Contrôle continu - Compilation (L3 Info)
Durée : 2h - documents interdits

Conventions

- Axiome : symbole non terminal de la partie gauche de la première production de la grammaire
- Symboles non terminaux : lettres *MAJUSCULES ITALIQUES*
- Symboles terminaux : lettres `minuscules true-type` ou caractères spéciaux simples

1 Vrai ou faux *(4 pts)*

Pour chacune des affirmations suivantes dites si elle est vraie ou fausse en justifiant brièvement et de façon convaincante votre réponse (seules les réponses accompagnées d'une justification seront prises en compte).

Q1.1 Les grammaires hors-contexte ambiguës ne peuvent pas être reconnues par un automate à pile. Faux

Q1.2 Le langage généré par la grammaire $\langle\{S, X\}, \{\mathtt{a}, \mathtt{b}\}, \{S \to X, X \to \mathtt{a}, X \to \mathtt{b}\}, S\rangle$ peut être reconnu par un automate à pile déterministe. Vrai

Q1.3 Il est possible qu'un même mot possède deux dérivations gauches différentes avec une grammaire ambiguë. Vrai

Q1.4 Seules les grammaires récursives peuvent générer des langages infinis. Vrai

2 Écriture de grammaire *(6 pts)*

MaTheX[1] est un langage permettant d'écrire des jolies équations mathématiques avec des commandes spécifiques. Par exemple, la commande `x exp 2 ind [i + 1] + y ind [n ind 2] + z ind i exp 3 exp 2` permet d'afficher l'expression $x_{i+1}^{2} + y_{n_2} + z_i^{32}$. Vous devez écrire une grammaire hors contexte non ambiguë qui reconnaît les expressions en MaTheX. Nous nous limitons au sous-ensemble d'opérateurs suivants :

1. Somme (`+`), comme habituellement dans une expression mathématique
2. Indices (en bas), signalés par l'opérateur `ind`, par exemple, `x ind i` donne x_i
3. Exposants (en haut), signalés par l'opérateur `exp`, par exemple, `y exp 2` donne y^2

Les indices et exposants sont prioritaires sur la somme, et tous les opérateurs sont associatifs à gauche. Les crochets `[` et `]` modifient la priorité (comme pour les parenthèses dans les expressions mathématiques).

Les éléments atomiques (simples) du langage sont des lettres isolées (`lt`) et des nombres entiers (`nb`) reconnus par l'analyseur lexical. Utilisez les terminaux `lt` et `nb` pour représenter ces éléments dans votre grammaire.

Q2.1 Écrivez une grammaire non ambiguë qui reconnaît des expressions formées par des sommes d'éléments atomiques (lettres et nombres). Ignorez, pour le moment, les exposants et les indices, ainsi que les crochets. (2pt)

$$\begin{array}{lcl} E & \to & E \ \mathtt{+} \ V \mid V \\ V & \to & \mathtt{nb} \mid \mathtt{lt} \end{array}$$

1. Il s'agit en réalité d'une adaptation fictive d'une sous-partie du langage LaTeX, avec une syntaxe plus explicite.

Q2.2 Écrivez une grammaire non ambiguë qui reconnait des élements atomiques isolés avec des indices et des exposants ~~(et des indices d'indices, des exposants d'indices, etc.)~~. Ignorez, pour le moment, la somme et les crochets.

(2pt)

$$\begin{array}{lcl} T & \rightarrow & \texttt{nb}\ T' \mid \texttt{lt}\ T' \\ T' & \rightarrow & \texttt{ind}\ T \mid \texttt{exp}\ T \mid \varepsilon \end{array}$$

Alternativement, plus simple :

$$\begin{array}{lcl} T & \rightarrow & T\ \texttt{ind}\ V \mid T\ \texttt{exp}\ V \mid V \\ V & \rightarrow & \texttt{nb} \mid \texttt{lt} \end{array}$$

Q2.3 Écrivez la grammaire non ambiguë complète qui mélange les deux grammaires ci-dessus et qui reconnaît des expressions avec des sommes, des exposants, des indices et des crochets combinés récursivement **(indices d'indices, exposants d'indices, etc.).** (2pt)

$$\begin{array}{lcl} E & \rightarrow & E\ +\ T \mid T \\ T & \rightarrow & \texttt{nb}\ T' \mid \texttt{lt}\ T' \\ T' & \rightarrow & \texttt{ind}\ T \mid \texttt{exp}\ T \mid \texttt{ind}\ [\ E\ ]\ T' \mid \texttt{exp}\ [\ E\ ]\ T' \mid \varepsilon \end{array}$$

Alternativement, plus simple (pas tout à fait équivalent, mais la spécification est suffisamment vague pour admettre cette interprétation) :

$$\begin{array}{lcl} E & \rightarrow & E\ +\ T \mid T \\ T & \rightarrow & T\ \texttt{ind}\ V \mid T\ \texttt{exp}\ V \mid V \\ V & \rightarrow & \texttt{nb} \mid \texttt{lt} \mid [\ E\ ] \end{array}$$

3 Premier, Suivant, dérivation, ambiguïté *(4 pts)*

Soit la grammaire G_1 :

$$\begin{array}{lcl} S & \rightarrow & E\ ;\ L \mid E\ ; \\ L & \rightarrow & E\ ;\ L \mid \varepsilon \\ E & \rightarrow & \langle\ M\ \rangle \\ M & \rightarrow & \texttt{0}\ M \mid \texttt{1}\ M \mid \texttt{0} \mid \texttt{1} \end{array}$$

Q3.1 Calculez Premier et Suivant pour les symboles non terminaux de G_1.

(2pt)

	Premier	Suivant
S	$\langle$	$\perp$
L	$\langle\ \varepsilon$	$\perp$
E	$\langle$	;
M	0 1	$\rangle$

Q3.2 Donnez une dérivation gauche du mot $\langle\texttt{01}\rangle;\langle\texttt{1}\rangle;$. (1pt)

$S \Rightarrow E;L \Rightarrow \langle M\rangle;L \Rightarrow \langle 0M\rangle;L \Rightarrow \langle 01\rangle;L \Rightarrow \langle 01\rangle;E;L \Rightarrow \langle 01\rangle;\langle M\rangle;L \Rightarrow \langle 01\rangle;\langle 1\rangle;L \Rightarrow \langle 01\rangle;\langle 1\rangle;$

Q3.3 Démontrez que la grammaire G_1 est ambiguë. (1pt)

Deux arbres de dérivations différents pour E; :

1. $S \Rightarrow E$;
2. $S \Rightarrow E; L \Rightarrow E$;

4 Création de la table SLR *(4 pts)*

Soit la grammaire G_2 :

1	A	$\rightarrow$	$B\ A$
2	A	$\rightarrow$	ε
3	B	$\rightarrow$	$C\ D$
4	C	$\rightarrow$	`a`
5	C	$\rightarrow$	`b`
6	D	$\rightarrow$	`c`
7	D	$\rightarrow$	ε

Q4.1 Décrivez en une phrase ou deux ou en notation mathématique le langage engendré par G_2 (1pt)

$L(G_2)$ est le langage régulier correspondant à l'expression régulière $((a+b)(c+\varepsilon))*$

En notation Linux `([ab]c?)*`

Q4.2 Construisez l'automate $LR(0)$ de G_2 à l'aide des fonctions FERMETURE et ALLER_À (2pt)

$I_0 = \{S \rightarrow \bullet\ A,\ A \rightarrow \bullet\ B\ A,\ A \rightarrow \bullet,\ B \rightarrow \bullet\ C\ D,\ C \rightarrow \bullet\ \mathtt{a},\ C \rightarrow \bullet\ \mathtt{b}\}$

$I_1 = \{S \rightarrow\ A\ \bullet\}$

$I_2 = \{A \rightarrow\ B\ \bullet\ A,\ A \rightarrow \bullet,\ B \rightarrow \bullet\ C\ D,\ C \rightarrow \bullet\ \mathtt{a},\ C \rightarrow \bullet\ \mathtt{b}\}$

$I_3 = \{B \rightarrow\ C\ \bullet\ D, D \rightarrow \bullet\ \mathtt{c}, D \rightarrow \bullet\}$

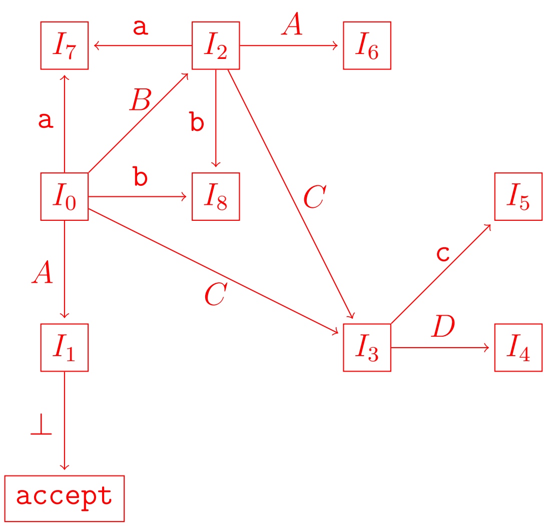
$I_4 = \{B \rightarrow\ C\ D\ \bullet\}$

$I_5 = \{D \rightarrow\ \mathtt{c}\ \bullet\}$

$I_6 = \{A \rightarrow\ B\ A\ \bullet\}$

$I_7 = \{C \rightarrow\ \mathtt{a}\ \bullet\}$

$I_8 = \{C \rightarrow\ \mathtt{b}\ \bullet\}$



Q4.3 Étant donné les ensembles SUIVANT de G_2 ci-dessous, construisez la table SLR de G_2

SUIVANT(A) = $\{\perp\}$
SUIVANT(B) = {`a`, `b`, $\perp$}
SUIVANT(C) = {`a` , `b`, `c` $\perp$}
SUIVANT(D) = {`a`, `b`, $\perp$}

(1pt)

	ACTION				GOTO			
	`a`	`b`	`c`	$\perp$	A	B	C	D
0	d7	d8		r2	1	2	3	
1				acc				
2	d7	d8		r2	6	2	3	
3	r7	r7	d5	r7				4
4	r3	r3		r3				
5	r6	r6		r6				
6				r1				
7	r4	r4	r4	r4				
8	r5	r5	r5	r5				

5 Analyse SLR *(2 pts)*

Soit la grammaire G_3 ci-dessous et sa table SLR correspondante :

1	N	$\rightarrow$	$S\ E\ D$
2	S	$\rightarrow$	`-`
3	S	$\rightarrow$	ε
4	E	$\rightarrow$	`c` E
5	E	$\rightarrow$	`c`
6	D	$\rightarrow$	`:` E

	`-`	`c`	`:`	`$`	N	S	E	D
0	s1	r3			2	3		
1		r2						
2				acc				
3		s4					5	
4		s4	r5	r5			6	
5			s7					8
6			r4	r4				
7		s4					9	
8				r1				
9				r6				

Q5.1 Simulez l'analyse LR du mot `-c:c`. Les configurations sont représentées par le triplet (α, w, y), où α est la pile avec le sommet à droite, w est la partie de la bande de lecture qui commence au caractère sous la tête de lecture (ce qui reste à analyser), et y est la séquence de symboles de sortie (numéros des productions appliquées lors des réductions). (2pt)

($-0,`-c:c$`,$\varepsilon$) $\vdash$
($-0-1,`c:c$`,$\varepsilon$) $\vdash$
($-0-3,`c:c$`,2) $\vdash$
($-0-3-4,`:c$`,2) $\vdash$
($-0-3-5,`:c$`,2-5) $\vdash$
($-0-3-5-7,`c$`,2-5) $\vdash$
($-0-3-5-7-4,`$`,2-5) $\vdash$
($-0-3-5-7-9,`$`,2-5-5) $\vdash$
($-0-3-5-8,`$`,2-5-5-6) $\vdash$
($-0-2,`$`,2-5-5-6-1) $\vdash$
($-0-2,`$`,2-5-5-6-1-acc)