SEMINAR 1

Saturday, November 6, 2021 4:38 PM

(1) ANALIZA LEXICALĂ

- specificații lexicale

(1) (1) (10|01)*0
(2) 011+
(3) 1(01|0)*

1010100010001000111
3 3 3

(2) (1) (a*b)*c Un șir care nu poate fi recunoscut? => b
(2) abc*

cabccbbcaabbbcabb
1 2 1 1 2

(2) ANALIZA SINTACTICĂ

ANALIZA TOP-DOWN

(1) S → Aa | b
A → Sb

- E recursivă la stânga?
- Ce înseamnă recursivitate la stânga?

RECURSIVITATE LA STANGA

$X \to^{+} Xa$

$X \to X\alpha \mid \beta$ (direct)

$\begin{cases} X \to Y\alpha \mid \beta \\ Y \to X\delta \end{cases}$ (indirect)

ELIMINARE REC. LA STÂNGA

(1) $A \to A\alpha \mid \beta$
$A \to \beta A'$
$A' \to \alpha A' \mid \varepsilon$

EXERCIȚII

⑤ $S \to Aa \mid d$ $\Leftrightarrow$

$A \to Sb$

$S \to Sba \mid d$ (α = ba, β = d) $\Rightarrow$ $S \to dS'$

$S' \to baS \mid \varepsilon$

PAȘI GENERALI:

1. * eliminarea producțiilor ε
2. * eliminarea ciclurilor ($S \stackrel{+}{\Rightarrow} S$)

} Foarte puțin probabil să fie nevoie

3. eliminarea recursivității

① * ELIMINARE PRODUCȚII ε

① $S \to XZ$ $\Rightarrow$ $S \to XZ \mid X \mid Z$ → cazul $X \to \varepsilon$ (X: cazul $Z \to \varepsilon$)

$X \to aXb \mid \varepsilon$ $\quad$ $X \to aXb \mid ab$ ⇒ cazul în care $X \to \varepsilon$

$Z \to aZ \mid ZX \mid \varepsilon$ $\quad$ $Z \to aZ \mid a \mid ZX \mid X$ → cazul $Z \to \varepsilon$ (a: cazul $Z \to \varepsilon$)

② $S \to AB$ $\Rightarrow$ $S' \to S \mid \varepsilon$

$A \to AaA \mid \varepsilon$ $\quad$ $A \to AaA \mid aA \mid Aa \mid a$

$B \to BbB \mid \varepsilon$ $\quad$ $B \to BbB \mid bB \mid Bb \mid b$

⇒ aici au fost eliminate toate producțiile ε, mai puțin cele pentru regula S' care nu apare în dreapta (RHS) niciunei alte producții

③ * ELIMINARE CICLURI

- O gramatică are un ciclu, atunci când există o producție S astfel încât $S \stackrel{+}{\Rightarrow} S$

exemplu: $S \to X \mid Xb \mid SS$

$X \to S \mid a$

⇒ ciclu $S \stackrel{+}{\Rightarrow} S$ și $X \stackrel{+}{\Rightarrow} X$

! Eliminarea ciclurilor se face numai **DUPĂ** eliminarea ε-producțiilor !

De ce avem nevoie să eliminăm ciclurile?

exemplu: $A \to A \mid Ab \mid b$ (α_1, α_2, β) $\Rightarrow$ $A \to bA'$ ⇒ nu merge să aplic algoritmul

$A' \to A' \mid bA' \mid \varepsilon$

CUM FACEM?

- eliminăm aparițiile variabilelor ciclice din dreapta

ex: $S \to X \mid Xb \mid SS$ $\Rightarrow$ $S \to a \mid ab \mid SS$

$X \to S \mid a$ $\quad$ $X \to Xb \mid SS \mid a$

③ ELIMINARE RECURSIVITATE

① $E \to E+T \mid T$ (α: $+T$; β: T)

$\Rightarrow E \to TE'$

$E' \to +TE' \mid \varepsilon$

② $A \to Ab\alpha \mid Aa \mid a$ (α_1: $b\alpha$; α_2: a; β: a)

$\Rightarrow A \to aA'$

$A' \to bcA' \mid aA' \mid \varepsilon$

③ $A \to Ac \mid Aad \mid Bd \mid C$ (α_1: c; α_2: ad; β_1: Bd; β_2: C)

$\Rightarrow A \to BdA' \mid CA'$

$A' \to cA' \mid adA' \mid \varepsilon$

④ CAZ SPECIAL $\Rightarrow$ ce se întâmplă dacă nu avem niciun β?

$A \to Aa \mid Ab \Rightarrow A \to aA' \mid bA' \mid \varepsilon$

⑤ $S \to Sa \mid S \mid \varepsilon \Rightarrow S' \to aS' \mid \varepsilon$

⑥ CAZ SPECIAL $\Rightarrow$ recursivitate indirectă

$S \to Sx \mid Ssb \mid xS \mid a$

$x \to Sa \mid xb$

$\Rightarrow S \to xSS' \mid aS'$ $\Rightarrow$ nu am eliminat recursivitatea

$S' \to xS' \mid sbS' \mid \varepsilon$ $\quad S \overset{+}{\Rightarrow} SaxSS'$

$x \to Sax'$

$x' \to bx' \mid \varepsilon$

$\Rightarrow$ CUM facem?

(ex) $S \to Sx \mid Ssb \mid xS \mid a$ $\Rightarrow$ nu avem prod. ε, nu avem cicluri

$x \to xb \mid Sa \mid b$

① se elimină recursivitatea la stânga din prima regulă (S)

$S \to xSS' \mid aS'$

$S' \to xS' \mid sbS' \mid \varepsilon$

$x \to xb \mid Sa \mid b$

② îl înlocuim pe S în x

$x \to xSS'a \mid aS'a \mid xb \mid b$

③ înlocuim recursivitatea pentru producția x

$x \to aS'ax' \mid bx'$

$x' \to SS'ax' \mid bx' \mid \varepsilon$

$\Rightarrow$

$S \to xSS' \mid aS'$

$S' \to xS' \mid sbS' \mid \varepsilon$

$x \to aS'ax' \mid bx'$

$x' \to SS'ax' \mid bx' \mid \varepsilon$

(7) $S \to Aa \mid b$

$A \to Ac \mid Sd \mid \varepsilon$

⇒ ① eliminăm rec. în $S \to Aa \mid b$ ⇒ nu avem

② înlocuim pe S în A ⇒

$A \to Ac \mid Aad \mid bd \mid \varepsilon$

③ eliminăm rec. din A ⇒ $A \to bdA' \mid A'$

$A' \to cA' \mid adA' \mid \varepsilon$

⇒
$S \to Aa \mid b$
$A \to bdA' \mid A'$
$A' \to cA' \mid adA' \mid \varepsilon$

(8) Eliminate left recursion from the following grammar:

$$S \to STS \mid ST \mid T$$
$$T \to Ta \mid Tb \mid U$$
$$U \to T \mid c$$

① gramatica are cicluri ⇒ $T \to U \to T$

⇒ $T \to Ta \mid Tb \mid c$ ⇒ $U \Leftrightarrow T$
$U \to Ta \mid Tb \mid c$

⇒ $S \to STS \mid ST \mid T$ ⇒ $S \to TS'$ ⇒ $S \to TS'$
$T \to Ta \mid Tb \mid c$ $S' \to TSS' \mid TS' \mid \varepsilon$ $S' \to TSS' \mid TS' \mid \varepsilon$
$T \to Ta \mid Tb \mid c$ $T \to cT'$
$T' \to aT' \mid bT' \mid \varepsilon$

FACTORIZARE (LA STANGA)

(1) 3. (a) Left factor the following grammar:

$$E \to int \mid int + E \mid int - E \mid E - (E) \mid E * E$$

(b) Eliminate left-recursion from the following grammar:

$$A \to A + B \mid B$$
$$B \to int \mid (A) \mid B * int$$

a) $E \to int\,X \mid EY$
$X \to +E \mid -E \mid \varepsilon$
$Y \to -(E) \mid *E$

b) $A \to BA'$
$A' \to +BA' \mid \varepsilon$
$B \to int\,B' \mid (A)B'$
$B' \to *int\,B' \mid \varepsilon$

(2)

S	→	bAb	|	bBa
A	→	aS	|	CB
B	→	b	|	BC
C	→	c	|	cC

→ eliminare rec. stg + factorizare

① Factorizare : Eliminare recursivitate

$S \to bX$
$X \to Ab \mid Ba$
$A \to aS \mid CB$
$B \to b \mid BC$ ⇒ $B \to bB'$
$B' \to CB' \mid \varepsilon$
$C \to cD$
$D \to \varepsilon \mid C$

③ E -> E*E | E+E | (E) | int

Câți arbori de parsare diferiți?

5*3 + (2*7)+4

=> în total 5

a) 5 ⊛ 3 + (2*7)+4 (rădăcină)
   [(   )   ] 2 arbori
        (   )

b) 5*3 ⊕ (2*7)+4 (rădăcină) 1 arbore

c) 5*3 + (2*7) ⊕ 4 (rădăcină) 2 arbori
   [(   )]
   ( )   (   )

④ Ce ar trebui să adăugăm gramaticii de mai sus pt. a nu mai fi ambiguă?

-> reguli de precedență și asociere

⑤ S -> aSb | Sb | b

Este ambiguă?

=> găsim un șir cu 2 arbori de derivare diferiți

abbb

S $\xrightarrow{1}$ aSb $\xrightarrow{2}$ aSbb $\xrightarrow{3}$ abbb

S $\xrightarrow{2}$ Sb $\xrightarrow{1}$ aSbb $\xrightarrow{3}$ abbb

RECURSIVE DESCENT

-> face backtracking cu producțiile în ordine până găsește o potrivire

S -> baSab | baS | b

șir babab

S -> baSab -> ba baSab ab -> ba ba baSab abab (nepotrivire)
   -> baba baS abab ×
   -> baba b abab
 -> ba baS ab .....
 -> ba b ab ✓

Care ar fi cea mai scurtă variantă posibilă?

S -> ba Sab -> ba b ab (neținând cont de recursive descent)

② S -> baS | b | baSab

S -> baS -> ba baS -> baba baS ×
 -> baba b ✓

FIRST, FOLLOW

FIRST

1. $First(\alpha) = \alpha$; $First(\alpha X) = \{\alpha\}$ (→ terminal !!)
2. $X \to \varepsilon \Rightarrow \varepsilon \in First(X)$
3. $X \to y_1 \dots y_k \Rightarrow First(X) \subseteq First(y_1 \dots y_k)$
4. $First(y_1 y_2 \dots y_k) =$
   a) $First(y_1)$, dacă $\varepsilon \notin First(y_1)$
   b) $First(y_1) \setminus \{\varepsilon\} \cup First(y_2 \dots y_k)$
   c) dacă $\varepsilon \in$ toate $First(y_1), First(y_2), \dots$
   $\Rightarrow First(y_1 \dots y_k) \cup \{\varepsilon\}$

FOLLOW

1. $\$ \in Follow(S)$
2. $A \to \alpha B \beta \Rightarrow Follow(B) \supseteq First(\beta) \setminus \{\varepsilon\}$
3. $A \to \alpha B \Rightarrow Follow(B) \supseteq Follow(A)$
4. $\begin{cases} A \to \alpha B \beta \\ First(\beta) \text{ conține } \varepsilon \end{cases} \Rightarrow Follow(B) \supseteq Follow(A)$

! α, β nu sunt neapărat terminali, ci sunt secvențe

EXERCIȚII

5.

$S \to A(S)B \mid \varepsilon$
$A \to S \mid SB \mid x \mid \varepsilon$
$B \to SB \mid y$

	First
S	ε, x, (, y
A	ε, x, (, y
B	x, y, (

FIRST

1. ε + terminali
   $First(S) \supseteq \{\varepsilon\}$
   $First(A) \supseteq \{x, \varepsilon\}$
   $First(B) \supseteq \{y\}$
2. $First(S) \supseteq First(A) \setminus \{\varepsilon\} \cup First('(')$
   $\supseteq \{x, (\}$
   $First(A) \supseteq First(S) \setminus \{\varepsilon\} \cup \{\varepsilon\}$
   $\supseteq First(S) \setminus \{\varepsilon\} \cup First(B) = \{x, (, y\}$
   $First(B) \supseteq First(S) \setminus \{\varepsilon\} \cup First(B)$
   $= \{x, (, y\}$

FOLLOW

$S \to A(S)B \mid \varepsilon$
$A \to S \mid SB \mid x \mid \varepsilon$
$B \to SB \mid y$

	First	Follow
S	ε, x, (, y	\$, (, x, y,),
A	ε, x, (, y	(
B	y, x, (	), \$, (, x, y

1. $\$ \in Follow(S)$
2. $Follow(A) \supseteq \{(\}$
   $Follow(S) \supseteq \{)\}$
3. $Follow(B) \supseteq Follow(S)$
   $Follow(S) \supseteq Follow(A)$
   $Follow(S) \supseteq First(B) \setminus \{\varepsilon\}$
   $Follow(B) \supseteq Follow(A)$
   $Follow(B) \supseteq Follow(B)$

! ε niciodată în Follow

TABEL LL

	x	y	(	)	\$
S	A(S)B ε	A(S)B ε	A(S)B ε	ε	ε
A	S SB x	S SB	S SB ε		
B	SB	SB y	SB		

TABEL LL(1) → Left to right, Leftmost derivation

LL(k) ⇒ avem nevoie pt. maxim k tokeni de lookahead pt. a selecta o producție

exemple

① Exemplu de LL(2) care nu este LL(1)

$A \to ab \mid ac$ → nu putem avea doar 1 token, pt că nu știm ce producție să alegem
→ dacă avem 2, e clar că putem alege

② Multe gramatici factorizabile la stânga nu sunt LL(1). De ce?
⇒ exemplul de mai sus ↑ (gramatică factorizabilă, nu știm ce să alegem)

③ Exemplu de gramatică factorizabilă la stg, dar LL(1)

$A \to X \mid Xa$
$X \to \varepsilon$

	First	Follow
A	a, ε	$
X	ε	a, $

⇒ nu avem conflict

	a	$
A	A→Xa	A→X
X	X→ε	X→ε

Gramatici non-*LL*(1)

- Valori **multiple** ale intrărilor din tabel în caz de gramatică:
  - Ambiguă
  - Recursivă la stânga
  - Nefactorizată la stânga
  - Precum și în alte situații

CONSTRUIRE TABEL

$S \to \langle AS \rangle \mid [BS] \mid (S)$
$A \to aA \mid \varepsilon$
$B \to bB \mid \varepsilon$

	First	Follow
S	<, [, (	$, >,],)
A	a, ε	<, [, (
B	b, ε	<, [, (

First

First(S) = {<, [, (}
First(A) = {a, ε}
First(B) = {b, ε}

Follow

① $ ∈ Follow(S)
② Follow(S) = {>,],)}
③ Follow(A) = First(S) \ {ε}
Follow(B) = First(S) \ {ε}

① Alegem o regulă (ex: S)
② Calculăm First(α) pentru fiecare producție $S \to \alpha_1 \mid \alpha_2 \mid .. \mid \alpha_m$
(ex: First(<AS>), First([BS]), First(S))
③ Dacă ε ∈ First(S) ⇒ ne uităm în Follow(S)
și punem ε sub toți terminalii din Follow(S)
④ Punem fiecare producție (α_i) la intersecția dintre:
- linia regulii (ex: S)
- coloana terminalului din First(α_i)

ex: First(<AS>) = {<} ⇒

	<	>	[	]	(	)	a	b	$
S	<AS>		[BS]		(S)				
A	ε		ε		ε		aA		
B	ε		ε		ε			bB	

Este LL(1)? ⇒ Da, nu avem 2 prod. într-o celulă.

ANALIZA BOTTOM-UP

SHIFT-REDUCE

↳ SHIFT = deplasarea cursorului "|" la dreapta

↳ REDUCE = aplicarea unei producții la stânga "|"

Conflicte LR(0) – Left to right, Rightmost derivation

- Conflict ***shift-reduce***, în cazul existenței unei stări cu câte un item de fiecare tip:

$$\begin{cases} \boxed{X_1 \to \beta_1 \bullet} & \to \text{reduce} \\ \boxed{X_2 \to \beta_2 \bullet t \gamma_2} & \to \text{shift} \end{cases}$$

- Conflict ***reduce-reduce***, în cazul existenței unei stări cu cel puțin doi itemi *reduce*:

$$\boxed{\begin{matrix} X_1 \to \beta_1 \bullet \\ X_2 \to \beta_2 \bullet \end{matrix}} \Rightarrow \text{ambii sunt reduce}$$

Analiză SLR(1)

- ***SLR***(1): *Simple LR*
- *SLR*(**1**): utilizarea unui *token* de ***lookahead*** pentru reducerea numărului stărilor conflictuale
- Îmbunătățire: ***Reduce*** cu $X \to \beta$, dacă s conține $X \to \beta \bullet$ și $t \in Follow(X)$

  ↳ terminalul care urmează

• SLR(1) aduce o îmbunătățire care spune că nu vom alege Reduce doar dacă tokenul curent din input $t \in Follow(B)$

.... $\beta \mid t$ ....

$X \to \beta \bullet$

dacă $t \notin Follow(X)$ nu are sens să facem Reduce (t nu poate urma niciodată după X)

OBS: Totuși, conflict SR încă putem avea → ex: $\begin{cases} X_1 \to \beta \bullet \\ X_2 \to \beta \bullet t \end{cases}$ conflict dacă $t \in Follow(X_1)$

..... $\mid t$ ..... ⇒ nu știu ce să aleg, pt. că amândouă merg

① $S \to Sb \mid a$ LL(1), SLR(1)?

① verificăm dacă e LL(1)

$First(S) = \{a\}$

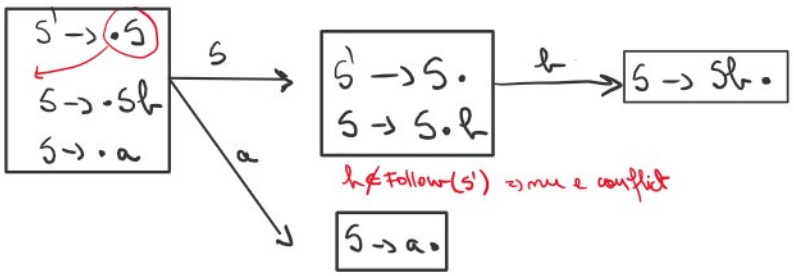
$Follow(S) = \{b, \$\}$

	a	b	$
S	Sb a		

⇒ 2 intrări sub a ⇒ nu este LL(1) (a anunță 2 producții)

② SLR(1) $S' \to S \Rightarrow Follow(S') = \{\$\}$

Construcție AFD:



② $S \to SbS \mid a$ LL(1) SLR(1)

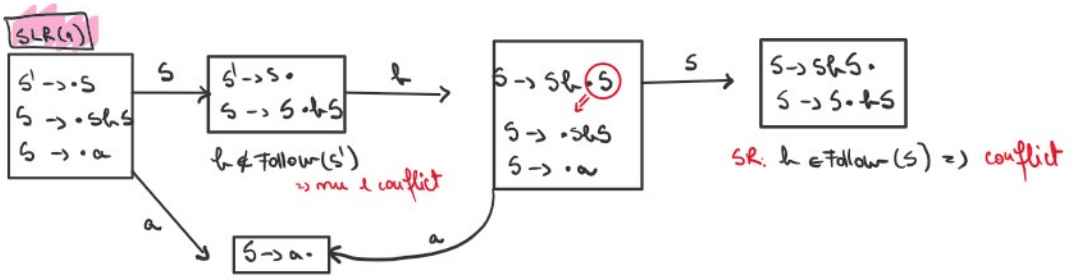
$First(S) = \{a\}$

$Follow(S) = \{\$, b\}$

$Follow(S') = \{\$\}$

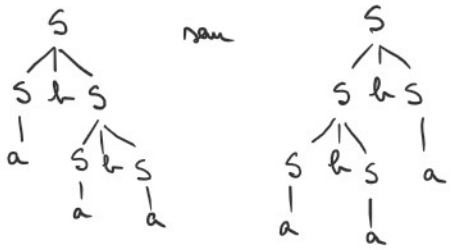
	a	b	$
S	SbS		
	a		

=> nu e LL(1)



OBS: Gramatică ambiguă, deci nu este nici LL, nici SLR !

ahaha



sau

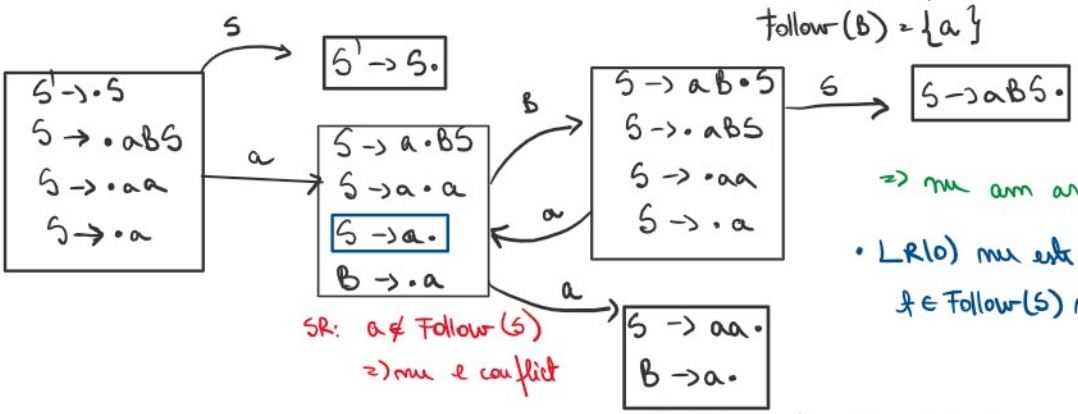
③ $S \to aBS \mid aa \mid a$ SLR(1), LR(0) ?

$B \to a$

$Follow(S') = \{\$\}$

$Follow(S) = \{\$\}$

$Follow(B) = \{a\}$



=> nu am avut conflicte => SLR(1)

- LR(0) nu este, pt că avem nevoie de euristica $t \in Follow(S)$ să știm dacă putem face reduce

(4) $S \to A(S)B \mid \varepsilon$
$A \to S \mid SB \mid x \mid \varepsilon$
$B \to SB \mid y$

SLR(1)

		Follow
S		$,), x, y,)
A		(
B		), $, (, x, y
S'		$

$S' \to \cdot S$
$S \to \cdot A(S)B$
$S \to \cdot$
$A \to \cdot S$
$A \to \cdot SB$
$A \to \cdot x$
$A \to \cdot$

=> există conflicte în starea asta?

RR: Follow(S) ∩ Follow(A) = {(} => pt tokenul '(' vom avea conflict RR

În ce caz putem avea conflict RR pt tokenul următor din input "t"?
=> dacă $t \notin$ Follow(S) ∩ Follow(A)

ex: $\varepsilon \mid t$ ....
A ↙ ↘ S

SR: $x \in$ Follow(S) => conflict SR

(5) De ce preferăm left recursion over right recursion în strategia shift-reduce?

(1) $S \to aS \mid \varepsilon$ vs (2) $S \to Sa \mid \varepsilon$; în aaaa .....

(1) => se shiftează tot inputul pe stivă, înainte să se facă reducerea

(2) => se face reducerea $S \to \varepsilon$, iar apoi se alternează shift 'a' cu reduce $S \to Sa$

⇓ se preferă left recursion pt shift-reduce parser pt. că fol. mai puțină stivă

(6) $S \to (T)$
$T \to T + int \mid int$

(int | + int + int + int)
(T + int | + int + int)
(T + int | + int)
(T + int |)
(T)
S

$S \to (T)$
$T \to int + T \mid int$

(int | + int + int + int)
int + int + int + T
T
T
(T)
S

(int + int + int + int)

=> nu putem face reduce pt că (T+) nu apare în gramatică