Context-Free Languages

COMPSCI 3331

Outline

- Motivation for Context-Free Languages.
- Definition of Context-Free Languages.
- Examples.
- Derivations and Ambiguity.

Non-Regular Languages

- Not every language is regular: $L = \{a^n b^n \; : \; n \geq 0\}$.
- Use grammars to define some languages which are not regular.
- Context-free grammars: define words through **rewriting**.

Grammars

- Grammars use **rewriting rules** to define words and languages.
- These rules work on symbols (**non-terminals**) that can be written with expressions.
- Rewriting rules (or **productions**) act as a recursive definition for showing how words are produced.

$$
\begin{aligned}
S &\rightarrow aSb \\
S &\rightarrow \varepsilon
\end{aligned}
$$

"To rewrite S, we can either replace it with aSb **or** replace it with ε."

Productions in a Grammar

Productions are interpreted as the ways we generate words in a language.

Python language specification (`docs.python.org/3/reference/grammar.html`)

```
if_stmt -> 'if' named_expression ':' block
if_stmt -> 'if' named_expression ':' block elif_stmt
if_stmt -> 'if' named_expression ':' block else_block
elif_stmt -> 'elif' named_expression ':' block
elif_stmt -> 'elif' named_expression ':' block elif_stmt
elif_stmt -> 'elif' named_expression ':' block else_block
else_block -> 'else' ':' block
```

CFGs: Formal Definitions

A CFG G is a 4-tuple $G = (V, \Sigma, P, S)$ where

- V is a finite set of non-terminals;
- Σ is the finite alphabet;
- P is the set of productions of the form $A \to \alpha$ where $A \in V$ and $\alpha \in (V \cup \Sigma)^*$.
- $S \in V$ is the start non-terminal.

Derivations

Given a CFG $G = (V, \Sigma, P, S)$, how do we derive a word?

- Start with the start symbol S.
- Apply rules from P to rewrite non-terminals (from V).
- Keep going until no non-terminals remain, and only have letters from Σ^*.
- Any word in Σ^* we get in this way is generated by the grammar G.

Derviations

- Formally, define $\Rightarrow_G$ as a relation between words in $(V \cup \Sigma)^*$
- $\alpha \Rightarrow_G \beta$ if we can write

$$\begin{aligned} \alpha &= \alpha_1 A \alpha_2 \\ \beta &= \alpha_1 \gamma \alpha_2 \end{aligned}$$

and $A \rightarrow \gamma$ is a production in P.

Derviations

- $\Rightarrow_G$ means that α can be rewritten to β using one production from P.
- $\Rightarrow_G^*$ means that α can be rewritten to β by using some number of productions.
  - The "transitive closure" of $\Rightarrow_G$.
- If G is understood, we leave it out: $\Rightarrow, \Rightarrow^*$.

Language Generated by a CFG

- A word $w \in \Sigma^*$ is **generated** by a CFG $G = (V, \Sigma, P, S)$ if $S \Rightarrow^* w$.
- The language generated by a CFG is the set of all words generated by G:

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

- If L is a language such that $L = L(G)$ for some CFG G, then we say that L is a **context-free language** (CFL).
- If $S \Rightarrow^* \alpha$ for some $\alpha \in (V \cup \Sigma)^*$, then we say that α is a **sentential form.**

Language Generated by a CFG

Example: $G = (\{S\}, \{a, b\}, P, S)$ with P given by:

$$\begin{array}{lcl} S & \to & aSa \mid bSb \\ S & \to & a \mid b \mid \varepsilon \end{array}$$

What can we derive using G?

What are some sentential forms in G?

Proofs involving CFGs

To show that $L = L(G)$ for some language L and some grammar G, we need to:

(a) Show that $L \subseteq L(G)$. This is usually proved by induction on the length of words in L.

(b) Show that $L(G) \subseteq L$. This can be done by using structural induction.

Example: $G = (\{S\}, \{a, b\}, P, S)$ with P given by:

$$\begin{array}{rcl} S & \to & aSa \mid bSb \\ S & \to & a \mid b \mid \varepsilon \end{array}$$

Prove that $L(G) = \{w \in \{a, b\}^* \ : \ w = w^R\}$.

Representing Derivations

We can represent derivations using a **parse tree.**

$S \Rightarrow aSa \Rightarrow abSba \Rightarrow ababa.$

Restricted Derivations

- Say that a derivation step is a **leftmost** derivation step if the leftmost nonterminal in the sentential form is rewritten.
- We denote a leftmost derivation step by $\Rightarrow_{lm}$.
- A **leftmost derivation** is a derivation in which every step is leftmost.

Example: if $A \rightarrow aa, C \rightarrow c$ are rules, and $bACb$ is a sentential form, then $bACb \Rightarrow_{lm} baaCb$, but not $bACb \Rightarrow_{lm} bAcb$.

Ambiguity

- A CFG $G = (V, \Sigma, P, S)$ is **ambiguous** if there exists $w \in L(G)$ such that w has two distinct leftmost derivations in G.
- Easier: If G is ambiguous, w will have two different parse trees.
- Example: set of all arithmetic expressions.

Inherent Ambiguity

- If every CFG, G with $L(G) = L$ is ambiguous, the CFL L is said to be **inherently ambiguous**.
- Note that ambiguity is a property of grammars, inherent ambiguity is a property of languages.
- There are inherently ambiguous languages:

$$L = \{a^n b^n c^m d^m \ : \ n, m \geq 1\} \cup \{a^n b^m c^m d^n \ : \ n, m \geq 1\}.$$

- L is a CFL (exercise). Proving it is inherently ambiguous is difficult.
- The difficult part: we can't assume anything about a grammar generating L.