Lecture 19 - July 29

Syntactic Analysis

Removing Left Recursions: Algorithm
Tracing: Removing Direct LRs

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

```
ALGORITHM: RemoveLR
  INPUT: CFG G = (V, Σ, R, S)
  ASSUME: G has no ε-productions
  OUTPUT: G′ s.t. G′ ≡ G, G′ has no
            indirect & direct left-recursions
PROCEDURE:
  impose an order on V: ⟨⟨A₁, A₂, . . . , Aₙ⟩⟩
  for i: 1 .. n:
    for j: 1 .. i−1:
      if ∃ Aᵢ → Aⱼγ ∈ R ∧ Aⱼ → δ₁ | δ₂ | . . . | δₘ ∈ R then
        replace Aᵢ → Aⱼγ with Aᵢ → δ₁γ | δ₂γ | . . . | δₘγ
      end
  for Aᵢ → Aᵢα | β ∈ R:
    replace it with: Aᵢ → βAᵢ′, Aᵢ′ → αAᵢ′ | ϵ
```

ε-productions: $A \to \varepsilon$

P1: eliminate indirect LRs (presumptuously)

P2: eliminate direct LRs.

γ: arbitrary mix of terminals and non-terminals

Criteria:
1. correct
2. left-recursion removed

left-recursive: $A_i \to A_i \alpha \mid \beta$

① $A_i \Rightarrow \beta$

② A_i "$\beta \alpha^*$"

$\Rightarrow A_i \alpha$

$\Rightarrow A_i \alpha \alpha$

$\Rightarrow \beta \alpha \alpha$

right-recursive: $A_i \to \beta A_i'$, $A_i' \to \alpha A_i' \mid \varepsilon$

Removing Left-Recursions (1a)

Why not checking Term (2) → Factor (3)?

```
1  ALGORITHM: RemoveLR
2    INPUT: CFG G = (V, Σ, R, S)
3    ASSUME: G has no ϵ-productions
4    OUTPUT: G' s.t. G' ≡ G, G' has no
5            indirect & direct left-recursions
6  PROCEDURE:
7    impose an order on V: ⟨(A1, A2, ..., An)⟩
8    for i: 1 .. n:
9      for j: 1 .. i − 1:
10       if ∃ Ai → Ajγ ∈ R ∧ Aj → δ1 | δ2 | ... | δm ∈ R then
11         replace Ai → Ajγ with Ai → δ1γ | δ2γ | ... | δmγ
12       end
13   for Ai → Aiα | β ∈ R:
14     replace it with: Ai → βAi', Ai' → αAi' | ϵ
```

nothing to do when i == 1

i: 1 Expr, 2 Term, 3 Factor

j: (j: 1..0 is empty) 1 Expr; 1, 2 Expr Term

Exercise replacement (P2)

i == 1 → P1 not applicable, P2 identifies Expr → Expr..

i == 2, j == 1

(P1) Ai → Aj... ? Term → Expr ? X → do nothing

(P2) Ai → Ai... ? α

Term → Term * Factor | Factor β

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

Directly Left-Recursive CFG:

(1)	Expr	→	Expr + Term	
		|	Term	
(2)	Term	→	Term * Factor	
		|	Factor	
(3)	Factor	→	(Expr)	
		|	a	

Ai β Ai'

Term → Factor Term'

Term' → * Factor Term' | ϵ

Ai' α Ai'

Removing Left-Recursions (1b)

Exercise

```
1  ALGORITHM: RemoveLR
2    INPUT: CFG G = (V, Σ, R, S)
3    ASSUME: G has no ϵ-productions
4    OUTPUT: G′ s.t. G′ ≡ G, G′ has no
5            indirect & direct left-recursions
6  PROCEDURE:
7    impose an order on V: ⟨(A1, A2, ..., An)⟩
8    for i: 1 .. n:
9      for j: 1 .. i − 1:
10       if ∃ Ai → Ajγ ∈ R ∧ Aj → δ1 | δ2 | ... | δm ∈ R then
11         replace Ai → Ajγ with Ai → δ1γ | δ2γ | ... | δmγ
12       end
13     for Ai → Aiα | β ∈ R:
14       replace it with: Ai → βA′i, A′i → αA′i | ϵ
```

Directly Left-Recursive CFG:

$$
\begin{array}{lcl}
\mathit{Expr} & \rightarrow & \mathit{Expr} + \mathit{Term} \\
 & | & \mathit{Expr} - \mathit{Term} \\
 & | & \mathit{Term} \\
\mathit{Term} & \rightarrow & \mathit{Term} * \mathit{Factor} \\
 & | & \mathit{Term} / \mathit{Factor} \\
 & | & \mathit{Factor}
\end{array}
$$