Analyse syntaxique automatique

LI035, 2ème semestre 2000/2001

Laura Kallmeyer

Solutions du Partiel du 7.6.2001

1. Parsing top-down, bottom-up et coin-gauche

   (a) i. top-down de gauche à droite : 1, 2, 3, 4, 5, 6, 7

   ii. bottom-up de gauche à droite : 3, 4, 2, 6, 7, 5, 1

   iii. coin-gauche : 3, 2, 4, 1, 6, 5, 7

   (b) $L(G_1) = L(G_2) = \{\mathrm{a}^n\mathrm{b}^m \mid n, m \geq 1\}$

   i. G_1 : top-down et coin gauche peuvent être utilisés parce que G_1 ne contient ni des récursions à gauche ni des cycles ni des productions $A \rightarrow \epsilon$.
   G_2 : top-down ne peut pas être utilisé parce que G_2 contient des récursions à gauche : A → A a. Coin gauche est possible parce qu'il n'y a ni des cycles ni des productions $A \rightarrow \epsilon$.

   ii. top-down : pas déterministe pour G_1 parce qu'il existe plusieurs productions avec les mêmes parties gauches : A → a A, A → a.
   coin gauche : ni pour G_1 ni pour G_2 déterministe : dans G_1, A → a A et A → a ont les mêmes coins gauches, dans G_2 S → A B et A → A a ont les mêmes coins gauches.

2. La relation $FIRST$.

   (a) {a}

   (b) {b, ϵ}

   (c) {a, b, c, ϵ}

   (d) {a, b, c}

3. Lookahead dans l'algorithme de Earley :

   (a) Il faut modifier `closure`. Nouvelle version de `closure` :

Procédure closure

Données globales : chart $chart$
Input : i, j, production divisée $A \rightarrow \alpha.\beta$
Output : -
Effet supplémentaire : modification de $chart$
Méthode :
  if $A \rightarrow \alpha.\beta \notin chart_{i,j} \wedge (FIRST(\beta) \cap \{t \mid \langle w_{j+1}, t\rangle \in D\} \neq \emptyset \vee \epsilon \in FIRST(\beta))$
  then
    $chart_{i,j} := chart_{i,j} \cup \{A \rightarrow \alpha.\beta\}$
    if $\beta = \epsilon$
    then complete(i, j, $A \rightarrow \alpha.\beta$)
    else expand(i, j, $A \rightarrow \alpha.\beta$)

   (b) i. l'algorithme Earley-2 sans lookahead :

	arc	procédure
1	NP $\rightarrow$. det n $\in chart_{0,0}$	
2	NP $\rightarrow$. det AP n $\in chart_{0,0}$	
3	NP $\rightarrow$. n $\in chart_{0,0}$	
4	det $\rightarrow$ le. $\in chart_{0,1}$	shift
5	NP $\rightarrow$ det . n $\in chart_{0,1}$	complete
6	NP $\rightarrow$ det . AP n $\in chart_{0,1}$	complete
7	AP $\rightarrow$. a $\in chart_{1,1}$	expand
8	AP $\rightarrow$. a AP $\in chart_{1,1}$	expand
9	a $\rightarrow$ petit . $\in chart_{1,2}$	shift
10	AP $\rightarrow$ a . $\in chart_{1,2}$	complete
11	NP $\rightarrow$ det AP . n $\in chart_{0,2}$	complete
12	AP $\rightarrow$ a . AP $\in chart_{1,2}$	complete
13	AP $\rightarrow$. a $\in chart_{2,2}$	expand
14	AP $\rightarrow$. a AP $\in chart_{2,2}$	expand
15	n $\rightarrow$ chien . $\in chart_{2,3}$	shift
16	NP $\rightarrow$ det AP n . $\in chart_{0,3}$	complete

ii. l'algorithme Earley-2 avec lookahead :
Les arcs 3, 5, 12, 13 et 14 ne sont pas entrés dans le chart. Pour les autres arcs, l'ordre est le même que dans i.

4. Des grammaires LR(0)

(a) G_1 est LR(0) :
$CG(\text{NP}) = \{\epsilon\}$, $CG(\text{AP}) = \{\text{det}\}$
$CG(\text{NP} \rightarrow \text{det n}) = \{\text{det n}\}$, $CG(\text{NP} \rightarrow \text{det AP n}) = \{\text{det AP n}\}$,
$CG(\text{AP} \rightarrow \text{a}) = \{\text{det a}\}$, $CG(\text{AP} \rightarrow \text{AP a}) = \{\text{det AP a}\}$.
Aucun élément d'un contexte gauche d'une production est préfixe d'un élément du contexte gauche d'une autre production.

(b) G_2 n'est pas LR(0) :
$CG(\text{NP}) = \{\epsilon\}$, $CG(\text{AP}) = \{\text{det a}^n \,|\, n \geq 0\}$
$CG(\text{NP} \rightarrow \text{det n}) = \{\text{det n}\}$, $CG(\text{NP} \rightarrow \text{det AP n}) = \{\text{det AP n}\}$,
$CG(\text{AP} \rightarrow \text{a}) = \{\text{det a}^n \,|\, n > 0\}$, $CG(\text{AP} \rightarrow \text{a AP}) = \{\text{det a}^n \text{ AP} \,|\, n > 0\}$
Chaque det a^n, $n > 0$ dans $CG(\text{AP} \rightarrow \text{a})$ est préfixe d'un élément dans $CG(\text{AP} \rightarrow \text{a AP})$, notamment de det a^n AP.

5. Table de parsing LR pour G_2 :
Construction des états :

C_0	`closure`$(\{\langle$S'$\to$ .NP, \$$\rangle\})$
	S' $\to$. NP, \$
	NP $\to$. det n, \$
	NP $\to$. det AP n, \$
C_1	`goto-states`(C_0, NP)
	S' $\to$ NP ., \$
C_2	`goto-states`(C_0, det)
	NP $\to$ det . n, \$
	NP $\to$ det . AP n, \$
	AP $\to$. a, n
	AP $\to$. a AP, n
C_3	`goto-states`(C_2, n)
	NP $\to$ det n ., \$
C_4	`goto-states`(C_2, AP)
	NP $\to$ det AP . n, \$
C_5	`goto-states`(C_2, a)
	AP $\to$ a ., n
	AP $\to$ a . AP, n
	AP $\to$. a , n
	AP $\to$. a AP, n
C_6	`goto-states`(C_4, n)
	NP $\to$ det AP n ., \$
C_7	`goto-states`(C_5, AP)
	AP $\to$ a AP ., n

Construction de la table :

état	*action*				*goto*	
	det	a	n	\$	NP	AP
0	s2				1	
1				acc		
2		s5	s3			4
3				r1		
4			s6			
5		s5	r3			7
6				r2		
7			r4			

6. Des grammaires LR(k) :

Deux dérivations droites dans G :

S $\Rightarrow$ A B $\Rightarrow$ A b A $\Rightarrow$ A b a B $\Rightarrow$ A b a b $\Rightarrow$ a b a b

S $\Rightarrow$ A B $\Rightarrow$ A b $\Rightarrow$ a B b $\Rightarrow$ a b A b $\Rightarrow$ a b a b

L'application de la définition d'une grammaire LR(k) avec $k = 1$ donne

$\alpha = \epsilon$, X = A, w = b a b, β = a, γ = a b, Y = A, x = b et y = b a b.

$init_1(w) = init_1(y)$ =b. Mais $\alpha X y$ = A b a b $\neq$ a b A b = $\gamma Y x$. Par conséquent G n'est pas LR(1) (ni LR(2) ni LR(3)).

En fait, pour tous les k, G n'est pas LR(k).