22/08/24.

⇒ Syntax Analysis

- Relationship b/w words i.e sentences.

PoS → relationship b/w different classes of words.

noun / verb.

Implementation:

1. Context form Grammar (CFG)
2. FSA / FST
3. Regular Expression (RE)

CFG:

- is a tuple consisting of the following.
  1. A start symbol S.
  2. a set of non-terminals - generalization of terminals.
  3. a set of terminals - words.
  4. a set of function rules of the form $\alpha \rightarrow \beta$

Non-terminal / terminal.

eg. G.

S → NP VP

NP → noun ~~nominal~~ nominal / noun

Nominal → noun

VP → V NP

NP → Adj Nominal

→ Various types of ambiguity in the phases of NLP.

1. Morphology.

- same word → same PoS
  ↓
  different meaning.

eg. Ram deposited his cash in a Central Bank.
                                                                    N
There is a river bank in Paris.
                        N

- same word → different ~~word~~ PoS.

eg. Book that flight
      V
the Book is on the table.
      N

Ram created an issue.

John will issue a cheque.

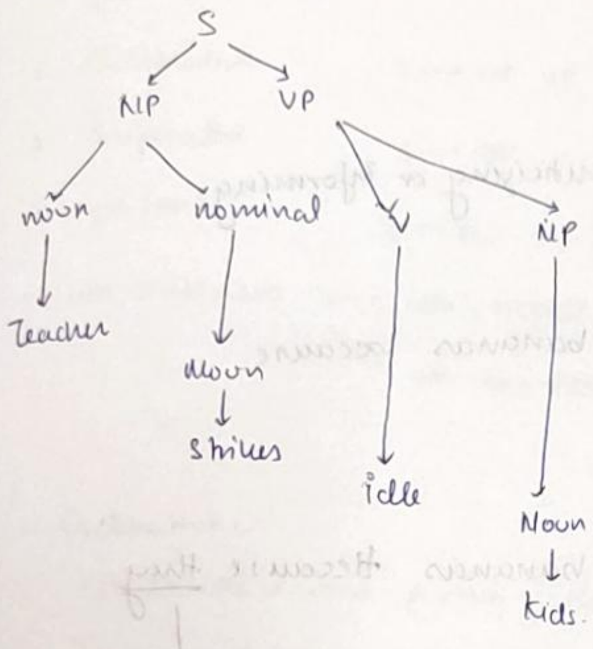
2. Syntax analysis.

- more than one parse tree generated for the same sentence.

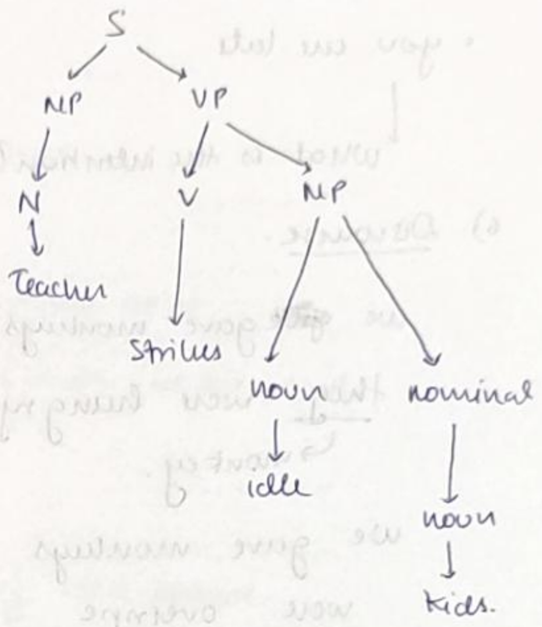
eg. Teacher strikes idle kids
      N   N   V   N

Teacher strikes idle kids.
      N   V   Adj   N.

Parse tree 1 (using G)



Parse tree 2 (using G)



3. <u>Attachment Ambiguity</u>.

- a constituent fits more than one place at the sentence.

eg. John saw <u>Mary with a telescope</u>.

→ John saw Mary having a telescope with her.

<u>John saw Mary</u> with a telescope

→ John saw Mary through his telescope.

4. <u>Semantic Ambiguity</u>.

- a sentence can be interpreted in more than one form.

eg. The car hits a hole while it was moving.

The car while moving hits a hole (car is moving)

The car hits a moving hole. (hole is moving)

5) Pragmatic

- you are late

what is the intention? → criticizing or informing.

6) Discourse.

we ~~gave~~ gave monkeys the bananas because they were hungry.
↳ monkey.

we gave monkeys the bananas because they were overripe
↓
bananas

Q. Draw the pronunciation networks for the following words using Markov model and HMM.

1) believe
2) receive
3) yellow
4) orange.
5) study.
6) slight
7) planning
8) natural

26/08/24.

→ Types of Sentences.

1. Declarative $S \rightarrow NP\ VP$
2. Imperative $S \rightarrow VP$
3. yes/no $S \rightarrow Aux\ Np\ Vp$.
4. wh sentences → wh subject $S \rightarrow wh\ NP\ VP$
   → wh non-subject $S \rightarrow wh\ NP\ Aux\ NP\ VP$.

1. Declarative.
   - starts with a noun phrase followed by verb phrase.

   eg. $\underbrace{\text{I}}_{NP}\ \underbrace{\text{prefer a morning flight.}}_{VP}$

   $\underbrace{\text{The book}}_{\text{noun p}}\ \underbrace{\text{is on the table}}_{VP}$

2. Imperative
   - starts with a verb.
   - actions / commands / suggestions.

   eg. $\underbrace{\text{Book that flight}}_{VP}$

   $\underbrace{\text{Show me the cheapest flight fare.}}_{VP}$

3. Yes/No.
   - Begins with an auxilary ~~aux~~ verb followed by NP & VP.

   eg. $\underbrace{\text{Does}}_{Aux}\ \underbrace{\text{this flight}}_{NP}\ \underbrace{\text{serve dinner}}_{VP}$?

4. wh sentences.

- begins with wh word - what why, which, when, whom, who, how.

4.1. wh subject.

- similar to declarative, preceded by wh.

eg. $\underbrace{\text{which}}_{\text{wh}}$ $\underline{\text{flight}}_{\text{NP}}$ $\underbrace{\text{serve dinner?}}_{\text{VP}}$

4.2. wh non subject.

- similar to yes/no preceded by wh NP.

eg. $\underbrace{\text{which}}_{\text{wh}}$ $\underbrace{\text{book}}_{\text{NP}}$ $\underbrace{\text{do}}_{\text{Aux}}$ $\underbrace{\text{you}}_{\text{NP}}$ $\underbrace{\text{borrow from library}}_{\text{VP}}$.

Q. Identify the types of the following sentence.

1. Does the flight include a meal?
   VBZ DT NN VB DT NN

2. John prefer an Indian flight.
   NN VB DT NN

3. I would like to fly in American Airlines.
   VBZ

4. I need to fly between Philedelphia & Atlanta.

5. I took a book from my friend.

6. I shot an elephant in my pyjamas.

7. which book do you have in your classroom?

8. Can you give me some information about Paris?

9. How can I go from chennai to trichy?

10. I told Harry to go London.

11. I want to go to eat some Italian food today.

12. When does flight 573 arrive in India?

13. Have you ever seen a ghost?

14. Do you like ice cream?
VB

15. Does that taste ok?
VB

16. Could you help me lift this?

17. Is she working very hard?

18. Should I open the book?

19. Had they visited Rome before?

20. Bring me a glass of water.

21. Play with intensity & courage.

22. Take a step & don't move.

23. Give me a pen & a paper.

24. Don't ever touch my phone.

25. Remember me when we are parted.

Q Write the Reg Expression for the following pattern.

a) set of all alphabet strings ending in aab. ^[A-Z a-z]* aab$

b) set of all strings with 2 consecutive repeated words. \b(\w+)\b\s+\1\b

c) all strings which start at the beginning of the line with an integer & which end at the end of the line with a word. ^\d+\b.*\b\w+\b$

d) a pattern which places first word of an English sentence in a register. Deal with punctuation. ^\s*(\w+)([.,!?;:]|\s)

e) A string with mn followed by 2 to 5 no. of Ps.
^mn p{2,5}

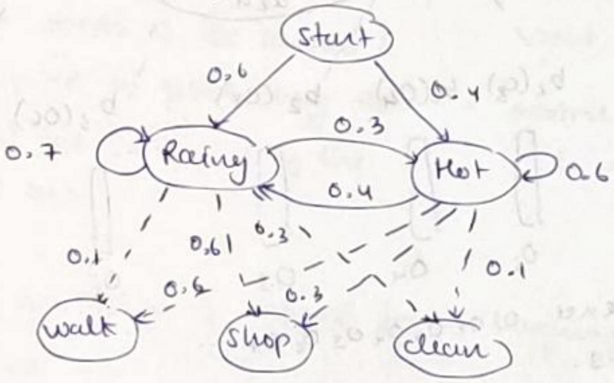
f) A string starts with 'a' followed by 0 or more copies of the sequence 'bcd'. ^a (bcd)*

g) A pattern that matches the string : cat, rat, bat.
\[c r b]at\

Hidden Markov Model (HMM)

1. Hidden state
2. Initial probability distribution - start state
3. Transition probability set - obtain from transition among states given the current state.
4. Emission probability - probability of observing the o/p given the hidden states.
5. observing sequence.

eg. process of predicting weather.



Hidden states - 2

{Rainy, Hot} - hidden because what is observed as the process output is whether the person is shopping, walking or cleaning

- Initial probability distribution
  - start probability
  - = {Rainy : 0.6, Hot : 0.4}

- Transition Probability

  {Rainy : {Rainy-Rainy : 0.7, Rainy-Hot : 0.3}
  Hot : {Hot-Hot : 0.6, Hot-Raing : 0.4} }

- Emission probability

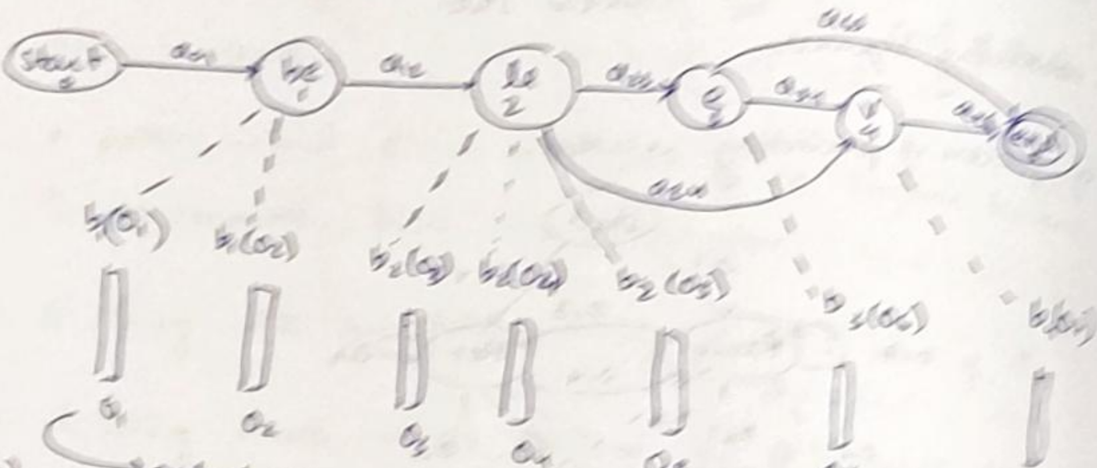
  { Rainy : {walk : 0.1, shop : 0.4, clean : 0.5}
  Hot : {walk : 0.6, shop = 0.3, clean : 0.1}
  }

- prob of observing the o/p - shop, walk & clean given the state - rainy & hot

Observations = {walk, clean, shop}

Pronunciation w/o using HMM:

believe.



$b_1(o_1)$ → output sequence $o_1 o_2 o_3 o_4 o_5 o_6 o_7$ ...

↳ generated from state 1.

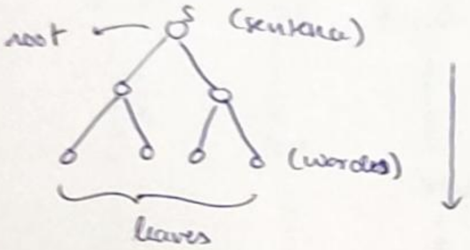
Application of HMM:

Used in applications that aim to recover data sequence where the next data sequence can't be observed by the next data depends on the old sequence.

2/07/24.

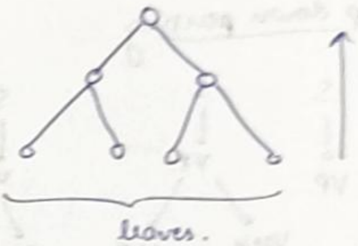
Types of parsing

Top-down



- from the root node S (sentence) down towards the leaves (meaningful words of the sentence)
- no time wasted in generating the subtree that are not part of the S rooted tree.

Bottom up



- Start the leaves (words of the sentence) moving upwards the root node S (S)
- subtree - that are not part of Ex also

Eg. Consider a grammar G defined for a language L.

S → NP VP

S → Aux NP VP

S → VP

NP → Det Nominal

Nominal → noun

Nominal → noun nominal

NP → proper noun.

VP → verb

VP → verb NP | verb NP PP

NP → NP PP

PP → P NP

NP → Adj Noun

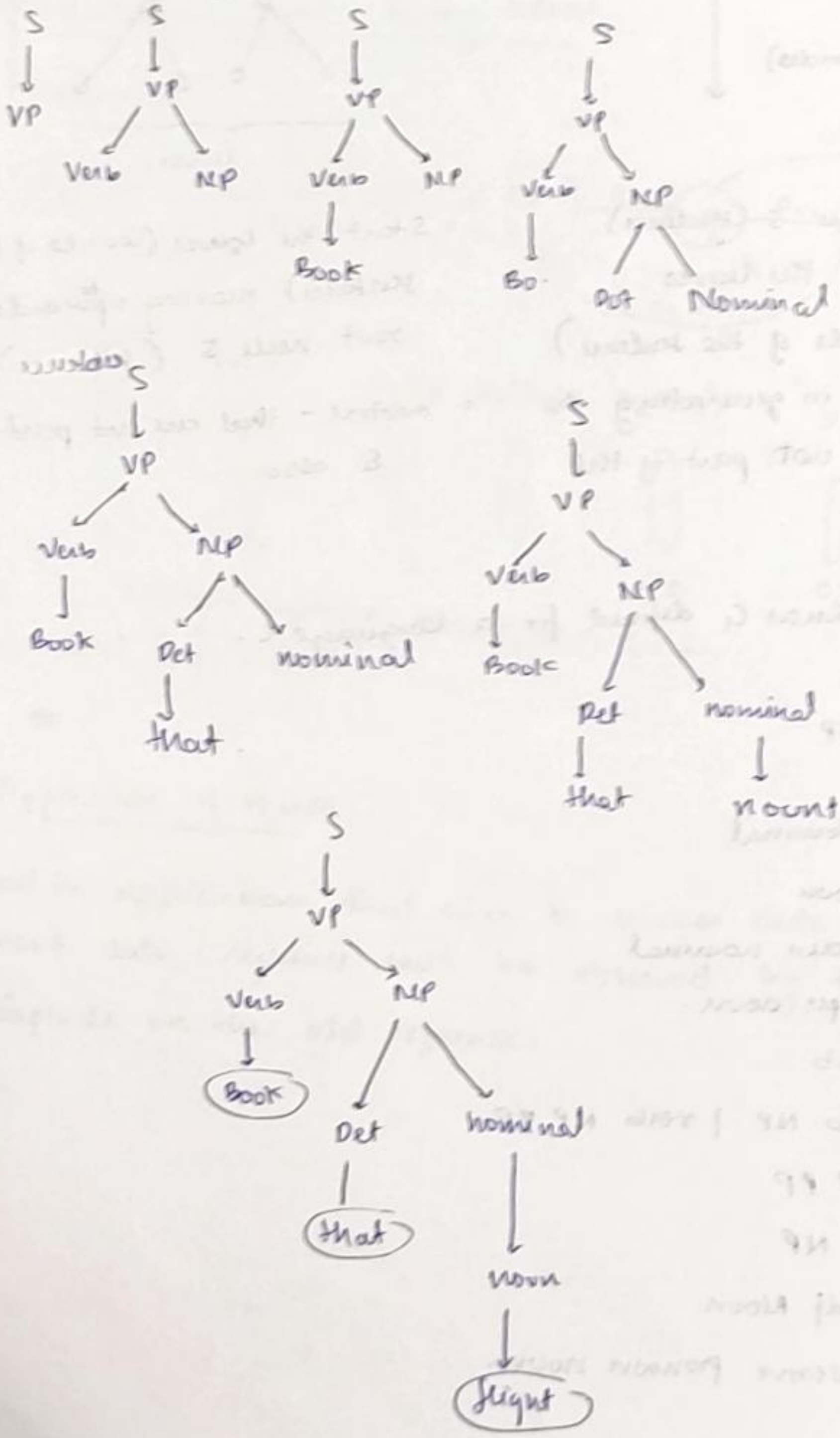
NP → Possessive pronoun noun.

Ex. Consider the sentence

S. Book that flight.

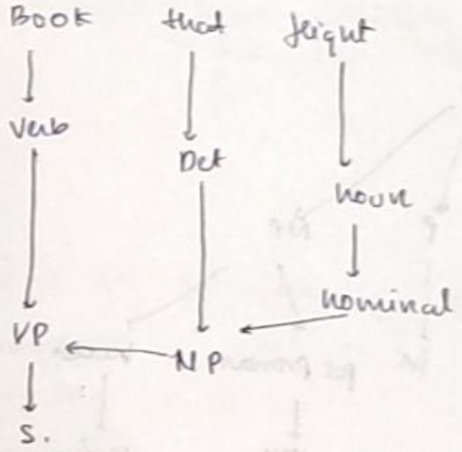
Check whether $S \in L(G)$ or not.

Top down parsing



~~Ex Consider the sentence~~

Bottom up parsing



→ Problem in Top down Parsing :

1. Left Recursion
2. Structural ambiguity.
3. Repeated parsing of subtree.

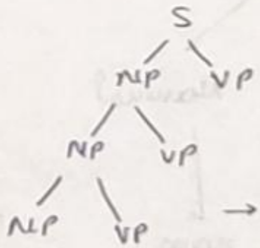
1. Left Recursion.

$A \longrightarrow A\beta \mid \alpha$

$\big\downarrow$ elimination

$A \longrightarrow \alpha A'$

$A' \longrightarrow \beta A' \mid \epsilon$



→ infinite no. of substitutions.

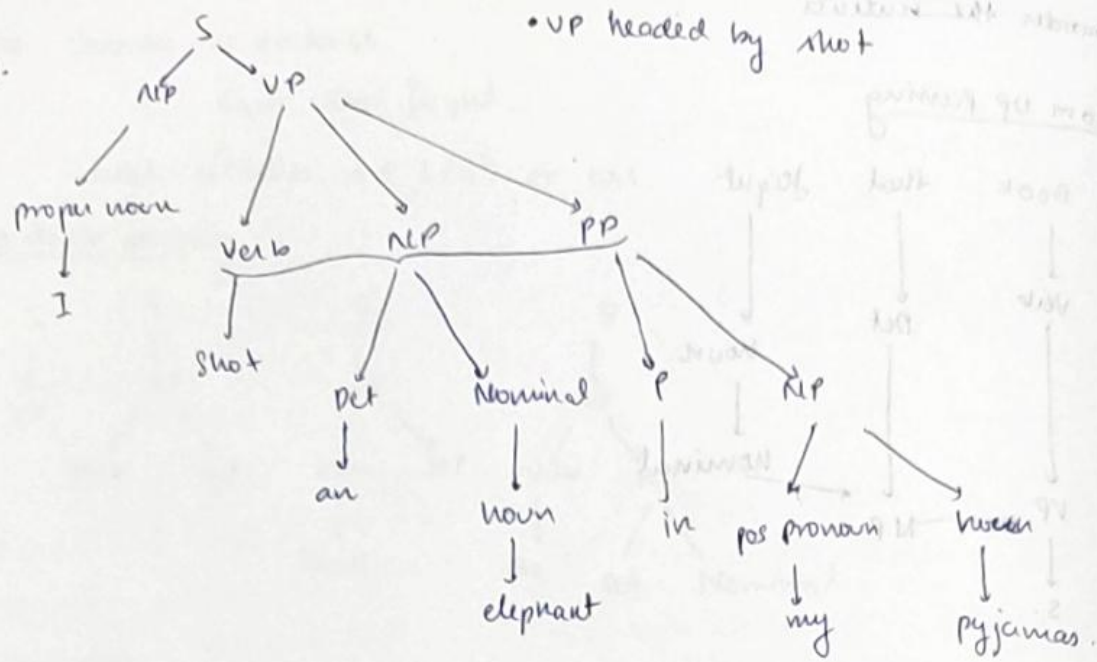
2. Structural Ambiguity.

- occurs when a parser generates more than one parse tree for the same sentence.

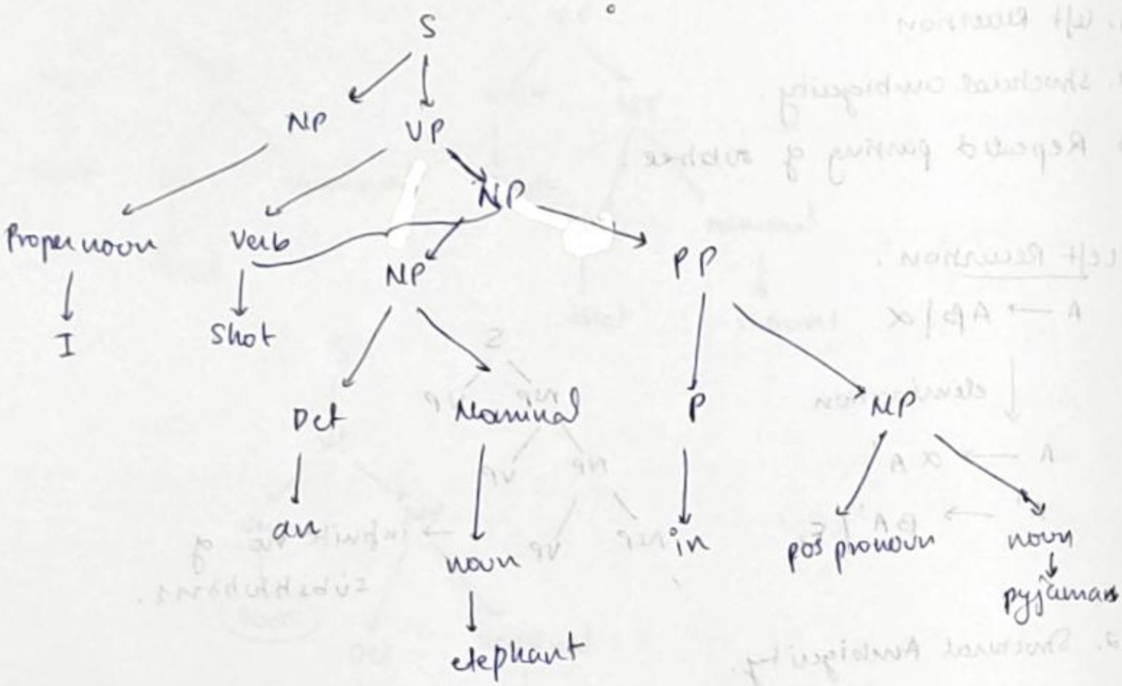
eg. I shot an elephant in my pyjamas

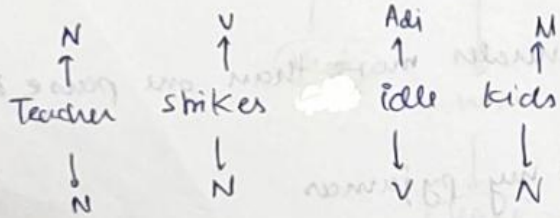
• VP headed by shot

a.



b.





3. Repeated parsing of subtree.

PP PP PP.

a flight (from Delhi) (to Chennai) (on Indian Airlines.)

P N P N P N

Q. Consider the grammar G defined for a language L.

S → NP VP

S → Aux NP VP

S → VP

S → wh NP VP

S → wh Aux NP VP

NP → Det Nominal

Nominal → Noun

Nominal → Noun Nominal

NP → Noun

VP → verb

VP → verb NP

VP → Verb NP PP

NP → NP PP

PP → P NP

NP → Poss Pronoun Noun.

NP → Adj Noun.

Check whether the following sentences ∈ L(G) or not using Top Down & Bottom Up Parsing.

1. Does the flight include a meal?
2. John prefers a morning flight.
3. I would like to fly in American Airlines.
4. I need to fly between Philadelphia & atlanta.

5. I took a book from my table

6. Show me the cheapest flight fare.

7. Which flight serves dinner?

8. Which book do you borrow from the library?

9. Which book do you have in your room?

10. Can you give me some information about paris?

5/9/24

→ Named Entity Recognition.

- classification of proper noun in the text & organised into predefined list of categories.
- Keypart of IE & IR.

Types of categories:

1. Person
2. Organisation
3. Location.

Problems:

1. Person VS Artifact
   eg. Hamsandwich pay the pill
   Bring me a Hamsandwich.
2. Operation VS Location.
   eg. India won the worldcup.
   The worldcup match took place in India.
3. Company VS Artifact.
   eg. Shares in MTV
   Watching MTV
4. Ambiguity in NE types
   Person VS company - Johnsmith
   Person VS Location — Washington.
   Person VS Month - August
   Date VS Time — 1945

Solutions:

- Task definition - must be clearly specified at the outset.
- Simplistic approach for disrespecting metonymy.

  Eg: England → always recognised as a location.
- for each category, there must be a list of examples, counter examples, guidelines, logic behind the intention.

Approach:

1. List lookup Approach
2. Shallow Parsing.

1. List lookup Approach.

- system that recognised only entities stored in the lists (gazette)

Adv: Simple, fast, efficient, language independent.

Disadv: * maintainance of larger lists.
* doesn't resolve ambiguity

2. Shallow Parsing.

→ Internal Evidence

- names often have internal structures.

eg. capword + {forest, city ...}

Bombay City.

Mangrove forest.

capword + {street, avenue, road}

Gandhi Road.

7st street

Market Avenue

→ External Evidence

- names often used in local predictive contexts.
  - locations.

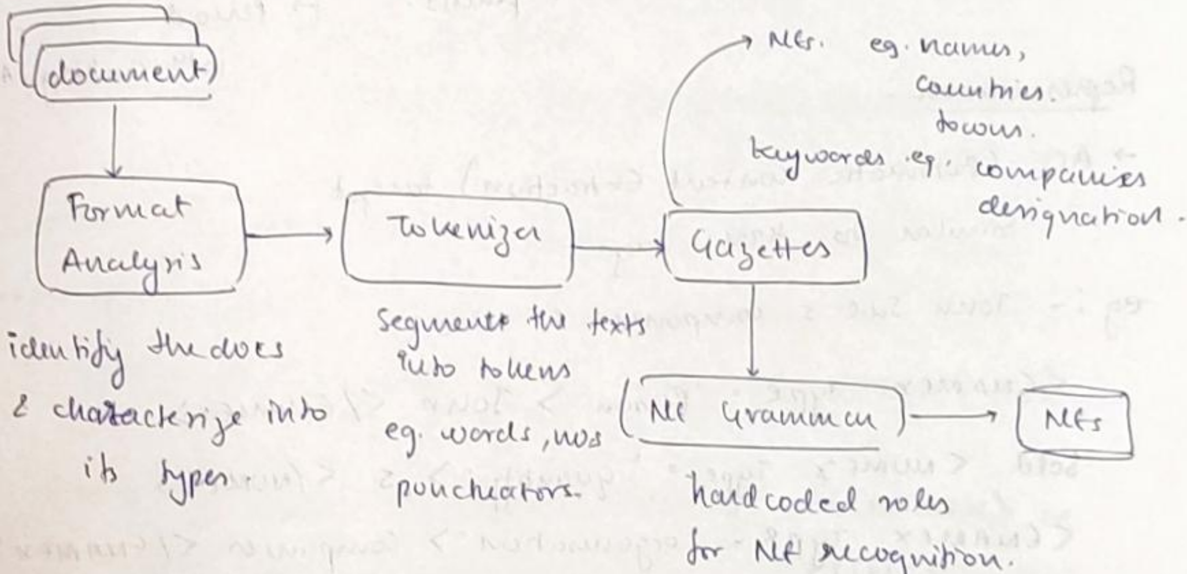
eg. - into the compass of capword. (compass → direction; capword → country)

into the south of India.

- capword - adjective.

eg. Chennai is a friendly city.

⇒ **NE System Architecture.??**



→ **NE Hierarchy**

1) Entity Name (ENAMEX)
2) Numerical Expression (NUMEX)
3) Temporal Expressions (TIMEX)

ENAMEX

- Person
- Organism
- locomotives
- Disease
- location
- Artifacts
- Organization
- Entertainment
- Facilities.

NUMEX

- Distance (km, mi, cm, m)
- Quantity (litres, grams, kgrams, tonnes)
- Money (RS, $, Euro)
- Count eg. 6 apples, 4 pencils.

TIMEX

- Time (am, pm, morning)
- Month (Jan-Dec)
- Date 7th Jan
- Year 1995, 2024
- Day Sun-Sat
- Special day (Independence day, Republic day)
- Period 10th Century, AD, BC

Representation:

→ ACE (Automatic Content Extraction) target.
Similar to HTML tags.

eg:- John Sold 5 companies in 2002

<ENAMEX Type = "Person"> John </ENAMEX>
Sold <NUMEX Type = "quantity"> 5 </NUMEX>
<ENAMEX Type = "organisation"> Companies </ENAMEX>
in <TIMEX type = "year"> 2002 </TIMEX>

9/9/24.

Q Convert the following sentences into corresponding named entity tag sets.

1. Ram took 3 books on NLP from the library in 2020.
2. This bag contains 8 kg of Rice.
3. Robert sold 7 cars in 2016.
4. John will buy 2 litres milk today from the market.
5. The mumbai Express train will start at 5 am from chennai.
6. This container can hold 8 litres of oil.
7. Ram got his Ph.D degree from Standford University at USA.
8. Jane wake up 5'o clock in the morning.
9. The table is made of wood.
10. The airport is 7 kms from the city bus stand.

1. <ENAMEX Type = "Person"> Ram </ENAMEX> took <NUMEX Type = "Count"> 3 </NUMEX> <ENAMEX Type = "Artifacts"> books </ENAMEX> on

<ENAMEX type = "location"> library </ENAMEX> in <TIMEX type = "Year"> 2020 </TIMEX>

2. This <ENAMEX type = "Artifacts"> bag </ENAMEX> contains <NUMEX type = "count"> 8 </Numex> <NUMEX type = "quantity"> kg </NUMEX> of <ENAMEX type = "artifacts"> Rice </ENAMEX>

3. <ENAMEX type = "Person"> Robert </ENAMEX> sold <NUMEX type = "Count"> 8 </ENAMEX> <ENAMEX type = "locomotives"> cars </ENAMEX> in <TIMEX type = "year"> 2016 </TIMEX>

4. <ENAMEX type = "person"> JOHN </ENAMEX> will buy
<NUMEX type = "count"> 2 </NUMEX> <NUMEX type = "quantity">
litres </NUMEX>
<ENAMEX type = "artifacts"> milk </ENAMEX>
today from the
<ENAMEX type = "location"> market </ENAMEX>.

10/4/24.

→ Shallow / Chunk / Partial Parsing.

- divide the sentence into a sequence of chunks.
- find the major but un embedded constituents like NPs, VPs, PPs.

<u>Characteristics</u> of <u>chunks</u>:

- non-overlapping region of text.
- non-recursive → a chunk can't contain another ~~cons~~ chunks.
- non-exhaustive → not all words are included.
- doesn't cross the boundaries of the larger constituents.

Types:

1. <u>VP checking</u>:

E. I [saw] the man who [was older] in the garden
VP VP

2. NP chunking.

[I saw] [a tall man] in [the Park]
NP NP NP

3. Question Answering

- wh-questions

which spanish emperor discovered the Mississippi River?

4. Prosodic Chunking

- deals with tone & rhythms of speech.

⇒ Applications

- Information Retrieval
- Extraction of specialized terms or multiwords.
- Bootstraping a complete parser.
- Constructing a tree bank (annotated text) which other applicants can use.

⇒ Unchunking

3 rules : 1) Apply PoS tagging to the sentences.

2) Apply chunk rule (NP, VP, PP)

3) Unchunking → identify tree pattern.

e.g. S: The little cat sat on the mat

1. Apply postagging

The	little	cat	sat	on	the	mat
det	adj	N	V	P	Det	N

2. Apply NP chunking

[The little cat] sat on [the mat]
NP NP

3. Unchunking

Pattern → [Det Adj N]

The little $\underline{\text{Cat}}$ ↳ noun chunk.

NP chunking

→ BIO tagging Parser

- uses trained classifier to classify each element of input in sequence

- follows I O B tagging
  - B – beginning of sequence
  - I → interior of any sequence
  - O → outside of

Consider the grammar & depict for the language L

S → NP VP

S → Aux NP VP

S → VP

S → wh-NP VP

S → wh-NP Aux NP VP

NP → Det Nominal

Nominal → Noun

Nominal → Noun Nominal

NP → Noun

VP → Verb

VP → Verb NP

VP → Verb NP PP

VP → VP PP

PP → P NP

NP → [illegible]

NP → [illegible]

12/9/24.

⇒ ML Learning Parser.

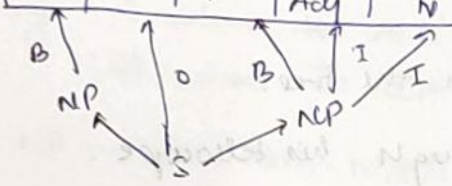
- uses trained classifier to classify each element of input in sequence.
- follows I-O-B tagging.

  B → beginning of sequence.

  O → Outside of any sequence.

  I → Internal to

S:	He	saw	the	big	dog
PoS:		V	Det	Adj	N



Consider the grammar G defined for a language L.

S → NP VP

S → Aux NP VP

S → VP

S → wh NP VP

S → wh NP Aux NP VP.

NP → Det Nominal

Nominal → Noun.

Nominal → Noun Nominal

NP → Noun

VP → verb

VP → verb NP

VP → Verb NP PP

NP → NP PP

PP → P NP

NP → Poss Pronoun Noun

NP → Adj Noun.

check whether the following sentences $\in L(G)$ or not by using

a) Chunk Parser
b) FST Parser → NP chunking / → VP chunking.
c) M/C learning Parser

1) Does this flight include a meal?
2) I took my book from ~~th~~ my table.
3) I shot an elephant in my pyjama.
4) Which book do you borrow from the library?
5) Paint that picture.
6) The boy pick the fruit from the tree.
7) John saw the moon through his telescope.
8) Show me the cheapest flight fare.

19/9

⇒ Earley Parser.

- named after its inventor.
- Jay Earley - to parse the string that belong to a particular CF language or not
- is a chart parser that uses DP techniques.
- Earley parsers are appealing because they work on all CF languages unlike LL & LR parsers - designed for use in compilers - restricted to particular class of languages.

Time Complexity

1. General case - cubic time - $O(n^3)$ $n \rightarrow$ the length of the parsed string.
2. For unambiguous grammars - quadratic time - $O(n^2)$

3) For deterministic context free grammars - linear time - $O(n)$.

→ n+1 ⟶ charts.
  ↳ no of words in the sentences.

→ Each chart consists of various states & states have dotted rules.

Dotted rule

- A dot (.) specified with in the RHS of a state's grammar rule indicating the progress made towards recognizing it.
- a state's position w.r.to its input is represented by 2 nos indicating 1) where the state begins and
  2) where its dot lies in the production rule.

eg. NP ⟶ Det. Nominal [1, 2]
  ↳ where . lies in the prod rule.
  ↳ where the state begins.

NP begins at position 1.
Det has been successfully parsed.
Nominal ⟶ expected next.

Three operators:

1. Predictor ⟶ non-terminal
2. Scanner
  ↳ right to the dots.
3. Completer

→ Predictor

- applied to any state consisting of a non-terminal immediately to the right of its dot (non-terminal ∉ PoS category).

  eg. S ⟶ .VP

→ Scanner

- applied to any state consisting of a PoS category to the right of its dot

  eg. VP ⟶ • verb
  VP ⟶ • verb NP
  VP ⟶ • verb PP

→ Completor

• applied to any state where its dot reached the end of its rule.

e.g. NP → Det Nominal.

Nominal → Noun.

Noun → book.

Q. Consider the grammar L(G)

S → NP VP

S → VP

NP → Det Nominal

Nominal → Noun.

VP → verb

VP → verb NP.

S: Book that flight.

Chart 0.

State	Rule	Start, end	Added by
S0	γ → . S	0, 0	Start state
S1	S → . NP VP	"	predictor
S2	S → . VP	"	"
S3	NP → . Det Nominal	"	"
S4	VP → . verb	"	"
S5	VP → . verb NP.	"	"

chart 1

state	rule	start → end	added by
S6	verb → book	0,1	scanner
S7	VP → verb.	"	completer
S8	VP → verb. NP	"	"
S9	S → VP	"	"
S10	NP → . Det nominal	1,1	predictor.

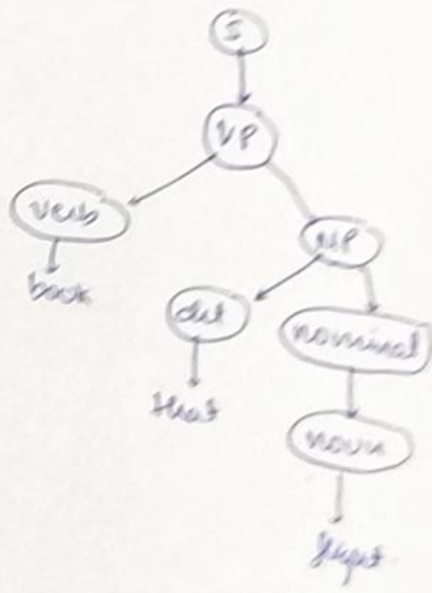
chart 2

state	rule	start, end	added by
S11	Det → that.	1,2	scanner
S12	NP → Det. nominal	1,2	completer
S13	Nominal → . noun.	2,2	predictor.

chart 3

state	rule	start, end	added by
S14	noun → flight.	2,3	scanner
S15	nominal → noun.	2,3	completer
S16	NP → det nominal.	1,3	"
S17	VP → verb NP.	0,3	"
S18	S → VP.	0,3	"

final parse tree:



⇒ Semantic Analysis.

- study of meaning of linguistic utterance.
- studies of algorithms that transform the linguistic utterances into meaningful representation.

⇒ Knowledge Representation Languages.

1. Logical Representation.
   - uses mathematical logic to represent the knowledge.
   eg. First Order Predicate Calculus.

2. N/W representation.
   - uses graphs to represent the knowledge.
   eg. Conceptual graphs.
   Semantic networks.

3. Structural representation
   eg. Scripts
   Frames.

⇒ Conceptual Graphs.

- finite, connected, bipartite graph.
- 2 types of nodes.
  1) concept nodes
     - represented by ▭.
     [a] concrete concept
        - ability to imagine images of them.
        eg. Chair, telephone, car.

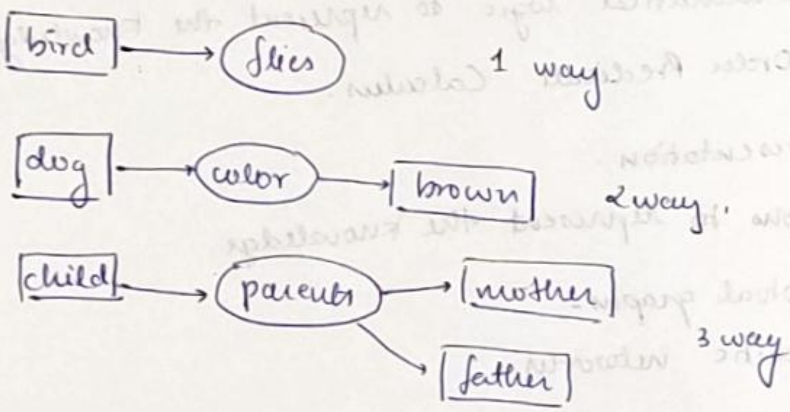
1b Abstract concepts.

- rep the characteristics.

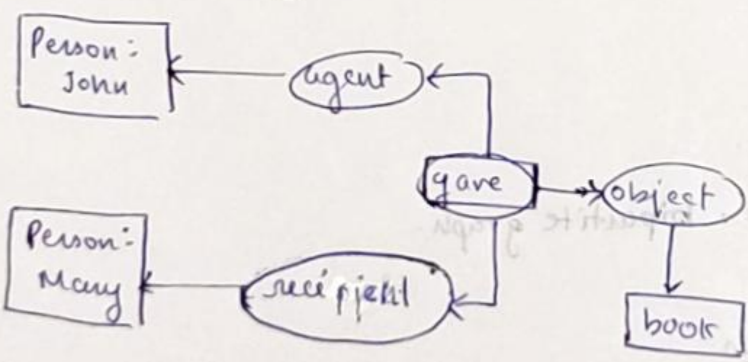
  eg. punctuality, logitivity.

2. Conceptual relation nodes.
   - rep by
   - rep relations btw one or more concepts.
   - no labelled arc.

eg.



John gave mary the book.

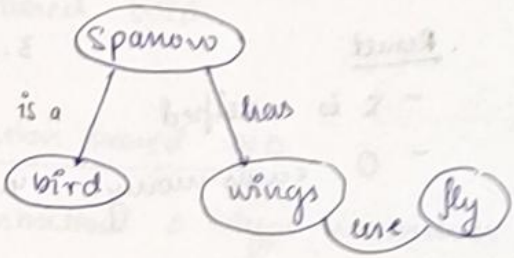


⇒ Semantic Networks

- labelled, directed graph.

  nodes → rep the concepts / objects / situations.
  rep by ⬭

  Links → rep the relations btw the nodes.
  ↳ Labels → types of relationship.



⇒ Scripts

sterotyped sequence of events in a particular context.

Components

1. Entry Condition - must be true for a script to be called on
2. Result - final situations after the exec of scripts.
3. Props - things involved in the script.
4. Roles — actions played by the participants.
5. Scenes —

= death or wile =

Scripts for a Restructured System.

RESTAURANT

Props:
- chairs
- tables
- plates
- spoons
- cups
- memo cards.

Entry condition
- Restaurant is Open
- S is hungry
- S has cash

Scenes:
1. ENTERING.
2. ORDERING.
3. EATING
4. EXITING

Roles:
- Customer : S
- Owner : O
- cook : C
- waiter : W
- manager : M

Result:
- S is satisfied
- O earns money.

Q. write the scripts for
1. Library system
2. Online Air Ticket Booking System.

Q. Construct the conceptual Graphs & semantic networks for the following ~~sentence~~ statements.
1. Parrot is a Green bird having wings to fly
2. Alligator is a species of crocodile that lives on both water and land.
3. Ram likes all kinds of vegetarian food.

10/4/24.

⇒ Word Sense DisAmbiguation (WSD)

- to examine the word token in context & specify which sense of each word is being used.
- to choose the correct POS for a given word.

2 approaches:

1. Selection Restriction based WSD
2. Robust based WSD.

1 Selection Restriction based WSD

- Selection Restriction & type hierarchies are the primary knowledge sources to perform WSD.
- used to rule out inappropriate senses & thereby reducing the ambiguity during semantic analysis.

eg.. In our home, everybody has a career, but none of them includes washing dishes.

↳ selection restriction word. → vessels/ utensils used for eating.

2. In her smart kitchen, Mary prepares several dishes.

↳ selection restriction word → food items for eating.

eg. based on the predicate serve.

1. John serve Green Pea Masala.
   ↳ serve ⇒ type of food item.
2. Which airline serve Paris?
   ↳ serve ⇒ a geographical entity.
3. which airlines serve breakfast?
   ↳ serve ⇒ meal designates

2 Robust based WSD

- is a stand alone task independent of & prior to component semantic analysis.
- 2 types:
  a. Machine learning approaches
  b. Dictionary based approaches.

1 Machine learning approaches.

- systems are to perform WSD
- What is learned in a classifier used to assign one of the fixed no. of senses.

Input:

feature vector - target word to be disambiguited along with its context.

processing of initial inputs

1. POS tagging
2. context may be replaced with smaller / larger segments.
3. Stemming.
4. Partial / depending parsing

- i/p is sliced in to fixed set of features.

1. Select the relevant linguistic features
2. Encode them in a form to be used in the learning algorithm.

⇒ 2 types of linguistic features.

1. Collocational features:

- quantifiable position specific solution between 2 ~~fixed~~ lexical items to the left & right of the target word.

eg. An electric guitar and bass player stand off.

bass → target word.

feature vector: $\begin{bmatrix} \text{guitar - N} \\ \text{and - CNJ} \\ \text{player - N} \\ \text{stand - VB} \end{bmatrix}$

2. Co-occurance features

- co-occurance data about the neighbouring words, ignoring the exact position

eg. Consider the features for bass w.r.t WSJ corpus
[fishing, big, sandal, player, fly, rod, sound, double, rows, playing, guitar, band]

vector: [0 0 0 1 0 0 0 0 0 0 1 0]

⇒ Types of learning approaches:

1. Supervised learning
2. Boot strapping
3. unsupervised learning.

⇒ Supervised Learning.

- instances - labelled
- output - is a classifier capable of ~~using~~ assigning labels to new features encoded inputs.

Methods:

1. Naive Bayes Classifier.
2. Decision trees.
3. Decision list
4. Neural Network
5. Logic learning system.
6. K nearest neighbour.

⇒ Boot Strapping

- major problem in supervised : large sense tagged training
- Bootstrapping eliminates this need by retagging on relatively small no. of instances of each senses for each lexeme.
- These labelled instances - needs to train an initial classifier using any one of the supervised methods.

⇒ Unsupervised:

- instances - unlabelled.
- grouped - form clusters based on similarity metric.

Agglomerative

Agglomerative Clustering

- used in language application.
- each of N training instances initialy assigned to ~~the~~ its own cluster.
- new clusters are then formed in Bottom up fashion by successively merging 2 clusters that are similar.
- process continuous until a specified no. of clusters is reduced.

Q. Apply the Selection Restriction based WSD for the following pairs of sentences.

1. John always wanted to become a holywood star.
   In the Milky way galaxy, there are millions of stars.
2. The bank will not accept cash on sundays
   The river overflowed the bank.
3. Ram finished his task within 5 minutes
   The chairman prepared the minutes of the board meeting.
4. Sachin buys a branded bat for his upcomming cricket match.
   Jack has a bat hagging on the banyan tree.

1/10/24.

Q. Consider the CNF Grammar G defined for a language L.

$S \rightarrow AB \mid BC$ (R1)

$A \rightarrow BA \mid a$ (R2)

$B \rightarrow CC \mid b$ (R3)

$C \rightarrow AB \mid a$ (R4)

Check whether the string baaba $\in L(G)$ or not using CYK parsing.

Row 1:

5					
4	{ϕ}	{S, A}			
3	{ϕ}	{B}	{B}		
2	{A, S}	{B}	{S, C}	{A, S}	
1	{B}	{A, C}	{A, C}	{B}	{A, C}
	1	2	3	4	5
	b	a	a	b	a

Row 2:

$$\boxed{\begin{aligned} V_{ij} &= V_{ik}\, V_{(i+k)(j-k)} \\ k &\rightarrow 1 \text{ to } j-1 \end{aligned}}$$

Find V_{12}:

$i = 1$ $\quad$ $k = 1$ to $2-1$

$j = 2$ $\quad$ $= \underline{1}$

$V_{12} = V_{ik} \cdot V_{(i+k)(j-k)}$

$= V_{11}\, V_{21}$

$= \{B\}\ \{A, C\}$

$= \{BA, BC\} = \{A, S\}$

{ by R_2, by R_1 }

Find V22.

$i=2,\ j=2.$

$k = 1$ to 1.

$V_{22} = V_{21}\, V_{31}$

$= \{A, C\}\{A, C\}$

$= \{AA, AC, CA, CC\}$

$\quad \phi \quad \phi \quad \phi$

$V_{22} = \{CC\}$

$\boxed{V_{22} = \{B\}}$

Find V_{32}.

$i = 3 \quad j = 2.$

$k = 1$ to 1.

$V_{32} = V_{31}\, V_{41}$

$= \{A, C\}\{B\}.$

$= \{AB, CB\}$

S (by R_1) C (by R_4)

Find V_{42}

$i=4 \quad j=2$

$k=1$

$V_{42} = V_{41}\, V_{51}$

$= \{B\}\{A, C\}$

$= \{BA, BC\}$

$= \{A, S\}.$

[by R_2, by R_1]

Row 3:

$i=1 \quad j=3.$

$k = 1$ to 2.

$V_{13} = V_{ik}\, V_{(i+k)(j-k)} \quad [k=1]$

$\cup$

$V_{ik}\, V_{(i+k)(j-k)} \quad [k=2]$

$= V_{11} V_{22} \cup V_{12} V_{31}$

$= \{B\}\{B\} \cup \{A, S\}\{A, C\}.$

$= \{BB\} \cup \{AA, AC, SA, SC\} = \phi \cup \{\phi, \phi, \phi, \phi\}$

$\boxed{V_{13} = \{\phi\}}$

find V_{23}

$i=2 \quad j=3$

$k = 1$ to 2.

$V_{23} = V_{ik}\, V_{i+1}\, V_{j-k} \cup V_{ik}\, V_{(i+1)(j-k)}$

$= V_{21} V_{32} \cup V_{22} V_{41}$

$= \{AS, AC, CS, CC\} \cup \{BB\}$

$V_{23} = \{CC\} \qquad \phi$

$= \{B\}$.

find V_{33}

$i=3 \quad j=3$

$k = 1$ to 2.

$V_{33} = V_{31} V_{42} \cup V_{32} V_{51}$

$= \{S, C\}\{A, S\} \cup \{S, C\}\{A, C\}$

$= \{SA, SS, CA, CS\} \cup \{SA, SC, CA, CC\}$

$= \{\phi, \phi, \phi, \phi\} \cup \{\phi, \phi, \phi, B\}$

$= \{B\}$

<u>Row 4:</u>

find $V_{1,4}$

$i=1 \quad j=4$

$k = 1$ to 3

$V_{14} = V_{ik}\, V_{(i+k)(j-k)} \quad [k=1]$

$\cup$

$V_{ik}\, V_{(i+k)(j-k)} \quad [k=2]$

$\cup$

$V_{ik}\, V_{(i+k)(j-k)} \quad [k=3]$

$= V_{11} V_{23} \cup V_{12} V_{32} \cup V_{13} V_{41}$

$= \{B\}\{B\} \cup \{A, S\}\{S, C\} \cup \{\phi\}\{B\}$

$= \{BB\} \cup \{AS, AC, SS, SC\} \cup \{\phi B\}$

$= \phi \cup \{\phi, \phi, d, \phi\} \cup \{\phi\} = \{\phi\}$

$V_{24} = V_{21} V_{33} \cup V_{22} V_{42} \cup V_{23} V_{41}$

$= \{A, C\}\{B\} \cup \{B, S\}\{A, S\} \cup \{B\}\{B\}$

$= \{AB, CB\} \cup \{BA, BS, SA, SS\} \cup \{B, B\}$

$= \{S, \phi\} \cup \{A, \phi, \phi, \phi\} \cup \{\phi\}$

$= \{S, A\}$

<u>Row 5:</u>

find. V_{15} $i = 1$ $j = 5$.

$k = 1, 2, 3, 4$

$V_{15} = V_{11} V_{24} \cup V_{12} V_{33} \cup V_{13} V_{42} \cup V_{14} V_{51}$

$= \{B\}\{S, A\} \cup \{A, S\}\{B\} \cup \{\phi\}\{A, S\} \cup \{\phi\}\{A, C\}$

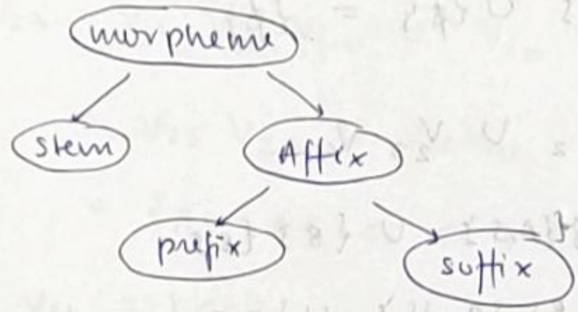
$= \{BS, BA\} \cup \{AB, SB\} \cup \{A, S\} \cup \{A, C\}$

7/10/24.

Lexical Semantics

Morphology → formation of new words using morphemes.

Morphemes → a word / part of a word that has meaning



stem : believe
prefix : un
suffix : able.
un believ able.

beauty + ful → Beautiful (adjective)

Bus + es → Buses (noun)

Lexeme: an entity in the lexicon table.

- pairing of orthographic & phonological form.

Lexicon - a table with finite list of lexemes.

Sense - lexemes meaning component.

Relations between lexemes & their senses:

1. Homonymy
2. Polysemy
3. Synonymy
4. Antonymy
5. Hyponymy
6. Hypernymy
7. Meronymy
8. Holonymy.

1. Homonymy :

- Relation between 2 lexeme with the same form & unrelated meanings.

eg. $\underbrace{\text{Bank} \to \text{river bank} \quad \& \quad \text{Bank} \to \text{financial bank}}_{\text{same PoS}}$

$\underbrace{\text{would} \to \text{auxillary} \quad \& \quad \text{wood} \to \text{noun}}_{\text{different PoS}}$

Be Bee

$\underbrace{\text{auxillary} \quad \text{noun}}_{\text{different POS}}$

2. Polysemy :

- A single lexeme with multiple meanings.

eg. Bank → river bank
→ blood bank
→ financial bank.

Register → CPU register
→ attendance register
→ stock register.

3. Synonymy

- Relation between 2 lexemes with the same meaning.

eg. big, large → How big is that building?
→ How large is that building.

price, fare → what is the cheapest fare?
→ what is the cheapest price?

4. Antonymy

- Relation between 2 lexemes with opposite meaning.

eg. big - small, rich - poor, rise - fall, up-down.

5) Hyponymy

- Relation between 2 lexeme - x & y where x is subclass of y.

eg. Car is a hyponymy of vehicle.
Mango is a hyponymy of fruit
Neem is a hyponymy of tree

6. Hypernymy:

- Relation between 2 lexeme - x,y where x is superclass of y.

eg. Vehicle is hypernymy of car.
Fruit is a hypernymy of mango.
Tree is a hypernymy of neem.

7. Mesonymy:

- Relation between 2 lexeme, x&y where x is member/ part of y.

eg. Finger is a mesonymy of hand.
Wheel is a mesonymy of vehicle.
leaf is a mesonymy of tree.

8. Holonymy:

- Opposite to that of mesonymy
- It is a relation between 2 lexemes x,y where x has y.

eg. Hand is a holonymy of finger (hand has finger)
Tree is a holonymy of leaf (tree has leaf).

8/10/24.

Word Net

- a lexical database for English language.
- a database of lexical relations.

2 ways to create such a database.

1) mining information from existing dictionaries & thesaurus
2) handcrafting a database from the scratch

wordnet
↳ developed using 2nd approach by Backwith et al, 1991

3 seperate database in wordnet:

1. for nouns
2. for verbs
3. for adjective & adverbs

noun - bass
↓
8 senses.

Scope of wordnet of 1.6 release.

category	unique forms	no. of senses
noun	94474	116317
verb	10319	22066
adjective	20170	29881
adverb	4546	5677.

- directly accessed with a browser (or) programmatically through C library files.

⇒ Noun Relations in WordNet:

→ Relation	Definition	Example
Hypernym	from concepts to supertypes.	vehicle ← car
Hyponym	from concepts to subtypes	car ← vehicle
Has - member	groups to members	Faculty to professors
Has - part	from whole to parts	table to legs
part of	from parts to whole	legs to table.
antonym	opposite.	leader to follower

⇒ verb Relations in wordnets

Relations	Definition	Example.
Hypernym	from events to superordinate events	fly → travel
Troponym	from events to their subtypes	walk → stroll
Antonym	opposite.	increase → decrease.

⇒ Adjective & adverb relations.

Relation	definition	example
Antonym	opposite	heavy → light quickly → slowly.

Q. Consider the CNF grammar G defined for a language L

S → AB | BC (R1)

A → BA | a (R2)

B → CC | b (R3)

C → AB | a (R4)

Check whether the following strings ∈ L(G) or not using CYK parser.

1) aba 3×3
2) aab 3×3
3) aaba 4×4.