Syntactic Parsing.

Syntactic parsing is the process of analyzing the grammatical structure of a sentence based on a set of grammar rules. It determines how words are arranged hierarchically to convey meaning.

Why is Syntactic Parsing Important?

*) Helps understand sentence structure in NLP application.

*) Essential for machine translation, question answering, sentiment analysis, and grammar checking.

*) Aids in resolving ambiguities in language processing.

Types of Syntactic Parsing:-

① constituency parsing (phrase - structure parsing)

↳ uses context-free Grammars (CFGs) to construct hierarchical parse trees.

② Dependency parsing:

→ Focuses on word-to-word relationships rather than phrase structure

Constituency parsing :-

→ Consjtituency parsing is a phrase-structure analysis technique in NLP that represents a sentence using a hierarchical tree based on CFG rules.

→ It breaks a sentence into constituents (phrases) such as noun phrases (NP), Verb phrases (VP), prepositional phrases (PP) etc.

CFG :- (write about CFG). also parsing
- top. down
- bottom up.

Real-Time NLP appln:- Chatbot & AI Assistants

ex:- "set an alarm for 7 A.M".

constituency parse tree :-

```
(S
  (VP (V set)
      (NP (Det an) (N alarm))
      (PP (P for)
          (NP (N 7 AM)))))
```

The AI assistant analyzes the structure to understand:

- Main Action (V) : "set"
- what to set (NP) : "an alarm"
- Additional context (PP) : "for 7 AM".

How Parsing Helps ?

*) Identifies the main action (set) and the object (alarm).

*) Extracts the time reference (7AM) from the prepositional phrase (PP)

*) Helps the AI accurately process under commands.

Dependency parsing.

Dependency parsing is a syntactic analysis technique in NLP that identifies relationships between words in a sentence.

Instead of focusing on phrase structure (like constituency parsing), dependency parsing represents a sentence as a directed graph where:

- Nodes represent words
- Edges (arcs) represents relationships ~~b/w~~ between words, called dependencies

Why is dependency parsing important?

- Helps in sentence structure analysis
- Used in machine translation, information extraction, chatbots and search engines.
- Resolve ambiguity in NLP by identifying the correct syntactic relations.

Key components of Dependency Grammar:

1. Head: A word that governs another word.
(eg. a verb governs a noun)

2. Dependent: A word that depends on another word.
(eg. an object depends on a verb)

3. Relations: Describe how words relate to each other. (eg subject_verb, verb-object).

Ex: sentence: "The dog chased the cat".

Head: "chased" (verb)
Dependent: "dog" (subject),
"cat" (object),
"the" (determiner)

Relations :-

"dog" → subject of "chased"

"cat" → object " "

"the" → modifies "dog" & "Cat"

Dependency Tree Representation:

chased

dog

cat

The

The.

Real_Time Ex: AI Chatbots

consider a voice assistant (eg siri, Alexa) where a user asks:

"Set an alarm for 7Am".

Dependency Tree:

set

alarm

for

an

7AM

Dependency Analysis:

"set" (verb) → Head of the sentence.

"alarm" (noun) → object of "set" (obj)

"for" (preposition) → connects "alarm" and "7Am" (Prep)

"7Am" → Time specification.

How parsing Helps?

→ Identifies the action ("set").

→ Determines the object ("alarm")

→ Extracts additional context (for 7Am)

→ Helps AI interpret user commands correctly.

Dependency parsing (DP) vs constituency parsing (CP)

Feature	DP	CP
focus	Direct relationship between words	phrase structure (NP, VP, etc)
Grammar used	Dependency grammar	Context-free grammar (CFG)
Representation	Tree with arrows (head-dependent)	Hierarchical tree (nested phrases)
Best used for	Information extraction, Question Answering	Machine translation, Speech recognition
Ex:- The cat sat on the mat	"sat" is the head	"The cat sat" → NP, VP, PP.

Dynamic parsing in NLP.

Dynamic parsing is an efficient parsing technique that uses dynamic programming to analyze the structure of sentences.

Instead of recomputing results for the same subproblems, dynamic parsing stores intermediate results & reuse them.

This significantly improves parsing efficiency., especially for longer and complex sentence.

Why is dynamic parsing important?

- Avoids redundant computations, making parsing faster.
- Helps in efficient syntactic parsing, particularly for large-scale NLP tasks

→ Used in probabilistic parsing where multiple parse tree exist for a sentence.

Types of Dynamic parsing in NLP:-

Dynamic parsing is commonly used in Context free Grammar (CFG) parsing and can be categorized into:-

① CYK Algorithm (Cocke - Younger - Kasami) Algorithm

→ Works for CNF (Chomsky Normal Form grammars)

→ uses bottom-up dynamic programming to parse sentences

→ Time Complexity : $O(n^3)$ where n is the sentence length.

② Early parsing Algorithm

→ works for any CFG (not just CNF)

→ uses both top-down & bottom-up parsing.

→ Efficient for both left recursive and right recusive grammars.

→ Time Complexity : $O(n^3)$ in worst case, & $O(n^2)$ for unambiguous grammar.

Ex :-

CYK Alg :-

The CYK alg works only for Chomsky Normal Form (CNF), where:

1. Rules must be in the form

$A \to BC$ or

$A \to a$

2. No empty rules (ϵ)

Ex :- CYK

Converted to CNF.

S → NP VP
NP → Det N
VP → V NP
Det → "The"
N → "dog" | "cat"
V → "chased"

Note:-
NP → Det Noun PP
⇓ to CNF
Nominal → Noun PP
NP → Det Nominal

Sentence to parse:
"The dog chased the cat".

CYK Table Construction:

Words	1st step (Terminals)	2nd Step	3rd step final parse.
The	Det	NP	S
dog	N	NP	
chased	V	VP	
the	Det	NP.	
cat	N.	NP.	

The top-right cell (S) indicates a valid parse.

Real-Time Application:- Machine Translation (Google Translate)

→ Sentence : "El perro persiguio' al gato". (spanish) → "The dog chased the cat".

→ CYK parsing ensures that grammar rules are applied correctly in translation.

Earley parsing Alg - Ex:

Unlike CYK, Earley parsing works for any CFG and handles both left and right - recursive rules.

Ex:- Grammar

S → NP VP
NP → Det N | N
VP → V NP
Det → "The"
N → "dog" | "cat"
V → "chased"

Sentence :- "The dog chased the cat".

Earley Table construction.

Step	Scanned word	Predicted Rules	completed rules.
		Det → "The"	NP → Det N
1	The	N → "dog"	NP → Det N
2	dog	V → "chased"	VP → V NP
3.	chased	Det → "The	NP → Det N
4.	the	N → "cat"	NP → Det N
5.	cat		
6.	End	S → NP VP	Sentence parsed.

Real-time Appli:- Speech Recognition (Siri, Alexa)

- Sentence : " Remaind me to call John at 5 PM".

- Earley parsing helps recognizes sentence structure dynamically, even if words are spoken differently.

Ex :- 2   CYK. X

S → NP VP
NP → Det N
VP → V NP

Syntatic (Grammar rules)

V → includes
Det → the
Det → a
N → meal
N → flight

Lexical rules [words]

Sentence :- the flight includes a meal.

Parse using CKY.

0 the   1 flight   2 includes   3 a   4 meal

[0,1] the   [1,2] flight   5 x 5.   [2,3] includes   [3,4] a   [4,5] me

	1	2	3	4	5
0	the Det	NP			S
1		Noun: flight			
2			Verb includes		VP
3				det a	NP
4					noun.

25/3/25 1/4th hr.

A :- 3, 18, 44, 56
B : 76, 99
C : 52, 55
D : 12, 23, 47, 44, 49, 73, 63, 69, 66, 2004, 68, 264, 226.
TOT : 15.

Probabilistic Context-Free Grammar (PCFG)

PCFGs are used when we want to find the most probable parsed structure of the sentence.

PCFGs are grammar rules, similar to what we have seen, along with probabilities associated with each production rule.

For eg:-

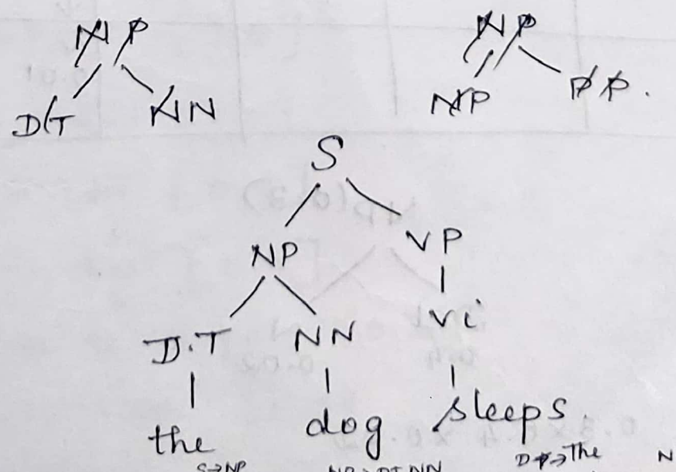
$NP \rightarrow Det\ N\ (0.5) \mid N\ (0.3) \mid N\ PP\ (0.2)$

eg:-

$S \rightarrow NP\ VP$	(1.0)
$VP \rightarrow V_t\ NP$	(0.8)
$VP \rightarrow V_i$	0.6.
$VP \rightarrow VP\ PP$	(0.2)
$NP \rightarrow DT\ NN$	(0.8)
$NP \rightarrow NP\ PP$	(0.2)
$PP \rightarrow IN\ NP$	(1.0)

$V_i \rightarrow$ sleeps	1.0
$V_t \rightarrow$ saw	1.0
$NN \rightarrow$ dog	0.5
$NN \rightarrow$ man	0.1
$NN \rightarrow$ woman	0.1
$NN \rightarrow$ telescope	0.3
$DT \rightarrow$ the	1.0
$IN \rightarrow$ with	0.6
$IN \rightarrow$ in	0.4

The dog sleeps:-



~~the~~ $\underset{S\to NP\,VP}{(1.0)} \times \underset{NP\to DT\,NN}{(0.8)} \times \underset{DT\to the}{(1.0)} \times \underset{NN\to dog}{0.5} \times \underset{VP\to vi}{0.6} \times \underset{vi\to sleeps}{1.0}$

final expression →

Probabilistic CYK

Cocke Younger Kasami

S → NP VP [0.30]
NP → Det N [0.3]
VP → V NP [0.20]

Syntactic Grammar rules

V → includes [0.05]
Det → the [0.4]
Det → a [0.4]
N → meal [0.01]
N → Flight [0.02]

Lexical Rule words.

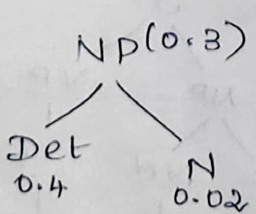
0 the 1 flight 2 includes 3 a 4 meal 5

	1	2	3	4	5
0	Det 0.4	NP 0.0024			S
1		N 0.02			
2			V 0.05		VP 0.0000012
3				Det 0.4	NP 0.0012
4					N 0.01

0.30 × 0.0024 × 0.000001

0.20 × 0.05 × 0.0012

0.3 × 0.4 0.01 = 0.001



0.3 × 0.4 × 0.02

0.0024

Naa

PCFG

Probabilistic Context Free Grammar

~~DOG saw the~~

The cat saw the dog with a telescope

This sentence has two major parses due to prepositional phrase (PP) attachment ambiguity.

A PCFG helps us resolve this by assigning probabilities to grammar rules based on how frequently they appear in training data (like a treebank)

Given :- Grammar rules with prob

S → NP VP [1.0]

NP → Det N [0.7]

NP → NP PP [0.3]

VP → V NP [0.6]

VP → VP PP [0.4]

PP → P NP [1.0]

Det → 'the' [1.0]

N → 'cat' [0.3]

N → 'dog' [0.4]

N → 'telescope' [0.3]

V → 'saw' [1.0]

P → 'with' [1.0]

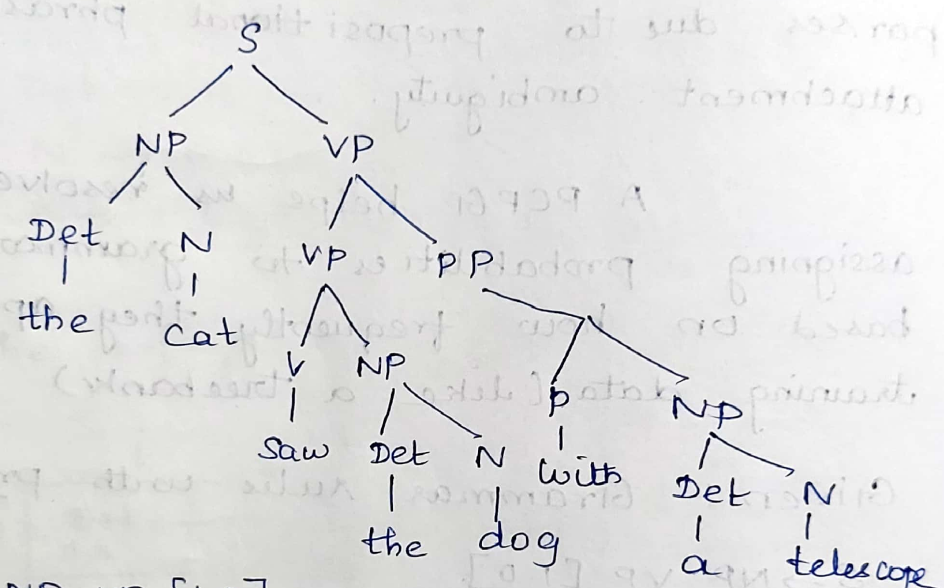
Det → 'a' [1.0]

Two Parse Trees [Ambiguity]

Parse 1: PP attaches to VP (the cat saw [the dog with a telescope]

VP → VP PP

(implies the cat used the telescope to see)



S → NP VP [1.0]
NP → Det N [0.7]
Det → 'the' [1.0]
N → 'cat' [0.3]

VP → VP PP [0.4]
VP → V NP [0.6]
V → 'saw' [1.0]
NP → Det N [0.7]
Det → 'the' [1.0]
N → 'dog' [0.4]
PP → P NP [1.0]
P → 'with' [1.0]

NP → Det N [0.7]
Det → 'a' [1.0]*
N → 'telescope' [0.3]

Multiply all the rule probabilities:

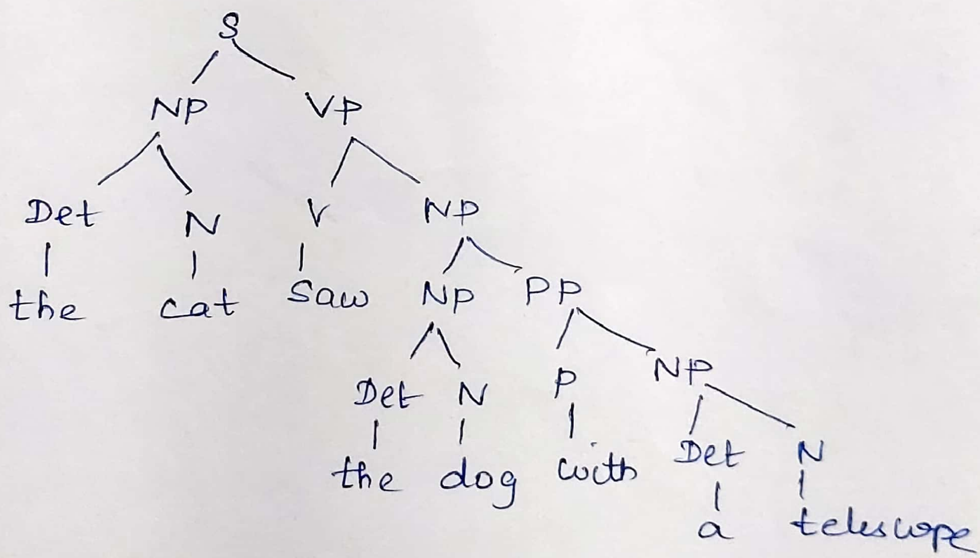
$$P_1 \simeq 1.0 \times 0.7 \times 0.3 \times 0.4 \times 0.6 \times 1.0 \times 0.7 \times 1.0 \times 0.4 \times 1.0 \times 1.0 \times 0.7 \times 1.0 \times 0.3 =\sim 0.0179$$

Parse 2:

PP attaches to NP [the cat saw [the dog who had a telescope])

NP → NP PP

(Implies the dog had the telescope)



prob:-

S → NP VP [1.0]
NP → Det N [0.7]
Det → 'the' [1.0]
N → 'cat' [0.3]
VP → V NP [0.6]
V → 'saw' [1.0]
NP → NP PP [0.3]

NP → NP PP [0.3]
NP → Det N [0.7]
Det → 'the' [1.0]
N → 'dog' [0.4]
PP → P NP [1.0]
P → 'with' [1.0]
NP → Det N [0.7]
Det → 'a' [1.0]
N → 'telescope [0.3]

Multiply all rule prob.

$P_2 \simeq 1.0 \times 0.7 \times 0.3 \times 0.6 \times 1.0 \times 0.3 \times 0.7$

$\times 1.0 \times 0.4 \times 1.0 \times 1.0$

$\times 0.7 \times 1.0 \times 0.3 = \sim 0.0138$

$P_1 > P_2$, so the parser using PCFG with prefer the VP attachment → that the cat used a telescope to see the dog.