Chapitre 6 : Analyse ascendante

6.1 Analyseurs LR (principe)

Cette section présente une technique efficace d'analyse syntaxique ascendante qui peut être utilisée pour analyser une large classe de grammaires non contextuelles. Cette technique est appelée analyse LR(K); "L" signifie parcours de l'entrée de gauche vers la droite" (left to right scanning of the input), "R" signifie "en construisant une dérivation droite inverse) et K indique le nombre de symboles de prévision utilisés pour prendre les décision s d'analyse. Quand (K) est omis, K est supposé être égal à un.

Après avoir présenté le fonctionnement d'un analyseur LR, nous présenterons trois techniques pour construire les tables d'analyse LR pour une grammaire.

- La première méthode appelée "simple LR" (SLR en abrégé), est la plus facile à implémenter, mais la moins puissante des trois. Pour certaines grammaires, la production des tables d'analyse peut échouer alors qu'elle réussirait avec d'autres méthodes.

- La seconde méthode, appelée LR canonique, est la plus puissante, mais elle est la plus coûteuse.

- la troisième méthode appelée "LookAheadLR" (LALR en abrégé) ou LR à prévision, a une puissance et un coût intermédiaires entre les deux autres.

6.1.1 Modèle d'analyse LR

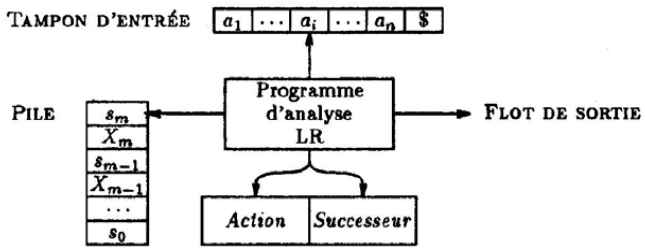

Figure 6.1 : Modèle d'un analyseur LR

La figure 6.1 montre qu'un modèle d'analyseur LR consiste en : un tampon d'entrée, un flot de sortie, une pile, un programme pilote et des tables d'analyse subdivisées en deux parties (Action et Successeur).

Le programme moteur est le même pour tous les analyseurs LR; seules les tables d'analyse changent d'un analyseur à l'autre. Le programme d'analyse lit les unités lexicales l'une après l'autre dans le tampon d'entrée, il utilise une pile pour y ranger les chaînes de la forme S0X1S1X2S2…XmSm, où Sm est au sommet. chaque Xi est un symbole de la grammaire et chaque Si est un symbole appelé état. La combinaison du numéro de l'état en sommet de pile et

du symbole d'entrée courant est utilisée pour indicer les tables et déterminer l'action d'analyse "décaler ou réduire" à effectuer.

Les tables d'analyse contiennent deux parties, une fonction d'actions d'analyse; Action et une fonction de transfert; Successeur.

6.1.2 Algorithme Analyse LR

Donnée : Une chaîne d'entrée w et des tables d'analyse LR (les fonctions Action et successeur) pour une grammaire G.

Résultat : Si w est dans L(G), une analyse ascendante de w, sinon une indication d'erreur.

Méthode : Initialement, l'analyseur a S0 en pile, où S0 est l'état initial et w$ dans son tampon d'entrée.

```
Initialiser le pointeur ps sur le premier symbole de w$
répéter indéfiniment début
        soit S l'état en sommet de pile et
        a le symbole pointé par ps;
        si Action [s,a] = décaler S' alors début
                empiler a puis S'
                avancer ps sur le prochain symbole d'entrée
        fin
        sinon si Action[s,a] = réduire par A →β alors début
            dépiler 2* | β |  symboles;
                    soit S' le nouvel état sommet de pile;
                    empiler A puis successeur [S',A];
                    emettre en sortie une identification de production A →β
        fin
        sinon si Action  [S,a] = accepter alors
            retourner
        sinon Erreur ()
Fin
```

Exemple

La table 5.1 présente les fonctions Action et Successeur des tables d'analyse LR pour la grammaire des expressions arithmétiques restreintes aux opérateurs binaires + et *.

(1) E →E+T
(2) E →T
(3) T →T * F
(4) T →F
(5) F → (E)
(6) F →id

Le codage des actions est :

1. di signifie décaler et empiler l'état i

2. rj signifie réduire par la production (j)

3. acc signifie accepter et

4. une case vide signifie une erreur

	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			

Tableau 6.1 : Table d'analyse pour la grammaire des expressions

Notons que l'état atteint par transition sur le symbole a depuis l'état s est identifié dans le champs Action[S,a] en même temps que l'action décaler, et que l'état atteint par transition sur le non terminal A depuis l'état S se trouve en successeur [S,A].

Sur le texte d'entrée id*id+id, la séquence des contenus de la pile et du tampon d'entrée est présenté à la figure 6.2:

PILE	ENTREE	ACTION
(1) 0	id * id + id $	décaler d5
(2) 0 id 5	* id + id $	réduire par F → id (r6)
(3) 0 F 3	* id + id $	réduire par T →F (r4)
(4) 0 T 2	* id + id $	décaler 7
(5) 0 T 2 * 7	id + id $	décaler 5
(6) 0 T 2 * 7 id 5	+ id $	réduire par F → id (r6)
(7) 0 T 2 * 7 F 10	+ id $	réduire par T →T * F (r3)
(8) 0 T 2	+ id $	réduire par E → T
(9) 0 E 1	+ id $	décaler 6
(10) 0 E 1 + 6	id $	décaler 5
(11) 0 E 1 + 6 id 5	$	réduire par F → id (r6)
(12) 0 E 1 + 6 F 3	$	réduire par T → F (r4)
(13) 0 E 1 + 6 T 9	$	réduire par E → E+T
(14) 0 E 1	$	accepter

Figure 6.2 : Transitions d'un analyseur LR sur id*id+id

A la ligne (1) l'analyseur LR est dans l'état 0 avec id comme premier symbole en entrée. L'entrée ligne 0 colonne id de la table Action de la figure 4 est d5 qui signifie décaler et empiler l'état 5. C'est ce qui a été fait à la ligne (2).

A ce moment, * devient le symbole d'entrée courant et l'action dans l'état 5 sur l'entrée * est réduire par F → id. Deux symboles sont dépilés, (un symbole d'état et un symbole de la grammaire). L'état 0 est donc au sommet de la pile. Comme la valeur du champ successeur pour l'état 0 sur F est 3, F et 3 sont empilés.