CSC 320 - Lecture 10a

#CNF #textbook #context-free #languages #grammars #CFL #CFG #ambiguous

---

Introduction Theory Computation Textbook

Chapter 2 - Page 101

Context-Free Languages

These grammars can describe certain features that have a recursive structure, which makes them useful in a variety of applications. We note its importance in **parsers** that extract the meaning of a program prior to generating the compiled code or performing the interpreted execution.

Context-Free Languages include all the regular languages and many additional languages.

We will also introduce **pushdown automate**, a class of machines recognizing context-free languages.

All strings generated in this way constitute the **language of the grammar**.

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

The **language of the grammar** is $\{w \in \Sigma^* | S \Rightarrow^* w\}$.

In grammar $G_1, V = \{A, B\}, \Sigma = \{0, 1, \#\}, S = A$, and R is the collection of three rules $A \longrightarrow 0A1|B$, and $B \longrightarrow \#$. In grammar G_2,

$$\begin{aligned} V = \{ & \langle SENTENCE \rangle, \langle NOUN - PHRASE \rangle, \langle VERB - PHRASE \rangle, \\ & \langle PREP - PHRASE \rangle, \langle CMPLZ - NOUN \rangle, \langle CMPLX - VERB \rangle, \\ & \langle ARTICLE \rangle, \langle NOUN \rangle, \langle VERB \rangle, \langle PREP \rangle \}, \end{aligned}$$

and $\Sigma = \{a, b, c, \ldots, z, \text{'' "}\}$. The symbol '' " is the blank symbol, placed invisibly after each word (a, boy, etc.), so the words won't run together.

Designing Context-Free Grammars

The design of context-free grammars requires creativity.

Example

Design a grammar for the language $\{0^n1^n|n \geq 0\} \cup \{1^n0^n|n \geq 0\}$. We first construct the grammar...

$$S_1 \longrightarrow 0S_11|\varepsilon$$

for the language $\{0^n1^n|n \geq 0\}$ and the grammar...

$$S_2 \longrightarrow 1S_20|\varepsilon$$

for the language $\{1^n0^n|n \geq 0\}$ and then add the rule $S \longrightarrow S_1|S_2$ to give the grammar

$$\begin{aligned} S &\longrightarrow S_1|S_2 \\ S_1 &\longrightarrow 0S_11|\varepsilon \\ S_2 &\longrightarrow 1S_20|\varepsilon \end{aligned}$$

Note. Construction a CFG for a language that happens to be regular is easy if you can first construct a DFA for that language. You can convert any DFA into an equivalent CFG.

Ambiguity

Sometimes a grammar can generate the same string in several different ways. This result may be undesirable for certain applications, such as programming languages, where a program should have a unique interpretation.

We say that if a grammar generates the same string in different ways that the string is derived *ambiguously*.

A grammar that is not *ambiguous* is *unambiguous*.

Note. When we say that a grammar generates a string ambiguously, we mean that the string has two different parse trees, not two different derivations. Thus, we use the following to help define: a derivation of a string w in a grammar G is a **leftmost derivation** if at every step the leftmost remaining variable is the one replaced.

Definition. A string w is derived **ambiguously** in context-free grammar G if it has two or more different leftmost derivations. Grammar G is **ambiguous** if it generates some string ambiguously.

Note. Some context-free languages can be generated only by ambiguous grammars. Such languages are called **inherently ambiguous**.

Chomsky Normal Form

Restricted (simplified) constrains on grammar.

Definition. A context-free grammar is in **Chomsky Normal Form** if every rule is of the form

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

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

Example

Let G be the following CFG and convert it to CNF by using the conversion procedure. The series of grammars presented illustrates the steps in the conversion. Rules shown in bold have just been added. Rules shown in ForestGreen have just been removed.

1. The original CFG G is shown on the left. The result of applying the first step to make a new start variable appears on the right.

$$\begin{aligned} S &\longrightarrow ASA|aB \\ A &\longrightarrow B|S \\ B &\longrightarrow b|\varepsilon \end{aligned} \qquad \begin{aligned} \boldsymbol{S_0} &\boldsymbol{\longrightarrow S} \\ S &\longrightarrow ASA|aB \\ A &\longrightarrow B|S \\ B &\longrightarrow b|\varepsilon \end{aligned}$$

Note. $A \longrightarrow B|S$ is not allowed in CNF because S can't be on the right side.

2. Remove ε-rules $B \longrightarrow \varepsilon$, shown on the left, and $A \longrightarrow \varepsilon$, shown on the right.

$$\begin{aligned} S_0 &\longrightarrow S \\ S &\longrightarrow ASA|aB|\boldsymbol{a} \\ A &\longrightarrow B|S|\boldsymbol{\varepsilon} \\ B &\longrightarrow b|\color{ForestGreen}{\varepsilon} \end{aligned} \qquad \begin{aligned} S_0 &\longrightarrow S \\ S &\longrightarrow ASA|aB|a|\boldsymbol{SA}|\boldsymbol{AS}|\boldsymbol{S} \\ A &\longrightarrow B|S|\color{ForestGreen}{\varepsilon} \\ B &\longrightarrow b \end{aligned}$$

For each $W \longrightarrow uBv$ and occurrence of B: add $W \longrightarrow uv$. For $W \longrightarrow B$: add $W \longrightarrow \varepsilon$ unless $W \longrightarrow \varepsilon$ was removed previously.

3. a) Remove unit rules $S \longrightarrow S$, show on the left, and $S_0 \longrightarrow S$, shown on the right.

$$\begin{aligned} S_0 &\longrightarrow S \\ S &\longrightarrow ASA|aB|a|SA|AS|\color{ForestGreen}{S} \\ A &\longrightarrow B|S \\ B &\longrightarrow b \end{aligned} \qquad \begin{aligned} S_0 &\longrightarrow \color{ForestGreen}{S}\color{black}|\boldsymbol{ASA}|\boldsymbol{aB}|\boldsymbol{a}|\boldsymbol{SA}|\boldsymbol{AS} \\ S &\longrightarrow ASA|aB|a|SA|AS \\ A &\longrightarrow B|S \\ B &\longrightarrow b \end{aligned}$$

Given $S_0 \longrightarrow S$: for each appearance of $S \longrightarrow u$ and $S_0 \longrightarrow u$, unless previously removed.

3. b) Remove unit rules $A \longrightarrow B$ and $A \longrightarrow S$.

$$\begin{aligned} S_0 &\longrightarrow ASA|aB|a|SA|AS \\ S &\longrightarrow ASA|aB|a|SA|AS \\ A &\longrightarrow B|S|\boldsymbol{b} \\ B &\longrightarrow b \end{aligned} \qquad \begin{aligned} S_0 &\longrightarrow ASA|aB|a|SA|AS \\ S &\longrightarrow ASA|aB|a|SA|AS \\ A &\longrightarrow S|b|\boldsymbol{ASA}|\boldsymbol{aB}|\boldsymbol{a}|\boldsymbol{SA}|\boldsymbol{AS} \\ B &\longrightarrow b \end{aligned}$$

$$\begin{aligned} S_0 &\longrightarrow ASA|aB|a|SA|AS \\ S &\longrightarrow ASA|aB|a|SA|AS \\ A &\longrightarrow b|ASA|aB|SA|AS \\ B &\longrightarrow b \end{aligned}$$

Given $A \longrightarrow B$: for each appearance of $B \longrightarrow u$ and $A \longrightarrow u$, unless previously removed.

Given $A \longrightarrow S$: for each appearance of $S \longrightarrow u$ add $A \longrightarrow u$, unless previously removed.

4. Convert the remaining rules into the proper form by adding additional variables and rules. The final grammar in CNF is equivalent to G.

$$\begin{aligned} S_0 &\longrightarrow AA_1|UB|a|SA|AS \\ S &\longrightarrow AA_1|UB|a|SA|AS \\ A &\longrightarrow b|AA_1|UB|a|SA|AS \\ A_1 &\longrightarrow SA \\ U &\longrightarrow a \\ B &\longrightarrow b \end{aligned}$$

Note. ASA is too long - only allowed two variables! We also don't like aB because it is a mix of terminals and variables.

Previous Lecture

[Lecture09]

Next Lecture

[Lecture10b]