \cup \bigcup_{q\in F_R\cup F_S}(q, \varepsilon, Q_F)\right\rangle$$



5. Q_0 est un nouvel état t.q. $Q_0 \notin Q$.