

# Lecture 13 – Parse Trees and Ambiguity

## COSE215: Theory of Computation

Jihyeok Park



2023 Spring

- A context-free grammar (CFG):

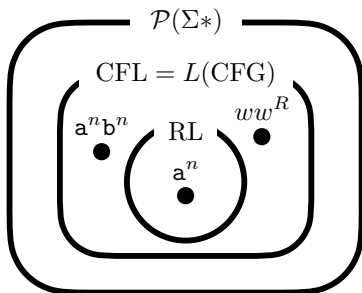
$$G = (V, \Sigma, S, P)$$

- The **language** of a CFG  $G$ :

$$L(G) = \{w \in \Sigma^* \mid S \Rightarrow^* w\}$$

- A language  $L$  is a **context-free language (CFL)**:

$$\exists \text{ CFG } G. L(G) = L$$



## 1. Parse Trees

- Definition

- Yields

- Relationship between Parse Trees and Derivations

## 2. Ambiguity

- Ambiguous Grammars

- Eliminating Ambiguity

- Inherent Ambiguity

Consider the following CFG for arithmetic expressions:

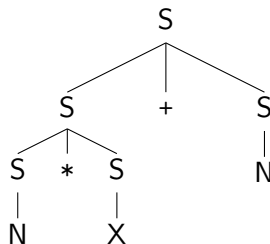
$$S \rightarrow N \mid X \mid S+S \mid S*S \mid (S)$$

$$N \rightarrow 0 \mid \dots \mid 9 \mid 0N \mid \dots \mid 9N$$

$$X \rightarrow a \mid \dots \mid z$$

A derivation and parse tree for a sentential form  $N*X+N$ :

$$\begin{aligned} S &\Rightarrow S+S \\ &\Rightarrow S*S+S \\ &\Rightarrow N*S+S \\ &\Rightarrow N*X+S \\ &\Rightarrow N*X+N \end{aligned}$$



## Definition (Parse Trees)

For a given CFG  $G = (V, \Sigma, S, P)$ , **parse trees** are trees satisfying:

- 1 The **root node** is labeled with the **start variable**  $S$ .
- 2 Each **internal node** is labeled with a **variable**  $A \in V$ .

If its children are labeled with:

$$X_1, X_2, \dots, X_k$$

from the left to the right, then  $A \rightarrow X_1X_2 \cdots X_k \in P$ .

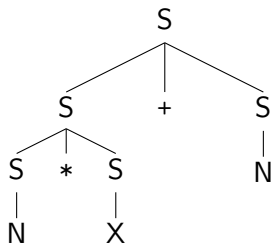
- 3 Each **leaf node** is labeled with a variable, symbol, or  $\epsilon$ . However, if a leaf node is labeled with  $\epsilon$ , it must be the only child of its parent.

$$S \rightarrow N \mid X \mid S+S \mid S*S \mid (S)$$

$$N \rightarrow 0 \mid \dots \mid 9 \mid 0N \mid \dots \mid 9N$$

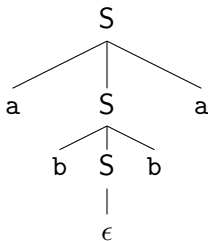
$$X \rightarrow a \mid \dots \mid z$$

A parse tree:



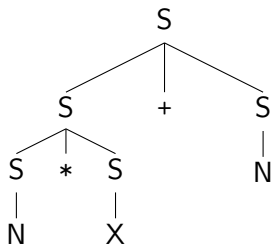
$$S \rightarrow \epsilon \mid aSa \mid bSb$$

A parse tree:

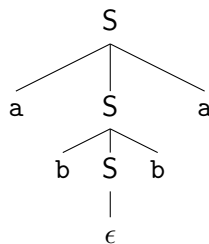


## Definition (Yields)

The sequence obtained by concatenating the labels (without  $\epsilon$ ) of the leaf nodes of a parse tree is called the **yield** of the parse tree.



Its yield is  $N*X+N$ .



Its yield is abba.

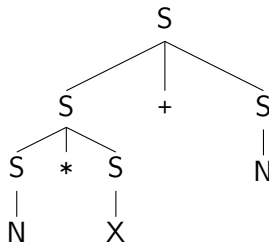


## Theorem (Parse Trees and Derivations)

For a given CFG  $G = (V, \Sigma, S, P)$ , for any sequence of variables or symbols  $w \in (V \cup \Sigma)^*$ :

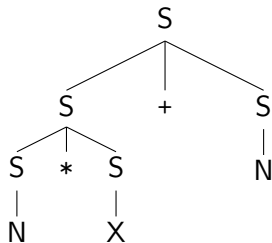
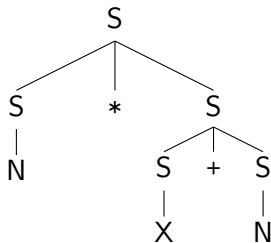
$$S \Rightarrow^* w \iff \exists \text{ parse tree } T. \text{ s.t. } T \text{ yields } w$$

$S \Rightarrow S+S$   
 $\Rightarrow S*S+S$   
 $\Rightarrow N*S+S$   
 $\Rightarrow N*X+S$   
 $\Rightarrow N*X+N$



$$\begin{aligned} S &\rightarrow N \mid X \mid S+S \mid S*S \mid (S) \\ N &\rightarrow 0 \mid \dots \mid 9 \mid 0N \mid \dots \mid 9N \\ X &\rightarrow a \mid \dots \mid z \end{aligned}$$

Actually, there are two distinct parse trees for a sentential form  $N*X+N$ :



## Definition (Ambiguous Grammar)

A **context-free grammar**  $G = (V, \Sigma, S, P)$  is **ambiguous** if there exist a word  $w \in \Sigma^*$  and two distinct parse trees whose yields are  $w$ . If not,  $G$  is **unambiguous**.

## Theorem

Let  $G = (V, \Sigma, S, P)$  be a CFG. Then, the following numbers are equal for any sequence of variables or symbols  $w \in (V \cup \Sigma)^*$ :

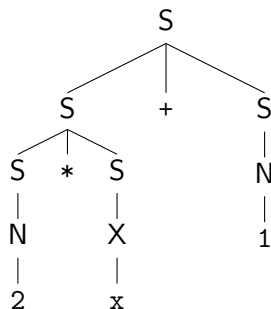
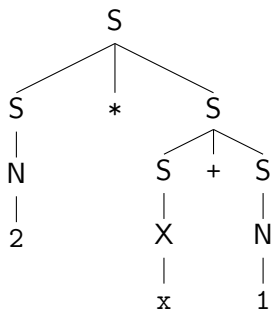
- 1 The number of parse trees whose yields are  $w$ .
- 2 The number of left-most derivations for  $w$ .
- 3 The number of right-most derivations for  $w$ .

$$S \rightarrow N \mid X \mid S+S \mid S*S \mid (S)$$

$$N \rightarrow 0 \mid \dots \mid 9 \mid 0N \mid \dots \mid 9N$$

$$X \rightarrow a \mid \dots \mid z$$

This grammar is **ambiguous** because there are **two** parse trees for  $2 * x + 1$ :



$$\begin{aligned} S &\rightarrow N \mid X \mid S+S \mid S*S \mid (S) \\ N &\rightarrow 0 \mid \dots \mid 9 \mid 0N \mid \dots \mid 9N \\ X &\rightarrow a \mid \dots \mid z \end{aligned}$$

There are **two** left-most derivations for  $2 * x + 1$ :

① Applying the production rule  $S \rightarrow S*S$  first:

$$\begin{aligned} S &\xRightarrow{\text{lm}} S*S && \xRightarrow{\text{lm}} N*S && \xRightarrow{\text{lm}} 2*S && \xRightarrow{\text{lm}} 2*S+S \\ &\xRightarrow{\text{lm}} 2*X+S && \xRightarrow{\text{lm}} 2*x+S && \xRightarrow{\text{lm}} 2*x+N && \xRightarrow{\text{lm}} 2*x+1 \end{aligned}$$

② Applying the production rule  $S \rightarrow S+S$  first:

$$\begin{aligned} S &\xRightarrow{\text{lm}} S+S && \xRightarrow{\text{lm}} S*S+S && \xRightarrow{\text{lm}} N*S+S && \xRightarrow{\text{lm}} 2*S+S \\ &\xRightarrow{\text{lm}} 2*X+S && \xRightarrow{\text{lm}} 2*x+S && \xRightarrow{\text{lm}} 2*x+N && \xRightarrow{\text{lm}} 2*x+1 \end{aligned}$$

Unfortunately,

- There is **NO** general algorithm to determine a CFG is ambiguous.
- There is **NO** general algorithm to remove ambiguity from a CFG.

Fortunately, there are well-known techniques to **eliminate** the ambiguity in a grammar commonly used in programming languages.

$$S \rightarrow N \mid X \mid S+S \mid S*S \mid (S)$$

$$N \rightarrow 0 \mid \dots \mid 9 \mid 0N \mid \dots \mid 9N$$

$$X \rightarrow a \mid \dots \mid z$$

An equivalent but unambiguous grammar:

$$S \rightarrow T \mid S+T$$

$$T \rightarrow F \mid T*F$$

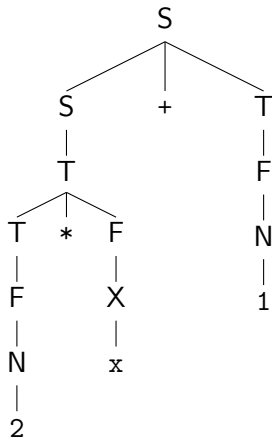
$$F \rightarrow N \mid X \mid (S)$$

$$N \rightarrow 0 \mid \dots \mid 9 \mid 0N \mid \dots \mid 9N$$

$$X \rightarrow a \mid \dots \mid z$$

Now, the unique parse tree for  $2 * x + 1$  is:

$S \rightarrow T \mid S + T$   
 $T \rightarrow F \mid T * F$   
 $F \rightarrow N \mid X \mid (S)$   
 $N \rightarrow 0 \mid \dots \mid 9 \mid 0N \mid \dots \mid 9N$   
 $X \rightarrow a \mid \dots \mid z$



First, analyze why the original grammar is ambiguous.

$$S \rightarrow N \mid X \mid S+S \mid S*S \mid (S)$$

$$N \rightarrow 0 \mid \dots \mid 9 \mid 0N \mid \dots \mid 9N$$

$$X \rightarrow a \mid \dots \mid z$$

- The **precedence** of + and \* is not specified.
  - For example, two parse trees for  $1 * 2 + 3$  interpreted as:

$$1 * (2 + 3) \quad \text{and} \quad (1 * 2) + 3$$

- Let's give \* higher precedence than + to interpret it as  $(1 * 2) + 3$ .
- The **associativity** of + (or \*) is not specified.
  - For example, two parse trees for  $1 + 2 + 3$  interpreted as:

$$1 + (2 + 3) \quad \text{and} \quad (1 + 2) + 3$$

- Let's give the left-associativity to + to interpret it as  $(1 + 2) + 3$ .



To enforce the **precedence**, define new variables  $F$  for factors and  $T$  for terms:

- A **factor** is a number, a variable, or a parenthesized expression:

$$42, \quad x, \quad (1 + 2), \quad \dots$$

In the grammar,  $F$  is defined as:

$$F \rightarrow N \mid X \mid (S)$$

- A **term** is the multiplication of one or more factors:

$$42, \quad 2 * x, \quad 2 * (1 + 2), \quad 1 * (x * y) * z, \quad \dots$$

In the grammar,  $T$  is defined as:

$$T \rightarrow F \mid T * F$$

- An **expression** is the addition of one or more terms:

$$42, \quad 1 + 2, \quad 1 + 2 * 3, \quad (1 + 2) * 3 + 4), \quad \dots$$

In the grammar,  $S$  is defined as:

$$S \rightarrow T \mid S + T$$

The unambiguous grammar is:

$$\begin{aligned} S &\rightarrow T \mid S+T \\ T &\rightarrow F \mid T*F \\ F &\rightarrow N \mid X \mid (S) \\ N &\rightarrow 0 \mid \dots \mid 9 \mid 0N \mid \dots \mid 9N \\ X &\rightarrow a \mid \dots \mid z \end{aligned}$$

- This grammar supports the **left-associativity** of + and \*. How?
- How to support the **right-associativity** of + and \*?

$$\begin{aligned} S &\rightarrow T \mid T+S \\ T &\rightarrow F \mid F*T \\ &\dots \end{aligned}$$

So far, we have discussed the **ambiguity** for grammars.  
We will now discuss the **inherent ambiguity** for languages.

## Definition (Inherent Ambiguity)

A language  $L$  is **inherently ambiguous** if all CFGs whose languages are  $L$  are ambiguous. (i.e. there is no unambiguous grammar for  $L$ )

For example, the following language is **inherently ambiguous**:

$$L = \{a^i b^j c^k \mid i, j, k \geq 0 \wedge (i = j \vee j = k)\}$$

An example of ambiguous grammar for  $L$  is:

$$\begin{aligned} S &\rightarrow L \mid R \\ L &\rightarrow A \mid Lc \\ A &\rightarrow \epsilon \mid aAb \\ R &\rightarrow B \mid aR \\ B &\rightarrow \epsilon \mid bBc \end{aligned}$$

- Midterm exam will be given in class.
- **Date:** 14:00-15:15 (1 hour 15 minutes), April 24 (Mon.).
- **Location:** 302, Aegineung (애기능생활관)
- **Coverage:** Lectures 1 – 13
- **Format:** short- or long-answer questions, including proofs
  - Closed book, closed notes
  - No questions about Scala code in the midterm exam.
- **Note that there is a lecture on April 26 (Wed.).**

## 1. Parse Trees

- Definition

- Yields

- Relationship between Parse Trees and Derivations

## 2. Ambiguity

- Ambiguous Grammars

- Eliminating Ambiguity

- Inherent Ambiguity

- Pushdown Automata (PDA)

Jihyeok Park

[jihyeok\\_park@korea.ac.kr](mailto:jihyeok_park@korea.ac.kr)

<https://plrg.korea.ac.kr>