SEMINAR 5

1) Sa se construiasca tabela LR(1) pentru gramatica G cu productiile:

E-> TR R->+TR | \*TR | lambda T->n

1: 2: 3: 4: 5:

Extindem gramatica: E’->E; se adauga simbolul terminal nou #

Se construiesc multimile canonice LR(1)

I0= E’->.E;# —>I1 // (I1=goto(I0,E)); inchidere pentru E

E->.TR;# —>I2 (=First(lambda.#)) //inchidere pentru T

T->.n;+|\*|# —>I3 (=First(R#)={+,\*,#})

I1= E’->E.;#

I2= E->T.R;#—>I4 //inchidere pentru R

R->.+TR;#—>I5

R->.\*TR;#—>I6

R->.;#

I3= T->n.;+|\*|#

I4 = E->TR.;#

I5= R->+.TR;#—>I7

T->.n;+|\*|#—>I3

I6 = R->\*.TR;#—>I8

T->.n;+|\*|#—>I3

I7= R->+T.R;#—>I9

R->.+TR;#—>I5

R->.\*TR;#—>I6

R->.;#

I8= R->\*T.R;#—>I10

R->.+TR;#—>I5

R->.\*TR;#—>I6

R->.;#

I9= R->+TR.; #

I10= R->\*TR.; #

Construim tabela LR(1) pentru G extinsa

Tabela action Tabela goto

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| M | + | | \* | n | # | E | T | R |
| 0 | error | | error | Shift 3 | error | 1 | 2 | error |
| 1 | error | | ... |  | accept |  |  |  |
| 2 | Shift 5 | | Shift 6 |  | Reduce 4 |  |  | 4 |
| 3 | Reduce 5  T->n | | Reduce 5 T->n |  | Reduce 5 T-> n |  |  |  |
| 4 |  | |  |  | Reduce 1 E->TR |  |  |  |
| 5 |  | |  | Shift 3 |  |  | 7 |  |
| 6 |  | |  | Shift 3 |  |  | 8 |  |
| 7 | Shift 5 | Shift 6 | |  | Reduce4 R>lambda |  |  | 9 |
| 8 | Shift 5 | Shift 6 | |  | Reduce4 R->lambda |  |  | 10 |
| 9 |  |  | |  | Reduce 2 R->+TR |  |  |  |
| 10 |  |  | |  | Reduce 3 R->\*TR |  |  |  |

M nu are intrari multiple (conflicte) <=> G este de tip LR(1)

Analizam sirul n\*n:

(0, n\*n#, lambda) ->(shift 3) (0n3, \*n#, lambda)-> (reduce5 T->n) (0T2, \*n#, 5)

2=goto(0,T)

-> (shift 6) (0T2\*6,n#,5) -> (shift 3) (0T2\*6n3,#,5) -> (reduce5 T->n) (0T2\*6T8,#, 55) -> (reduce4 R->lambda) (0T2\*6T8R’10’,#, 455)-> (reduce3 R->\*TR) (0T2R4,#,3455) -> (reduce 1 E->TR) (0E1,#,13455) -> accept

2) Se da gramatica G2: E-> E+E | E\*E | n

1: 2: 3:

Sa se construiasca tabela SLR(1) pentru G2.

Se extinde G2: E’->E se adauga simbolul terminal nou #

* Se calculeaza multimile Follow(X), X neterminal; se init Follow(net\_de\_start)={#}

|  |  |
| --- | --- |
| Follow(E) | #, +, \* |

E->**E**+E, Follow(**E**) += First (+E.Follow(E))= {+}

E->**E\***E, Follow(**E**) += First (\*E.Follow(E))= {\*}

* Se calculeaza multimile canonice LR(0)

I0= E’->.E —>I1

E->.E+E —>I1

E->.E\*E —>I1

E->.n —>I2

I1= E’->E.

E->E.+E —>I3

E->E.\*E —>I4

I2 E->n. // reducere 3 pentru orice in Follow(E)

I3 = E->E+.E —>I5

E->.E+E —>I5

E->.E\*E —>I5

E->.n —>I2

I4= E->E\*.E —>I6

E->.E+E —>I6

E->.E\*E —>I6

E->.n —>I2

I5= E->E+E. // reduce 1 pentru orice in Follow(E)

E->E.+E —>I3

E->E.\*E —>I4

I6= E->E\*E. // reducere 2 pt orice in Follow(E)

E->E.+E —>I3

E->E.\*E —>I4

Tabela SLR(1) pentru G2

Tabela action goto

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| M2 | + | \* | n | # | E |
| 0 |  |  | Shift 2 |  | 1 |
| 1 | Shift 3 | Shift 4 |  | accept |  |
| 2 | Reduce 3 | Reduce 3 |  | Reduce 3 |  |
| 3 |  |  | Shift 2 |  | 5 |
| 4 |  |  | Shift 2 |  | 6 |
| 5 | Reduce 1/  Shift 3 | Reduce 1/  Shift 4 |  | Reduce 1 |  |
| 6 | Reduce 2/  Shift 3 | Reduce 2/  Shift 4 |  | Reduce 2 |  |

M2 are intrari multiple <=> G2 nu este SLR(1).

Acesta este un caz particular pentru care se pot rezolva conflictele intr-un parser LR astfel:

Presupunem ca +, \* sunt asociative la stanga, \* prioritar fata de +

Conflictele se rezolva:

Reduce 1/ shift 3 in starea 5 pentru ‘+’: alegem reduce 1

Reduce 1/ shift 4 in starea 5 pentru ‘\*’: alegem shift 4

Reduce 2/ shift 3 in starea 6 pentru ‘+’: alegem reduce 2

Reduce 2/ shift 4 in starea 6 pentru ‘\*’: alegem reduce 2