# Context-Free Grammars

Chanathip Namprempre

Computer Science
Reed College

# Outline

# Outline

- We learned that some languages are not regular. Finite automata cannot handle them.
- A more powerful method to describe some of these languages is to use context-free grammars.
- When you learn a new computer language, you must learn the syntax for that language (e.g. `printf(‘‘%d\n.’’, x);`). The syntax is often specified in the form of grammars.
- A compiler must use grammars to parse a given program to figure out what the program means so that it can generate machine code for the program.
- The languages associated with context-free grammars are called context-free languages. They include regular languages and many more.

# Example of a Context-Free Grammar

This is what a grammar looks like:

$$
\begin{aligned}
A &\rightarrow 0A1 \\
A &\rightarrow B \\
B &\rightarrow \#
\end{aligned}
$$

The components are

- substitution rules, aka productions
- variable
- terminals
- start variable

# Example of a Derivation

This is what a grammar looks like:

$$\begin{array}{rcl} A & \to & 0A1 \\ A & \to & B \\ B & \to & \# \end{array}$$

This grammar generates the string $000\#111$. The sequence of substitutions to obtain this string is called a derivation.

$$A \Rightarrow 0A1 \Rightarrow 00A11 \Rightarrow 000A111 \Rightarrow 000B111 \Rightarrow 000\#111$$

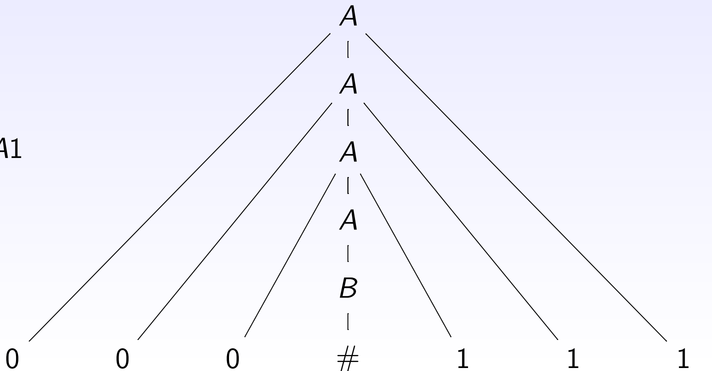
## Derivation and parse tree

The derivation

$$A \Rightarrow 0A1 \Rightarrow 00A11 \Rightarrow 000A111 \Rightarrow 000B111 \Rightarrow 000\#111$$

for 000#111 can be represented using a parse tree.

$$\begin{aligned} A &\rightarrow 0A1 \\ A &\rightarrow B \\ B &\rightarrow \# \end{aligned}$$



# Language of a Grammar

All strings generated from a grammar is called the language of that grammar.

Let $G_1$ be the grammar:

$$\begin{array}{rcl} A & \rightarrow & 0A1 \mid B \\ B & \rightarrow & \# \end{array}$$

Then, the language of $G_1$, denoted $L(G_1)$, is $\{0^n\#1^n \mid n \geq 0\}$.

Any language that can be generated by some context-free grammar is called a context-free language (CFL).

# More example of a Context-Free Grammar

The following grammar $G_2$ describes a fragment of the English language.

$$
\begin{array}{rcl}
<\text{sentence}> & \rightarrow & <\text{noun-phrase}><\text{verb-phrase}> \\
<\text{noun-phrase}> & \rightarrow & <\text{cmplx-noun}> \mid <\text{cmplx-noun}><\text{prep-phrase}> \\
<\text{verb-phrase}> & \rightarrow & <\text{cmplx-verb}> \mid <\text{cmplx-verb}><\text{prep-phrase}> \\
<\text{prep-phrase}> & \rightarrow & <\text{prep}><\text{cmplx-noun}> \\
<\text{cmplx-noun}> & \rightarrow & <\text{article}><\text{noun}> \\
<\text{cmplx-verb}> & \rightarrow & <\text{verb}> \mid <\text{verb}><\text{noun-phrase}> \\
<\text{article}> & \rightarrow & \text{a} \mid \text{the} \\
<\text{noun}> & \rightarrow & \text{cat} \mid \text{dog} \mid \text{paw} \\
<\text{verb}> & \rightarrow & \text{touches} \mid \text{likes} \mid \text{sees} \\
<\text{prep}> & \rightarrow & \text{with}
\end{array}
$$

# More example of a Context-Free Grammar

An example of a derivation for $G_2$ on the string “a cat sees”.

$$
\begin{aligned}
<\text{sentence}> &\Rightarrow\ <\text{noun-phrase}><\text{verb-phrase}> \\
&\Rightarrow\ <\text{cmplx-noun}><\text{verb-phrase}> \\
&\Rightarrow\ <\text{article}><\text{noun}><\text{verb-phrase}> \\
&\Rightarrow\ \text{a} <\text{noun}><\text{verb-phrase}> \\
&\Rightarrow\ \text{a cat} <\text{verb-phrase}> \\
&\Rightarrow\ \text{a cat} <\text{cmplx-verb}> \\
&\Rightarrow\ \text{a cat} <\text{verb}> \\
&\Rightarrow\ \text{a cat sees}
\end{aligned}
$$

# More examples of a Context-Free Grammar

The following grammar $G_3$ describes a language of balanced parentheses.

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

Try deriving ()(())((())).

The following grammar $G_4$ describes a language of mathematical expressions.

$$\begin{aligned} <\text{expr}> \quad &\to \quad <\text{expr}> + <\text{term}> \mid <\text{term}> \\ <\text{term}> \quad &\to \quad <\text{term}> \times <\text{factor}> \mid <\text{factor}> \\ <\text{factor}> \quad &\to \quad (<\text{expr}>) \mid a \end{aligned}$$

Try deriving $a + a \times a$ and $(a + a) \times a$.

# Formal definition of a context-free grammar

## Definition

A context-free grammar is a 4-tuple $(V, \Sigma, R, S)$ where

1. $V$ is a finite set called the variables,
2. $\Sigma$ is a finite set, disjoint from $V$, called the terminals,
3. $R$ is a finite set of rules, with each rule being a variable or a string of variables and terminals, and
4. $S \in V$ is the start variable.

# Example

How would you write the grammar $G_1$ formally?

$$
\begin{aligned}
A &\rightarrow 0A1 \mid B \\
B &\rightarrow \#
\end{aligned}
$$

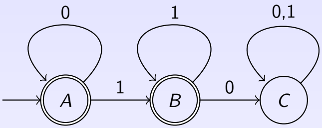
$G_1 = (V, \Sigma, R, S)$ where

1. [Variables] $V = \{A, B\}$
2. [Terminals] $\Sigma = \{0, 1, \#\}$
3. [Rules] $R = \{\ A \rightarrow 0A1 \mid B\ ,\ B \rightarrow \#\ \}$
4. [Start variable] $S = A$

# Outline

# Let's design CFGs

A grammar for $0^*1^*$?




$$\begin{aligned} A &\rightarrow 0A \mid 1B \mid \varepsilon \\ B &\rightarrow 1B \mid 0C \mid \varepsilon \\ C &\rightarrow 0C \mid 1C \end{aligned}$$

## Idea

- For every transition $\delta(q_i, a) = q_j$, add rule $q_i \rightarrow aq_j$.
- For every accept state $q$, add rule $q \rightarrow \varepsilon$.

# Let's design CFGs

1. the language of balanced parentheses
2. $\Sigma^*$
3. $\{0^n1^n \mid n \geq 0\} \cup \{1^n0^n \mid n \geq 0\}$
4. $\{w \mid w$ has an even number of 1s $\}$
5. $\{w \mid w$ contains at least three 1s $\}$
6. $\{w \mid w$ starts and ends with the same symbol $\}$
7. $\{w \mid$ the length of $w$ is odd $\}$
8. $\{w \mid$ the length of $w$ is odd and its middle symbol is 0 $\}$
9. $\{w \mid w = w^{\mathcal{R}}$, that is, $w$ is a palindrome $\}$
10. $\{\}$

# Outline

# Ambiguous Grammars

Some grammars are ambiguous. This means that there is some string that the grammar generates ambiguously. For example, consider the following grammar $G_5$:

$$< \text{expr} > \; \rightarrow \; < \text{expr} > + < \text{expr} > \mid < \text{expr} > \times < \text{expr} > \mid (< \text{expr} >) \mid a$$

This grammar generates the string $a + a \times a$ ambiguously. This means that there are at least two ways to parse the string using the given grammar.

$G_5$ generates exactly the same language as that generated by $G_4$ but $G_4$ is unambiguous.

# Ambiguity Defined

## Definition

A string $w$ is derived ambiguously in a CFG $G$ if it has two or more different leftmost derivations. Grammar $G$ is ambiguous if it generates some strings ambiguously.

$G_2$ is also ambiguous. Try constructing the parse trees for the sentence "the girl touches the boy with the flower".

# Outline

# Chomsky Normal Form

## Definition

A CFG is in Chomsky Normal Form if every rule is of the form

$$\begin{aligned} A &\to BC \\ A &\to a \end{aligned}$$

where $a$ is a terminal and $A, B, C$ are any variables except that $B$ and $C$ may not be the start variable. In addition, we permit the rule $S \to \varepsilon$, where $S$ is the start variable.

## Theorem

*Any CFL is generated by a CFG in Chomsky Normal Form.*

# Converting CFG into Chomsky Normal Form

1. Add new start variable $S_0$ and add a rule $S_0 \to S$ where $S$ is the start variable.
2. Remove all $\varepsilon$-rules $A \to \varepsilon$ not involving the start variable.
3. Remove all unit rules $A \to B$.
4. Convert all rules into the two proper forms.

# Converting any CFGs into Chomsky Normal Form

1.

$$\begin{aligned} S &\rightarrow ASA \mid aB \\ A &\rightarrow B \mid S \\ B &\rightarrow b \mid \varepsilon \end{aligned}$$

2.

$$\begin{aligned} A &\rightarrow BAB \mid B \mid \varepsilon \\ B &\rightarrow 00 \mid \varepsilon \end{aligned}$$

# Outline

# Closure properties

The class of context-free languages are closed under union, reversal, concatenation, and star.

The class of context-free languages are not closed under complementation and intersection.

[Consider the languages $\{a^m b^m c^n \mid m, n \geq 0\}, \{a^m b^n c^n \mid m, n \geq 0\}, \{a^n b^n c^n \mid n \geq 0\}$]