LFTC – **Seminar 9**

1. **FIRST1, FOLLOW1**
2. Determinați FIRST1 and FOLLOW1

pentru neterminalele gramaticii următoare:

S → abA

S → e

A → Saa

A → b

|  |  |  |
| --- | --- | --- |
|  | FIRST1 | FOLLOW1 |
| S | a, e |  |
| A | a, b |  |

Explicații informale First1 (» primul terminal de după →, se aplică recursiv):

S → abA

S → e

A → Saa

A → b

|  |  |  |
| --- | --- | --- |
|  | FIRST1 | FOLLOW1 |
| S | a, e |  |
| A | b |  |

Pentru producția (3), care începe cu S, se va înlocui S cu “First1” său

A → Saa (3) devine A → a...aa (3’), respectiv A → aa (3”)

|  |  |  |
| --- | --- | --- |
|  | FIRST1 | FOLLOW1 |
| S | a, e |  |
| A | b, a |  |

S → abA

S → e

A → Saa

A → b

|  |  |  |
| --- | --- | --- |
|  | FIRST1 | FOLLOW1 |
| S | a, e | $, a |
| A | b, a |  |

Pentru un neterminal care se află pe ultima poziție într-o regulă de producție, sau care este urmat de o secvență de neterminale care îl au pe e în First1 (deci ajunge pe ultima poziție), se adaugă Follow1 al Neterminalului producției curente. Deoarece ce va urma după neterminalul respectiv va urma după neterminalul de pe ultima poziție.

S → ab**A**

Deci vom adăuga în Follow1(A) pe Follow1(S), deoarece în momentul în care îl vom înlocui pe S cu această producție, tot ce va urma după S va urma după A.

De exemplu, avem coada de mai jos:

Albu, Mara, S, Nastratin

Albu, Mara, ~~S~~ ab**A**, Nastratin

|  |  |  |
| --- | --- | --- |
|  | FIRST1 | FOLLOW1 |
| S | a, e | $, a |
| A | b, a | $, a |

Observație! Deși veți avea impresia că $-ul apare tot timpul în Follow1, el este tot timpul în Follow1(S), este doar o coincidență că ajunge să fie inclus în Follow1 al altor neterminale.

Observație! e nu apare în Follow1.

**2. Analiza sintactică descendentă : LL(1)**

1. Dându-se gramatica:

S → if c then S endif

S → if c then S else S endif

S → stmt

Dacă înlocuim**: *if c then*** cu ***a***, **else** cu ***b***,

***endif*** cu ***c***, și ***stmt*** cu ***i*** avem:

S → a S c

S → a S b S c

S → i

Pentru una dintre cele 2 gramatici de mai sus:

a) Verificați dacă gramatica este LL(1).

b) Încercați să transformați gramatica în una echivalentă LL(1) aplicând factorizarea la stânga. Verificați dacă noua gramatică este LL(1).

c) Folosind un analizor descendent verificați dacă secvența: if c then if c then stmt else stmt endif endif (sau echivalenta ei scrisa cu a,b,c,i) aparține limbajului generat de gramatică.

...

1. Fie gramatica ambiguă:

S → if c then S else S

S → if c then S

S → stmt

Dacă înlocuim**: *if c then*** cu ***a***, ***else*** cu ***b***,

***endif*** cu ***c***, și ***stmt*** cu ***i*** avem:

S → a S b S

S → a S

S → i

Pentru una dintre cele 2 gramatici de mai sus:

a) Verificați dacă gramatica este LL(1).

b) Încercați să transformați gramatica în una echivalentă LL(1) aplicând factorizarea la stânga. Verificați dacă noua gramatică este LL(1).

c) Discutați, împreună cu cadrul didactic, cum se poate modifica tabelul de analiză astfel încât să se elimine conflictele.

d) Folosind analizorul LL(1) dacă secvența: if c then if c then stmt else stmt (sau echivalenta ei scrisa cu a,b,c,i) aparține limbajului generat de gramatică.

...

1. Dându-se gramatica:

E → T + E | T

T → T \* F | F

F → (E) | a

a) Verificați dacă gramatica este LL(1).

b) Încercați să transformați gramatica în una echivalentă LL(1) aplicând factorizarea la stânga. Verificați dacă noua gramatică este LL(1).

c) Folosind un analizor descendent verificați dacă secvența: a+a aparține limbajului generat de gramatică.

...

1. Fie gramatica:

List → id

List → id sep List

a) Verificați dacă gramatica este LL(1).

b) Încercați să transformați gramatica în una echivalentă LL(1) aplicând factorizarea la stânga. Verificați dacă noua gramatică este LL(1).

1. Fie gramatica:

S → begin SList end

S → stmt

SList → S

SList → S ; SList

a) Verificați dacă gramatica este LL(1).

b) Încercați să transformați gramatica în una echivalentă LL(1) aplicând factorizarea la stânga. Verificați dacă noua gramatică este LL(1).

...

4. a) List → id (1)

List → id sep List (2)

|  |  |  |
| --- | --- | --- |
|  | FIRST1 | FOLLOW1 |
| List | id | $ |

Tabelul LL(1)

|  |  |  |  |
| --- | --- | --- | --- |
|  | id | sep | $ |
| List | (id,1) (id,2) |  |  |
| id |  |  |  |
| sep |  |  |  |
| $ |  |  |  |

Avem conflict, deci Gratica nu e de tip LL(1)

b) Factorizarea la stânga (se păstrează doar partea comună + un neterminal nou. Neterminal nou va genera părțile lipsă):

List → id

List → id sep List

List → id A (1)

A → e (2)

A → sep List (3)

Determinăm First & Follow

|  |  |  |
| --- | --- | --- |
|  | FIRST1 | FOLLOW1 |
| List | id | $ |
| A | e, sep | $ |

Tabelul LL(1)

|  |  |  |  |
| --- | --- | --- | --- |
|  | id | sep | $ |
| List | (id A,1) |  |  |
| A |  | (sep List,3) | (e,2) |
| id | pop |  |  |
| sep |  | pop |  |
| $ |  |  | acc |

c) Verificați dacă cuvântul id sep id este acceptat de gramatică.

(**id** sep id $,**List**$,e) |- (push 1) (id sep id $,id A $,1) |- (pop) ( **sep** id $, A $,1) |- (push 3) ( sep id $, sep List $,13) |- (pop)

(id $, List $,13) |- (push 1) (id $, id A $,131) |- (pop) ($, A $,131) |- (push 2) ($, $,1312) |- (acc) acc

Deci id sep id Î L(G)

1. Fie gramatica:

S → begin SList end

S → stmt

SList → S

SList → S ; SList

a) Verificați dacă gramatica este LL(1).

b) Încercați să transformați gramatica în una echivalentă LL(1) aplicând factorizarea la stânga. Verificați dacă noua gramatică este LL(1).

|  |  |  |
| --- | --- | --- |
|  | FIRST1 | FOLLOW1 |
| S | begin, stmt | $ , ; , end |
| SList | begin, stmt | end |