Laborator 7

Implementare Translator Push Down pt. Expresii Aritmetice

1.Notiuni teoretice:

Gramatici LR(1) pentru expresii aritmetice

Expresiile aritmetice simple pot fi generate si verificate pe baza urmatoarei gramatici de tip LR(1)

G=(N,T,S,P) cu:

N={E,T,F} E semnifica expresie, T termen F factor

T={a,+,-,*,/,(,)} a este identificator de variabile sau constanta numerica

S={E} E este simbol initial

P={ 1. E → E+T,
11. E → E-T,
2. E→ T,
3. T→ T*F,
31. T→ T/F,
4. T→F,
5. F→(E),
6. F→a,
51. F→-(E) }

un exemplu de expresie genarata de aceata gramatica este : a+a*a

Pe baza acestei gramatici se poate genera, conform teoriei limbajelr independente de context de tip LR(k), Automatul Push Down (APD) care recunoaste siruri generate de graatica G. Pentru gramatica care are doar productiile 1,2,3,4,5,6, APD va avea urmatoarele tabele de actiuni si salt:

Nr	Tabel de actiuni : TA						Tabel de salt : TS		
stare	a	+	*	(	)	$	E	T	F
0	d_5			d_4			1	2	3
1		d_6				a_{cc}			
2		r_2	d_7		r_2	r_2			
3		r_4	r_4		r_4	r_4			
4	d_5			d_4			8	2	3
5		r_6	r_6		r_6	r_6			
6	d_5			d_4				9	3
7	d_5			d_4					10
8		d_6			d_{11}				
9		r_1	d_7		r_1	r_1			
10		r_3	r_3		r_3	r_3			
11		r_5	r_5		r_5	r_5			

Starile de pe prima coloana se vor gasii in varful stivei APD iar terminalele de pe prima linie din TA reprezinta urmatorul caracter de pe banda de intrare(=sirul de anticipare p_1 din definitia gramticii LR(k)). Spatiul gol din TA sau TS reprezinta eroare.

Detailând funcționarea APD care recunoaste gramatica LR(1) prin compararea vârfului stivei cu banda de intrare rezultă următoarele actiuni:

-d_i = deplasare peste un terminal în şirul de intrare, depunere în stivă a terminalului din şirul de intrare şi a stării i din tabelul de actiuni, salt la starea i ;
d_6: deplasare şi salt la 6;

- r_i = reducere cu producția i , se scot n*2 simboluri din stivă (n=lungime membru drept al producției i) şi se înlocuiesc cu membrul stâng al producției i (= un neterminal) şi apoi se face o comparare a vârfului anterior al stivei(cel dinaintea simbolurilor scoase) şi a neterminalului stâng(depus) şi se depune în stivă starea rezultată din tabela de salt TS.

Se scot n*2 simboluri din stivă pentru că pe lângă simbolurile din membrul drept al producției = n simboluri în stivă se mai află şi stări depuse în urma acțiunilor de deplasare şi reducere.
Stiva se inițializează cu $0 (0 este prima stare a automatului) iar cuvântul de intrare se termină cu $

1.2 Scheme de Translatare Orientate de Sintaxa (STOS) pentru Gramatici LR(1)

Daca completam actiunile de reducere ale automatului APD de mai sus cu actiuni de generare de cod intermediar se obtine o schema de translatare orintata de sintaxa STOS cara va fi capapila sa genereza cod intermdiar pentru evalauarea expresiei aritmetice. In paralel cu analiza sintactica. Implementarea schemei de translatare va genera Translatorul Push Down pentru expresii aritmetice.

Schema de translatare pentru gramatica G va arata astfel:

1.	$E \rightarrow E_1 + T$	{E.p:=newtemp; emit(E.p ´:=´ E_1.p ´+´ T.p) }
11.	$E \rightarrow E_1 - T$	{E.p:=newtemp; emit(E.p ´:=´ E_1.p ´-´ T.p) }
2.	$E \rightarrow T$	{E.p:= T.p) }
3.	$T \rightarrow T_1 * F$	{T.p:=newtemp; emit(T.p ´:=´ T_1.p ´*´ F.p) }
31.	$T \rightarrow T_1 / F$	{T.p :=newtemp; emit(T.p ´:=´ T_1.p ´/´ F.p) }
4.	$T \rightarrow F$	{T.p:= F.p) }
5.	$F \rightarrow (E)$	{F.p:=E.p}
51.	F = - (E)	{F.p:=newtemp; emit(F.p ´:=´ ´uminus´ E.p) }
6.	$F \rightarrow a$	{F.p:=a}

Intre acolade s-au trecut actiunile semantice de generare de cod intermediar care se vor executa impreuna cu actiunea de reducere a APD. S-au folosit indici inferiori pentru a diferentia neterminalele care apar de mai multe ori intr-o productie.

Fiecare neterminal are asociat cate un atribut *place* notat cu .p care semnifica locul/eticheta/variabila unde este memorata valoarea intermediara calculata si asociata neterminalului. De xemplu T.p este variabila temporara asociata lui T care memoreaza valoarea calculata a lui T.

Functia *newtemp* genereaza si returneaza o noua variabila temporara. Se pot genera variabile tempoarare cu numele t1,t2,...

Functia *emit* genereaza cod intermediar cu trei adres intr-un fisier de iesire. Textul dintre ´ ´ se emite ca si text. Atributele .p se inlocuiesc cu valoarea lor ca si text.

2. Creinte:

Se va implementa printr-un proram Translatorul Push Down care genereaza cod intermediar pentruu expresii aritmetice.

Pentru sirul de intrare $a_1+a_2*a_3$ codl intermediar generat va fi :

$t1=a_2*a_3$

$t2=a_2+t1$

Variabila t2 va memora valaorea expresiei.

2.1 Indicatii de programare:

Artibutele .p sunt atribute sintetizte. Valorile atributelor .p se pot memora pe o stiva de atribute. Intotdeauna in varful stivei se vor afla atributele necesare funtiti *emit*.

2.1 Exemplu de functionare a Translatorului Push Down pentru sirul a+a*a

Stivă APD	cuvânt de intrare	acțiune rezultată	Stiva atribute	Actiune generare de cod	Cod generat
\$ 0	$a_1+a_2*a_3\$$	d_5			
\$ 0 $\underline{a_1 5}$	$+a_2*a_3\$$	$r_6 \Rightarrow F+TS(0,F)$	a_1	F.p=a_1; [push(F.p)]	
\$ 0 $\underline{F3}$	$+a_2*a_3\$$	$r_4 \Rightarrow T+TS(0,T)$	a_1	T.p=F.p [T.p=pop(); push(T.p);]	
\$ 0 $\underline{T2}$	$+a_2*a_3\$$	$r_2 \Rightarrow E+TS(0,E)$	a_1	E.p=T.p [E.p=pop(); push(E.p)]	
\$ 0 E1	$+a_2*a_3\$$	d_6	a_1		
\$ 0 E1+6	$a_2*a_3\$$	d_5	a_1		
\$ 0 E1+6 $\underline{a_2 5}$	$*a_3\$$	$r_6 \Rightarrow F+TS(6,F)$	a_1	F.p=a_2; [push(F.p)]	
\$ 0 E1+6 $\underline{F3}$	$*a_3\$$	$r_4 \Rightarrow T+TS(6,T)$	a_1 a_2	T.p=F.p [T.p=pop(); push(T.p);]	
\$ 0 E1+6 T9	$*a_3\$$	d_7	a_1 a_2		
\$ 0 E1+6 T9*7	$a_3\$$	d_5	a_1 a_2		
\$ 0 E1+6 T9*7 $\underline{a_3 5}$	\$	$r_6 \Rightarrow F+TS(7,F)$	a_1 a_2	F.p=a_3; [push(F.p)]	
\$ 0 E1+6 $\underline{T9*7\ F10}$	\$	$r_3 \Rightarrow T+TS(6,T)$	a_1 a_2 a_3	T.p=t1; [F.p=pop(); T_1.p=pop();] emit($t_1=a_2*a_3$); [push(T.p)]	
\$ 0 $\underline{E1+6\ T9}$	\$	$r_1 \Rightarrow E+TS(0,E)$	a_1 t_1	E.p=t2; [T.p=pop(); E_1.p=pop();] emit($t2=a_1+t1$); [push(T.p)]	$t1=a_2*a_3$
\$ 0 E1	\$	acceptare	t_2		$t2=a_1+t_1$

La actiunile de generare de cod s-au trecut intre paranteze drepte procedurile de lucru cu stiva.

Se observa ca intotdeauna avem in varful stivei atributele necesare actiunii semantice si nu este nevoie sa memoram separat aceste atribute pentru fiecare neterminal. Mai mult la actiuni de tip atribuie simpla de atribute de genul T.p=F.p nu este necesara nici o actiune pe stiva pentru ca pop() si push() scot si pun aceaiasi valoare pe stiva.