8: Context-Free Languages

Context-free grammars

- A language generator that is more powerful than regular expressions.
- Example: $\Sigma = \{a, b\}$

  $L = \{$strings that begin with a and end with b and that contain a string of a's or a string of b's in between$\}$

- Regular expression: $a(a^* \cup b^*)b$.
- By a set of rules: L consists of all strings S such that

$S \to aMb$	S begins with a and ends with b and has a middle part M.
$M \to A$	The middle part can be a string A of a's, or
$M \to B$	a string B of b's.
$A \to e$	A string of a's can be the empty string, or
$A \to aA$	an a followed by a string of a's.
$B \to e$	A string of b's can be the empty string, or
$B \to bB$	a b followed by a string of b's.

8: Context-Free Languages

In the second description of the language L,

- $V = \{a, b, M, A, B\}$ — alphabet used.
- $\Sigma = \{a, b\}$ — subset of symbols that appear in the target strings. (**Terminals**, $\Sigma \subseteq V$).
- $R = \{S \to aMb, M \to A, \ldots\}$ — the set of **rules**, subset of $(V - \Sigma) \times V^*$.
- S — the **start symbol**, an element of $V - \Sigma$.

The quadruple $G = (V, \Sigma, R, S)$ is called a **context-free language**.

8: Context-Free Languages

Derivation in a context-free grammar:

- $R = \{\ S \rightarrow aMb,\ M \rightarrow A,\ M \rightarrow B,$
  $A \rightarrow e,\ A \rightarrow aA,$
  $B \rightarrow e,\ B \rightarrow bB\}$
- $S \Rightarrow aMb \Rightarrow aAb \Rightarrow aaAb \Rightarrow aaaAb \Rightarrow aaab.$
- $S \Rightarrow^* aaab.$ Multiple step derivation.
- $S \Rightarrow aMb \Rightarrow aBb \Rightarrow abBb \Rightarrow abbBb \Rightarrow abbb.$
- $S \Rightarrow^* abbb.$
- Called **context-free** because the applications of the rules do not depend on the context.
- The **language generated** by G is

$$L(G) = \{w \in \Sigma^* : S \Rightarrow^* w\}.$$

- A **context-free language** is a language can be generated by some context-free grammar.

8: Context-Free Languages

Example: $G = (V, \Sigma, R, S)$

- $V = \{a, b, S\}$, $\Sigma = \{a, b\}$.
- $R = \{S \to aSb,\ S \to e\}$.
- Possible derivations: (there may be many)
  - $S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aabb$.
  - $S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aaaSbbb \Rightarrow aaabbb$.
  - $\ldots$
- $L(G) = \{a^n b^n : n \geq 0\}$.
- $L(G)$ is not regular. Context-free grammars can describe languages that are not regular.

8: Context-Free Languages (CFLs)

- CFLs are very important in **parsing** a program.
- Example: CFL for legal arithmetic expressions
  E — expression, T — term, F — factor

$$
\begin{aligned}
&V = \{x,\ 1,\ 2,\ +,\ *,\ (,\ ),\ S,\ T,\ F,\ E\} \\
&\Sigma = \{x, 1, 2, +, *, (,)\} \\
&R = \{\ S \to E, && \text{(R0)} \\
&\qquad E \to E + T, && \text{(R1)} \\
&\qquad E \to T, && \text{(R2)} \\
&\qquad T \to T * F, && \text{(R3)} \\
&\qquad T \to F, && \text{(R4)} \\
&\qquad F \to (E), && \text{(R5)} \\
&\qquad F \to x1, && \text{(R6)} \\
&\qquad F \to x2\} && \text{(R7)}
\end{aligned}
$$

An expression is legal iff it can be derived.

$(x1 * x2 + x1)$ is legal

$*x1$ and $x1 + ($, are not legal.

8: Context-Free Languages (CFLs)

CFL for checking balanced left and right parentheses:

$$
\begin{aligned}
V &= \{S, (,)\} \\
\Sigma &= \{(,)\} \\
R &= \{S \to e, \\
&\qquad S \to SS, \\
&\qquad S \to (S)\}.
\end{aligned}
$$

()(()()) is balanced

()(() is not balanced.

8: Context-Free Languages (CFLs)

Parse trees

Pictorial representation of derivations.

The derivation

$$S \Rightarrow SS \Rightarrow (S)S \Rightarrow ((S))S \Rightarrow (())S \Rightarrow (())(S) \Rightarrow (())()$$

can be represented by the following parse tree:

Figure 3-5 here, with the bottom arcs removed.

One tree can represent may derivations, e.g.,

$$S \Rightarrow SS \Rightarrow (S)S \Rightarrow (S)(S) \Rightarrow ((S))(S) \Rightarrow (())(S) \Rightarrow (())()$$

$$S \Rightarrow SS \Rightarrow S(S) \Rightarrow S() \Rightarrow (S)() \Rightarrow ((S))() \Rightarrow (())()$$

$(())()$ is called the **yield** of the parse tree.

8: Context-Free Languages (CFLs)

Formal inductive definition of parse trees

Given $G = (V, \Sigma, R, S)$.

1. For any $A \in V$,

   is a parse tree rooted at A with yield A.

2. For any $A \to e \in R$,

   is a parse tree rooted at A with yield e.

3. For any $A \to A_1 A_2 \ldots A_n \in R$, if

   are parse trees rooted at A_1, A_2, ..., A_n with yields y_1, y_2, ..., y_n respectively, then

   is parse tree rooted at A with yield $y_1 y_2 \ldots y_n$.

4. Nothing else is a parse tree.

8: Context-Free Languages (CFLs)

Regular grammars:

- A context-free grammar is **regular** if the right-hand side of every rule contains at most one non-terminal, which, if present, must be the last symbol in the string.
- Example:

$$\begin{aligned}
&V = \{S, A, B, a, b\}\\
&\Sigma = \{a, b\}\\
&R = \{S \to bA,\\
&\qquad S \to aB,\\
&\qquad A \to abaS,\\
&\qquad B \to babS,\\
&\qquad S \to e\}
\end{aligned}$$

  $L(G) = (abab \cup baba)^*$.

- Revisit earlier examples.
- Formally, $G = (V, \Sigma, R, S)$ is **regular** if

$$R \subseteq (V - \Sigma) \times \Sigma^*((V - \Sigma) \cup \{e\}).$$

8: Context-Free Languages (CFLs)

Theorem 1 *A language is regular iff it is generated by a regular grammar.*

Notes:

- This implies that any language that can be described using a regular expression can also be described using a CFG.
- There are languages that can be described using CFGs but are not regular expressions, e.g. $\{a^n b^n \,:\, n \geq 0\}$.

Proof:
IF

1. Pictorial representation of regular grammar $G = (V, \Sigma, R, S)$. Will look very much like an NFA.
   - One node for each nonterminal plus extra node f.
   - S — initial node, f — final node.
   - For all nonterminals A and B and any $w \in \Sigma^*$

     Draw $A \xrightarrow{w} B$ if $A \to wB \in R$,
     Draw $A \xrightarrow{w} f$ if $A \to w \in R$.

2. Example: Figure 3-1

3. Diagram looks like an NFA but has a different interpretation.

4. Derivations in G are mimicked by moves in M.

   $S \Rightarrow bA \Rightarrow babaS \Rightarrow babaaB \Rightarrow babaababS \Rightarrow babaabab$

$$\begin{aligned}(s, babaabab) &\vdash (A, abaabab) \vdash (S, abab) \\ &\vdash (B, bab) \vdash (S, e) \vdash (f, e)\end{aligned}$$

5. Hence, L generated by G implies that L accepted by M.

<u>ONLY IF</u>

6. By interpreting the arcs as grammar rules, a DFA can also be viewed as representing a regular grammar.

7. Hence L accepted by a DFA implies L generated by a regular grammar. □