UNIT 3

1.Context-Free Grammar (CFG)

Definition:

A **Context-Free Grammar (CFG)** is a formal system used to describe the syntax of a language. It defines a set of rules that specify how sentences in a language can be formed.

Components of CFG:

A CFG is defined as a 4-tuple:
G = (V, Σ, R, S)

Component	Description	Example
V (Variables / Non-terminals)	Symbols that can be replaced by other symbols	S (Sentence), NP (Noun Phrase), VP (Verb Phrase)
Σ (Alphabet / Terminals)	Symbols that appear in actual sentences	"dog", "runs", "barks"
R (Rules / Productions)	Rules that describe how non-terminals can be expanded	S → NP VP NP → Det N VP → V NP Det → "the"
S (Start Symbol)	The symbol from which sentence generation begins	Typically S

How CFG Works:

- Sentences are generated by replacing non-terminals step by step using the production rules.
  Example:

Start: S

S → NP VP   => NP VP

NP → Det N   => Det N VP

Det → "the", N → "dog"   => "the dog" VP

VP → V NP   => "the dog" V NP

V → "chases", NP → Det N → "a cat"   => "the dog chases a cat"

- CFG ensures that generated sentences are **syntactically valid** according to the rules.

Applications of CFG in NLP:

1. **Parsing:** Produces parse trees representing hierarchical relationships of words.
   Example Parse Tree for "the dog chases":

```
    S
   / \
  NP  VP
  |   / \
 Det  N   V
  |   |   |
 the  dog chases
```

2. **Syntax Checking:** Validates if a sentence is grammatically correct.
3. **Machine Translation:** Helps map syntactic structures between languages.
4. **Speech Recognition:** Defines valid sentence structures to improve accuracy.
5. **Information Extraction:** Identifies components like noun phrases, verbs, etc.

Advantages:

- Simple, formal way to describe syntax.
- Can generate parse trees for structured understanding.
- Useful for rule-based NLP applications.

Limitations:

- Cannot capture all natural language features, especially context-dependent meanings.
- Too rigid for complex languages.
- Modern NLP often uses **Probabilistic CFGs (PCFGs)** or neural networks to handle ambiguity.

---

2. Grammar Rules for English

Basic Components:

Phrase Type	Description	Example Rules
Sentence (S)	A complete thought	S → NP VP
Noun Phrase (NP)	Subject or object	NP → Det N NP → Det Adj N NP → NP PP Examples: "the big dog", "a cat on the roof"
Verb Phrase (VP)	Action or predicate	VP → V VP → V NP VP → V NP PP Examples: "runs", "chased the cat", "gave a gift to her"
Prepositional Phrase (PP)	Connects nouns, verbs, or sentences	PP → P NP Examples: "on the table", "in the park"
Modifiers & Determiners	Words modifying nouns	Determiners: "the", "a", "an" Adjectives: "big", "small", "red"

Example Sentence Generation:

- Sentence: "The big dog chased a cat in the park."
  Parse Steps:

S → NP VP

NP → Det Adj N → "The big dog"

VP → V NP PP → "chased a cat in the park"

PP → P NP → "in the park"

Purpose in NLP:

- Basis for **parsing**, **sentence generation**, and **syntax checking**.
- Essential for creating **parse trees** and **treebanks**.

---

3. Treebanks

Definition:

A **Treebank** is a linguistic resource where sentences are annotated with their **syntactic structure**. Each sentence is represented as a **parse tree**.

Purpose:

- Provides training data for NLP tasks like parsing, syntax analysis, and grammar checking.
- Helps study natural language structure systematically.

Structure:

- **Non-terminal nodes:** syntactic categories (e.g., NP, VP)
- **Terminal nodes:** actual words in the sentence

Types of Treebanks:

1. **Constituency Treebanks:** Show how words group into phrases (CFG-based)
   Example: Penn Treebank
2. **Dependency Treebanks:** Show dependencies between words (head-dependent relationships)

Example Constituency Tree for "The cat chased the mouse":

```
      S
    /  \
   NP    VP
  / \  / \
 Det  N  V   NP
 |   |  |   / \
The cat chased Det N
              | |
             the mouse
```

Applications in NLP:

- **Syntactic Parsing:** Train parsers to predict structure.
- **Grammar Checking:** Detect grammatical errors.
- **Machine Translation:** Ensure correct sentence structure.
- **Information Extraction:** Identify subject, object, and verb relations.

---

4. Normal Forms for Grammar

Normal Forms in NLP

Definition:

A **Normal Form** is a standardized way of writing grammar rules. It simplifies the structure of rules so that **parsing algorithms** can process them more efficiently and consistently.

- It ensures rules follow a specific pattern.
- Helps reduce ambiguity and complexity in syntactic analysis.

Common Normal Forms:

1. Chomsky Normal Form (CNF):

- Every production rule is in one of the following forms:
    1. **A → BC** → A non-terminal produces exactly **two non-terminals**
    2. **A → a** → A non-terminal produces a **single terminal**

Example:

Original Rules:

S → NP VP

NP → Det N

VP → V NP

Det → "the"

N → "dog"

V → "chased"

CNF Rules (already mostly in CNF):

S → NP VP

NP → Det N

VP → V NP

Det → "the"

N → "dog"

V → "chased"

- **Usage:** Required for the **CYK parsing algorithm**.

- **Benefit:** Ensures parsing can be done in **polynomial time**.

---

2. Greibach Normal Form (GNF):

- Every production starts with a **terminal symbol**, optionally followed by **non-terminals**.
- **Form:** A → aB (terminal first, then non-terminals)

Example:

A → aB

B → bC

- **Usage:** Useful for **top-down parsing** (predictive parsers).
- **Benefit:** Simplifies **predictive parsing** by knowing the first terminal to expect.

Applications of Normal Forms in NLP:

1. **Parsing Algorithms:**
    - CNF is essential for **CYK parsing**, which systematically finds all possible parses.
2. **Automated Grammar Checking:**
    - Standardized rules reduce ambiguity.
3. **Computational Efficiency:**
    - Simplifies the implementation of syntactic parsers.
    - Reduces the variety of rule types the parser must handle.
4. **Treebank Processing:**
    - Standardizes tree structures for machine learning models.

---

5. Dependency Grammar (DG)

Definition:

Dependency Grammar analyzes syntactic structure based on **relationships between words** rather than phrase groupings.

Key Concepts:

- **Head and Dependent:** Every word (except root) depends on a head word.
- **Dependency Relation:** Grammatical relationship between head and dependent (e.g., nsubj, dobj, det)
- **Root of Sentence:** Main verb often serves as root.

Example:

Sentence: "The cat chased the mouse."

Word	Head	Relation
The	cat	det
cat	chased	nsubj
chased	ROOT	root
the	mouse	det
mouse	chased	dobj

Dependency Tree:

```
   chased
   /  \
 cat  mouse
 /      \
The      the
```

Advantages:

- Shows **direct relationships** between words.
- Flexible for **free word-order languages**.
- Useful for NLP tasks:
    - Information extraction
    - Machine translation
    - Semantic parsing