6 CHAPTER NATURAL LANGUAGE PROCESSING

6.0 INTRODUCTION

A **natural language** is any of the languages naturally used by humans that is not artificial or man-made language. Natural Language Processing (NLP) is a subfield of AI and linguistics. NLP provides methods for performing useful tasks with natural languages. To give a feel of a human expert, computer is needed to interact with the user in natural languages. **What is done in NLP?** The answer is here.

S1 : User inputs in form of a natural language.

S2 : It goes to the Natural Language Interface (NLI).

S3 : Output is obtained in a language form that is understood by the application program.

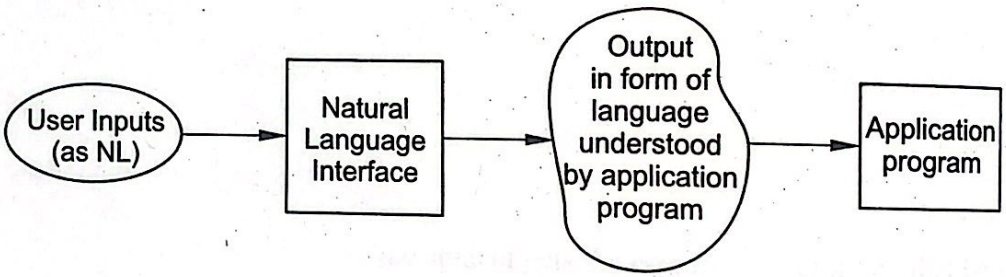


Fig. 6.1 NLP system.

KBS need query languages for information retrieval. A good system should interact with database in natural language. To interact with the database in natural languages, computer is required to have basic knowledge of alphabets, lexicon, grammar, words formation, sentence formation rules of that language. Also required are general knowledge, commonsense knowledge, domain specific knowledge etc for sentence interpretation in a proper way—

NLP includes—

(*a*) Speech Synthesis.

(*b*) Speech Recognition.

(*c*) Natural language understanding.

(*d*) Natural language generation.

(*e*) Machine Translation.

Speech synthesis: Synthesis of natural sounding speech is complex as it needs some understanding of what is being spoken to ensure for correct information.

Speech recognition: Basically, the reduction of continuous sound waves to discrete words.

Natural language understanding
It involves moving isolated words to meaning.
Natural language generation
It involves generating appropriate NL responses to unpredictable inputs.
Machine Translation (MT): Translation of one NL into another.

Please note that natural language understanding is sometimes referred to as an AI-complete problem because natural language recognition seems to require extensive knowledge about the outside world and the ability to manipulate it.

6.1 NLP

6.1.1 Levels of Knowledge Used in Language Understanding

Language understanding needs—

(*a*) **Phonological knowldge**

A phoneme is the smallest unit of sound and relates to the sound of word.

(*b*) **Syntactic knowledge**

It relates to how words are put together to form grammatically correct sentence.

(*c*) **Semantic knowledge**

It relates to the meaning of words and phrases and how they combine to form a meaningful sentence.

(*d*) **Morphological knowledge**

It relates to word construction from basic units called **morphemes.**

(*e*) **Pragmatic knowledge**

It relates to the use of sentence in different contexts and how the contexts affect the meaning of sentences.

(*f*) **Word knowledge**

It relates to the language a user must have in order to understand and carry on a conversation.

6.1.2 Phases of Natural Language Understanding

1. **Morphological analysis:** Individual words are analyzed into their components, and nonword tokens, such as punctuation, are separated from the words.
2. **Syntactic analysis:** Linear sequences of words are transformed into structures that show how the words relate to each other. Some word sequences may be rejected if they violate the language's rules for how words may be combined. For example, an English syntactic analyzer would reject the sentence "Boy the go the to store".

Syntactic analysis determines: Whether the sentence is a legal sentence of the language or generates legal sentences using a grammar and lexicon and if so returns a fare tree for the sentence representing its structure. It is the process of parsing that is computers, equivalent of diagramming a sentence.

3. **Semantic analysis:** The structures created by the syntactic analyzer are assigned **meanings.** In other words, a mapping is made between the syntactic structures and objects in the task domain. Structures for which no such mapping is possible may be rejected. **For example,** in most universes, the sentence "Colorless green ideas sleep furiously" would be rejected as **semantically anomalous.**

Thus, it is the process of extracting the meaning of an utterance as interpreted by a speaker. It takes the parse tree for the sentence and interprets it according to the possible meanings of its constituent parts. A representation of semantics may include information about different meanings of words and their characteristics.

4. **Discourse integration:** The meaning of an individual sentence may depend on the sentences that precede it and may influence the meanings of the sentences that follow it. **For example**, the word "it" in the sentence, "John wanted it" depends on the prior discourse context while the word "John" may influence the meaning of later sentences (such as, "He always had".)

A discourse is any string of language usually which is more that one sentence long. For eg., Textbooks novel, weather reports and conversations are all discourses. The meaning of an individual sentence may depend on the sentences which precede it and may also influence the meaning of the sentences which follow it.

For eg. the word 'it' in the sentence, "**Mayur wanted it**" depends on the prior discourse, it may refer to a bike, ,Mayur is interested in purchasing and he purchased it. While the word, 'Mayur' may influence the meaning of later sentences such as "He purchased the bike" This type of interpretation is of a pronoun or a definite noun phrase which refers to an object in a world. The resolution is based on knowledge of the world and of the previous parts of the discourse. Usually, resolution is just a matter of selecting a referent from a list of candidates but sometimes it involves the creation of new candidates.

Please understand that choosing the best referent is a process of disambiguation which relies on combining. a variety of semantic, syntactic and pragmatic information. But some clues in form of constraints are required.

The rhythm and intonation of a language refers to **Prosody**. Rhythm is often used in the babbling of infants and children's wordplay.

5. **Pragmatic analysis:** The structure representing what was said is reinterpreted to determine what was actually meant. **For example**, the sentence "Do you know what time it is?" should be interpreted as a request to be told the time.

The boundaries between these five phases are often very fuzzy. The phases are sometimes performed in sequence, and they are sometimes performed all at once. If they are performed in sequence, one may need to appeal for assistance to another. **For example**, part of the process of performing the syntactic analysis of the sentence "Is the glass jar peanut butter?" is deciding how to form two noun phrases out of the four nouns at the end of the sentence (giving a sentence of the form "Is the x >>?"). All of the following constituents are syntactically possible: glas, glass jar, glass jar peanut, jar peanut better, peanut butter, and butter. A syntactic process or on its own has no way to choose among these, and so any decision must be made by appealing to some model of the world in which some of these phrases make sense and others do not. If we do this, then we get a syntactic structure in which the constituents "glass jar" and "peanut butter" appear. Thus although it is often useful to separate these five processing phases to some extent, they can all interact in a variety of ways, making a complete separation impossible.

6.1.3 Parsing and Its Types

The process of converting a sentence into a tree that represents a sentences syntactic structure is known as parsing. It tell us whether it is a valid sentence as defined by our grammar. If a sentence is not a valid sentence then it can't be parsed.

For example : Consider the following sentence

"The white dog crossed the road". Its parse tree is as shown in Fig. 6.2.

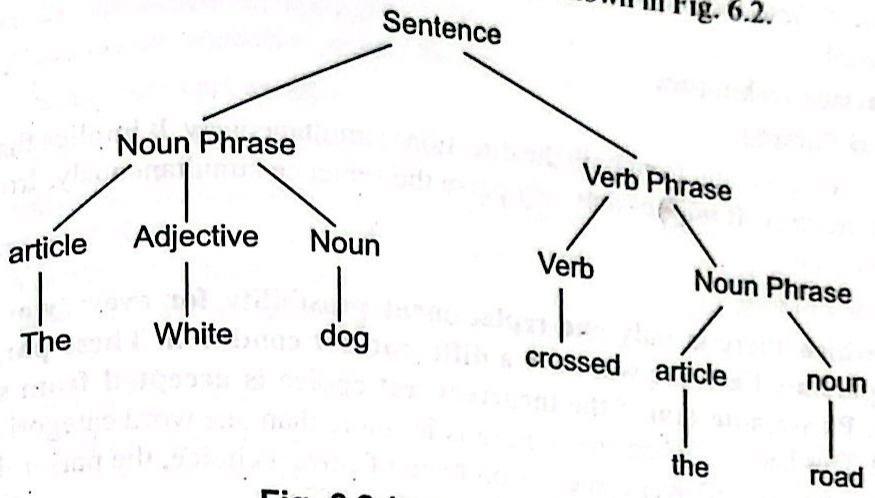


Fig. 6.2 A parsed sentence.

It is clear from the above Fig. 6.2 that how a sentence is made up of a **noun phrase** and a **verb phrase**. The noun phrase consists of an article, an adjective and a noun. The verb phrase consist of a verb and a noun phrase, in turn, consists of an article and a noun. Parsing is, thus, a typical **AI search problem** in which —

(*a*) **The initial state** is input sequence of words.

(*b*) **The goal state** is a complete tree representing the whole sentence structure.

(*c*) **The production rules** are grammar rules.

(*d*) **The parser** can be viewed as searching through the space of all possible parse trees to find the correct parse tree for the sentence in consideration.

Parsing is of two types—

(*a*) **Top-down parsing.**

(*b*) **Bottom-up parsing.**

Parse trees can be built in a top-down fashion or in a bottom-up fashion. Let us understand this now.

(*a*) **Top-down Parsing**

Building a parse tree from the top-down involves starting from a sentence and determining which of the possible rewriters for a sentence can be applied to the sentence that is being parsed. Hence, in this passing the sentence would be rewritten using the following rule:

Sentence ⟶ Noun phrase, Verb phrase

Then the verb phrase and the noun phrase would be broken down **recursively** in the same way, until only terminal symbols are left.

(*b*) **Bottom-up parsing**

To build a parse tree from the bottom-up, the terminal symbol of the sentence are first replaced by their corresponding non-terminals (e.g. dog is replaced by noun) and then these non-terminals are combined to match the right-hand sides of rewrite rules.

Fog example: 'the' and 'road' (see Fig. 6.2) will be combined using the following rewrite rule:

NounPhrase ⟶ Article, Noun

But neither of these techniques adequately exploits the constraints presented by the grammar and the input word.

Some other Parsing techniques

1. Bidirectional Parsing

These parsers start the parsing from both the directions simultaneously. It implies that they parse the sentence simultaneously. It implies that they parse the sentence simultaneously, from 'left to right' and from 'right to left'.

2. Deterministic Parsing

A parsing in which there is only one replacement possibility for every word is known as deterministic parsing. Each arc will have a different test condition. These parsers cannot do back tracking. Please note that if the incorrect test choice is accepted from some state, the parser will fail. This happens when one word satisfies more than one word categories like noun and verb or adjective and verb. To over come this problem of correct choice, the parser uses **look ahead mechanism**. But even then designing such parsers is not difficult.

3. Non-Deterministic Parsing

A parsing that allows different arcs to be labeled with the same test is known as Non-Deterministic parsing. The next test from any state may not be uniquely determined by the state and the current input word. The parser guesses at the proper constituents and then do backtrack if guess is proven to be wrong.

6.1.4 Transition Networks and Its Types

A transition network is a finite state automaton that is used to represent a part of a grammar. A transition network parser uses a number of these transition network to represent its entire grammar. Each network represents one non-terminal symbol in the grammar.

Actually, **transition network is a method of parsing which represents the grammar as a set of finite state machines (FSM).** A FSM is a model of computational behaviour where each node represents an internal state of the system and the **arcs** are the means of moving between the states. They are used in automata theory to represent grammar. In case of parsing of natural language, the arcs in the networks represent either a terminal or a non-terminal symbol. Rules in the grammar correspond to a path through a network. Each non-terminal is represented by a different network.

When moving from one state to the next through the network, the parser tests the **label** on the **arc.**

Case 1. If it is a terminal symbol, the parser will check whether it matches the next word in the input sentence.

Case 2. If it is a non-terminal symbol, the parser moves to the network for that symbol and attempts to find a path through that.

Case 3. If it finds a path through that network it returns to the higher-level network and continues. If the parser fails to find a path through that network, it backtracks and attempts another path. If it succeeds in finding a path, the sentence is a **valid one.**

Please understand that the parsing of the sentence continues in this fashion until the top-level sentence network is successfully traversed. Also note that the transition network allows each non-terminal to be represented in a single network rather than by numerous rules. This makes this approach more concise than grammars. But there is a disadvantage too. These transition network does not produce a parse tree for sentences. Also tracing the path through the network can be unclear for complex sentences.

How do we represent transition networks?

Transition networks are based on application of directed graphs (or digraphs) and finite state automata. This graph is now used to represent the syntactic structure of the sentence.

Nodes—Represent various states.

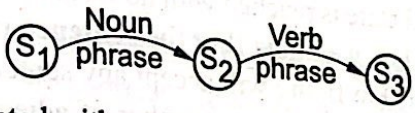
Arcs—Represent transition from certain states to final state.

We start at a given node, transverse an arc if the current word in the sentence is in the category on the arc. **Please note that if an arc is followed, the current word is updated to the next word. Also note that a sentence is accepted by transition network, if there is a path from the start node to a final node with arc accounting for every word in a sentence.**

How it works?

In each transition network, S_1 is the start state. When a phrase is applied to a transition network, the first word is compared against one of the arcs leading from the first state. If this word matches one of those arcs then the network moves to the next state.

For e.g.: Consider the following transition network—



When this network is presented with a noun phrase, it will move from state S_1 to State S_2.

Please understand that if a phrase is presented to a transition network and no match is found from the current state then that network cannot be used and another network must be tried. Also remember that transition networks can be used to determine whether a sentence is grammatically correct, at least as per the rules of the grammar the networks represent. And parsing using transition involves exploring a search space of possible parses in a depth-first fasion. Let us consider an example to understand this **"A dog sat".**

We begin in state S1 in the sentence transition networks. To proceed, we must follow the arc that is labeled as Noun phrase. Thus, we move out from the sentence network into the Noun phrase network. The first arc of the Noun phrase network is labeled as **Noun**. So, we move into the Noun networks. We now follow each of the arc in the Noun network and discover that our first word, 'A', does not match any of them. Hence, we **backtrack** to the next arc in the Noun phrase network. This arc is labeled as Article, so we move onto the Article transition network. Here, on examining the second label we find that the first word i.e. matched by the terminal symbol on this arc. Therefore, we consume the word, "A", and move on to the state S_2 in the **Article network**. Because this is a success node, we are able to return to the **Noun phrase network** and move on to the state S_2 in this network. We now have an arc labeled Noun. As before, we move into the Noun network and find that our next word, "dog" matches. So, we move to state S_4 in the Noun phrase network. This is a success node and so we move back to the sentence network and repeat the process for the verb phrase arc. **Please note that it is possible for a system to use transition networks to generate a derivation tree for a sentence and also to determine whether the sentence is grammatically valid, it parses it fully to obtain further information by semantic analysis from the sentence.** This can be done by simply having the system build up the tree by noting which arc it successfully followed—

For e.g.: When it successfully follows the Noun phrase arc in the sentence network, the system generates a root node labeled sentence and an arc leading from that node to an new node labeled Noun phrases when the system follows the Noun Phrase network and identifies an article and a noun, these are similarly added to the tree. In this way, a full parse tree for a given sentence can be generated using transition networks.

As explained earlier that the starting point in studying ATN'S is the finite–state transition diagram (FSTS) as shown below.

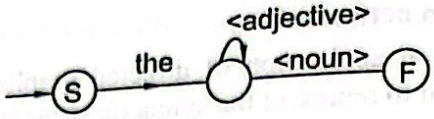


Fig. The FSTD that can process any noun phrase starting with "the".

FSTD diagrams consist of a set of **nodes** and **arcs** connecting the node. The rules followed are:

1. All terminal symbols (words of the sentence) are represented as **arcs.**
2. One state (or node) is defined as 'S' the START state and one subset of states as the FINAL state 'F'.
3. A FSTD specifies a (dynamic) **process** for operating on arbitrary sentences instead of a static relationship between words as in a semantic network. **That is why an ATN is sometimes called as a finite state machine.**

The FSTD processes the input sentence, one word at a time, checking to see if the word matches the prescribed structure . **If it matches**, the word is placed on the appropriate arc and removed from the input sentence. When the final state is reached with no words in the sentence left over, we say the FSTD has accepted the sentence i.e., we **know that the sentence matches the syntactic structure specified by the FSTD**. The FSTD in figure will accept any sentence segment beginning with the, ending with a noun' and containing an arbitrary number of adjectives inbetween. Like, the big white, dog would be accepted by this FSTD.

The main problem with FSTD is that it can recognize only sentences written as a regular language (type 3) which is the most restrictive of chomsky's four categories of language. It cannot handle sentences from the more general context-free (type -2) grammar.

Merits and demerits of parsing using transition networks

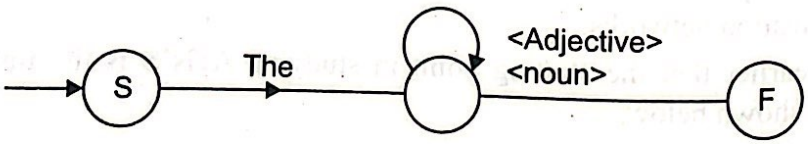
This type of parsing is simple but not very efficient. No attention is given to potential ambiguities or the need of words to agree with each other.

Other variants (types) of transition networks

Augmented Transition Networks (ATNs)

ATN or augmented transition network was given by William Woods in 1970. The ATN method of parsing sentences integrates many concepts from Chomsky's (1957) formal grammar theory with a matching process resembling a dynamic semantic network. Such parsing schemes may be considered as recursive pattern matching in which the string of words of the sentence are mapped onto a meaningful syntactic pattern. **Please note that the characteristics of an ATN can be understood by studying the progression from finite-state transition diagrams through recursive transition network to augmented transition networks. Also note that each level of this progression represents increased levels of incorporating options for semantic input into the formal syntactic structure of the parsing program.** The objective is to take an arbitrary input sentence and assign each word of the sentence its proper part of speech.

1. It is a modified form of transition networks.
2. It uses top down parsing procedure to gather various types of information to be later used for understanding system.
3. ATNs have the ability to apply tests to arcs.
4. ATN for a sentence will be



But, here, the arc from node S_2 to S_3 would be conditional. **The condition here, will be that the number of verbs should be same as the number of nouns, only then S_2 to S_3 transition is possible.** The conditions on the arc are calculated by the procedures that are attached to the arcs. The procedure attached to an arc is called when the network reaches that arc. These procedures, as well as, carrying out checks on agreement, are able to form parse tree from the sentence that is being analyzed.

5. ATNs not only do parsing but also collects the sentence features for further analysis.
6. ATNs represent sentence structure using a slot filler representation which reflects more of the functional role of phrases in a sentence.

Recursive Transition Networks (RTNs)

RTN is a recursive transition network which permits arc labels to refer to other networks (including the networks own name) and they in turn may refer back to the referring network rather than just permitting word categories used previously.

For e.g., Consider the RTN shows in Fig. 6.5 where the main network calls two sub networks—a NP and a PP network.

The top network in the figure is the top level (sentence) network and the lower level networks are for NP and PP arc states. The arcs corresponding to these states will be transversed only if the corresponding sub networks (b) or (c) are successfully transversed. Note that the arc named POP is used as a dummy arc to signal the successful completion of the sub network and a return to the node following the arc from which it was called.

The capability for recursion gives the RTN a considerable advantage over its predecessor, the FSTD. There is no restriction on the level of hierarch or **recursion with RTNs.** That is, a **sub-network may call another sub-network which may, in turn, call another and so on. Recursion is supported by allowing a sub-network to call itself.**

There is also no restriction on the number of sub-networks which may be called from a certain arc. Several sub-structures may be investigated simultaneously, and thus RTNs have the capability of parallel processing. Often, of course, many of the parallel sub-networks will fail to parse the phrase they are examining because it may not fit the structure specified by the RTN.

However, even context-free grammars cannot handle all English sentence structure. To extend the capabilities of syntactic parsers, the augmented transition network (ATN) was invented.

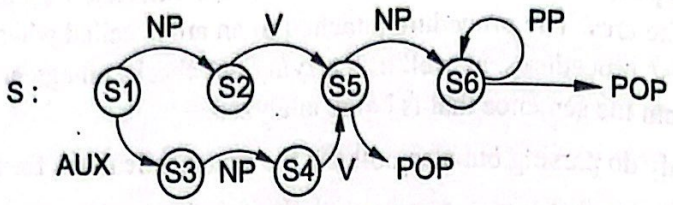
To further extend the capacity of finite state transition diagrams to handle more general context-free language, a recursion mechanism was added to the FSTD and it became a **recursive transition network** (RTN). This extension is like adding a subroutine capability at each arc. Now, in addition to containing a terminal (word), an are may contain the name of a sub-network to which the control may be transferred to parse like a prepositional phrase. When the sub-network is finished, control returns to the subsequent node and the processing continues.

Advantages of RTN

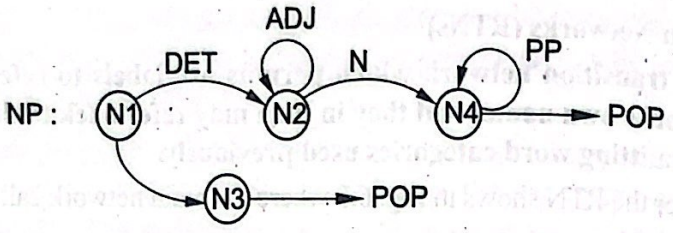
1. No restriction on the level of hierarchy or **recursion** with RTNS. **It means that a sub-network may call another sub-network which may call another and so on. This process is known as recursion.** Recursion is supported by allowing a sub-network to call itself.
2. No restriction on the number of sub-networks which may be called from a certain arc. Several sub-structures may be investigated simultaneously. **Thus, RTNs have the capability of parallel processing too.**

Many parallel sub-networks will fail to parse the phrase they are examining because it may not fit the structure specified by the RTN.

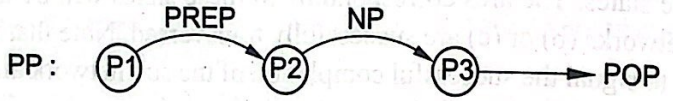
NOTE: Even context free grammars cannot handle all English sentence structures. To further extend the capabilities of syntactic parsers, ATNS were invented.



(a) Top level RTN



(b) Noun phrase subnetwork



(C) Prepositional phrase subnetwork

Fig. 6.5 Recursive transition network (RTN).

6.1.5 Chart Parsing

The chart is a record of all the substructures build during parsing. It is also known as a **well-formed substring table.** Chart parsing works on the **principle of dynamic pgoramming** i.e., it uses and stores solutions of smaller part of a sentence to solve larger problems. Chart parsing works **incrementally** i.e., word by word. **In worst case,** chart parsing will parse a sentence of n words in $O(n^3)$ time. In many cases, it will perform better than this and will parse most of the sentences in $O(n^2)$ or even O(n) time.

For example, say the sentence is —

"The cat eats a big rat"

It's initial chart is as follows—

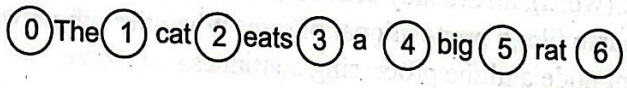


Fig. 6.6 Initial chart for given example.

How its chart parsing is done?

The chart of Fig. 6.6 show seven verticals which will become connected to each other by edges. The edges will show how the constituents of the sentence combine together.

The chart parser starts by adding the following edge to the chart—

$$[0, 0, \text{Target} \longrightarrow \bullet \text{Sentence}]$$

This notation means that the edge connects vertex 0 to itself. The first two numbers in the square brackets show which vertical the edge connects. Target is the target that we want to find which is really just a place holder to enable us to have an edge that requires us to find a whole

sentence. The **arrow** indicates that in order to make what is on its LHS (target) we need to find what is on its RHS (sentence). The **dot** shows what has been found already on its LHS, and what is yet to be found on its RHS.

Now, consider the following edge, show in chart of Fig. 6.7.

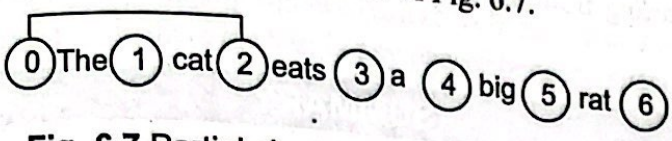


Fig. 6.7 Partial chart for the given sentence.

Now, the chart parser adds the following edge—

$$[0, 2, \text{Sentence} \longrightarrow \text{Nounphrase} \bullet \text{Verphrase}]$$

This means that an edge exists connecting nodes 0 and 2. The dot shows us that we have already found a Noun phrase (the cat) and that we are looking for a verb phrase. Once we have found that verb phrase, we will get what is on the LHS of the arrow i.e., a sentence.

The chart parser can add edges to the chart using the following three rules:

Rule 1: **If we have an edge [x, y, A → B · C], which needs to find a C, then an edge can be added that supplied that C i.e., the edge [x, y,(c) → E], where E is some sequence of terminals or non-terminals which can be replaced by C.**

Rule 2: **If we have two edges [x, y, A → B · CD] and [y, x, C → E] then these two edges can be combined together to form a new edge i.e. [x, z, A→ BC · D].**

Rule 3: **If we have an edge [x, y, A → B · C] and the word at vertex y is of type C, then we have found a suitable word for this edge and so we extend the edge along to the next vertex by adding the following edge**

$$[y, y + 1, A \rightarrow BC \bullet]$$

Let us apply these three rules to our example now.

We start with the edge [0, 0, Target . sentence] which means that to find our target, we must first find a sentence.

Use rule-1, so we can add the following edge to chart:–

[0, 0, sentence → Noun phrase Verb phrase]

This means that we must now find a Noun phrase and a Verb phrase. We now apply rule-1 again, to try to find a suitable Noun phrase, which involves adding the following edge:

[0, 0, Noun phrase → · Article Noun phrase]

Now, we can apply rule-3 because the word at the end of this edge (form vertex 0 to vertex 0) is "The", which is an article. This would be determined by looking the word up in a lexicon. Hence, we can now add the following edge:

[0, 1, Noun phrase → Article · Noun phrase]

Now, we are looking for another Noun phrase, so we use rule-1 again to add the following edge:

[0, 1, Noun phrase → · Noun]

We can now use rule-3 again because the next word is indeed a Noun, to add the following edge to the chart:

[0, 2, Noun phrase → Noun •]

This process continues, until we have reached an edge in which we have found everything we need. So, the **final edge** will be –

[0, 6, Sentence → Noun phrase Verbphrase •]

To build a parse tree from the chart, we modify rule-2 so that when it combines two edges together, it stores in the new edge information about the two edges that were combined to form the children edges. Then, when the parse has completed, we can obtain the parse tree directly from the edges of the tree by starting from the first edge and **recursively** examining the children edges of each node.

SUMMARY

English, Hindi, French etc are all natural languages but computers cannot understand them. NLP or computational Linguistics is the scientific study of languages from a computational perspective. A computer is said to be intelligent only if it can understand the commands given in natural languages.

MULTIPLE CHOICE QUESTIONS [MCQS]

1. KBS stands for

(*a*) Knowledge based software (*b*) Knowledge basis system

(*c*) Knowledge based system (*d*) None of the above.

2. The process of converting a sentence into a tree that represents a sentence's syntatic structure is known as

(*a*) Parsing (*b*) Grammar

(*c*) Discourse integration (*d*) None of the above.

3. ATN stands for

(*a*) Automated Teller Network (*b*) Automatic Truth Network

(*c*) Augmented Transition Network (*d*) None of the above.

4. Which is the following is NOT Transition Network—

(*a*) ATN (*b*) RTN (*c*) ATM (*d*) None of the above.

5. Chart parsing works on the principle of

(*a*) Dynamic programming (*b*) Backtracking

(*c*) Greedy method (*d*) None of the above.

6. In worst case, the time complexity of chart parsing is

(*a*) O(n) (*b*) $O(n^2)$ (*c*) $O(n^3)$ (*d*) All of the above.

ANSWERS

1. (*c*) **2.** (*a*) **3.** (*c*) **4.** (*c*) **5.** (*a*)
6. (*c*)

CONCEPTUAL SHORT QUESTIONS WITH ANSWERS

Q. 1. What are the various passes in natural language processing system? Explain the problem (syntactic, semantic or pragmatic), if any, with the following sentences—

(*a*) Mohan a letter Mary sent to. **(*b*) He takes a tea with milk.**

(*c*) Rice flies like an apple. **(*d*) A boy study in a college.**

[MDU, BE (CSE)-6th Sem., Dec. 2007, 2008 and GGSIPU, B. Tech (CS)-8th Sem., May 2010]

Ans. NLP phases are as follows—

1. **Speech segmentation**

In most spoken languages, the sounds representing successive letters blend into each other, so the conversion of the ana log signal to discrete characters can be a very difficult process. Also in natural speech, there are hardly any pauses between successive words; the location of those boundaries usually must take into account grammatical and semantic constraints, as well as the context.

2. **Text segmentation**

Some written languages like Chinese, Japanese and Thai do not have single - word boundaries either, so any significant text parsing usually requires the identificaiton of work boundaries, which is often a non-trivial task.

3. **Part-of-speech tagging**

Many words have more than one meaning; we have to select the meaning which makes the most sense in context.

4. **Syntactic ambiguity**

The grammar for natural languages is ambigous i.e., there are often multiple possible parse trees for a given sentence. Choosing the most appropriate one usually requires semantic and contextual information. Specific problem components of syntactic ambiguity include sentence boundary disambiguation.

5. **Imperfect or irregular input**

Foreign or regional accents and vocal impediments in speech, typing or grammatical errors, OCR errors in texts.

6. **Speech acts and plans**

A sentence can often be considered as an action by the speaker. The sentence structure alone may not contain enough information to define this action.

Let us answer for the given sentences now—

(*a*) **Mohan a letter Mary sent to**

This sentence is **syntactically** and **semantically** wrong but **pragmatically** correct.

(*b*) **He take a tea with milk**

This sentence is **syntactically** wrong.

(*c*) **Rice flies like an apple**

This sentence is **pragmatically** and **syntactically** wrong.

(*d*) **A boy study in a college**

This sentence is semantically incorrect.

Q. 2. Name some NLP systems along with their year of development.

Ans. (*a*) ELIZA System (1966-67)—By Weizenbaun at MIT.

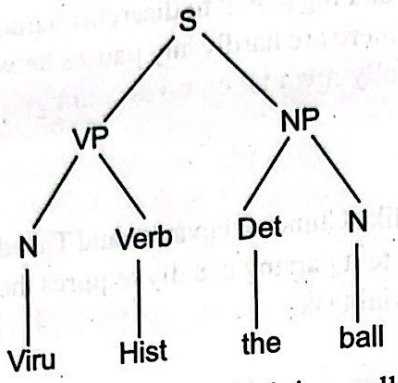
(*b*) LUNAR System (1970)

(*c*) SHRDLU System (1970).

LUNAR is a question–answering system in AI. SHRDLU was a dialogue system which could converse with a human user.

Q. 3. Draw a parse tree for the sentence—"Viru hits the ball". **[UPTU, B. Tech (CSE) 6th Sem., 2004-05]**

Ans.



This is a CFG i.e., context-free grammar which is a collection of productions of the form S → NP VP i.e., a constituent S can consist of sub-constituents NP and VP.

Q4. Explain top-down and bottom-up parsing for the following grammar—

S → NP VP

NP → ART N

VP → V NP

N → boy/frog

V → ate/kissed

ART → the/a

where S is initial symbol for sentences

NP is a Noun phrase

VP is Verb phrase

N is Noun

V is Verb

and ART is article

Apply this grammar on the following sentence— "A boy ate the frog" and write top-down and bottom-up parsing. **[UPTU, B. Tech (CSE)-6th Sem., 2004-05]**

Top-down parsing		Bottom-up parsing
S → NP VP	→	the/a boy/frog ate/kissed the/a boy/frog
→ ART N VP	→	the/a boy/frog ate/kissed the/a N
→ the/a N VP	→	the/a boy/frog ate/kissed NP
→ the/a boy/frog VP	→	the/a boy/frog V NP
→ the/a boy/frog V NP	→	the/a boy/frog VP
→ the/a boy/frog ate/kissed NP	→	the/a N VP
→ the/a boy/frog ate/kissed ART N	→	ART N VP
→ the/a boy/frog ate/kissed the/a N	→	NP VP
→ the/a boy/frog ate/kissed the/a boy/frog	→	NP VP
	→	S

Applying the grammar to given sentence gives—

Top-down parsing		Bottom-up parsing
S → NP VP	→	A boy ate the frog
→ ART N VP	→	A boy ate the N
→ A N VP	→	A boy ate ART N
→ A boy VP	→	A boy ate NP
→ A boy V NP	→	A boy V NP
→ A boy ate NP	→	A boy VP
→ A boy ate ART N	→	A N VP
→ A boy ate the N	→	ART N VP
→ A boy ate the frog	→	NP VP
	→	S

Q5. What is Fillmore's grammar? Give an example.

[Coachin University of Science & Technology, B. Tech. (CSE)-8th Sem., Oct 2000]

Ans.

1. Fillmore grammar is also called as **case grammar**. Case grammar provides a different approach to the problem of how syntactic and semantic interpretation can be combined. Grammar rules are written to describe syntactic rather than semantic regularities. But the structures, rules are produced correspond to semantic relations rather than to stricitly syntactic ones. As an example, consider the two sentences and the simplified forms of their conventional parse trees shown in Fig. 6.9.
2. Case grammar describes the relationship between verbs and their arguments parsing using a case grammar is usually expectation-driven. Once verb of the sentence is located, it can be used to predict the noun phrases that will occur and to determine the relationship of those phrases to the rest of the sentences.

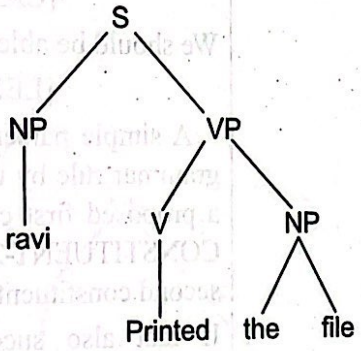


Fig. 6.9 Parsing

3. A case grammar describes the correct set of deep cases. Some of the cases are given as follows:
   - Agent – Investigator of the action.
   - Instrument – cause of the event.
   - Dative – entity affected by the action.
   - Factitive – object.
   - Locative – place of the event.
   - Source – place from which something moves.

Q.6. NLP is a classical AI problem. Explain how.

Ans. **1. Minimal input data**

The natural language statement does not contain the message but is a minimal specification to allow an intelligent reader to construct the message.

2. Knowledge based

The interpretation of the message is based on the large part of the knowledge that the reader already has.

3. Local ambiguity

Many wrong interpretations are superficially consistent with the input.

4. Global constraints

There are many different kinds of constraints on the interpretation of the input.

5. Capturing the infinite

A language understanding system must capture inifinite form — rules sufficient to understand a potentially infinite set of statements.

Q. 7 What is the difference between logical unification and graphical unification?

Ans.

Logical Unification	Graphical Unification
1. The inputs to the logical unifications are treated as logical formulas.	1. The inputs to the graph unification must be treated as sets.
2. Order in which attribute-value pairs are stated matters.	2. Order does not matter.
For eg f(g(a), h(b)) is different than f(h(b), g(a))	**For eg** if a rule is described as a constituent as follows [CAT : DET LEX : {1}] We should be able to match this constituent as [LEX : the CAT : DET] A simple parser can use the method to apply a grammar rule by unifying CONSTITUENT-1 with a proposed first constituent. If that succeeds then CONSTITUENT-2 is **unified** with a proposed second constituent. If that also succeeds then a new constituent corresponding to the value of BUILD IS PRODUCED. If there are variables on the value variables in the value of BUILD which were bound during the matching of constituents then those binding will be used to build the new constituent.

Q. 8. Derive the parse tree for the sentence "Bill loves the frog where the following rules are used.

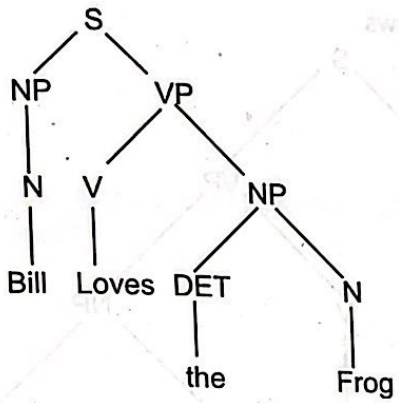
$S \rightarrow NP\ VP$

$NP \rightarrow N$

$NP \rightarrow DET\ N$

CP → V NP
DET → the
V → loves
N → bill frog

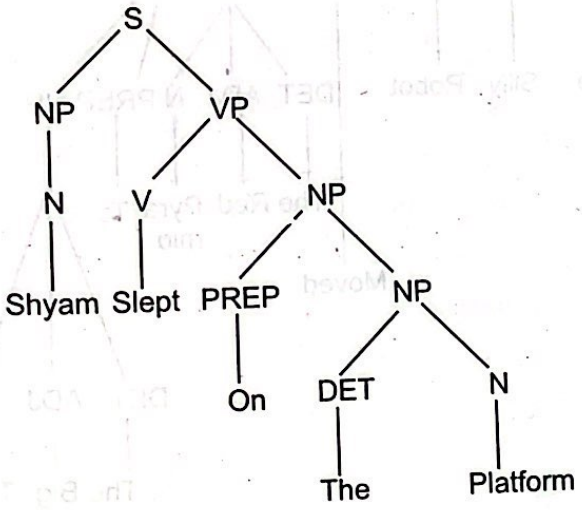
Ans.



Q. 9. Develop a parse tree for the sentence "Shyam slept on the platform" using following rewrite rules.

S → NP VP
NP → N
NP → DET N
VP → V PP
PP → PREP NP
N → Shyam | platform
DET → the
PREP → on

Ans.



Q. 10. Draw the parse tree for sentence "Mary likes a chocolate" where rewrite rules are given as below:

S → NP VP
NP → N

NP → DET N

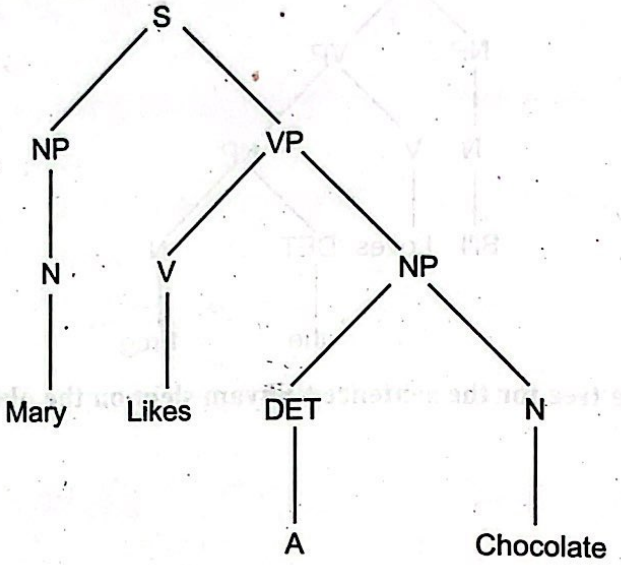
VP → V PP

DET → a

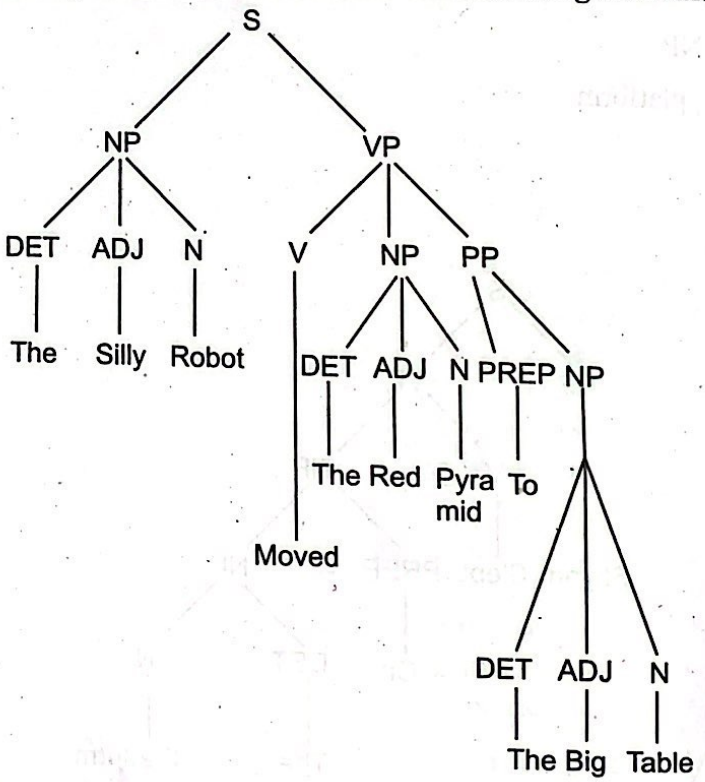
V → likes

N → Mary/chocolates.

Ans. The pares tree is as follows



Q. 11. Given the following parse tree for the sentence— "The silly robot moved the pyramid to the big table." Write down the context free grammar.



Ans. Its context free grammar is

S → NP VP

NP → DET ADT N

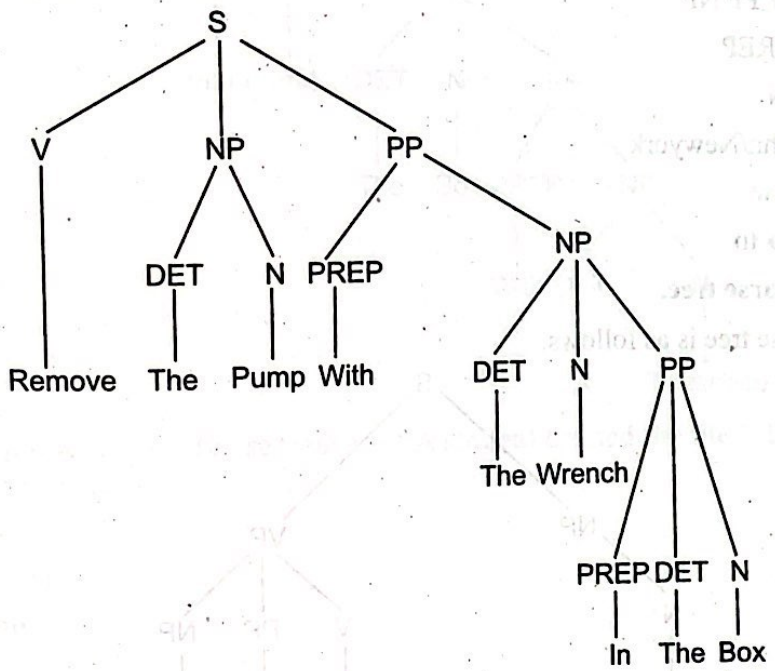
VP → V NP PP

NP → V PP
NP → DET ADJ N
PP → PREP NP
NP → DET ADJ N
DET → the
ADJ → silly/red/big
N → robot/pyramid/table
V → moved
PREP → to

Q. 12. Given are the rewrite rules, draw a parse tree for the sentence — "Remove the pump with the wrench in the box."

S → V NP VP
NP → DET N
PP → PREP NP
NP → DET N PP
PP → PREP NP
NP → DET N
DET → the
V → remove
N → pump/box
PREP → with/in

Ans. The parse tree is a follows.



Q. 13. Given the rewrite rules for the sentence — "The boy kissed a rabbit". Draw the parse tree.

S → NP VP
NP → ART N

VP → V NP

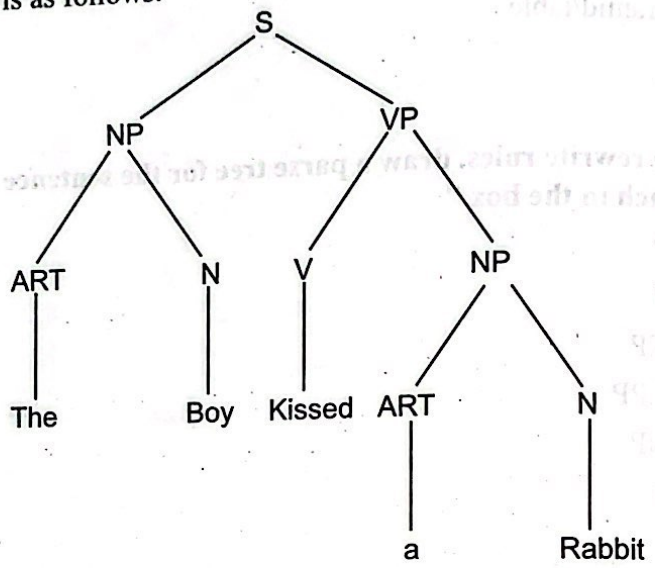
NP → ART N

V → kissed

ART → the/a

N → boy/rabbit.

Ans. The parse tree is as follows.



Q. 14. Given is a sentence "John flew to Newyork" Rewrite rules are given as:-

S → NP VP

NP → N

VP → V PP NP

PP → PREP

NP → N

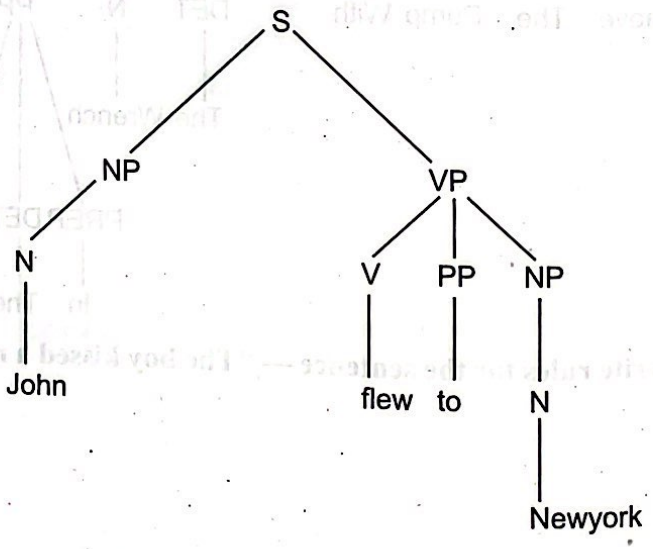
N → John/Newyork

V → flew

PREP → to

Draw parse tree.

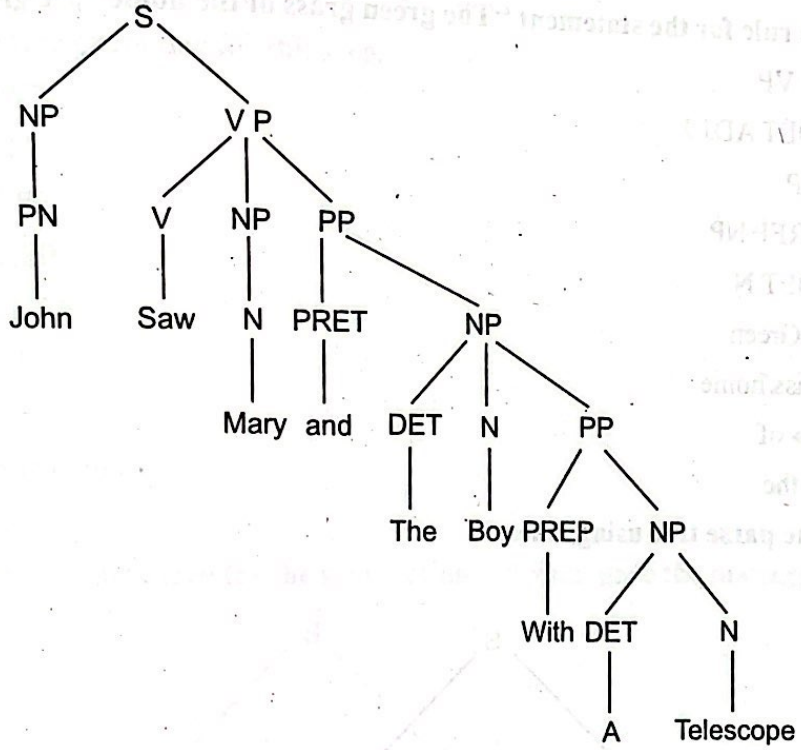
Ans. The parse tree is as follows.



Q. 15. Draw the syntax (parse tree) when rewrite rules are given for the sentence "John saw Mary and the boy with a telescope".

S → NP VP
NP → N
VP → V NP PP
NP → N
PP → PREP NP
NP → DET N PP
NP → DET N
PP → PREP NP
N → John/Mary/boy/telescope
V → Saw
PREP → and/With
DET → a/the.

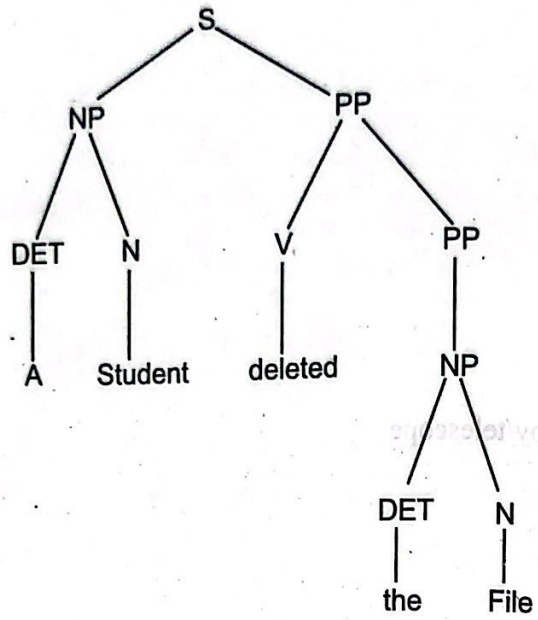
Ans. The parse tree is.



Q. 16. Draw the parse tree for the sentence— "A student deleted the file". The rewrite rules are as given below:-

S → NP VP
NP → DET N
VP → V PP
PP → NP
NP → DET N
DET → a/the
N → student/file
V → deleted.

Ans.



Q. 17. Rewrite rule for the statement "The green grass of the home" are given as follows

$S \rightarrow NP\ VP$

$NP \rightarrow DET\ ADJ\ J$

$VP \rightarrow PP$

$PP \rightarrow PREP\ NP$

$NP \rightarrow DET\ N$

$ADJ \rightarrow Green$

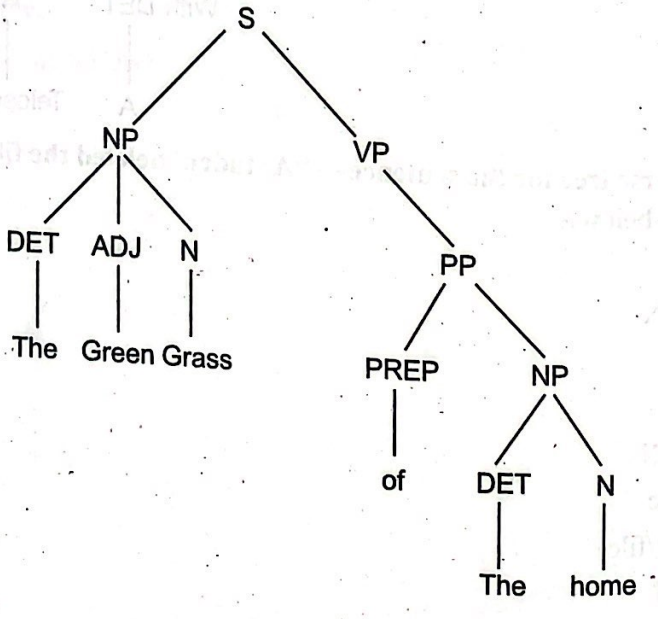
$N \rightarrow grass/home$

$PRET \rightarrow of$

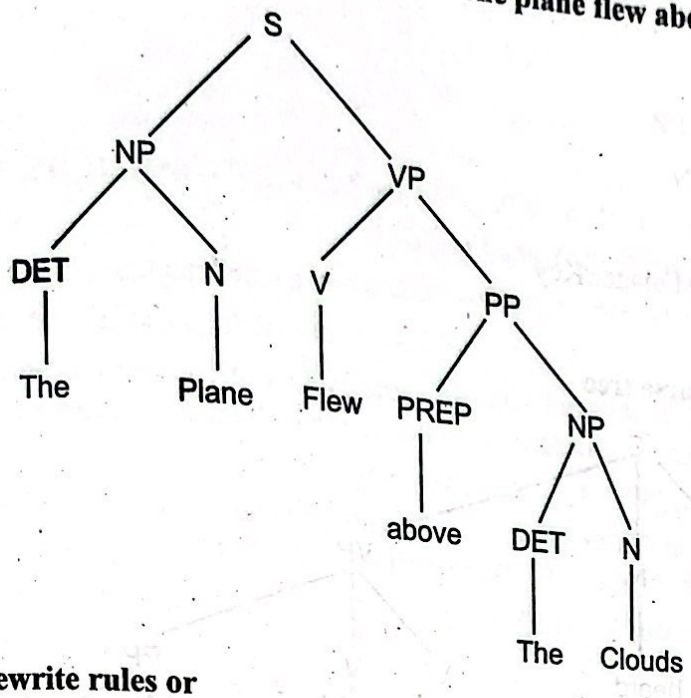
$DET \rightarrow the$

Draw the parse tree using rules.

Ans.



Q.18. Given is the parse tree for the sentence as— "The plane flew above the Clouds."



Write the rewrite rules or

Context free grammar for this tree.

Ans. S → NP VP

NP → DET N

VP → V PP

PP → PREP NP

NP → DET N

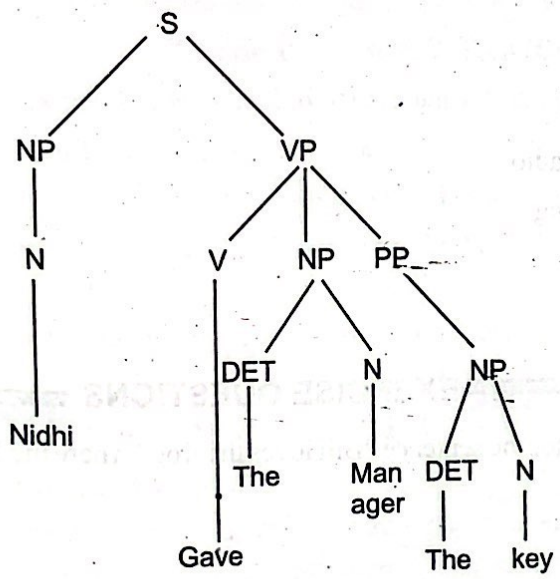
DET → the

PREP → above

N → plane/clouds

V → flew.

Q.19. Given is the parse tree for the sentence as— "Nidhi gave the manager the key."



Ans. S → NP VP

NP → N

VP → V NP PP

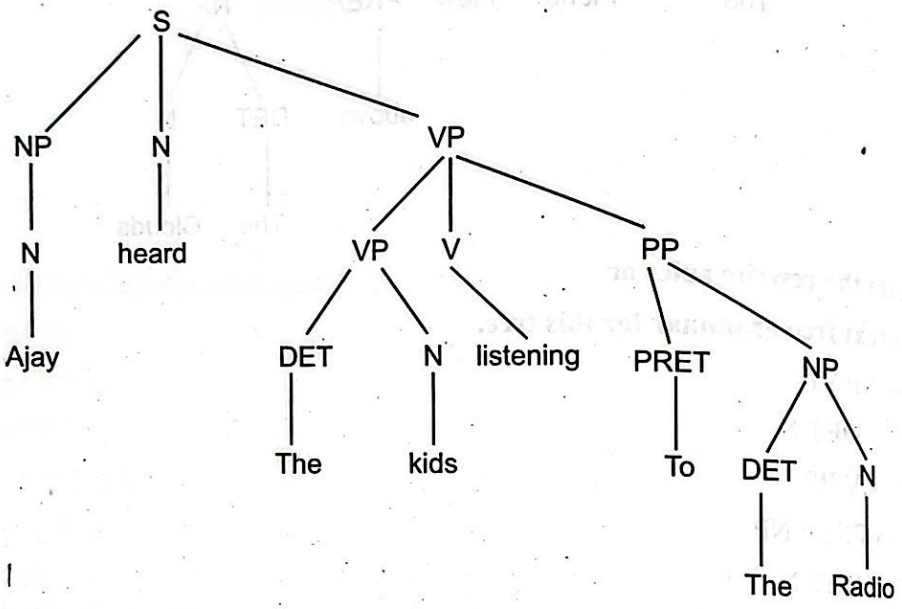
NP → DET N

DET → the

N → Nidhi/Manager/Key

V → gave

Q. 20. Given the parse tree



Write its context free Grammar.

Ans. S → NP VP

NP → N

VP → NP V PP

NP → DET N

PP → PREP NP

NP → DET N

N → Ajay/kids/radio

V → heard/listenig

DET → the

PREP → to.

EXERCISE QUESTIONS

Q. 1. Give a parse tree for the sentence "Bill loves the frog" where the following rules are used—

S → NP VP

NP → N

NP → DET N

$VP \rightarrow V\ NP$

$DET \rightarrow$ the

$V \rightarrow$ loves

$N \rightarrow$ bill/frog

[GGSIPU, B. Tech (CSE)-8th Sem., May 2008 & PTU, MCA-4th Sem., 2009]

Q.2. Describe ATN. **[GGIPU, B. Tech (CSE) 8th Sem., May 2009]**

Q.3. (*a*) What are important challenges in Natural Language Understanding?

(*b*) What are the various applications of NLP?

(*c*) Explain the various activities which are performed during Morphological Analysis and Syntactic processing with the help of following sentence: I want to see Raju's photo.

[GGSIPU, B. Tech (CSE)-8th Sem., May 2010]

Q.4. What are the different ways in which ambiguity results in natural langauge statements? Give an example of each type. **[RTM, Nagpur University BE (IT)-7th Sem., Summer 2003]**

Q.5. Develop a parse tree for the sentence- "Jack slept on the table" using the following rules—

$S \rightarrow NP\ VP$

$NP \rightarrow DET\ N$

$PP \rightarrow PEP\ NP$

$V \rightarrow$ Slept

$PRBP \rightarrow$ on

$NP \rightarrow N$

$VP \rightarrow VPP$

$N \rightarrow$ Jack/table

$DET \rightarrow$ the

[RTM, Nagpur University, BE (IT)-7th Sem., Summer 2008 & Winter 2008]

Q.6. (*a*) Explain types of a grammars used in NLP.

(*b*) What are the levels of knowledge used in Natural Language Understanding.

[RTM Nagpur University, BE (IT)-7th Sem., Summar 2009]

Q.7. (*a*) Explain the problems you may face in parsing Indian Languages?

(*b*) Design an ATN for recognizing the language $\{ a^n b^{2n} c^{3n}$, for $n > 0\}$.

[Cochin University, B. Tech (CSE)-8th Sem., May 2004]

Q.8. (*a*) Explain how lexical, semantic and Syntactic analysis are carried out in NLP.

(*b*) Briefly explain different parsing techniques.

[CUSAT, B. Tech (CSE)-8th Sem., Oct. 2004]

Q.9. (*a*) Using CFG, draw and explain a parse tree for the English Sentence—"The man bites the dog".

(*b*) Explain how can we combine syntac and semantic knowledge. Use sentence, if required.

[CUSAT, B. Tech (CSE)-8th Sem., Oct 2004]

Q.10. What are the various phases of NLP? **[MDU, BE (CSE)-6th Sem., Dec. 2008]**

Q.11. (*a*) Describe chowsky hierarchy of languages.

(*b*) Differentiate between syntactic and Semantic processing.

[GGSIPU, B. Tech (CSE)-8th Sem., May 2011]