Limbaje formale, automate şi compilatoare

Curs 11

Gramatici LR(1)

- **Definiţie**
  - Fie G = (V, T, S, P) o gramatică redusă. Un articol LR(1) pentru gramatica G este o pereche ($A \to \alpha\bullet\beta$, a), unde $A\to\alpha\beta$ este un articol LR(0), iar a $\in$ FOLLOW(A) (# în loc de ε).
- **Definiţie**
  - Articolul ($A \to \beta 1 \bullet \beta 2$, a) este valid pentru prefixul viabil $\alpha\beta 1$dacă are loc derivarea
    - S dr$\Rightarrow$*αAu $\Rightarrow$ $\alpha\beta 1\beta 2$u
    - iar a = 1:u (a = # dacă u = ε).
- **Teorema**
  - O gramatică G = (V, T, S, P) este gramatică LR(1) dacă şi numai dacă oricare ar fi prefixul viabil φ, nu există două articole distincte, valide pentru φ, de forma($A \to \alpha\bullet$, a), (B $\to\beta\bullet\gamma$, b) unde a $\in$FIRST(γb).

Gramatici LR(1)

- Nu există conflict deplasare/reducere. Un astfel de conflict înseamnă două articole ($A \to \alpha\bullet$, a) şi ($B \to \beta\bullet a\beta'$, b) valide pentru acelaşi prefix.
- Nu există conflict reducere/reducere. Un astfel de conflict înseamnă două articole complete($A \to \alpha\bullet$, a) şi ($B \to \beta\bullet$, a) valide pentru acelaşi prefix
- Pentru a verifica dacă o gramatică este LR(1) se construiește automatul LR(1) în mod asemănător ca la LR(0):
  - Automatul are ca stări mulțimi de articole LR(1)
  - Tranzițiile se fac cu simboluri ce apar după punct
  - Închiderea unei mulțimi de articole se bazează pe faptul că dacă articolul ($B \to \beta\bullet A\beta'$, b) este valid pentru un prefix viabil φ atunci toate articolele de forma ($A \to \bullet\alpha$, a)unde a $\in$FIRTS(αa) sunt valide pentru același prefix.

Procedura de închidere LR(1)

```
flag= true;
while( flag) {
    flag= false;
    for ( (A→ α●Bβ, a) ∈ I) {
        for B → γ ∈ P)
            for( b ∈ FIRST(βa)){
                if( (B → •γ , b) ∉ I) {
                    I = I ∪ {(B → •γ , b)};
                    flag = true;
                }//endif
            }//endforb
        }//endforB
    }//endforA
}//endwhile
return I;
```

Automatul LR(1)

- t0 = închidere((S′ → • S,#));T={t_0};marcat(t_0)=false;
- while(∃ t∈T&& !marcat(t)){ // marcat(t) = false
  - for(X ∈ Σ) {
  - t′ = Φ;
    - for((A→ α • Xβ ,a) ∈ t)
      - t′ = t′ ∪{(B→ αX • β ,a) | (B B→ α • Xβ,a) ∈ t};
      - if(t′ ≠ Φ){
        - t′ = închidere(t′);
        - if(t′T) {
          - T= T ∪{ t′ };
          - marcat(t′) = false;
        - }//endif
        - g(t, X) = t′;
      - } //endif
  - } //endfor
  - marcat(t) = true;
- } // endwhile

Automatul LR(1)

- **Teorema**
  - Automatul M construit în algoritmul 2 este determinist şi L(M) coincide cu mulţimea prefixelor viabile ale lui G. Mai mult, pentru orice prefix viabil γ, $g(t_0,\gamma)$ reprezintă mulţimea articolelor LR(1) valide pentru γ.
- Automatul LR(1) poate fi folosit pentru a verifica dacă G este LR(1)
  - Conflict reducere/reducere: Dacă în T există o stare ce conţine articole de forma $(A \rightarrow \alpha\bullet, a)$, $(B \rightarrow \beta\bullet, a)$ atunci gramatica nu este LR(1);
  - Conflict deplasare/reducere: Dacă în Texistă o stare ce conţine articole de forma $(A \rightarrow \alpha\bullet, a)$ şi $(B \rightarrow \beta_1\bullet a\beta_2, b)$, atunci G nu este LR(1).
  - O gramatică este LR(1) dacă orice stare $t \in T$ este liberă de conflicte

Exemplu

- $S \rightarrow L{=}R \mid R,\ L \rightarrow {*}R \mid a,\ R \rightarrow L$

0

- $(S' \rightarrow \bullet S, \#)$
- $(S \rightarrow \bullet L{=}R, \#)$
- $(S \rightarrow \bullet R, \#)$
- $(L \rightarrow \bullet {*}R, \{=,\#\})$
- $(L \rightarrow \bullet a, \{=,\#\})$
- $(R \rightarrow \bullet L, \#)$

1

- $(S' \rightarrow S\bullet, \#)$

2

- $(S \rightarrow L\bullet{=}R, \#)$
- $(R \rightarrow L\bullet, \#)$

3

- $(S \rightarrow R\bullet, \#)$

4

- $(L \rightarrow {*}\bullet R, \{=, \#\})$
- $(R \rightarrow \bullet L, \{=, \#\})$
- $(L \rightarrow \bullet {*}R, \{=, \#\})$
- $(L \rightarrow \bullet a, \{=, \#\})$

5

- $(L \rightarrow a\bullet, \{=,\#\})$

6

- $(S \rightarrow L{=}\bullet R, \#)$
- $(R \rightarrow \bullet L, \#)$
- $(L \rightarrow \bullet {*}R, \#)$
- $(L \rightarrow \bullet a, \#)$

7

- $(L \rightarrow {*}R\bullet, \{=,\#\})$

8

- $(R \rightarrow L\bullet, \{=,\#\})$

9

- $(S \rightarrow L{=}R\bullet, \#)$

10

- $(R \rightarrow L\bullet, \#)$

11

- $(L \rightarrow {*}\bullet R, \#)$
- $(R \rightarrow \bullet L, \#)$
- $(L \rightarrow \bullet {*}R, \#)$
- $(L \rightarrow \bullet a, \#)$

12

- $(L \rightarrow a\bullet, \#)$

13

- $(L \rightarrow {*}R\bullet, \#)$

Tabela de tranziţie

g	a	=	*	S	L	R
0	5		4	1	2	3
1						
2		6				
3						
4	5		4		8	7
5						
6	12		11		10	9
7						
8						
9						
10						
11	12		11		10	13
12						
13						

Construcţia tabelei de analiză LR(1)

- for(t ∈ T)
  - for (a ∈ T) ACTIUNE(t, a) = "eroare";
  - for (A ∈ V) GOTO(t, A) = "eroare";
- for(t ∈ T}{
  - for((A→ α•aβ, L) ∈ t)
    - ACTIUNE(t,a)="D g(t, a)";//deplasare in g(t, a)
  - for((B→ γ•, L) ∈ t){// acceptare sau reducere
    - for(c ∈ L) {
      - if(B == 'S') ACTIUNE(t, a) = "acceptare";
      - else ACTIUNE(t,c)="R B→γ";//reducere cu B→γ
      - }//endfor
  - }// endfor
  - for (A ∈ N) GOTO(t, A) = g(t, A);
- }//endfor

Exemplu

- 0:S'→S, 1:S →L=R, 2:S →R, 3:L →*R, 4:L →a, 5:R →L

STARE	ACȚIUNE				GOTO		
	a	=	*	#	S	L	R
0	D5		D4		1	2	3
1				acceptare			
2		D6		R5			
3				R2			
4	D5		D4			8	7
5		R4		R4			
6	D12		D11			10	9
7		R3		R3			
8		R5		R5			
9				R1			
10				R5			
11	D12		D11			10	13
12				R4			
13				R3			

Exemplu

- Fie cuvintele
  - ***a
  - a=**a
  - *a=**a
- Analiza LR(1)?

Gramatici LALR(1)

- **Definiţie**
  - Fie t o stare în automatul LR(1) pentru G. Nucleul acestei stări este mulţimea articolelor LR(0) care apar ca prime componente în articolele LR(1) din t.

- **Defininiţie**
  - Două stări t_1 şi t_2 ale automatului LR(1) pentru gramatica G sunt echivalente dacă au acelaşi nucleu.

Gramatici LALR(1)

- Fiecare stare a automatului LR(1) este o mulţime de articole LR(1). Pornind de la două stări t_1 şi t_2 putem vorbi de starea t1 $\cup$ t2.
  - Fie $t_1 = \{(L \rightarrow *R., \{=, \#\})\}$, $t_2 = \{(L \rightarrow *R., \#)\}$, atunci $t_1 \cup t_2 = t_1$ pentru că $t_2 \subset t_1$.

- **Definiţie**
  - Fie G gramatică LR(1) şi M = (Q, Σ, g, t0, Q) automatul LR(1) corespunzător. Spunem că gramatica G este LALR(1) (Look Ahead LR(1)) dacă oricare ar fi perechea de stări echivalente t_1, t_2 din automatul LR(1), starea $t_1 \cup t_2$ este liberă de conflicte.

Tabela de analiză LALR(1)

- **Intrare**: Gramatica $G = (N, T, S, P)$ augmentată cu $S' \to S$;
- **Ieşire**: Tabela de analiză LALR(1) (ACŢIUNE şi GOTO).
- **Algoritm**:
  - 1. Se construieşte automatul LR(1), $M = (Q, \Sigma, g, t_0, Q)$ Fie $Q = \{t_0, t_1, \ldots, t_n\}$. Dacă toate stările din Q sunt libere de conflict, urmează 2, altfel algoritmul se opreşte deoarece gramatica nu este LR(1).
  - 2. Se determină stările echivalente din Q şi, prin reuniunea acestora, se obţine o nouă mulţime de stări $Q' = \{t'_0, t'_1, \ldots, t'_m\}$
  - 3. Dacă în Q' există stări ce conţin conflicte, algoritmul se opreşte deoarece gramatica G nu este LALR(1).

Tabela de analiză LALR(1)

- **4.** Se construieşte automatul $M' = (Q', \Sigma, g', t'0, Q')$, unde $\forall\ t' \in Q'$:
  - **5.** Dacă $t' \in Q$ atunci $g'(t', X) = g(t, X)$, $\forall X \in \Sigma$;
  - **6.** Dacă $t' = t_1 \cup t_2 \cup \ldots$, $t_1, t_2, \ldots \in Q$, atunci
    - **7.** $g'(t', X) = g(t1, X) \cup g(t2, X) \cup \ldots.$
- **8.** Se aplică algoritmul pentru construirea tabelei de parsare LR(1) pornind de la automatul M'. Tabela obţinută se numeşte tabela LALR(1) pentru gramatica G.

Exemplu

- Pentru gramatica discutată anterior avem $4 \cup 11 = 4$, $5 \cup 12 = 5$, $7 \cup 13 = 7$, $8 \cup 10 = 8$

STARE	ACȚIUNE				GOTO		
	a	=	*	#	S	L	R
0	D5		D4		1	2	3
1				acceptare			
2		D6		R5			
3				R2			
4	D5		D4			8	7
5		R4		R4			
6	D5		D4			8	9
7		R3		R3			
8		R5		R5			
9				R1			

Exemplu

- Există gramatici LR(1) care nu sunt LALR(1).
  - S →aAb | bAd | aBd | bBb
  - A →e
  - B →e