5

Production Grammars

A *production grammar* (also known as a generative grammar, and which I will call simply grammar from now on) is a rewrite system that describes how to *generate* strings using a set of production rules.

Example 1: Here is a simple grammar given by two rules:

$$\begin{aligned} S &\to \mathtt{a}S\mathtt{b} \\ S &\to \epsilon \end{aligned} \tag{5.1}$$

These rules say that you can rewrite symbol S into $\mathtt{a}S\mathtt{b}$, or into the empty string. Here is a sequence of rewrites showing that these rules, starting with symbol S, can generate the string $\mathtt{aaabbb}$:

$$\begin{aligned} \underline{S} &\Longrightarrow \mathtt{a}\underline{S}\mathtt{b} \\ &\Longrightarrow \mathtt{aa}\underline{S}\mathtt{bb} \\ &\Longrightarrow \mathtt{aaa}\underline{S}\mathtt{bbb} \\ &\Longrightarrow \mathtt{aaabbb} \end{aligned}$$

(At every step, I indicated which symbol gets rewritten by underlining it.) It's not too difficult to see that such a grammar can generate all strings of the form $\mathtt{a}^n\mathtt{b}^n$ for any $n \geq 0$.

Example 2: Here is a slightly more complicated grammar, given by five rules:

$$\begin{aligned}
S &\to TB \\
T &\to \mathtt{a}T\mathtt{b} \\
T &\to \epsilon \\
B &\to \mathtt{b}B \\
B &\to \epsilon
\end{aligned} \tag{5.2}$$

Here is a sequence of rewrites showing that these rules, starting with symbol S, can generate the string aabbb:

$$\begin{aligned}
\underline{S} &\Longrightarrow \underline{T}B \\
&\Longrightarrow \mathtt{a}\underline{T}\mathtt{b}B \\
&\Longrightarrow \mathtt{aa}\underline{T}\mathtt{bb}B \\
&\Longrightarrow \mathtt{aabb}\underline{B} \\
&\Longrightarrow \mathtt{aabbb}\underline{B} \\
&\Longrightarrow \mathtt{aabbb}
\end{aligned}$$

Symbols S, T, B are intermediate (or nonterminal) symbols used during the rewrites, as opposed to a and b which are symbols in the strings that we care about generating. Again, it is not difficult to see that this grammar generates strings of the form $\mathtt{a}^n\mathtt{b}^m$ where $m \geq n \geq 0$.

All of the above grammars have the characteristic that the left-hand side of every rule consists of a single nonterminal symbol, that is, every rule is of the form $A \to \ldots$ for some nonterminal A. We call grammars made up of such rules *context-free grammars*, and they are an important class of grammars.

Example 3: Here is a grammar that is *not* context-free (also called unrestricted):

$$\begin{aligned}
S &\to AB \\
B &\to XbB\mathtt{c} \\
B &\to \epsilon \\
\mathtt{b}X &\to X\mathtt{b} \\
A &\to AA \\
A &\to \epsilon \\
AX &\to \mathtt{a} \\
\mathtt{a}X &\to X\mathtt{a}
\end{aligned} \tag{5.3}$$

Intuitively, these rules let us expand the initial A into a sequence of As, and the initial B into a sequence of `b` along with the same number of Xs on the left of the `b`s and `c`s on the right. Rules allow the Xs to "migrate" to the nearest As and interact with them to produce `a`s.

Here is a sequence of rewrites showing how to generate `aabbcc`:

$$
\begin{aligned}
\underline{S} &\Longrightarrow A\underline{B} \\
&\Longrightarrow AX\mathtt{b}\underline{B}\mathtt{c} \\
&\Longrightarrow AX\mathtt{b}X\mathtt{b}\underline{B}\mathtt{cc} \\
&\Longrightarrow AX\underline{\mathtt{b}X}\mathtt{bcc} \\
&\Longrightarrow \underline{A}XX\mathtt{bbcc} \\
&\Longrightarrow A\underline{AX}X\mathtt{bbcc} \\
&\Longrightarrow A\underline{\mathtt{a}X}\mathtt{bbcc} \\
&\Longrightarrow \underline{AX}\mathtt{abbcc} \\
&\Longrightarrow \mathtt{aabbcc}
\end{aligned}
$$

This grammar generates all strings of the form $\mathtt{a}^n\mathtt{b}^n\mathtt{c}^n$ for $n \geq 0$.

Formal Definitions

Definition: A production grammar is a tuple $G = (N, \Sigma, R, S)$ where

- N is a finite set of nonterminal symbols;
- Σ is a finite set of terminal symbols;
- R is a finite set of rules, each of the form $w_1 \to w_2$ where $w_1, w_2 \in (N \cup \Sigma)^*$ and w_1 has at least one nonterminal symbol;
- $S \in N$ is a nonterminal symbol called the start symbol.

Rewriting using a grammar G is defined using a relation $\Longrightarrow_G$:

$$uw_1v \Longrightarrow_G uw_2v \text{ if } w_1 \to w_2 \in R$$

and generalizing to the multi-step rewrite relation $\Longrightarrow_G^*$ defined by taking $w_1 \Longrightarrow_G^* w_2$ if $w_1 = w_2$ or $\exists u_1, \ldots, u_k$ such that $w_1 \Longrightarrow_G u_1$, $u_1 \Longrightarrow_G u_2, \ldots,$ $u_{k-1} \Longrightarrow_G u_k$, and $u_k \Longrightarrow_G w_2$. We usually drop the G when it's clear from context.

The language $L(G)$ of grammar G is the set of all strings of terminals that can be generated from the start symbol of the grammar:

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

Context-Free Languages

An important class of languages is the *context-free languages*. A language is *context-free* if there exists a context-free grammar that can generate it.

It is easy to see that regular languages are context-free. To show that, it suffices to show that to every deterministic finite automaton there exists a context-free grammar that generates the language accepted by the automaton.

Let $M = (Q, \Sigma, \delta, s, F)$ be a deterministic finite automaton. Construct the grammar $G_M = (N, \Sigma, R, S)$ by taking $N = Q$ and $S = s$, and by having one rule in R of the form

$$p \to aq$$

for every transition in M of the form $\delta(p, a) = q$, and one rule in R of the form

$$p \to \epsilon$$

for every $p \in F$.

Since $\{\mathtt{a}^n\mathtt{b}^n \mid n \geq 0\}$ is context-free by grammar (1) above but not regular, the class of context-free languages is a strictly larger class of languages than the regular languages.

[redacted]

[redacted]