Journal Impact Factor (JIF)™ 2022: 0.2 | 5 Year: 0.2 | *Source: Journal Citation Report™ 2023 (Clarivate Analytics)*

AESTHETIX MEDIA

RUPKATHA JOURNAL ON INTERDISCIPLINARY STUDIES IN HUMANITIES

ISSN 0975-2935 | Indexed in the Web of Science Core Collection™ Emerging Sources Citation Index (ESCI)

https://doi.org/10.21659/rupkatha.v16n1.03g | Volume 16, Number 1, 2024 | Dynamic Impact Dimensions

Research article

Semantic Model for Fragment of Hindi (Part 1)

Vivek Tripathi[1]* ✉ & Dinesh Rathod[2] ✉

[1]Research Scholar, Indian Institute of Technology (BHU) Varanasi. *Corresponding author.

[2]Research Scholar, Indian Institute of Science,

Abstract

This paper proposes a formal model for syntactic and semantic analysis for the Hindi language using context-free grammar. In this paper, we developed a syntactic parser that generates syntactic trees for Hindi sentences based on rules of propositional logic, and gender conventions. The context-free rules we have written follow a top-down approach with a sentence that goes on self-arrangement. A set of experiments were run based on the corpus we have created, and significant results are presented in this paper. In addition to the above, the model characterizes lexical items in terms of individuals and sets for the syntactic distribution for well-formed formulas.

Keywords*:* Natural Languages Processing. Hindi Language Processing. Parser. Context-Free Grammar. Parse Tree. Context-Free Rules for Hindi. Montague Grammar. Look Ahead LR Parser.

SUSTAINABLE DEVELOPMENT GOALS **Quality Education**

Introduction

Language, as a quintessential facet of human communication, has been the subject of extensive study, analysis, and modeling in the field of linguistics and computer science. The quest to understand the structure and semantics of natural languages has driven researchers to develop formal approaches that can capture the essence of linguistic expression with mathematical rigour. Among the luminaries in this field, Richard Montague stands tall for his pioneering work on Montague semantics, a theory that aimed to bridge the gap between the complexities of language and the precision of formal logic. While Montague's work primarily focused on the English language, the applicability of formal semantic models extends far beyond any single language. He developed English-like fragments to show isomorphism between a mathematical artificial language with natural languages. In the realm of linguistics and computational linguistics, these models provide a foundation for understanding and representing the semantics of various natural languages, including Hindi, a language renowned for its intricate structures and rich linguistic nuances.

Article History: Received: 26 November 2023. Revised: 10 February 2024. Accepted: 11 February 2024. Published: 12 February 2024

Citation: Tripathi, V. & Rathod, D. (2024). Semantic Model for Fragment of Hindi (Part 1). *Rupkatha Journal* 16:1. https://doi.org/10.21659/rupkatha.v16n1.03g

After the 19th century, inspired by Euclid's approach, mathematicians aimed to reduce all of mathematics into a collection of theorems that could be logically derived from a limited set of axioms. Thus, Frege pioneered modern logic. Bertrand Russell and Alfred North Whitehead used the foundational techniques of logic in Principia Mathematica, which they considered the logical foundations of mathematics. Later, Montague (around the 1970s) used this logic to develop formal languages (commonly called Logical Representation Language (LRL)) for semantic analysis. He mapped these LRLs onto Fragments (commonly called Object Language (OL)) through his well-known papers PTQ (Dowty et al.,1981) and EFL (Montague, 1975). These fragments are 'natural-language-like' (or near natural language) languages having the capacity to be characterized by truth-conditions. For example, English-like Fragments (near English), Japanese-like Fragments (near Japanese), Dutch-like Fragments (near Dutch) etc. Within the generative grammar paradigm, this approach is usually called Montague grammar or Montague semantics. Thus, Montague semantics is also called truth-conditional semantics. In contemporary times, researchers (see section 2) are exploring the possibility of using truth-conditional semantics to derive logical inferences from any natural language.

The foundational works of Chomsky (Chomsky, 2009), Montague (Dowty et al., 1982), and Barbara Partee (Partee, 1973 and Partee, 1984) have provided the fundamental underpinnings for this research. Chomsky's insights into generative grammar, Montague's contributions to formal semantics, and Partee's work on compositionality have all deeply influenced the development of this study. In Chomsky's "Aspects of the Theory of Syntax[1]", he introduced the theory of transformational-generative grammar, which revolutionized the study of syntax and formal language theory, which motivated us to the development of efficient phrase structure rules (as reported in Part 1 of this research). Montague's work, notably in "Montague Grammar" (Dowty et al., 1982), laid the groundwork for the application of formal logic and semantics to natural language, creating a bridge between language and mathematical models. Partee's "Semantics and the Syntax of Quantifiers" is pivotal in our understanding of how quantifiers and compositional semantics operate, contributing significantly to the field of linguistic semantics.

Syntactic Distribution

Context-free grammar (CFG) is a formal system used to describe the syntax of a language. It is a set of rules that specify how strings of symbols, typically characters or tokens, can be combined to form valid sentences or programs in a language. Context-free grammars are a fundamental concept in computer science, linguistics, and formal language theory. CFGs provide rules for the identification and parsing of syntactic categories of any language. Thus, with CFGs, a machine can generate all possible well-formed formulas (wff), i.e., grammatical strings of a language and eliminate non-wffs. CFGs are used to define the syntax of programming languages, markup languages, and many other formal languages using a set of key components such as Start, Terminals and Non-terminal symbols, and Production rules.

[1] Chomsky, N. (1965). Aspects of the Theory of Syntax (50th ed.). The MIT Press. http://www.jstor.org/stable/j.ctt17kk81z

In this paper's first section, we would like to present a parser system using the proposed CFG. In contrast, in the second section, we present the idea of semantic entailment generation in the form of CPI using a formal set theoretical method. The system deals with many morphological and syntactical issues such as gender, post-positions, etc. As far as these are concerned, Hindi differs significantly from Germanic or Slavic languages. Hindi is an inflectional language that is a Verb-final language and uses free word order for its object and subject placement. The parser system presented here tries to evolve by updating its OL, i.e., Hindi.

We present here the first fragment (henceforth L_{0H}) of simple Hindi without quantification. This fragment involves the rules of propositional calculus (propositional logic or sentential logic). It provides a means to analyze arguments, construct valid proofs, and model real-world situations using key components such as truth tables, logical equivalences, inference rules, tautologies and contradictions, etc. To keep our OL simple, we have the singular number of non-referential noun phrases (NPs) and present tense in the auxiliaries of Hindi propositions. However, it can be intuitively extended for referential NPs and other tenses and modalities.

Many of the language scientists have tried a variety of ways to formalize the Hindi language through formal rules. Many researchers find it challenging to come up with formal rules for the Hindi syntax due to its complex inflectional and derivational mechanisms. Ambati et al. (2010) presented their work on investigating the local morphosyntactic features such as chunk type, head information etc. to incorporate dependency parsing of Hindi sentences. Ranjan Ray et al. (2003) have used free order of Hindi sentences by fixing the order of word group and POS (part of speech) tagging to create a parser of the Hindi language. While identifying the morphologically rich and relatively free word? order language (MoR-FWO) feature of Hindi like Turkish, Basque, Czech, Arabic, etc., Jain et al. (2012) proposed a two-stage approach for Hindi dependency parsing using a parser named MaltParser. Other attempts were made to develop the parser of varieties like clausal-parsing by Husain et al. (2011) and micro-parsing of Hindi by Joshi et al. (2016). Further, CFILT, IIT Bombay corpus[2] has inspired many new researchers (including us) to work in the field of formal treatment of the Hindi language.

However, the current research in the field of Hindi NLP is mainly based on a computational approach (using machine learning and deep learning methods) that focuses on developing the efficiency of computational algorithms. However, it does not seem concerned with a formal grammatical (i.e., syntactic and semantic) description of the Hindi language. It is one of the lacking verticals in Hindi linguistics that we are willing to fill and experiment with knowledge-representation of Hindi.

Syntactic Parsing

Parsing is the process of resolving the input sentence into its constituents using CFG. CFGs provide a formal framework to capture hierarchical sentence structures and generate parse trees. Researchers have leveraged CFGs in creating parsers for Hindi and other languages (Manning & Schütze, 1999). Parsing of the Hindi language has been a focus of research due to the language's

[2] Kunchukuttan, A., Mehta, P., & Bhattacharyya, P. (2017). The iit bombay english-hindi parallel corpus. arXiv preprint arXiv:1710.02855.

complexity and growing importance in NLP. Studies have explored various parsing techniques, including dependency parsing and constituency parsing, to dissect sentence structures and enable accurate language understanding (Kumar & Rose, 2012). A CFG is a function of 4 tuples[3]:

$$G = f (V, T, P, S)$$

where, G stands for the grammar of any formal language, T for a finite set of terminal symbols, V for a finite set of non-terminal symbols or variables (meta-symbols), P for a finite set of production rules, and S is the start symbol for Sentence. This methodological approach helps efficient part of speech (POS) tagging to assign the syntactic category (noun, verb, and particle) to each word in the input sentence.

Our parser checks whether a given Hindi sentence is syntactically well-formed or not. It further checks the gender rules to identify correct auxiliary verbs corresponding to the gender of the subject (i.e., Noun Phrase) of the sentence. This parser, which uses recursive rules (check Table 1), can deal with a sentence of any length, and detects syntactic and other errors. The parse tree for different corpus-based sentences is discussed in detail. Finally, we present a variety of experimental cases consisting of verb types, subject types, etc. As mentioned earlier, Hindi has a relatively free word order; the parser is accounting for? wffs only in rule-based word order in the current research design to accommodate argument structure in semantic analysis; however, parsing can be easily extended by involving such syntactical rules. The presence of modifying (as per direct and oblique cases) personal pronouns in combination with post-positions increases the difficulty not only for developing a comprehensive and wide-ranging parsing system but also for building efficient CFG rules. For example, see the translation of 'I' adopted in [s1] and [s2]:

[s1] Maĩ ek laṛakā hū̃.
I a boy be.prs.1p
I am a boy.

[s2] Mere se jyādā lambā koī nahī̃ hai.
I from more tall anybody neg be.prs.3p
No one is taller than I.

In the context of this research paper, the scope of investigation does not encompass the myriad issues encompassed within the broader landscape of linguistic analysis. Rather, the study is primarily oriented towards the development of a parser tailored to a simple and potentially infinite wffs-generating fragment of the Hindi language. Notably, Sagar et al. (2010) have previously elucidated the principles of context-free grammar as a formal system for the representation of natural languages (for Kannada language). This framework delineates the structure of a language by specifying the derivation of valid textual constructs (legal texts) from a distinguished symbol, denominated as the "sentence symbol." It is worth noting that this formalism has garnered

[3] See Hopcroft, John E.; Ullman, Jeffrey D. (1979), Introduction to Automata Theory, Languages, and Computation, Addison-Wesley.

widespread acceptance among researchers specializing in the domain of common language families, as exemplified by the work of Alqrainy et al. in 2012.

Alqrainy et al. (2012) employed the NLTK (natural language toolkit) recursive descent parser to assess the grammatical correctness of Arabic sentences, employing a top-down parsing technique. A series of empirical experiments were conducted using a corpus comprising 150 Arabic sentences, which had been previously annotated and tagged using a proprietary AMT tagger into noun (N), verb (V), and particle (P) categories before parsing.

These experimental designs represent instances of deliberate integration into the formal structure of natural language. We have assimilated the design insights derived from such experiments and are now presenting context-free grammar (CFG) rules, as detailed in Table 1, which can generate theoretically infinite well-formed formulas for a fragment of Hindi with finite lexical units.

R_1	$S \rightarrow NP\ VP$
R_2	$NP \rightarrow \begin{cases} N_m \\ N_f \end{cases}$
R_3	$VP \rightarrow \begin{cases} (NP_o)\ Verb_t \\ Verb_i \end{cases}$
R_4	$Verb_t \rightarrow V_t\ Aux$
R_5	$Verb_i \rightarrow V_i\ Aux$
R_6	$Aux \rightarrow \begin{cases} \text{tā - hai } \textit{in the context } N_m \\ \text{tī - hai } \textit{in the context } N_f \end{cases}$
R_7	$NP_o \rightarrow$ NP Object
R_8	$S \rightarrow S$ conj S
R_9	$S \rightarrow$ neg S

Table 1. CFG rules for formalization of the Hindi language

R1, i.e., the first rule is to be read as 'S goes to NP VP.' S, NP and VP stand for 'Sentence', 'Noun Phrase' and 'Verb Phrase' respectively, while Aux stands for 'Auxiliary'. In the above abbreviation symbols, N_m, N_f, NP_o, $Verb_t$, $Verb_i$, Object, conj and neg stand for masculine nouns, feminine nouns, object-based case-marked noun phrases, non-terminal nodes for transitive verbs, non-terminal node for intransitive verb, case-marker, conjunction/disjunction and negation respectively. Intuitively speaking, V_i and V_t are one-place and two-place predicates respectively. The brackets are to be interpreted following Dowty et al. (1982).

Moreover, it is essential to note that these context-free grammar (CFG) rules are proficient in effectively managing recursive nesting, auxiliary mapping, as well as the specification of verb phrase transitivity or intransitivity. In the context of affirmative sentences within the Hindi language, akin to their counterparts in numerous other natural languages, the initial elementary top branches encompass noun phrases (NP) and verb phrases (VP) from which the entire syntactic parse tree emanates.

These recursive syntactic rules possess the inherent capacity to accommodate sentences of varying lengths in a theoretically unlimited manner. However, for experimentation, we deem it necessary to impose a constraint concerning the number of dyadic connectives (referred to as "conj") that can be employed within a sentence in our object language. It is stipulated that, within our experimental framework, a sentence may feature at most two dyadic connectives. Let us consider the following example:

p **and** q **or** y

This string is ambiguous. It can either be read as:

(p and q) or y

or as:

p and **(q or y)**

Our parser will give both readings. But it will not parse a sentence that has more than two dyadic connectives. The following section describes the parse tree and the parsing system in more detail for each experimental case.

Parse Tree Generation

The fundamental objective underlying the generation of a parse tree is to demonstrate the presence of a hierarchical structure inherent in all languages and sentences. To put it differently, the parse tree corresponding to a given sentence 'S' serves as a visual representation of the constituent elements comprising 'S,' organized in a hierarchical arrangement. In the context of this research, parse trees are constructed utilizing the parser, developed using the Python programming language.

It is noteworthy that all the conventional parsed examples presented herein incorporate standard diacritical marks while showing the rules' application. However, the software facilitating parse tree generation permits user input both with and without diacritical marks. As a result, the output in each example case remains consistent with the chosen input method.

All the syntactic analysis cum interpretation provided below in the form of parse-trees uses Table 2 as the glossary reference of the syntactic category of Hindi sentences.

N_m	Masculine Noun	rām, śyām
N_f	Feminine Noun	sītā
V_t	Transitive Verb	jān, dekh
V_i	Intransitive Verb	cal, so
Aux	Auxiliary	tā-hai, tī-hai
Object	Object	ko
conj	Conjunction	yā, aur
neg	Negation	aisā nahī̃ hai ki

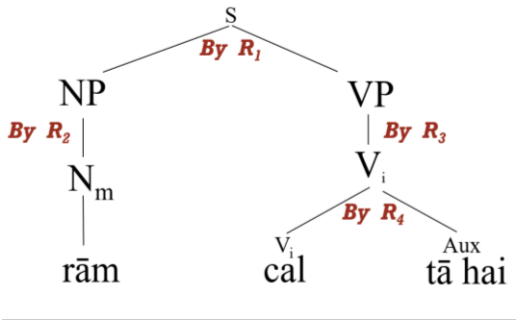
Table 2. Tagging of Lexical Item Glossary

Example 1 presents a sentence with an intransitive verb while Example 2 has a sentence with a transitive verb. Example 3 focuses on recursion involving logical connectives (e.g., conj) while Example 4 shows the use of separate phrase-transformation rules to develop negative sentences. Example 5 uses gender rules for Hindi NPs and identifies a proposition as non-wff using CFG rules.

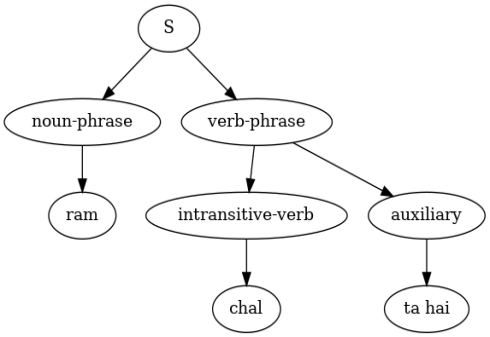
Example 1- Case of Intransitive Verb

[s3]	rām	cal	tā hai.
	rām	move	Aux_{mas} [4]
	rām	walk	prs.3p

rām walks.



1.1 Rule Basis of Parse Tree



1.2 Software generated Parse Tree

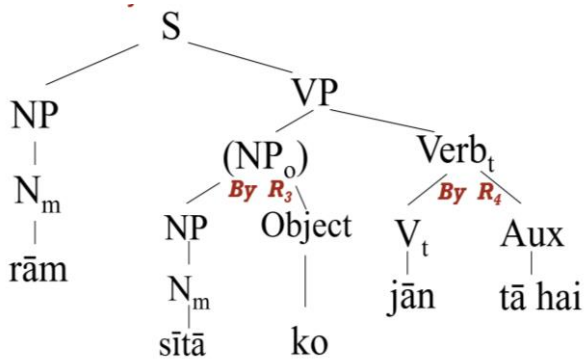
The terminal node 'S' designates 'rām' as a noun phrase (NP) labeled with a null case marker and an intransitive verb as its predicate. Within the syntactic distribution of this proposition, the sentence is recognized as a well-formed formula (wff) for the given linguistic fragment.

Example 2- Case of Transitive Verb

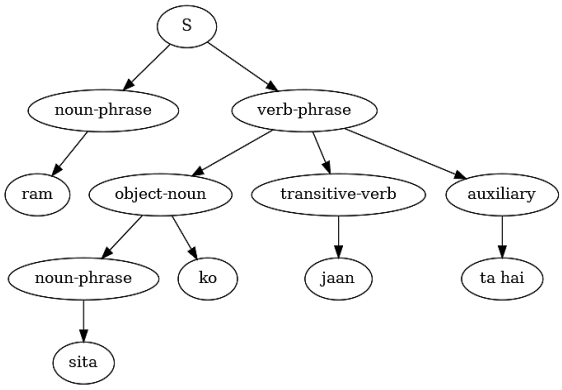
[s4]	rām	sītā ko	jān	tā hai.
	rām	sītā + Object	know	Aux_{mas} [5]
	rām	sītā -acc	know	prs.3p

rām knows sītā.

[4] Aux_{mas} stands for 'Aux in the context of N_m' and Aux_{fem} stands for 'Aux in the context of N_f'.
[5] Aux_{mas} stands for 'Aux in the context of N_m' and Aux_{fem} stands for 'Aux in the context of N_f'.



2.1 Rule Basis of Parse Tree



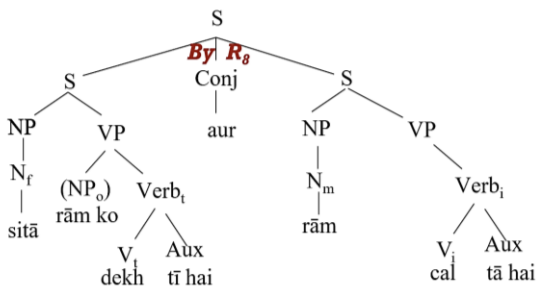
2.2 Software generated Parse Tree

The terminal node 'S' designates 'rām' as a noun phrase (NP) labelled with a null case marker, while 'sītā ko' as case marked noun phrase (NPo) and transitive verb as its predicate. Within the syntactic distribution of this proposition, NPo is the daughter branch of VP and thus, the sentence is recognized as a well-formed formula (wff) for the given linguistic fragment.

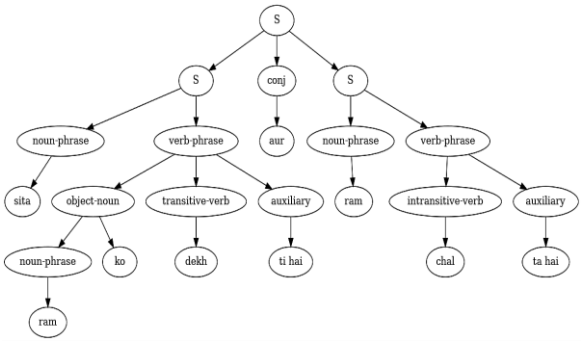
Example 3- Case of Conjunction

[s5]	sītā	rām ko	dekh	tī hai	aur	rām	cal	tā hai.
	sītā	rām + Object	see	Aux_{fem} [6]	conj	rām	walk	Aux_{mas}
	sītā	rām -acc	see	fem prs.3p	and	rām	walk	prs.3p

sītā sees rām and rām walks.



3.1 Rule Basis of Parse Tree



3.2 Software generated Parse Tree

The terminal node 'S' bifurcates the provided sentence into two distinct propositions, each delineated by two sub-nodes labeled as 'S.' The left sub-node 'S' encompasses 'rām' functioning as a noun phrase (NP) coupled with a transitive verb serving as its predicate, while the right sub-

[6] Aux_{mas} stands for 'Aux in the context of N_m' and Aux_{fem} stands for 'Aux in the context of N_f'.

node consists of 'rām' as a noun phrase (NP) with an intransitive verb as its predicate. In the context of this syntactic structure, the sentence is identified as a well-formed formula (wff) within the given linguistic fragment. This recognition is grounded in the presence of the conjunction 'aur,' which connects the two sub-nodes, thereby forming a coherent linguistic unit.

Example 4- Case of Negation

[s6]	rām	sītā ko	nahī̃	jān	tā hai.
	rām	sītā + Object	not	know	Aux_{mas}
	rām	sītā -acc	neg	know.m	prs.3p
	rām does not know sītā.				

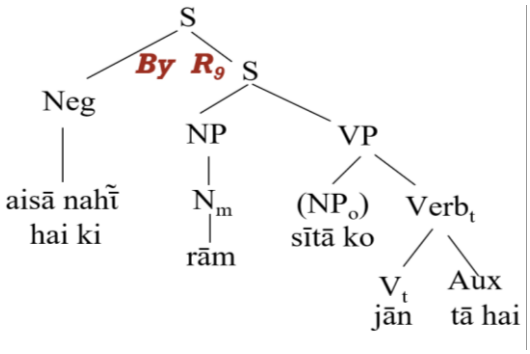
In case of negation, we propose the application of CFG rules, after applying the structure transformation rule that changes the sentence arrangement pattern in the following method:

X (nahī̃) Y —> (aisā nahī̃ hai ki) X Y

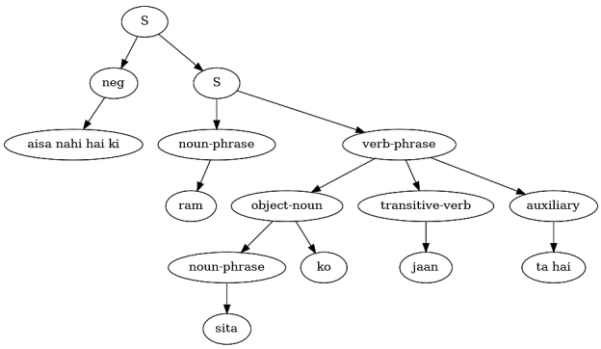
or as,

X (nahī̃) Y —> neg X Y

Parser identifies any instance of nahī̃ (ie. not) negation element found anywhere between the subject X and predicate Y and modifies the input string using above transformation rule and thus, then able to create the parse tree as per Rule R9.



4.1 Rule Basis of Parse Tree



4.2 Software generated Parse Tree

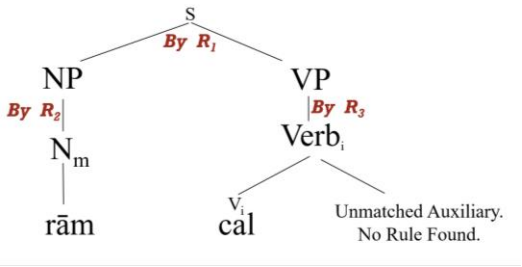
The terminal node 'S' partitions the given sentence into two discrete propositions, with one being denoted by a sub-node labeled'neg,' and the other sub-node designated as 'S.' The sub-node on the left includes the phrase 'aisā nahī̃ hai ki,' which serves the semantic function of negating the value of 'S,' while, from a syntactic standpoint, it employs the transformation rule mentioned earlier for parsing. In the context of this syntactic structure based on negation, the sentence is recognized as a well-formed formula (wff) within the specified linguistic framework.

Example 4- Case of Gender Mismatch

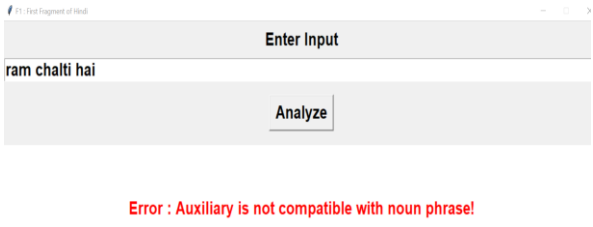
[s7] rām cal tī hai.

Rām walk Aux_{fem}

Invalid sentence.



5.1 Rule Basis of Parse Tree



5.2 Software generated Parse Tree

The parser system can successfully detect the lexical tokens, but as a consequence of an erroneous combination of these tokens, the syntactic structure is not correctly constructed, leading to the generation of the message "auxiliary is not compatible with the noun phrase." Consequently, this indicates the system's capability to identify grammatically incorrect sentences within the given linguistic fragment.

Example 6- Case of Dyadic Connectives

Conjunction serves the purpose of merging two adjectives into a single modifier, as well as combining two NPs, two VPs, two PPs, and more. However, in the present context, the L_{0H} fragment restricts conjunction to the combination of two sentences only.

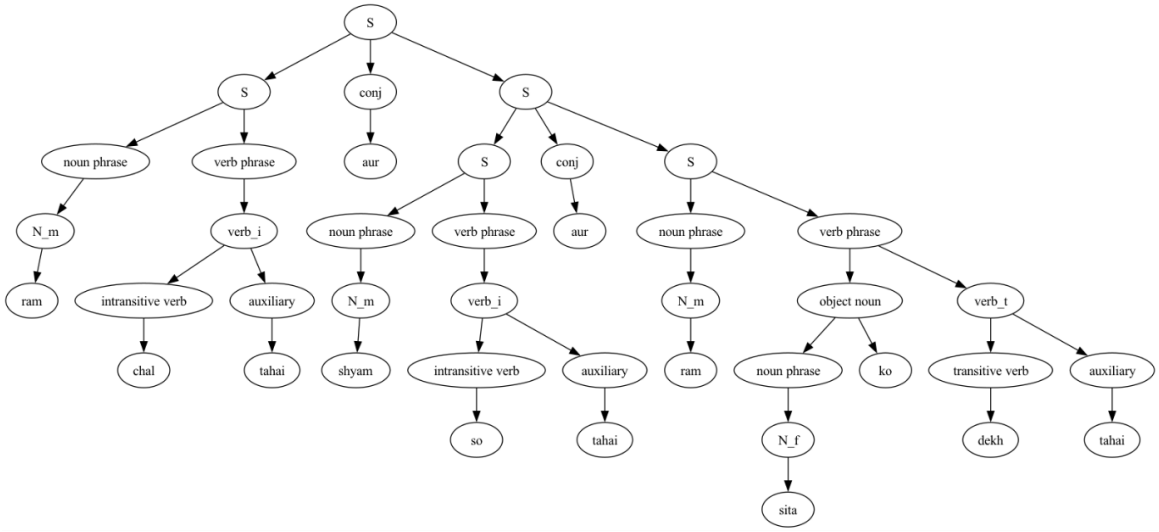
[s8] rām cal tā hai aur śyām so tā hai aur rām sitā ko dekh tā hai.

rām walk Aux_{mas} conj śyām sleep Aux_{mas} conj rām sitā Object see Aux_{mas}

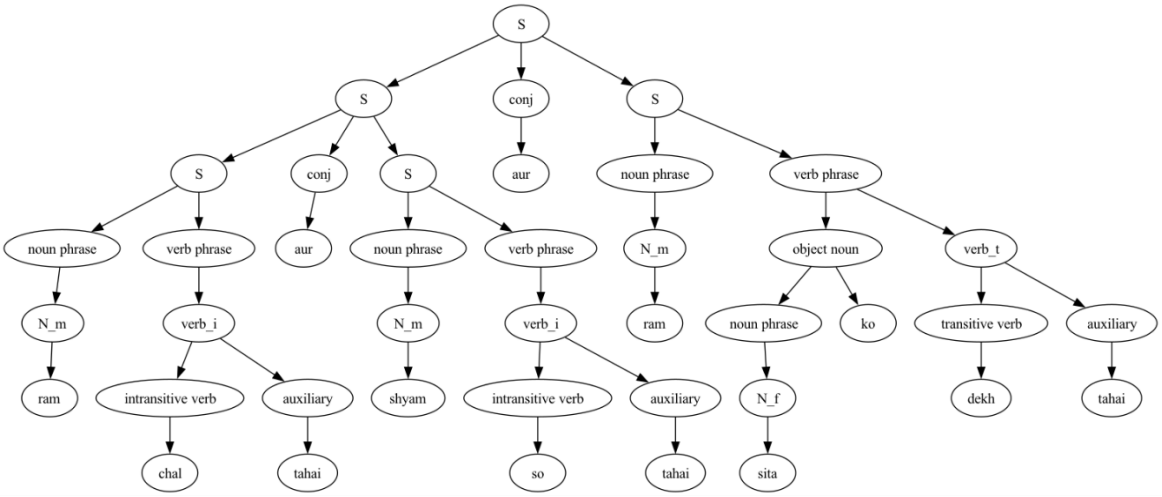
rām walk prs.3p and śyām sleep prs.3p and rām sitā -acc see prs.3p

rām walks and śyām sleeps and rām sees sitā.

The parser system, as outlined in parsing procedure section, regards the aforementioned sentence as a non-ambiguous, well-formed formula through the application of bracketing, also referred to as bounding. Consequently, the result yields two abstract syntax trees, as presented below.



6.1.1 Software generated Parse Tree (1st Parse Tree)



6.1.2 Software generated Parse Tree (2nd Parse Tree)

The primary objective accomplished through this syntactic analysis is to ascertain the grammatical feasibility of generating an input sentence in various forms using a provided grammar, either through parsing software or a computer system. A prospective augmentation could involve the formulation of a viable combination of sentences, generated through the automated application of self-correction rules. However, it is essential to emphasize that the current scope of our research paper is confined to the development of a parser with a specific focus on parsing objectives to develop proposition within the scope of current fragment L_{0H}, and does not encompass the aforementioned extension.

Parsing Procedure

The aforementioned syntactic analysis is acquired through tokenization or a process commonly referred to as part-of-speech (POS) tagging. The POS tagging system holds pivotal significance

as an initial step for any parsing system, as it aids in categorizing the constituents of a sentence. The primary objective of the POS tagging system is to assign specific lexical categories, such as NP for noun phrases, VP for verb phrases, and Aux for auxiliaries, within the parsed sentences. An illustrative representation of the tagging procedure can be found in Figure 7, which has been developed as part of the lexical analysis.

The Lexical analyzer transforms the provided input string into a sequence of tokens, with each token representing an indivisible linguistic unit or word. The tool produces a token when the character sequence aligns with user-defined token patterns. Each token is composed of a sequence of characters and is assigned a specific type. Our token types encompass masculine nouns, feminine nouns, transitive verbs, intransitive verbs, auxiliaries, negations, conjunctions, and objects. Subsequently, these tokens are forwarded to the parser to generate a parse tree. It is worth noting that this tool is an integral component included within the PLY Python package.

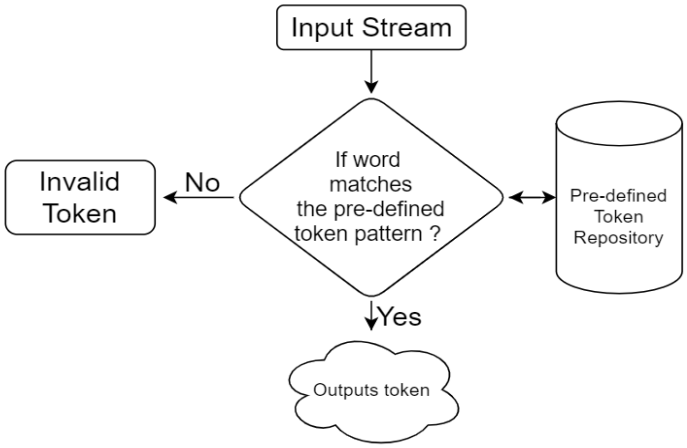


Figure 7. Lexical Analyzer and procedure of POS Tagging

The fundamental objective of any CFG is to generate all potential well-formed formulas within a given formal language. In other words, if we consider a formal language L, the rules of a CFG for Language (CFGL) should ideally have the capacity to generate all the grammatically correct sentences found within L. Nonetheless, the capabilities of a parser are inherently limited due to practical constraints within experimental designs, its functionality can be enhanced through the incorporation of additional rules and linguistic elements.

As a result, it is possible to transform a simple language, initially comprised of a few nouns, predicates, and logical connectives, into a more intricate language of first-order predicate calculus. This evolution entails the introduction of rules governing predicate logic, variables, and quantifiers, and potentially the inclusion of tense and modal operators at a later stage. It is important to note that the foundational parser developed and presented herein does not support quantification. Additionally, it is equipped with only the simple present tense and lacks modal operators. However, it possesses the adaptability to accommodate tense operators by expanding its repertoire of auxiliary elements.

When the object language undergoes changes in modality or tense, the inflection of auxiliaries within the parser will also evolve accordingly. Nonetheless, they will consistently retain their essential property as auxiliaries, following the CFG rules outlined earlier. Consequently, any rules established for the treatment of auxiliaries within the CFG framework will remain equally applicable to any potential new inflection of auxiliaries, should such modifications occur due to changes in tense.

The primary objective of a parser is to apply established rules and create a hierarchical parse tree that corresponds to a given sentence. There exist several strategies for constructing parse trees, with one of the most prevalent approaches being the top-down strategy. In this method, the input sentence is initially tagged as 'S,' and subsequently, the rules specified by the Context-Free Grammar (CFG) are systematically applied. Our parser for fragment L_{0H}, follows a sequence of steps to construct a parse tree, consistently adhering to recursive rules until the parse tree's terminal nodes encompass the entire length of the input sentence. Such a parser system can be categorized within the framework of LR or shift-reduce parsers. It employs the following procedure to generate syntactic categories cum lexical entries as per given CFG rules:

- LR parsing is a bottom-up technique that recognizes the right-hand side of various grammar rules. L denotes left-to-right scanning of the input, and R denotes building a rightmost derivation in reverse.
- Whenever a valid right-hand side of a grammar rule is found in the input, the appropriate action code is triggered, and the grammar symbol on the left-hand side replaces the grammar symbols.
- It is generally implemented by shifting grammar symbols onto a stack and peeking at the stack and the subsequent input token for patterns that match one of the grammar rules.

The presently developed parser operates as a look-ahead left-to-right (LALR) parser, with the primary objective of evaluating the grammatical correctness of a Hindi sentence based on multiple criteria, including the alignment of auxiliaries with gender and considerations of sentence plurality or singularity.

Conclusion

After conducting a series of experiments outlined in this paper, we have reached several significant conclusions regarding our proposed formalization of Hindi with a basic fragment called L_{0H}. This allows for syntactic analysis using context-free grammar and semantic analysis using a model-theoretic system (part 2 of this research). In summary, the essential findings and insights from our syntactic experiments are as follows:

- Distributional Criteria and Word Order: Our syntactic parser successfully elucidates the distributional criteria of the lexical items from the Hindi language, particularly concerning word order. The parsing rules we developed, following a 'top-down approach,' effectively handle the self-arrangement of sentences. This capability is crucial for understanding the structural aspects of Hindi sentences.

- Propositional Logic and Gender Conventions: The rules of propositional logic and gender conventions in the Hindi language were effectively incorporated into our model. This ensures that the system can analyze and generate sentences that adhere to these grammatical and syntactic principles.
- Corpus-Based Experiments: A set of experiments was conducted using the corpus we created for this research. These experiments provided empirical validation of the effectiveness of our model and its ability to parse and generate sentences in Hindi accurately. However, an extensive corpus containing a featured data set can be the future agenda of our current research.
- Conjunction, Negation, and Conditionals: Our model effectively handles complex linguistic constructs such as conjunction, negation, and conditionals. Incorporating idempotent laws ensures that classical derivations of entailments are approximated accurately.

The parser designed for fragment L_{0H} employs a look-ahead left-to-right (LALR) approach to assess the grammatical correctness of sentences, considering parameters such as auxiliary-gender mapping and the distinction between plurality and singularity within the sentence. In the subsequent phase of development, the parser aims to expand its capabilities by incorporating semantic analysis based on Montague grammar.

Our upcoming work, which constitutes Part 2 of this article, will provide comprehensive insights into the semantic rules and the methodology for evaluating truth conditions and truth values of lexical tokens. This evaluation process is grounded in the principle of compositionality.

Acknowledgement

On behalf of all contributors to this paper, I (Vivek Tripathi) would like to extend my sincerest gratitude to Dr. Nirmalya Guha for his invaluable feedback and guidance throughout developing this paper. The best aspects of this paper owe much to his direction, while any remaining mistakes are solely my own. I further thank Dr. Sanjukta Ghosh and Dr. V. Ramanathan for their constructive criticism and insightful suggestions that have significantly improved work quality. (All named persons are faculty at the Indian Institute of Technology, BHU, Varanasi).

Declaration of Conflicts of Interests

The author(s) declared no potential conflicts of interest.

References

Alqrainy, S., Muaidi, H., & Alkoffash, M. S. (2012). Context-free grammar analysis for Arabic sentences. *International Journal of Computer Applications*, 53(3).

Ambati, B. R., Husain, S., Jain, S., Sharma, D. M., & Sangal, R. (2010, June). Two methods to incorporate' local morphosyntactic features in hindi dependency parsing. In *Proceedings of the NAACL HLT 2010 First Workshop on Statistical Parsing of Morphologically-Rich Languages* (pp. 22-30).

Sagar, B. M., Shobha, G., & Kumar, R. (2010, August). Context Free Grammar (CFG) analysis for simple Kannada sentences. In *Proceedings of the International Conference [ACCTA-2010] on Special Issue of IJCCT (Vol. 1, No. 2, 3, 4)*.

Chomsky, Noam. "Syntactic structures." In *Syntactic Structures. (2009)*. De Gruyter Mouton.

Dowty, D. (1982). Grammatical relations and Montague grammar. In *The nature of syntactic representation* (pp. 79-130). Dordrecht: Springer Netherlands.

Dowty, D. R., Wall, R. E., Peters, S., Dowty, D. R., Wall, R. E., & Peters, S. (1981). The Grammar of PTQ. *Introduction to Montague Semantics*, 179-251.

Dowty, D. R., Wall, R., & Peters, S. (2012). Introduction to Montague semantics (Vol. 11). Springer Science & Business Media.

Husain, S., Gadde, P., Nivre, J., & Sangal, R. (2011, November). Clausal parsing helps data-driven dependency parsing: Experiments with Hindi. In *Proceedings of 5th International Joint Conference on Natural Language Processing* (pp. 1279-1287).

Jain, N., Singla, K., Tammewar, A., & Jain, S. (2012, December). Two-stage approach for hindi dependency parsing using maltparser. In *Proceedings of the Workshop on Machine Translation and Parsing in Indian Languages* (pp. 163-170).

Joshi, B. K., & Kushwah, K. K. (2016). Micro-parsing of Hindi words. *International Journal of Computer Science and Information Security*, 14(11), 261.

Kumar, N., & Rose, C. (2012). "Hindi Dependency Treebank." In "*Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC 2012)*."

Manning, C. D., & Schütze, H. (1999). "Foundations of Statistical Natural Language Processing." MIT Press.

Montague, R. (1975). English as a formal language. In J. Moravcsik (Ed.), *Logic and philosophy for linguists: A book of readings* (pp. 94-121). Berlin, Boston: De Gruyter Mouton. https://doi.org/10.1515/9783111546216-007

Partee, B. (1984). Compositionality. *Varieties of formal semantics*, 3, 281-311.

Partee, B. (1973). Some transformational extensions of Montague grammar. *Journal of Philosophical Logic*, 2, 509-534.

Ranjan, P., & Basu, H. V. S. S. A. (2003). Part of speech tagging and local word grouping techniques for natural language parsing in Hindi. In *Proceedings of the 1st International Conference on Natural Language Processing (ICON 2003)*.

Vivek Tripathi is a linguistics scholar affiliated with the Indian Institute of Technology (BHU) Varanasi. The primary focus of his research is to investigate the syntax and semantics of South Asian Languages. He is currently dedicated to understanding Hindi as a contemporary language and Sanskrit as a classical language from linguistic perspectives and frameworks. The areas of his research are syntax, semantics, modality and negative polarity items (NPI).

Dinesh Rathod is a researcher at the Computer Science and Automation Department of the Indian Institute of Science, Bangalore. He is dedicated to researching computer graphics, algorithmic problem-solving and software development. Dinesh has published papers in computer science and related domains such as geomechanics.