1 Context-Free Grammars (2.1, Continued)

We continue our notes on context-free grammars.

1.1 Relation Between CFGs and Regular Languages

We know that every regular language is also a context-free language (*however*, not every context-free language is a regular language). There are two approaches to show that this is the case.

1. Start with an arbitrary DFA M, then build a CFG that generates $L(M)$.
2. Build CFGs for $\{\mathtt{a}\}$, $\{\epsilon\}$, and $\emptyset$. Then, show that the class of context-free languages is closed under the regular operations (union, concatenation, Kleene star).

1.1.1 First Approach

Proposition. *Given any DFA M, there is a CFG whose language is $L(M)$.*

Proof. Given a DFA $M = (Q, \Sigma, \delta, q_0, F)$, we can define the CFG $G = (V, \Sigma, R, S)$ where

- $V = \{S_i \mid q_i \in Q\}$
- $R = \{S_i \mapsto \mathtt{a}S_j \mid \delta(q_i, \mathtt{a}) = q_j\} \cup \{S_i \mapsto \epsilon \mid q_i \in F\}$
- $S = S_0$

From this, we need to prove correctness. □

1.1.2 Second Approach

We can build CFGs for $\{\mathtt{a}\}$, $\{\epsilon\}$, and $\emptyset$. Then, show that the class of context-free languages is closed under the regular operations (union, concatenation, Kleene star).

- If $L = \{\mathtt{a}\}$, where $\mathtt{a}$ is some arbitrary character in the alphabet, then we have the CFG (V, Σ, R, S) where
  - $V = \{S\}$
  - $R = \{S \mapsto \mathtt{a}\}$
- If $L = \{\epsilon\}$, then we have the CFG (V, Σ, R, S) where
  - $V = \{S\}$
  - $R = \{S \mapsto \epsilon\}$
- If $L = \emptyset$, then we have the CFG (V, Σ, R, S) where
  - $V = \{S\}$
  - $R = \emptyset$ (or $R = \{S \mapsto S\}$)

Suppose we have $G_1 = \{V_1, \Sigma, R_1, S_1\}$ and $G_2 = \{V_2, \Sigma, R_2, S_2\}$, where G_1 describes the language L_1 and G_2 describes the language L_2, then we can describe $L_1 \cup L_2$ by combining the grammars to make the grammar $G = (V, \Sigma, R, S)$ where:

- $V = V_1 \cup V_2 \cup \{S\}$ (where we assume that $V_1 \cap V_2 = \emptyset$)
- $R = R_1 \cup R_2 \cup \{S \mapsto S_1 \mid S_2\}$

1.2 More Examples of CFG Construction

We now discuss some examples of CFG construction.

1.2.1 Example 1: Basic Construction

Build a CFG to describe the language $\{\mathtt{0}^n\mathtt{1}^n \mid n \geq 0\} \cup \{\mathtt{1}^n\mathtt{0}^n \mid n \geq 0\}$.

First, construct the grammar for $\{\mathtt{0}^n\mathtt{1}^n \mid n \geq 0\}$. We note that some strings in this grammar are ϵ, 01, 0011, and so on. So, the CFG would be

$$S_1 \mapsto \mathtt{0}S_1\mathtt{1} \mid \epsilon$$

Likewise, construct the grammar for $\{\mathtt{1}^n\mathtt{0}^n \mid n \geq 0\}$, which gives us

$$S_2 \mapsto \mathtt{1}S_2\mathtt{0} \mid \epsilon$$

So, our solution is

$$\begin{aligned} S &\mapsto S_1 \mid S_2 \\ S_1 &\mapsto \mathtt{0}S_1\mathtt{1} \mid \epsilon \\ S_2 &\mapsto \mathtt{1}S_2\mathtt{0} \mid \epsilon \end{aligned}$$

1.2.2 Example 2: Advanced Construction

Build a CFG to describe the langauge $\{0^n1^m2^n \mid n, m \geq 0\}$.

We begin by considering some strings that are generated by this CFG. In particular, we have 02, 1, 012, 0112, 00122, and so on. We know that there are an equal number of 0's and 2's, so our start rule must have at least

$$S \mapsto \mathtt{0}S\mathtt{2}$$

What if we have no more n though? We now need to consider the 1s. So, we can create another rule

$$T \mapsto \mathtt{1}T$$

But, since we can have 0 1's, we also need to include the empty string; therefore, our rule for T is

$$T \mapsto \mathtt{1}T \mid \epsilon$$

And, thus, our final set of rules are

$$S \mapsto \mathtt{0}S\mathtt{2}$$

$$T \mapsto \mathtt{1}T \mid \epsilon$$

To show that this works, note that we will always have the same number of 0's and 2's. After we expend all of those, then we can consider some number of 1's (which we may not necessarily have the same amount of as 0's and 2's). Finally, we note that S can map straight to ϵ, implying that the empty string is something that is generated from this language (and, indeed, this is true if $n = m = 0$).

1.2.3 Example 3: Wild Generations

Consider the CFG $G = (V, \Sigma, R, S)$ defined by

- $V = \{E\}$
- $\Sigma = \{\mathtt{1}, +, \times, (,)\}$
- $R = \{E \mapsto E + E \mid E \times E \mid (E) \mid \mathtt{1}\}$
- $S = E$

Which of the following strings is/are generated by this CFG?

a. E

b. $\texttt{11}$

c. $\texttt{1} + \texttt{1} \times \texttt{1}$

d. ϵ

The answer is **C**. The reason why is because we can perform the following substitutions:

$$\begin{aligned} E &\implies E + E && \text{By the first rule.} \\ &\implies \texttt{1} + E && \text{By the last rule.} \\ &\implies \texttt{1} + E \times E && \text{By the second rule.} \\ &\implies \texttt{1} + \texttt{1} \times E && \text{By the last rule.} \\ &\implies \texttt{1} + \texttt{1} \times \texttt{1} && \text{By the last rule.} \end{aligned}$$

Note that there are potentially other possible ways this string can be generated. Regardless, the answer is not A because a variable is not a valid string. The answer is not B because there is no way to expand E out twice. The answer is not D because you cannot generate the empty string with the given rules.

1.3 Ambiguity

Sometimes, a grammar can generate the same string in several different ways.

Definition 1.1

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

[a] We say that 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.

For example, consider the last example discussed above. There are several ways to derive $\texttt{1} + \texttt{1} \times \texttt{1}$. Some examples are shown below.

- Derivation 1:

$$\begin{aligned} E &\implies E + E && \text{By the first rule.} \\ &\implies \texttt{1} + E && \text{By the last rule.} \\ &\implies \texttt{1} + E \times E && \text{By the second rule.} \\ &\implies \texttt{1} + \texttt{1} \times E && \text{By the last rule.} \\ &\implies \texttt{1} + \texttt{1} \times \texttt{1} && \text{By the last rule.} \end{aligned}$$

- Derivation 2:

$$\begin{aligned} E &\implies E + E && \text{By the first rule.} \\ &\implies E + E \times E && \text{By the second rule.} \\ &\implies 1 + E \times E && \text{By the last rule.} \\ &\implies 1 + 1 \times E && \text{By the last rule.} \\ &\implies 1 + 1 \times 1 && \text{By the last rule.} \end{aligned}$$

Note that some context-free languages can be generated only by ambiguous grammars; in this case, we say that these languages are **inherently ambiguous**.

1.4 Chomsky Normal Form

We can use Chomsky normal form to "simplify" a context-free grammar.

Definition 1.2

A context-free grammar is in **Chomsky normal form** if every rule is of the form

$$A \mapsto BC$$
$$A \mapsto a$$

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

A somewhat important theorem is as follows:

Theorem 1.1

Any context-free language is generated by a context-free grammar in Chomsky normal form.