Chapitre 6 : Analyse ascendante

6.3 Analyse SLR(1)

Nous donnerons trois méthodes qui diffèrent par leur puissance et leur facilité d'implémentation. La première appelée "Simple LR" ou SLR en abrégé, est la moins puissante des trois en termes du nombre de grammaires pour lesquelles elle réussit mais elle est la plus simple à implémenter.

Des tables d'analyse Construites par cette méthode sont appelées Tables SLR et l'analyseur est appelé analyseur SLR. Une grammaire pour laquelle il est possible de construire un analyseur SLR est appelée grammaire SLR.

Tables d'analyse SLR

On va voir comment construire les fonctions **Action** et **Successeur** à partir du DFA qui reconnaît les préfixes viables.

Algorithme Construction des Tables d'analyse SLR

Donnée : Une grammaire augmentée G'

Résultat : Les tables d'analyse SLR des fonctions Action et Successeur pour G'

Méthode :

1. Construire C = {I0,I1,…,In}, la collection des ensembles d'items LR(0)
2. L'état i est construit à partir de Ii. Les actions d'analyse pour l'état i sont déterminés comme suit:
   a) si [A →α•aβ] est dans Ii et Transition (Ii,a) = Ij, remplir Action[i,a] avec "décaler j". Ici a doit être un terminal.
   b) si [A →α•] est dans Ii, remplir Action [i,a] avec "réduire par A →α" pour tous les a de suivant (A); ici A ne doit pas être S'.
   c) si [S' →S•] est dans Ii, remplir Action[i,$] avec "accepter"

   si les règles précédentes engendrent des actions conflictuelles, nous disons que la grammaire n'est pas SLR(1). Dans ce cas, l'algorithme échoue et ne produit pas d'analyseur.
3. On construit les transitions successeur pour l'état i pour tout non terminal A en utilisant la règle: si Transition (Ii,A) = Ij, alors Successeur [i,A] = j
4. Toutes les entrées non définies par les règles (2) et (3) sont positionnées à "erreur"
5. L'état initial de l'analyseur est celui qui est construit à partir de l'ensemble d'items contenant [S' →•S].

Les tables d'analyse formées des fonctions Action et successeur déterminées par l'algorithme 3 sont appelées tables SLR(1) pour G. Un analyseur LR utilisant les tables SLR(1) pour G est appelé analyseur SLR(1) pour G et une grammaire ayant des tables d'analyse SLR(1) est

SLR(1). Nous omettons en general le (1) apres SLR, car nous ne considérerons pas ici d'analyseurs utilisant plus d'un symbole de prévision.

Exemple

Construisons les tables SLR pour la grammaire G' de l'exemple 5.1.

La collection canonique des ensembles d'items LR(0) pour G' est représentée dans l'exemple 5.1.

Considérons tout d'abord l'ensemble d'items I0:

I_0 : $E' \rightarrow \bullet E$ $E \rightarrow \bullet E+T$ $E \rightarrow \bullet T$ $T \rightarrow \bullet T*F$	l'item $F \rightarrow \bullet$ (E) produit l'entée Action[0,(] = décaler 4 et l'item $F \rightarrow \bullet$id produit l'action[0,id] =

$T \rightarrow \bullet F$ $F \rightarrow \bullet (E)$ $F \rightarrow \bullet$ id	décaler5 les autres items de I0 ne produisent aucune action.

Considérons I1

$E' \rightarrow E\bullet$ $E \rightarrow E\bullet + T$	Le premier Item produit Action [1,$] = accepter et le second item produit Action[1,+] = décaler 6.

Considérons maintenant I2

$E \rightarrow T\bullet$ $T \rightarrow T\bullet * F$	comme Suivant(E) = {$,+,)}, le premier item produit Action[2,$] = Action[2,+] = Action[2,)] = réduire par $E \rightarrow T$ Le second item produit Action [2,*] = décaler 7 (car transition (I1,*) = I7 dans le DFA)

En continuant ainsi, nous obtenons les tables Action et successeur suivantes:

	Action						Successeur		
	id	+	*	(	)	$	E	T	F
0	d5			d4			1	2	3
1		d6				Acc			
2		r2	d7		r2	r2			
3		r4	r4		r4	r4			
4	d5			d4			8	2	3
5		r6	r6		r6	r6			
6	d5			d4				9	3
7	d5			d4					10
8		d6			d11				
9		r1	d7		r1	r1			
10		r3	r3		r3	r3			
11		r5	r5		r5	r5			