Automata, Languages and Computation

Chapter 5 : Context-Free Grammars and Languages

Master Degree in Computer Engineering
University of Padua
Lecturer : Giorgio Satta

Lecture based on material originally developed by :
Gösta Grahne, Concordia University

1. Context-free grammars : we consider devices defining structures more complex than regular languages

2. Parse trees : tree representation of a derivation

3. CFGs and ambiguity : some strings might have more than one parse tree

4. Relation with regular languages : CFGs can simulate FAs or regular expressions

Informal example of CFL

Let $L_{pal} = \{w \mid w \in \Sigma^*,\ w = w^R\}$, also called the language of all **palindrome** strings

Example : (ignore case, spaces, and punctuation characters)
`"Madam I'm Adam"` is a palindrome;
`"A man, a plan, a canal, Panama!"` is a palindrome

Informal example of CFL

Let $\Sigma = \{0, 1\}$ and assume L_{pal} is a regular language

Let n be the constant from the pumping lemma. We pick $w = 0^n 1 0^n \in L_{pal}$, $w \geqslant n$

Let $w = xyz$ be such that $y \neq \epsilon$ and $|xy| \leqslant n$

If $k = 0$, $xz \notin L_{pal}$: the number 0's to the left of 1 is smaller than the number of 0's to its right

Informal example of CFL

We **inductively** define L_{pal}

Base ϵ, 0, and 1 are palindrome strings

Induction
If w is a palindrome strings, then $0w0$ and $1w1$ are also palindrome strings

Nothing else is a palindrome string

CFG example

CFGs are a formalism for **recursively** defining languages such as L_{pal}, using **rewriting rules**

1. $P \rightarrow \epsilon$
2. $P \rightarrow 0$
3. $P \rightarrow 1$
4. $P \rightarrow 0P0$
5. $P \rightarrow 1P1$

P is a **variable** representing strings of a language. In this grammar P is also the initial symbol

Compare variables with recursive functions in programming languages

Definition

A **context-free grammar** (CFG for short) is a tuple

$$G = (V, T, P, S)$$

where

- V is a finite set of **variables** (also called **nonterminals**)
- T is a finite set of **terminal symbols**, representing the language alphabet
- P is a finite set of **productions** having the form $A \rightarrow \alpha$, where A (head, or left-hand side) is a variable and α (body or right-hand side) is a string in $(V \cup T)^*$
- S is a variable called **initial symbol**

Example

A CFG for palindrome strings is

$$G_{pal} = (\{P\}, \{0, 1\}, A, P)$$

with

$$A = \{P \rightarrow \epsilon, P \rightarrow 0, P \rightarrow 1, P \rightarrow 0P0, P \rightarrow 1P1\}$$

Compact notation

Usually, productions with a common head are grouped together

Example : Productions $A \rightarrow \alpha_1$, $A \rightarrow \alpha_2$, ..., $A \rightarrow \alpha_n$ can be written in a more compact notation

$$A \rightarrow \alpha_1 \mid \alpha_2 \mid \cdots \mid \alpha_n$$

Test

Define a CFG for each of the following languages

- $L = \{a^n b^n \mid n \geqslant 1\}$
- $L = \{a^n b^m \mid n \geqslant m \geqslant 1\}$

$S \leftarrow S_1 S_2 \quad S_1 \leftarrow a \mid aS_1$

$S_2 \leftarrow b \mid bS_2$

$S \leftarrow ab \mid aSb \mid aS_1b$

$S_1 \leftarrow a \mid aS_1$

Derivation

In order to generate strings using a CFG, we define a binary relation $\underset{G}{\Rightarrow}$ over $(V \cup T)^*$, called **rewrites**

Let $G = (V, T, P, S)$ be a CFG, $A \in V$, $\{\alpha, \beta\} \subset (V \cup T)^*$. If $A \to \gamma \in P$ then

$$\alpha A \beta \underset{G}{\Rightarrow} \alpha \gamma \beta$$

and we say that $\alpha A \beta$ **derives in one step** $\alpha \gamma \beta$

If G is understood from the context, we use the simplified notation

$$\alpha A \beta \Rightarrow \alpha \gamma \beta$$

Derivation

We define $\overset{*}{\Rightarrow}$ as the reflexive and transitive closure of $\Rightarrow$

Base Let $\alpha \in (V \cup T)^*$. Then $\alpha \overset{*}{\Rightarrow} \alpha$

Induction If $\alpha \overset{*}{\Rightarrow} \beta$ and $\beta \Rightarrow \gamma$, then $\alpha \overset{*}{\Rightarrow} \gamma$

Relation $\overset{*}{\Rightarrow}$ is called **derivation**

We often write derivations by indicating all of the **intermediate steps**

Leftmost derivation

In derivations, we can avoid the choice of variables to be rewritten if we stick to some **canonical** derivation form

The relation $\underset{lm}{\Rightarrow}$ always rewrites the leftmost variable with some production

We also use the reflexive and transitive closure of $\underset{lm}{\Rightarrow}$, written $\overset{*}{\underset{lm}{\Rightarrow}}$, and call it **leftmost derivation**

Example

Leftmost derivation of $a * (a + b00)$:

$$
\begin{aligned}
E &\underset{lm}{\Rightarrow} E * E \underset{lm}{\Rightarrow} I * E \underset{lm}{\Rightarrow} a * E \underset{lm}{\Rightarrow} a * (E) \underset{lm}{\Rightarrow} a * (E + E) \\
&\underset{lm}{\Rightarrow} a * (I + E) \underset{lm}{\Rightarrow} a * (a + E) \underset{lm}{\Rightarrow} a * (a + I) \underset{lm}{\Rightarrow} a * (a + I0) \\
&\underset{lm}{\Rightarrow} a * (a + I00) \underset{lm}{\Rightarrow} a * (a + b00)
\end{aligned}
$$

We conclude that $E \overset{*}{\underset{lm}{\Rightarrow}} a * (a + b00)$

Rightmost derivation

The relation $\underset{rm}{\Rightarrow}$ always rewrites the rightmost variable with the body of a production

We use the reflexive and transitive closure of $\underset{rm}{\Rightarrow}$, written $\overset{*}{\underset{rm}{\Rightarrow}}$, called **rightmost derivation**

Example

Rightmost derivation :

$$
\begin{aligned}
E &\underset{rm}{\Rightarrow} E * E \underset{rm}{\Rightarrow} E * (E) \underset{rm}{\Rightarrow} E * (E + E) \underset{rm}{\Rightarrow} E * (E + I) \\
&\underset{rm}{\Rightarrow} E * (E + I0) \underset{rm}{\Rightarrow} E * (E + I00) \underset{rm}{\Rightarrow} E * (E + b00) \\
&\underset{rm}{\Rightarrow} E * (I + b00) \underset{rm}{\Rightarrow} E * (a + b00) \underset{rm}{\Rightarrow} I * (a + b00) \\
&\underset{rm}{\Rightarrow} a * (a + b00)
\end{aligned}
$$

We conclude that $E \underset{rm}{\overset{*}{\Rightarrow}} a * (a + b00)$

Notation for CFGs

We use the following conventions

- $a, b, c, \ldots$ terminal symbols
- $A, B, C, \ldots$ variables (nonterminal symbols)
- u, v, w, x, y, z terminal strings
- X, Y, Z terminal or nonterminal symbols
- $\alpha, \beta, \gamma, \ldots$ strings over terminal or nonterminal symbols

Language generated by a CFG

Let $G = (V, T, P, S)$ be some CFG. The **generated language** of G is

$$L(G) = \{w \in T^* \mid S \overset{*}{\underset{G}{\Rightarrow}} w\}$$

that is, the set of all strings in T^* that can be derived from the start symbol

$L(G)$ is a **context-free language**, or CFL for short

Example : $L(G_{pal})$ is a CFL

Test

Consider the language L of all strings over "(" and ")" where parentheses are always **well balanced** (assume $\epsilon \notin L$)

- for the following CFG

$$G = (\{S\}, \{(,)\}, P, S)$$

  specify the set P such that $L(G) = L$
- produce a derivation for string

$$w = (\,(\,)\,(\,(\,)\,)\,)$$

$$S \underset{lm}{\Rightarrow} (S) \underset{lm}{\Rightarrow} (SS) \underset{lm}{\Rightarrow} (()S) \underset{lm}{\Rightarrow} (()(S)) \underset{lm}{\Rightarrow} (()(()))$$