Context-free grammars (CFG) and languages (CFL), Chomsky normal forms

Chapter 3: Grammars

Prof. Riddhi Atulkumar Mehta
Assistant Professor
Department of Computer Science and Engineering

Parul® University

NAAC A++

Content

INDEX

Introduction

- Context-Free Grammars (CFG) describe context-free languages (CFLs)
- CFGs are widely used in compiler design, parsers, and natural language processing
- CNF simplifies CFGs for algorithmic analysis

What is a Context-Free Grammar?

A CFG is defined as a 4-tuple:

G = (V, **Σ**, R, S)

- V → Set of variables (non-terminals)
- Σ → Set of terminals
- R → Set of production rules (A → α)
- S → Start symbol (S ∈ V)

Example of CFG

Grammar G = ({S}, {a, b}, R, S) with:

- S → aSb | ε

Generates strings:

- E
- Ab
- Aabb
- Aaabbb
- ✅ Language: $L = \{ a^n b^n \mid n \geq 0 \}$

What is a Context-Free Language (CFL)?

- A language is CFL if it can be generated by a CFG
- CFLs are recognized by Pushdown Automata (PDA)
- CFLs can include:
    - Balanced parentheses
    - Palindromes
    - $a^n b^n$, $a^n b^n c^n$ (NOT CFL)

Derivations in CFGs

Leftmost derivation: Expand the leftmost non-terminal

Rightmost derivation: Expand the rightmost non-terminal

Parse tree visualizes the derivation

Example for $S \rightarrow aSb \mid \varepsilon$

Derivation for aaabbb:

$S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aaaSbbb \Rightarrow aaabbb$

Ambiguity in CFGs

- A grammar is ambiguous if a string has more than one parse tree
- Ambiguity is undesirable in programming languages

Example:

Grammar:

$S \rightarrow S + S \mid S * S \mid a$

String: $a + a * a$

Two parse trees possible ⇒ ambiguous

Normal Forms in CFGs

- Normal forms are restricted forms of CFGs used in algorithms like parsing and simplification.
- Two main types:

1. Chomsky Normal Form (CNF)
2. Greibach Normal Form (GNF)

Chomsky Normal Form (CNF)

Every production is of the form:

1. $A \rightarrow BC$ (non-terminals only)
2. $A \rightarrow a$ (terminal only)
3. $S \rightarrow \varepsilon$ (if $\varepsilon \in L$)

Restrictions:

- $B, C \in V$ (non-terminals), $B \neq$ start symbol
- $a \in \Sigma$ (terminal)

Steps to Convert CFG to CNF

1. Remove **ε**-productions (except S → ε)
2. Remove unit productions (A → B)
3. Remove useless symbols
4. Convert to binary productions (A → BC)
5. Terminal replacements (replace terminals in RHS of long productions)

Steps to Convert CFG to CNF

Step 1 – Eliminate start symbol from right hand side (RHS)

If the start symbol S is at the right-hand side of any production,

Create a production as follows –

S1->S

Where, S1 is the new start symbol

Step 2 – In the grammar try to remove the null, unit and useless productions.

Step 3 – Eliminate terminals from RHS of the production if they exist with other non

terminals or terminals.

Example – S->aA can be decomposed as follows –

S->RA

R->a

Finally, it is nothing but S->aA only.

Steps to Convert CFG to CNF

Step 4 – Eliminate the RHS with more than two non-terminals.

Example – S->ABS can be decomposed as given below –

S->RS

R->AB

Key Takeaways

A context free grammar is in CNF, if the production rules satisfy one of the following conditions

- If there is start Symbol generating ε. Example – A-> ε
- If a non-terminal generates two non-terminals. Example – S->AB
- If a non-terminal generates a terminal. Example – S->a

Parul® University | NAAC GRADE A++

https://paruluniversity.ac.in/