# 01204211 Discrete Mathematics
# Lecture 11b: Context-free languages and grammars (2)[1]

Jittat Fakcharoenphol

September 19, 2024

[1]Based on lecture notes of *Models of Computation* course by Jeff Erickson.

# Review: Definition

A **context-free grammer** consists of the following components:

- a finite set $\Sigma$, a set of *symbols* (or *terminals*),
- a finite set $\Gamma$ disjoint from $\Sigma$, a set of *non-terminals* (you can think of them as variables),
- a finite set $R$ of *production rules* of the form $A \to w$ where $A \in \Gamma$ and $w \in (\Sigma \cup \Gamma)^*$ is a string of symbols and variable, and
- a *starting* non-terminal (usually the non-terminal of the first production rule).

# Review: Applying the rules

If you have strings $x, y, z \in (\Sigma \cup \Gamma)^*$ and the production rule

$$A \to y,$$

You can apply the rule to the string $xAz$. This yields the string

$$xyz.$$

We use the notation

$$xAz \rightsquigarrow xyz$$

to describe this application.

# Review: Derivation

We say that $z$ derives from $x$ if we can obtain $z$ from $x$ by production rule applications, denoted by $x \rightsquigarrow^* z$.
Formally, for any string $x, z \in (\Sigma \cup \Gamma)^*$, we say that $x \rightsquigarrow^* z$ if either

- $x = z$, or
- $x \rightsquigarrow y$ and $y \rightsquigarrow^* z$ for some string $y \in (\Sigma \cup \Gamma)^*$.

# Review: $L(w)$

The *language* $L(w)$ of string $w \in (\Sigma \cup \Gamma)^*$ is the set of all strings in $\Sigma^*$ that derive from $w$, i.e.,

$$L(w) = \{x \in \Sigma^* \mid w \rightsquigarrow^* x\}.$$

The language **generated by** a context-free grammar $G$, denoted by $L(G)$ is the language of its starting non-terminal.
A language $L$ is **context-free** if there exists some context-free grammar $G$ such that $L(G) = L$.

# Review: Parse tree

- 00011

$$
\begin{aligned}
S &\rightarrow A \mid B \\
A &\rightarrow 0A \mid 0C \\
B &\rightarrow B1 \mid C1 \\
C &\rightarrow \varepsilon \mid 0C1
\end{aligned}
$$

# Ambiguity

- $1 + 1 + 1 + 1 + 1$

$$S \rightarrow 1 \mid S + S \mid S * S$$

- A string $w$ is **ambiguous** with respect to a grammar $G$ if more than one parse tree for $w$ exists.
- A grammar $G$ is **ambiguous** if some string is ambiguous with respect to $G$.

# More example

Palindrome in $\{0, 1\}^*$

$$S \rightarrow 0S0 \mid 1S1 \mid 1 \mid 0 \mid \varepsilon$$

Consider the following grammar

$$S \longrightarrow \mathtt{0}S\mathtt{1} \mid \varepsilon$$

To show that

$$L(S) = \{\mathtt{0}^n\mathtt{1}^n \mid n \geq 0\},$$

we have to prove

- $L(S) \supseteq \{\mathtt{0}^n\mathtt{1}^n \mid n \geq 0\}$, and
- $L(S) \subseteq \{\mathtt{0}^n\mathtt{1}^n \mid n \geq 0\}$.

Consider the grammar $S \longrightarrow 0S1 \mid \varepsilon$.

## Lemma 1

*$S \rightsquigarrow^* 0^n1^n$ for every non-negative integer $n$.*

## Proof.

Consider any non-negative integer $n$.
**Induction Hypothesis:** Assume that for every non-negative integer $k < n$, $S \rightsquigarrow^* 0^k1^k$.
There are two cases to consider.

- Case 1: $n = 0$.
- Case 2: $n > 0$. From I.H., we know that

$$S \rightsquigarrow^* 0^{n-1}1^{n-1}.$$

We can apply rule $S \longrightarrow 0S1$ to obtain $0^n1^n$, i.e.,

$$S \longrightarrow 0S1 \rightsquigarrow^* 00^{n-1}1^{n-1}1 = 0^n1^n.$$

In both cases, we conclude that $S \rightsquigarrow^* 0^n1^n$, as required.

Consider the following grammar

$$S \longrightarrow 0S1 \mid \varepsilon$$

## Lemma 2

$L(S) = \{0^n 1^n \mid n \geq 0\}$

## Proof.

Consider any string $w \in L(C)$. We show that $w = 0^n 1^n$ for some non-negative integer $n$.
**I.H.:** Assume that for any string $x \in L(C)$ such that $|x| < |w|$, $x = 0^k 1^k$ for some non-negative integer $k$.
There are 2 cases:
**Case 1:** $w = \varepsilon$.
**Case 2:** $w = 0x1$ for some $x \in L(C)$. Since $|x| = |w| - 2 < |w|$, we can apply I.H., and get that $x = 0^k 1^k$;thus $w = 00^k 1^k 1$, i.e., $w = 0^n 1^n$ where $n = k + 1$, as required. □

# Careful

- When using inductive proof, you have to ensure that each part of the string $w$ is shorter than $w$.
- Consider this grammar

$$S \longrightarrow \varepsilon \mid SS \mid 0S1 \mid 1S0.$$

- When $w$ is created by rule $S \to SS$, we know that $w = xy$ for $x, y \in L(S)$.
- Do we know that $|x| < |w|$ and $|y| < |w|$?
- We can consider a **minimum-length derivation** in the proof to avoid this problem.

Consider grammar $S \longrightarrow \varepsilon \mid SS \mid 0S1 \mid 1S0$. For every string $w \in L(S)$, we have $\#(0, w) = \#(1, w)$, where $\#(a, w)$ is the number of occurrences of $a$ in $w$.

**Proof.**

Consider $w \in L(S)$. Fix a minimum-length derivation of $w$.
Induction Hypothesis: Assume that for any string $x \in L(S)$ such that $|x| < |w|$, we have $\#(0, x) = \#(1, x)$.
There are four cases to consider, depending on the first production in this derivation.

- Case 1: The first production is $S \longrightarrow \varepsilon$.
- Case 2: The first production is $S \longrightarrow 0S1$. Case 3: The first production is $S \longrightarrow 1S0$.
- Case 4: The first production is $S \longrightarrow SS$. In this case $w = xy$ for some $x, y \in L(S)$. Since we assume the minimum-length derivation, $x$ and $y$ cannot be $\varepsilon$ because in that case we can shorten the derivation of $w$.

  From I.H., we know that $\#(0, x) = \$(1, x)$ an $\#(0, y) = \#(1, y)$; thus,

$$\begin{aligned} \#(0, w) &= \#(0, x) + \#(0, y) \\ &= \#(1, x) + \#(1, y) = \#(1, w) \end{aligned}$$

In all cases, we conclude that $\#(0, w) = \#(1, w)$.

# Examples: Not palindromes

Strings in $(0+1)^*$ that are not palindromes.

$$
\begin{aligned}
S &\longrightarrow 0S0 \mid 1S1 \mid 0Z1 \mid 1Z0 \\
Z &\longrightarrow \varepsilon \mid 0Z \mid 1Z
\end{aligned}
$$

Why does this work?

# Strings with the same number of 0s and 1s

$$S \longrightarrow \varepsilon \mid SS \mid 0S1 \mid 1S0.$$

We already show that every string in $L(S)$ contains the same number of 0s and 1s. Why does it contain all possible required strings?

# Strings in which the number of 0s is greater than or equal to the number of 1s

We can start with the previous grammar

$$S \longrightarrow \varepsilon \mid SS \mid 0S1 \mid 1S0.$$

And try to add more rules.

$$S \longrightarrow \varepsilon \mid SS \mid 0S1 \mid 1S0 \mid 0S \mid S0.$$

# Strings with different numbers of 0s and 1s

We can start with the previous grammar $E$ of strings with equal number of 0 and 1.

$$E \longrightarrow \varepsilon \mid EE \mid 0E1 \mid 1E0.$$

There are two cases.

$$S \longrightarrow O \mid I$$

$$O \longrightarrow E0O \mid E0E$$

How about $I$?

$$I \longrightarrow E1I \mid E1E$$

# Balanced parentheses

$$S \longrightarrow (S) \mid SS \mid \varepsilon$$

$$S \longrightarrow (S)S \mid \varepsilon$$

# Mutual induction

Consider grammar

$$S \longrightarrow \mathtt{0}A\mathtt{1} \mid \varepsilon \qquad\qquad A \longrightarrow \mathtt{1}S\mathtt{0} \mid \varepsilon$$

What is $L(S)$?

From inspection, we may guess that $L(S) = (\mathtt{01})^*$. But how can we prove that?

To prove $L(S) = (\mathtt{01})^*$, we must also prove $L(A) = (\mathtt{10})^*$ *at the same time*.