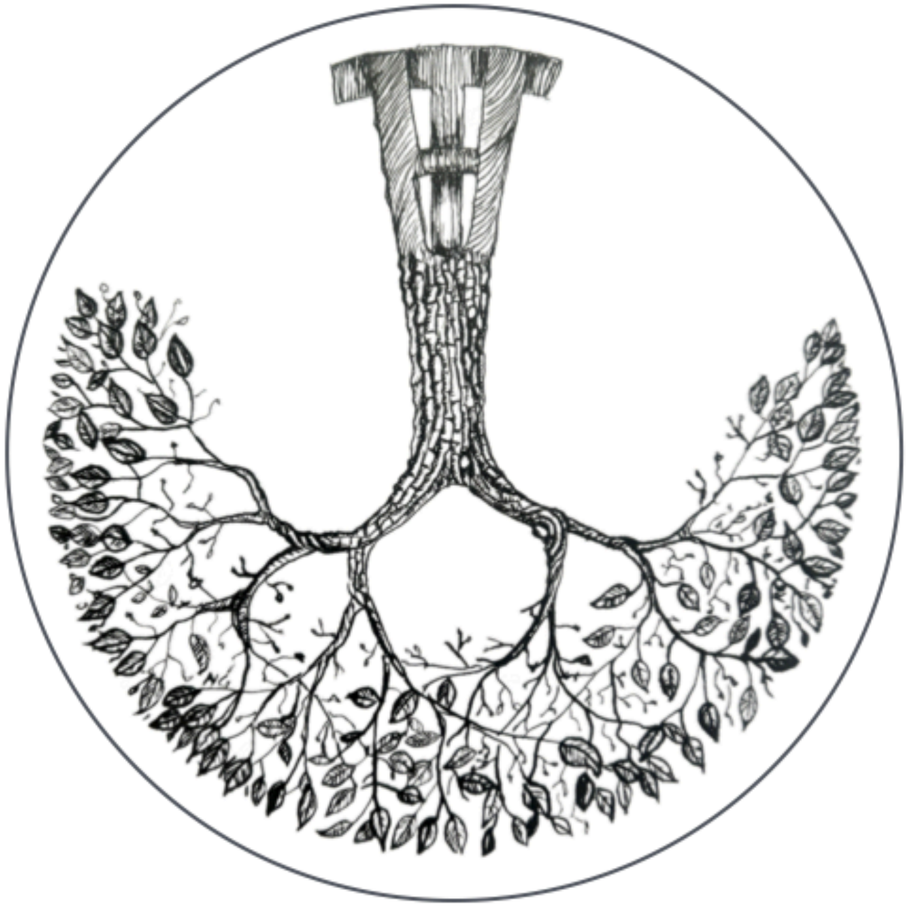
||urdhvamūlam adhaḥśākham||



The Hindi Tree

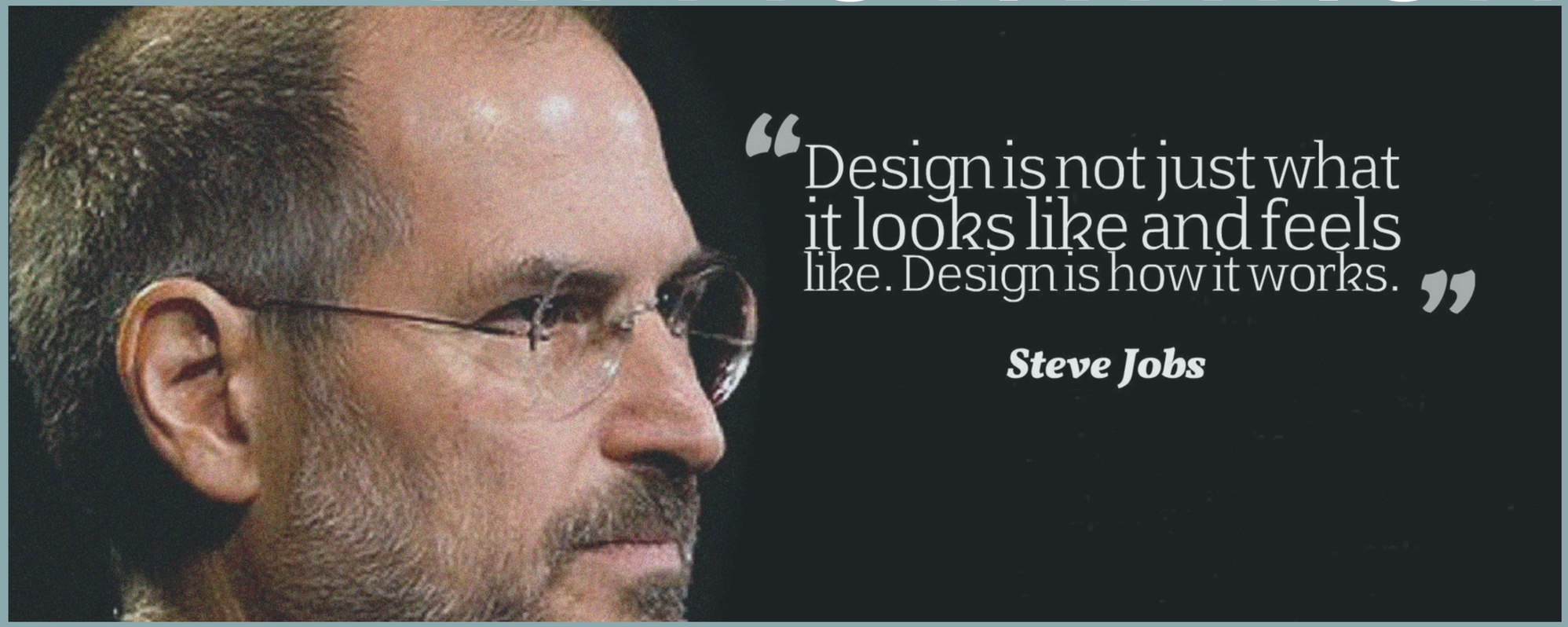
TECH-ING the THEORY

THE HINDI TREE

A Parsing Tool For Hindi

TABLE OF CONTENT

MY MOTIVATION



INTRODUCTION

We are proud to introduce you to The Hindi Tree (THT), a software solution to parse sentences of Hindi Presently, THT can deal with a small fragment of Hindi.

Our parser works differently from other available parsers in its algorithm for the syntactic parsing. It facilitates the logical treatment of lexical items at the semantic stage.

Design : Nirmalya
Sketch : Nirmalya & Rajit

FOR THE BUSYBEES

All current and future updates will be reported through our project website, which is hosted on Github.

Weblink:
https://iamalinguist.github.io/hinditree/

Developers - Vivek Tripathi | Dinesh Rathod

01.

WORD COMBINATION

Syntactic Arrangement

PROBLEM

Think of an imaginary situation, when my colleague, let's say Rajit, fathered a son. The nurse came and said :

- Your son is born.
- आपके बेटे का जन्म हुआ है.
- पुत्रस्ते जातः.

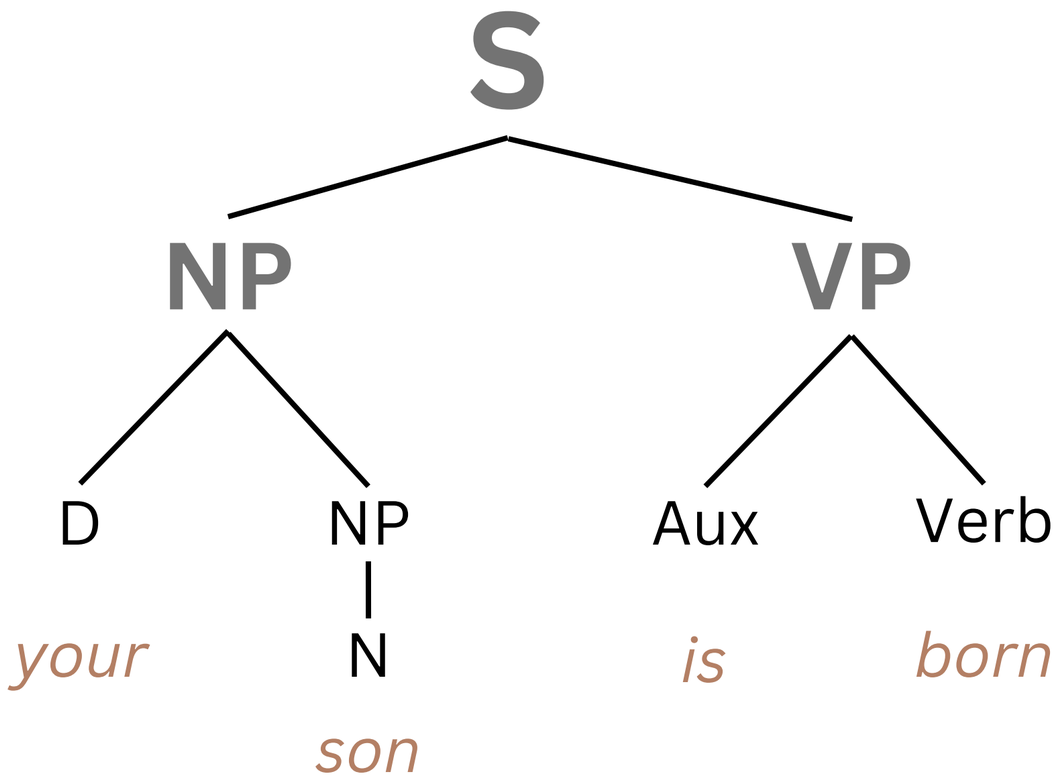
Rajit Smiled..... But Why ?

SOLUTION

When Rajit received those words, a process happened inside his cognitive faculties.

Rajit checked the combination to see if they were grammatical or not.

The valid configuration of a sentence is called its syntactic arrangement.



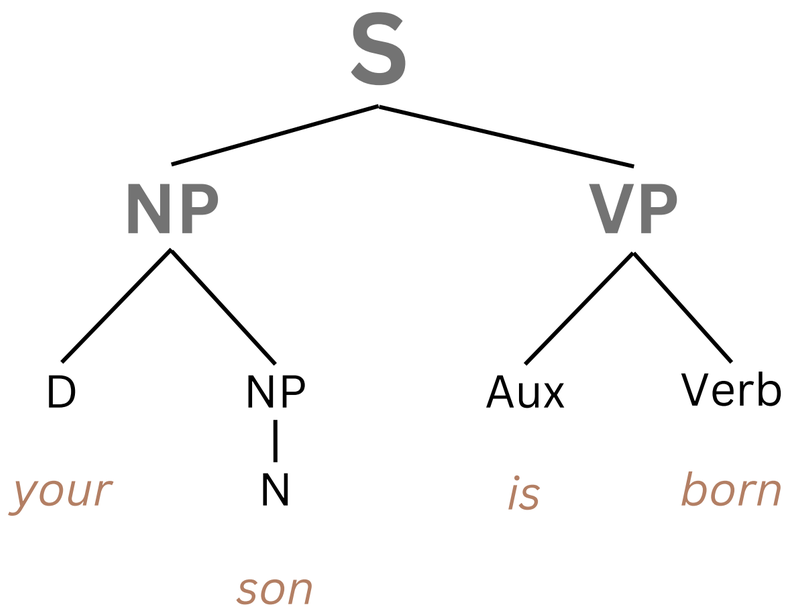
SOLUTION

When Rajit received those words, a process happened inside his cognitive faculties.

Rajit checked the combination to see if they were grammatical or not.

The valid configuration of a sentence is called its syntactic arrangement.

SOLUTION

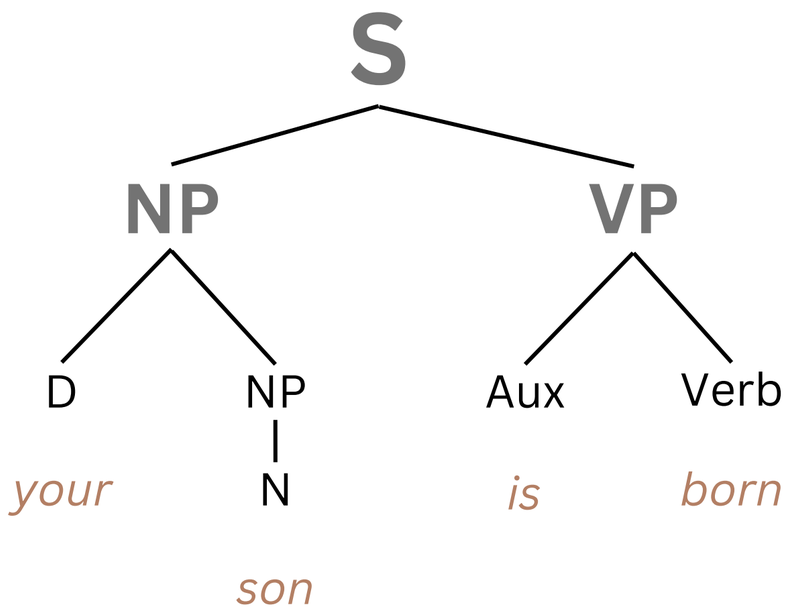


Checkpoints:

- ‘your son’ combining with ‘is born’ is a valid configuration in this setup.

- not ‘your son + are born’,
- not ‘your son + are borning’
- nor ‘your son + born are’’

or any such other ungrammatical forms.



SOLUTION

Thus, the combination should follow some rules. We DEVELOPED SUCH RULES first (to form grammatical sentences) and developed a software tool which can do this for a small set of data to begin with.

- Individuals: rāma, śyāma, sītā
- Verbs: cal, dekh, so, jān

Given as Glossary Here:
https://iamalinguist.github.io/hinditree/demo.html

The Journey So Far...

But did that word arrangement make Rajit happy?

The answer is No. This is not the end of his or our cognitive process. This is just the beginning of understanding what is happening inside his mind ‘meaning-wise’ after the arrangement of words has been corectly done.

02.

MEANING COMBINATION

Semantic Judgment

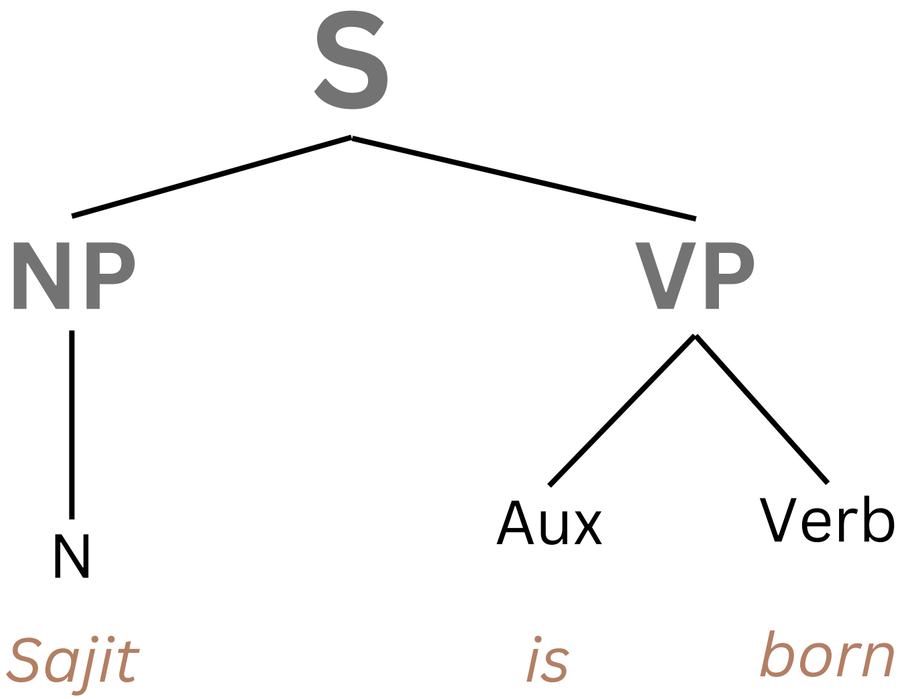
NO FACT CHECKING

was performed with the real world whether a son was born to Rajit or not. So, what made Rajit happy ?

PRINCIPLE OF COMPOSITIONALITY

was applied to understand ‘meaning of sentence’. It says **semantic composition is function application (FA).** The meaning of a sentence is composed from the **meanings of its individual words** and the way those words are syntactically combined through FA.

MEANING



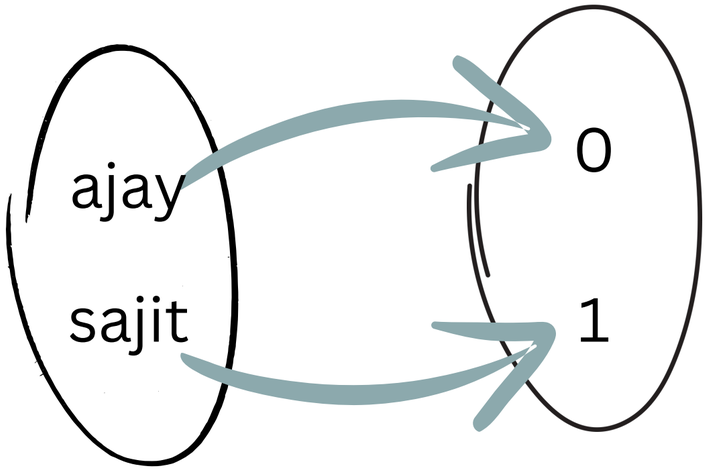
(let say, son's name is Sajit)

Rajit identifies the correctly drawn syntactical arrangement into two components, to understand its meaning:

- Saturated Component
- Unsaturated Component

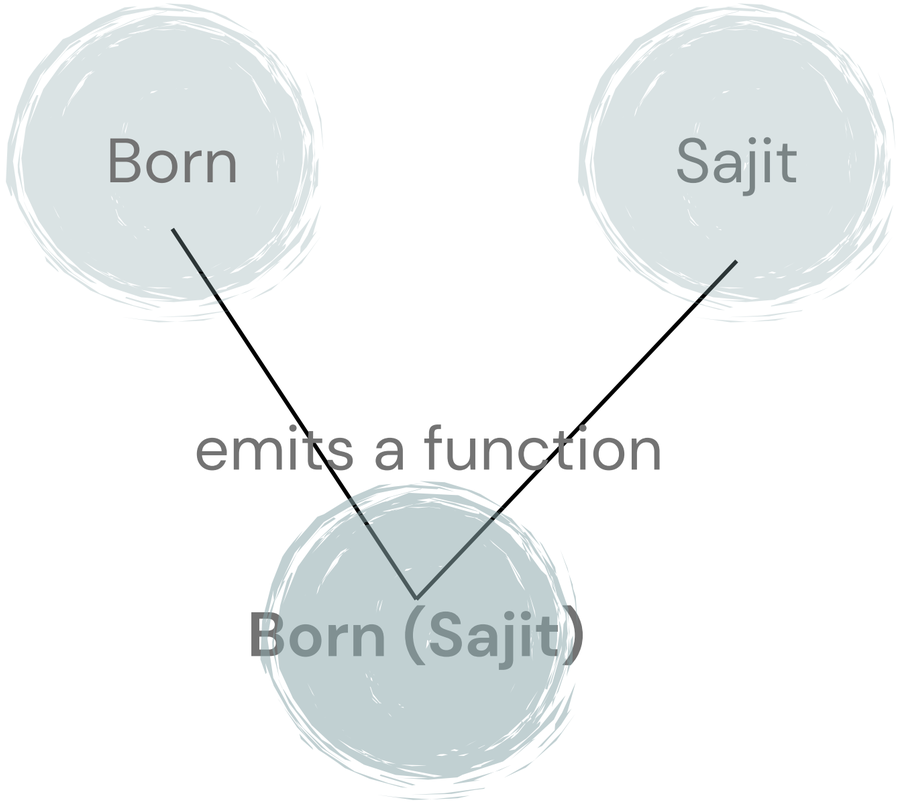
'Sajit' is a saturated component and 'Born' is unsaturated.

Saturated Lexical Items
as ‘Argument’



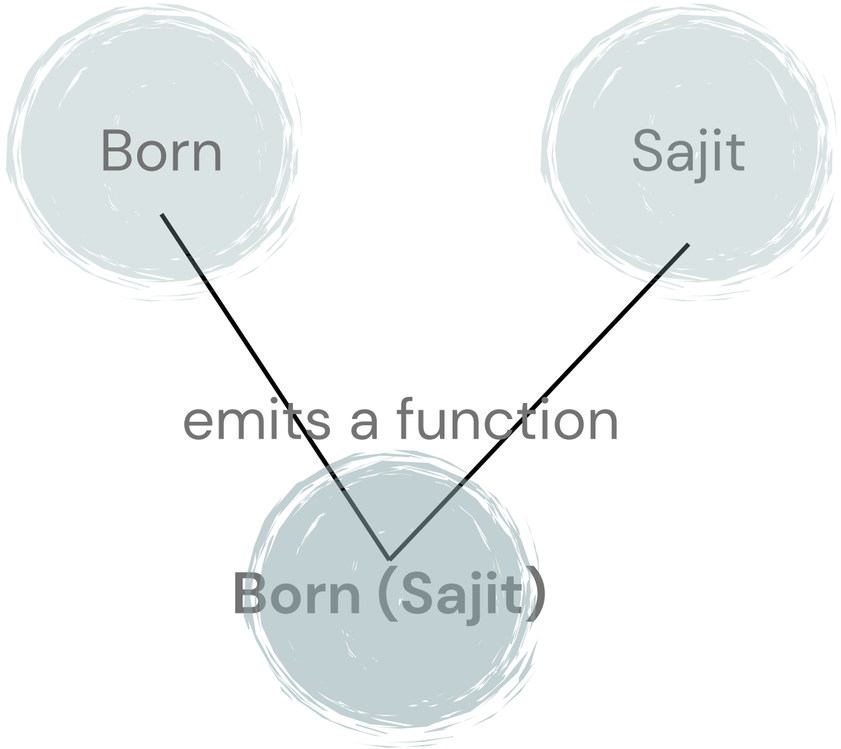
Unsaturated Lexical Item
‘Born’ as a Function

- Each thing that is ‘Born’ is mapped to the truth-value 1, thus they are ‘born’.
- All thing that is not ‘Born’ are mapped to the truth-value 0.
- Thus, born is actually a function which maps ‘individual’ (said as thing above) to 1 if they are member of ‘Born’ set and to 0 if they are non-members.



- Every composition can be seen as combination of an ‘unsaturated function’ and a ‘saturated argument’.
- This is called as ‘principle of compositionality’.
- Now, if the emitted function at the last node is giving value 1, then Rajit is happy else Rajit is not.

,



- However, it is very significant to mention that the value of Born(Sajit) is evaluated only when certain truth conditions are met.

- In this situations, truth conditions are:
  - there is atleast one individual called Rajit and that one individual called Sajit is part of our system (thus are arguments).
  - A relationship holds between them as part of our system (thus a functional relationship).

WE APPLIED THESE PRINCIPLES TO DEVELOP GRAMMAR AND FUNCTIONAL RELATIONSHIP FOR HINDI.

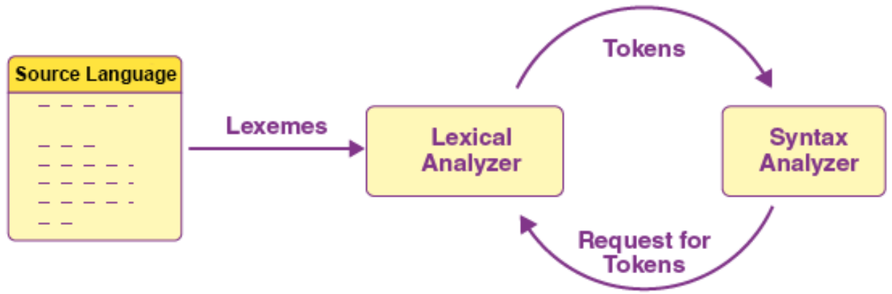
THUS, WE PRESENT TO YOU
‘THT PARSER’.

03.

DEMONSTRATION

Launching the Product

METHODOLOGY



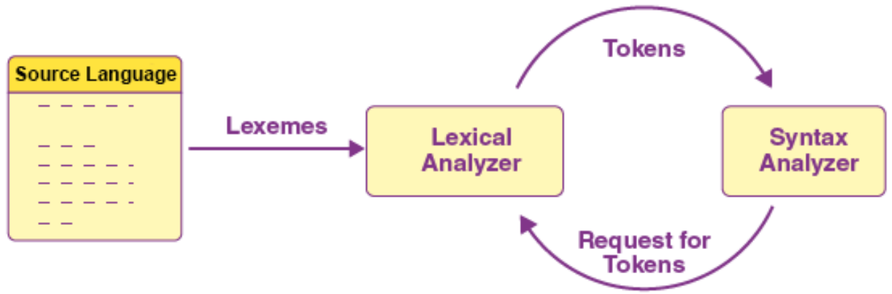
- It is accountable for terminating the comments and white spaces from the source language.
- It helps in identifying the tokens.
- Categorization of lexical units.

The program is written in Python language and it consists of two units.

Lexical Analyser: Lexical analysis or Lexical analyzer is the initial stage or phase of the compiler. This phase scans the source langauge, and transforms them into a series of a token.

- A token is basically the arrangement of characters that defines a unit of information in the source language.
- **NOTE:** In computer science, a program that executes the process of lexical analysis is called a scanner, tokenizer, or lexer.

METHODOLOGY



- Note syntax errors.
- Helps in building a parse tree.
- Acquire tokens from the lexical analyzer.
- Scan the syntax errors, if any.

Syntax Analyser: In the compilation procedure, the Syntax analysis is the second stage. Here the provided input string is scanned for the validation with the grammar rules we defined.

- Basically, in the second phase, it analyses the syntactical structure and inspects if the given input is correct or not in terms of grammar.
- It accepts tokens as input and provides a parse tree as output.
- It is also known as parsing in a compiler.

HOW TO DOWNLOAD

Search Just the Docs Template

Overview
About
Demo

Download The Hindi Tree 1.0.0

What's new: The semantic version 1.1.0 release adds a semantic interface

Masculine Noun	rāma, śyāma	ram, shyam
Feminine Noun	sītā	sita
Transitive Verb	jāna, dekha	jaan, dekh
Intransitive Verb	cal, so	chal, so
Auxiliary	tā-hai, tī-hai	ta hai, ti hai
Object	ko	ko
Conjunction	yā, aura	ya, aur
Negation	aisā nahī̃ hai ki	aisa nahi hai ki

Some Hint Sentences

- rām caltā hai.
- rām sītā ko jāntā hai.
- rām sītā ko nahī̃ jāntā hai.
- rām sītā ko dekhtā hai.
- aisā nahī̃ hai ki sītā caltī hai.
- rām caltā haiyā śyām rām ko sotā hai.
- rām caltā haiyā śyām sotā hai.
- sītā rām ko dekhtī hai aur rām caltā hai.
- śyām rām ko jāntā hai.
- aisā nahī̃ hai ki sītā sotī hai.
- aisā nahī̃ hai ki sītā sotī hai aur rām sitā ko dekhtā hai.

04.

FEEDBACK / QUESTIONS

You wish to suggest any improvement

05.

THANKYOU FOR YOUR SUPPORT AND COOPERATION.

Contact: sopan.tripathi@gmail.com