1 Context-Free Grammars

Consider the following context-free grammar G_1:

$$A \mapsto \texttt{0}A\texttt{1}$$
$$A \mapsto B$$
$$B \mapsto \texttt{\#}$$

Here, we note the following:

- A grammar consists of a collection of **substitution rules** (also called productions). Each rule appears as a line in the grammar, comprising a symbol and a string separated by an arrow.
- The symbol (e.g. A, B) is called a **variable**. They are often represented by capital letters.
- The string consists of variables and other symbols (e.g. `0`, `1`, `#`) called **terminals**. They are analogous to the input alphabet and often are represented by lowercase letters, numbers, or other special symbols. In other words, these are symbols that cannot be replaced (substituted).
- One variable, usually (but not always) the top-left one in the list of rules (e.g. A), is designated as the **start variable**.
- When we have multiple rules with the same left-hand variable, we may condense them into a single line where the right-hand sides are separated by a `|` (as an "or"). So, we can rewrite the above rules like so:

$$A \mapsto \texttt{0}A\texttt{1} \mid B$$
$$B \mapsto \texttt{\#}$$

We can use a grammar to describe a language by generating each string of that language like so:

1. Write down the start variable.
2. Find a variable that is written down and a rule that starts with that variable. Then, replace the written dow nvariable with the right-hand side of that rule.
3. Repeat step 2 until no variables remain.

One **derivation** (the sequence of substitutions needed to obtain a string) is as follows:

$$\begin{aligned} A &\implies \texttt{0}A\texttt{1} && \text{By first rule.} \\ &\implies \texttt{00}A\texttt{11} && \text{By first rule.} \\ &\implies \texttt{000}A\texttt{111} && \text{By first rule.} \\ &\implies \texttt{000}B\texttt{111} && \text{By second rule.} \\ &\implies \texttt{000\#111} && \text{By third rule.} \end{aligned}$$

We say that all strings generated in this way constitute the **language of the grammar**, and write $L(G)$for the language of grammar G. So, for our introductory example above, we have that

$$L(G_1) = \{\texttt{0}^n\texttt{\#1}^n \mid n \geq 0\}$$

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

1.1 Formal Definition

We now introduce the formal definition of a context-free grammar.

Definition 1.1

A **context-free grammar** is a 4-tuple (V, Σ, R, S) where

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

Suppose u, v, and w are strings of variables and terminals.

Yielding	Deriving
Suppose $A \mapsto w$ is a rule of the grammar. Then, we say that uAv **yields** uwv, written $uAv \implies uwv$.	We say that u **derives** v, written $u \overset{*}{\implies}$, if $u = v$ or if a sequence $u_1, u_2, \ldots, u_k$ exists for $k \geq 0$ and $$u \implies u_1 \implies u_2 \implies \cdots \implies u_k \implies v$$

So, it follows that the **language of the grammar** is given by

$$\{w \in \Sigma^* \mid S \overset{*}{\implies} w\}$$

1.1.1 Example 1: Identifying Language

Consider the grammar $G_3 = (\{S\}, \{\mathtt{a}, \mathtt{b}\}, R, S)$, with R being the rule

$$S \mapsto \mathtt{a}S\mathtt{b}|SS|\epsilon$$

Describe the language of this context-free grammar.

This grammar generates strings like `abab`, `aaabbb`, and `aababb`. We can describe the language $L(G_3)$ as all strings where, for any `a`, there is a corresponding `b`. Analogously, if we let `a` be `(` and `b` be `)`, then this is the language of all properly nested parentheses.

1.2 Describing Context-Free Languages

Many context-free languages are the union of simpler context-free languages. So, if you need to construct a context-free grammar for a context-free language, break it into simpler pieces, create the corresponding grammars, and then merge them into a grammar for the original language by combining their rules and then adding the new rule

$$S \mapsto S_1 \mid S_2 \mid \cdots \mid S_k$$

where the variables S_i are the start variables for the individual grammars.

1.2.1 Example 1: Constructing Context-Free Grammar

Build a CFG to describe the language $\{\mathtt{abba}\}$.

There are several ways to go about this. Consider

$$G_a = (\{S\}, \{\mathtt{a}, \mathtt{b}\}, \{S \mapsto \mathtt{abba}\}, S)$$

which maps S to `abba`, exactly what we wanted.

Another example is

$$G_b = (\{S, T, V, W\}, \{\mathtt{a}, \mathtt{b}\}, R, S)$$

where R is defined by the rules

$$\begin{aligned} S &\mapsto \mathtt{a}T \\ T &\mapsto \mathtt{b}V \\ V &\mapsto \mathtt{b}W \\ W &\mapsto \mathtt{a} \end{aligned}$$

So, if we applied the substitution rules, we would get

$$\begin{aligned} S &\mapsto \mathtt{aT} && \text{By first rule.} \\ &\mapsto \mathtt{ab}V && \text{By second rule.} \\ &\mapsto \mathtt{abb}W && \text{By third rule.} \\ &\mapsto \mathtt{abba} && \text{By fourth rule.} \end{aligned}$$

1.2.2 Example 2: Constructing Context-Free Grammar

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

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