

$w \in L(G) \Leftrightarrow S^* \rightarrow w, w \in \Sigma^*$

$$|w|=n$$

25. Martie 2019

Curs 6

GRAMATICA SI LIMITAIE DE TIP L-R (left-right)

- urmărușul analizat urtează parcursa stanga-dreapta
- deoarece obț. derivații drepte pt. orice urme analizate

DEF: Fie $G = (N, \Sigma, S, P)$ g.u.c. spunem că G este de tip $LR(k)$, $k \geq 0$ dacă și c. $\forall i \in \mathbb{N}$ și $\forall j \in \mathbb{N}$ derivații drepte:

$$S \xrightarrow{*} \alpha Aw \xrightarrow{*} \alpha \beta w \quad (w \in \Sigma^*)$$

$$S \xrightarrow{*} \underset{j}{\alpha} \underset{j}{\beta} \underset{j}{w} \xrightarrow{*} \alpha \beta y \quad (\beta \rightarrow \delta \in P, j \in \mathbb{N}, j \delta x = \alpha \beta y, y \in \Sigma^*)$$

a.i. $\text{First}_K(w) = \text{First}_K(y)$, atunci

$$\alpha = \beta, A = B, w = y$$

Prop 1: Dacă G este de tip $LR(k)$, atunci G este neambigă.

Dem: P.p. că $G = (N, \Sigma, S, P)$ este $LR(k)$ și este ambigă

At. $\exists z \in L(G)$ cu 2 derivații drepte distincte:

$$S \xrightarrow{*} \alpha Aw \xrightarrow{*} \alpha \beta w \xrightarrow{*} z$$

$$S \xrightarrow{*} \underset{j}{\alpha} \underset{j}{\beta} \underset{j}{w} \xrightarrow{*} \underset{j}{\alpha} \underset{j}{\beta} \underset{j}{y} \xrightarrow{*} z, j \delta y = \alpha \beta w \text{ și}$$

$$A \rightarrow \beta \neq B \rightarrow \beta, A \rightarrow \beta, B \rightarrow \delta \in P$$

Dacă G este $LR(k)$ atunci $j \delta y = \alpha \beta w \Rightarrow \alpha = \beta$
 $A = B, y = w$

Din $\beta \delta y = \alpha \beta w \Rightarrow \beta = \delta$, deci $A \rightarrow \beta = B \rightarrow \delta \cancel{\Rightarrow}$

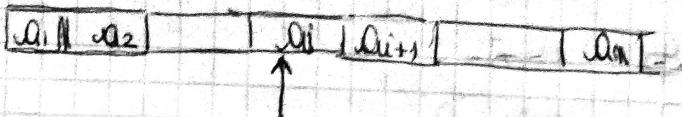
Prop 2: Orice gram. de tip $LR(k)$, $k \geq 2$ poate echiv. cu o gram de tip $LR(1)$.

ALGORITM DE ANALIZĂ SINTACTICĂ DE TIP LR

$$G = (N, \Sigma, S, P) \text{ g.u.c.}$$

$$G' = (N \cup \{S'\}, \Sigma \cup \{\$\}, S', P \cup \{S' \rightarrow S\})$$

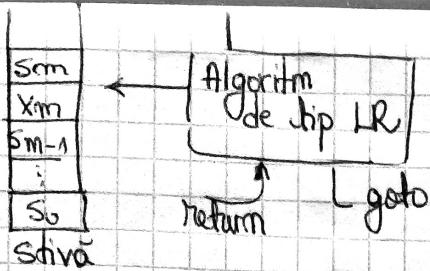
Input $\xrightarrow{\longrightarrow}$



$$w_1, \dots, w_n \in \Sigma^*$$

$$x_1, \dots, x_m \in \Sigma^N$$

$$S_0, S_1, \dots, S_m - Stările$$



Def: Fie $w \in (N \cup \Sigma)^*$, $S \xrightarrow{*} w$
(w este o forma scurta a lui G)

Pp. că ultima prod. aplicată în $S \xrightarrow{*} w$ a fost $A \rightarrow \beta$
Dc $w = \alpha \beta \gamma$. Numărul prefix viabil pt. w este prefixul lui $\alpha \beta$.

- Configurația initială pt. valo L₀(λ , s_0 , a_1 , ..., a_m , $\$, \lambda$)
- Schimbările de config
 - 1) $(s_0 x_1 s_1, \dots, s_m s_m, a_1 a_2 \dots a_m \$, \pi)$ pt. prefix. var
dă dă action [s_m, a_i] = shift s
 - 2) $(s_0 x_1 s_1, \dots, s_{j-1} x_j a_j s_j \dots s_m s_m, a_1 \dots a_{j-1}, a_j \$, \pi)$
 $\vdash (s_0 x_1 s_1, \dots, s_{j-1} A s_j, a_1 \dots a_{j-1} \$, \pi)$, unde q: $A \rightarrow X_j \dots X_m$ și goto [$s_j + 1, A$] = λ și action [s_m, a_i] = reduce
 $A \rightarrow X_j \dots X_m$
 - 3) $(s_0 x_1 s_1, \dots, s_m s_m, \$, \pi)$ \vdash accept, dă dă action [$s_m, \$$] = accept.
 - 4) $(s_0 x_1 s_1, \dots, s_m s_m, a_1 \dots a_m \$, \pi)$ \vdash error,
dă: - action [s_m, a_i] = error sau
- action [s_m, a_i] = reduce

$A \rightarrow X_j \dots X_m$ și
goto [$s_j + 1, A$] = errors

```

}
j ← 1;
while (1)
  a ← a[i];
  i ← v.f. nrvei;
  if (action[1, a] = shift s) {
    push a's spult nrvei; i ← i + 1;
  } else if (action[s, a] = reduce A → B) {
    pop 2 * |p| simboluri;
    s' ← v.f. nrvei;
  }
}
  
```

```

if ( foto [S', A] = error )
    error();
else
    push foto (S', A);
    printf ("A → P");
}
else if (action [s, a] = accept)
    return;
else
    error();

```

ALGORITMUL LR PENTRU GRAMATICI LR(1)

$G = (N, \Sigma, S, P)$ p.d.c. cu poz. numerotate 1, ..., 1 pl.

$G' = (N \cup \{S'\}, \Sigma \cup \{\$\}, S', P \cup \{S' \rightarrow S\})$.

Numește configurație LR(1):

$$A \rightarrow \alpha \cdot \beta, a$$

$$A \rightarrow \alpha \beta \in P$$

$a \in \Sigma \cup \{\$\}$ c.m. simbol lookahead

A este lungimea lui α

Dacă $\beta = \lambda$, $A \rightarrow \alpha$, va configurație finală, $a \in \text{Follow}(A)$ și reducerea $A \rightarrow \alpha$ se poate aplica când urmatorul sb. de intrare este a .

Suntem că $A \rightarrow \alpha \cdot \beta$ este validă pt. prefixul variabil și, dacă:

a) $\exists S \xrightarrow{*} \sigma Aw \Rightarrow \exists \cdot \alpha \beta w$

b) $\gamma = \delta \alpha$.

c) $a = \text{First}(w)$ ($w \in \Sigma^*$, dacă $w = \lambda$, at. $a = \$$),

atât fel a este primul simbol al lui w)

CONFIGURAȚII CANONICE LR(1) ASOCIAȚE LUI G' :

function closure (I)

// I este mt. de configurații LR(1)

{ $y \leftarrow I;$

do } for (fiecare $A \rightarrow \alpha \cdot \beta$, $\alpha \in J$)

 for (fiecare $B \rightarrow \gamma \in P$)

 for (fiecare $b \in \text{First}'(\beta, a)$)

if ($B \rightarrow \cdot x \rightarrow b \in J$)

adaugă $B \rightarrow \cdot x \xrightarrow{b} J$;

} while (se adaugă noi configurații);

return J ;

}

function goto (I, x) {

// $x \in N \cup \Sigma$, I mult. de config. LR(1)

$J \leftarrow \{ A \rightarrow \alpha \cdot x \cdot p, \alpha A \rightarrow \alpha \cdot x p, a \in I \}$

return closure (J);

}

function config () {

// returnată mult. canonice LR(1) pt. G'

$C_{G'} \leftarrow \{ \text{closure } (\{ S' \rightarrow \cdot S, \$ \}) \}$

do,

for (fiecare $I \in C_{G'}$)

for (fiecare $x \in N \cup \Sigma$)

if ($\text{goto } (I, x) \neq \emptyset \wedge \text{goto } (I, x) \notin C_{G'}$)

adăgăm $\text{goto } (I, x)$ în $C_{G'}$;

} while (se adaugă noi configurații);

return $C_{G'}$;

}

ALGORITM PT. DETERMINAREA TABELELOR

ACTION și GOTO

1. Construim $C_{G'} = \{ I_0, I_1, \dots, I_m \}$, $I_0 = \text{closure } (\{ \} \rightarrow \cdot S, \$)$

2. Pe fiecare $I_j \in C_{G'}$ se adaugă stocă j , $j \in \{ 0, \dots, m \}$

2.1. Dacă $A \rightarrow \alpha \cdot x p$, $b \in I_j$ și $\text{goto } (I_j, a) = I_k$,

atunci action $[j, a] = k$

2.2. Dacă $A \rightarrow \alpha$, $a \in I_j$, $A \neq S'$, atunci

action $[j, a] = \text{reduce } A \rightarrow \alpha$

Obs: Dacă tabelă action nu are multă multiple \Leftrightarrow
 G este LR(1).

2.3. Dacă $S \rightarrow S_0, \$ \in I_j$ ast. action $[f_j, \$] = \text{accept}$

2.4. action $[f_j, \cdot] = \text{error}$, în rest

2.5. Dacă $\text{goto}(I_j, A) = I_k$, adună goto $[f_j, A] = k$

/goto $(I_j, A) = \emptyset$, adună goto $[f_j, A] = \text{error}$

Exemplu:

$$E' \rightarrow E$$

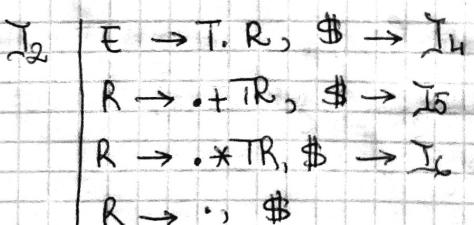
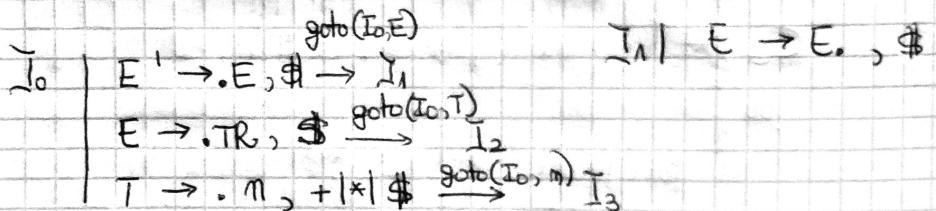
$$1) E \rightarrow .TR$$

$$2) R \rightarrow +TR$$

$$3) R \rightarrow *TR$$

$$4) R \rightarrow \lambda$$

$$5) T \rightarrow m$$



$$I_3 \quad | \quad T \rightarrow m., +|*| \$$$

$$I_4 \quad | \quad E \rightarrow TR., \$$$

$$I_5 \quad | \quad R \rightarrow +.TR, \$ \rightarrow I_4$$

$$T \rightarrow .m, +|*| \$ \rightarrow I_3$$

$$I_6 \quad | \quad R \rightarrow *.TR, \$ \rightarrow I_8$$

$$T \rightarrow .m, +|*| \$ \rightarrow I_3$$

$$I_7 \quad | \quad R \rightarrow +T.R, \$ \rightarrow I_9$$

$$R \rightarrow .+TR, \$ \rightarrow I_5$$

$$R \rightarrow .*TR, \$ \rightarrow I_6$$

I_8	R
	R
	R
	R
	acc
	+
	0
	1
2	55
3	15
4	
5	
6	
7	55
8	55
9	
10	

unde

$(0, n \times$
 $s_6)$

$(OT2^*$

$(OT2R)$

$R \rightarrow \cdot, \$$

$$I_8 \quad \left| \begin{array}{l} R \rightarrow T.R, \$ \rightarrow I_{10} \\ R \rightarrow \cdot + R, \$ \rightarrow I_5 \\ R \rightarrow \cdot * TR, \$ \rightarrow I_6 \\ R \rightarrow \cdot, \$ \end{array} \right.$$

$I_9 \quad R \rightarrow +TR \cdot, \$$

$I_{10} \quad R \rightarrow *TR \cdot, \$$

action

goto

	+	*	m	\$	E	T	R
0			π_3		1	2	
1				acc			
2	π_5	π_6		π_4			4
3	π_5	π_5		π_5			
4				π_1			
5			S_3			7	
6			S_3			8	
7	S_5	S_6		π_4			9
8	S_5	S_6		π_4			10
9				π_2			
10				π_3			

unde nu e completat avem eroare.

$$(0, m * m, \$, \lambda) \xrightarrow{S_3} (0m_3, *m\$, \lambda) \xrightarrow{\pi_5} (0T_2, *m\$, 5)$$

$$\xrightarrow{S_6} (0T_2 * 6, m\$, 5) \xrightarrow{S_3} (0T_2 * 6m_3, \$, 5) \xrightarrow{\substack{\text{goto}(0, T) \\ \pi_5}}$$

$$(0T_2 * 6T_8, \$, 55) \xrightarrow{T_4} (0T_2 * 6T_8R^{10}, \$, 455) \xrightarrow{\pi_3}$$

$$(0T_2R^4, \$, 3455) \xrightarrow{m} (0E1, \$, \underline{13455}) \xrightarrow{\substack{\text{goto}(S, R) \\ \text{acceptat}}}$$

derunarea
dreptă unică