

Lecture 19 - July 29

Syntactic Analysis

***Removing Left Recursions: Algorithm
Tracing: Removing Direct LR's***

Announcements/Reminders

- **WrittenTest** being graded
- **Project Report Template** released
- **Assignment 3** (Top-Down Parsing) released
- **Top-Down Parsing** Algorithm Tracing: Walkthrough Video Released
- **Exam**: 9 AM, Wednesday, August 13 (DB 0007)
- **Makeup & Exam Study** lecture (Bottom-Up Parsing) to be released

Removing Left-Recursions: Algorithm

ϵ -productions

$A \rightarrow \epsilon$

left-recursive

$A_i \rightarrow A_i \alpha$
 $\quad \quad \beta$

① $A_i \Rightarrow \beta$
 ② $A_i \Rightarrow \beta \alpha^*$
 $\Rightarrow A_i \alpha$
 $\Rightarrow A_i \alpha \alpha$
 $\Rightarrow \beta \alpha \alpha$

$A_i \rightarrow \beta A_i'$
 $A_i' \rightarrow \alpha A_i' \mid \epsilon$

```

1  ALGORITHM: RemoveLR
2  INPUT: CFG  $G = (V, \Sigma, R, S)$ 
3  ASSUME:  $G$  has no  $\epsilon$ -productions
4  OUTPUT:  $G'$  s.t.  $G' \equiv G$ ,  $G'$  has no
5             indirect & direct left-recursions
6  PROCEDURE:
7  impose an order on  $V$ :  $\langle A_1, A_2, \dots, A_n \rangle$ 
8  for  $i: 1 \dots n$ 
9    for  $j: 1 \dots i-1$ :
10     if  $\exists A_i \rightarrow A_j \gamma \in R \wedge A_j \rightarrow \delta_1 \delta_2 \dots \delta_m \in R$  then
11       replace  $A_i \rightarrow A_j \gamma$  with  $A_i \rightarrow \delta_1 \delta_2 \dots \delta_m \gamma$ 
12     end
13   for  $A_i \rightarrow A_i \alpha \mid \beta \in R$ :
14     replace it with:  $A_i \rightarrow \beta A_i', A_i' \rightarrow \alpha A_i' \mid \epsilon$ 
    
```

start with a known ordering (last action)

distinct vars arbitrary mix of terminals and non-terminals

eliminate indirect LR's (previously)

eliminate direct LR's.

Criteria:
 1. correct
 2. left-recursive removed

right-recursive

Why not checking Term \rightarrow Factor (1a)

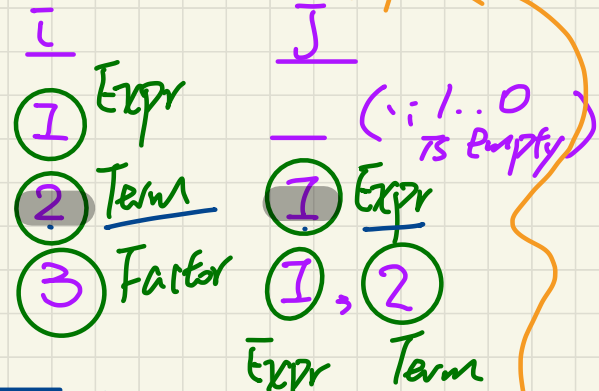
Removing Left-Recursions (1a)

1 **ALGORITHM:** RemoveLR
 2 **INPUT:** CFG $G = (V, \Sigma, R, S)$
 3 **ASSUME:** G has no ϵ -productions
 4 **OUTPUT:** G' s.t. $G' \equiv G$, G' has no
 5 **indirect** & **direct** left-recursions
 6 **PROCEDURE:**
 7 impose an order on V : $\langle\langle A_1, A_2, \dots, A_n \rangle\rangle$
 8 for i : 1 .. n :
 9 for j : 1 .. $i-1$:
 10 if $\exists A_i \rightarrow A_j \gamma \in R \wedge A_j \rightarrow \delta_1 \mid \delta_2 \mid \dots \mid \delta_m \in R$ then
 11 replace $A_i \rightarrow A_j \gamma$ with $A_i \rightarrow \delta_1 \gamma \mid \delta_2 \gamma \mid \dots \mid \delta_m \gamma$
 12 end
 13 \rightarrow for $A_i \rightarrow A_i \alpha \mid \beta \in R$:
 14 replace it with: $(A_i) \rightarrow \beta A'_i, A'_i \rightarrow \alpha A'_i \mid \epsilon$

Exercise
Re-order
variables
and
re-exec.
algo.

nothing to do when $i=1$

Exercise replacement (P2?)



$\bar{i} = 1$ P1 not applicable

$\bar{i} = 2, \bar{j} = 1$

P2 identifies
Expr \rightarrow Expr.

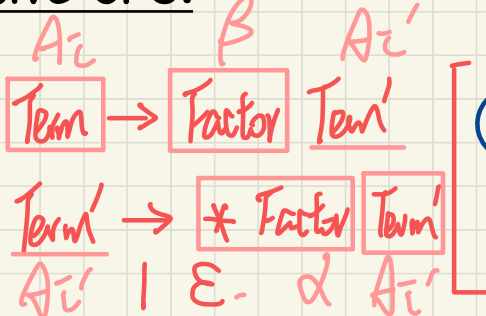
P1 $A_i \rightarrow A_j \dots ?$

Term \rightarrow Expr ? $\times \rightarrow$ nothing

P2 $A_i \rightarrow A_i \cdot A_i ?$
 $\text{Term} \xrightarrow{A_i} \text{Term} * \text{Factor}$
 | Factor β

Directly Left-Recursive CFG:

Expr	\rightarrow	Expr + Term
		Term
Term	\rightarrow	Term * Factor
		Factor
Factor	\rightarrow	(Expr)
		a



Removing Left-Recursions (1b)

Exercise

```
1  ALGORITHM: RemoveLR
2  INPUT: CFG  $G = (V, \Sigma, R, S)$ 
3  ASSUME:  $G$  has no  $\epsilon$ -productions
4  OUTPUT:  $G'$  s.t.  $G' \equiv G$ ,  $G'$  has no
5           indirect & direct left-recursions
6  PROCEDURE:
7      impose an order on  $V$ :  $\langle\langle A_1, A_2, \dots, A_n \rangle\rangle$ 
8      for  $i$ : 1 ..  $n$ :
9          for  $j$ : 1 ..  $i-1$ :
10             if  $\exists A_i \rightarrow A_j \gamma \in R \wedge A_j \rightarrow \delta_1 \mid \delta_2 \mid \dots \mid \delta_m \in R$  then
11                 replace  $A_i \rightarrow A_j \gamma$  with  $A_i \rightarrow \delta_1 \gamma \mid \delta_2 \gamma \mid \dots \mid \delta_m \gamma$ 
12             end
13         for  $A_i \rightarrow A_i \alpha \mid \beta \in R$ :
14             replace it with:  $A_i \rightarrow \beta A'_i, A'_i \rightarrow \alpha A'_i \mid \epsilon$ 
```

Directly Left-Recursive CFG:

```
Expr  → Expr + Term
      | Expr - Term
      | Term
Term   → Term * Factor
      | Term / Factor
      | Factor
```