Linguistic Inputs in building a dependency parser for Marathi: A Paninian Approach

A Thesis Submitted to the University of Hyderabad in Partial Fulfilment of the Requirements for the Award of the Degree of Doctor of Philosophy in Applied Linguistics

By

UMALE YOGESH VIJAY

(Reg. No. 12HAPH06)

Supervisor

PROF. K. RAJYARAMA

Centre for Applied Linguistics& Translation Studies

School of Humanities

University of Hyderabad

Hyderabad – 500046, India

December, 2022

Centre for Applied Linguistics and Translation Studies

School of Humanities, University of Hyderabad

Hyderabad – 500046, India

DECLARATION

I, UmaleYogesh Vijay, hereby declare that I have carried out the research embodied in the present thesis entitled **"Linguistic Inputs in building a dependency parser for Marathi: A Paninian Approach"** for the full period prescribed under the Ph.D. ordinances of University of Hyderabad. I also declare to the best of my knowledge that no part of this thesis was earlier submitted for the award of any research degree to any other university or institution.

I, agree that my thesis can be deposited in SHODHAGANGA INFLIBNET. A report on plagiarism statistics from the University Librarian is enclosed.

(Prof. K. Rajyarama)
Supervisor

(Umale Yogesh Vijay)
(Reg. No. 12HAPH06)

Head, CALTS

Dean, School of Humanities

Centre for Centre for Applied Linguistics and Translation Studies

School of Humanities, University of Hyderabad

Hyderabad – 500046, India

CERTIFICATE

This is to certify that the thesis entitled **"Linguistic Inputs in building a dependency parser for Marathi: A Paninian Approach"** submitted by **Umale Yogesh Vijay**, with Regd. No. **12HAPH06** in partial fulfilment of the requirements for the award of the degree of Doctor of Philosophy in **Applied Linguistics and Translation Studies**, School of Humanities is a bonafide work carried out by her under my supervision and guidance.

This thesis is free from plagiarism and has not been submitted previously in part or full to this or any other university or institution for the award of any degree or diploma. Parts of the thesis have been published in the following publications:

1. **"Dependency framework for Marathi Parser"** Language in India online Journal, ISSN 1930-2940, Volume 16:1 January 2016.

Further, the student has passed the following courses towards the fulfilment of course work requirements for the Ph. D. degree.

S. No	Course Code	Name	Credits	Pass/Fail
1.	ELS-801	Research Methodology	4 Credits	Pass
2.	ELS-803	Advance Topics in Computational Linguistics	4 Credits	Pass
3.	ELS- 821	Readings in Applied Linguistics (Research Oriented Readings)	4 Credits	Pass

Supervisor Head, CALTS Dean, of School of Humanities

ACKNOWLEDGEMENTS

I wish to express my sense of gratitude to all of my teachers who have enlightened me in drafting the thesis. Firstly, I would like to express my sincere gratitude to my supervisor Prof. **K. Rajyarama** for the support of my Ph.D. work, for her patience, motivation, and immense knowledge. Her guidance helped me for the writing of this thesis, and it was a great pleasure and privilege to work under her guidance.

Umale Yogesh Vijay
12HAPH06
31 Dec. 2022

ABBREVIATIONS

-∅ - zero marker
1- First person
2- Second person
3- Third person
abl- Ablative
acc- Accusative
AnnCorr- Annotated Corpora
ATB-Arabic Treebank
CL- Corpus/Computational linguistics
conj-Conjunction
CT- Constituency Treebank
CTB-Chinese Treebank
dat- Dative
DT-Dependency Treebank
erg- Ergative
f- Feminine
gen- Genitive/Possessive
inf- Infinitive
int – Instrument
m- Masculine
N/n- Neutral
NLP- Natural Language Processing
nom- Nominative
PDT-Prague Dependency Tree-bank
pl – Plural
POS - Parts of speech tagging
pp- Postpositions
SAC-Syntactically Annotated Corpus
SB-Syntactic bracketing
sg –Singular
SSF-Shakti Standard Format
TIL-Treebank for Indian Languages
TUT-Turin University Treebank
UD-Universal Dependencies

List of Figures and Tables

Marathi WX Notation

Vowels

अ	आ	इ	ई	उ	ऊ	ए	ऐ	ओ	औ
a	A	i	I	u	U	e	E	o	O

Sonorants

ऋ	ॠ	ऌ
q	Q	L

Anusvāra and visarga

ं	ः
M	H

Consonants

क्	ख्	ग्	घ्	ङ्	**Velar**
k	K	g	G	f	
च्	छ्	ज्	झ्	ञ्	**Palatal**
c	C	j	J	F	
ट्	ठ्	ड्	ढ्	ण्	**Retroflex**
t	T	d	D	N	
त्	थ्	द्	ध्	न्	**Dental**
w	W	x	X	n	
प्	फ्	ब्	भ्	म्	**Labial**
p	P	b	B	m	
य्	र्	ल्	व्		**Semi-vowel**
y	r	l	v		
श्	ष्	स्	ह्		**Fricative**
S	R	s	h		

Marathi Roman Transliteration Chart

अ	आ	इ	ई	उ	ऊ	ऋ	ऋ	ए	ऐ
a	**ā**	**i**	**ī**	**u**	**ū**	**ṛ**	¤	**e**	**ē**
ओ	औं	ः							
o	**ō**	**ṃ**	**ḥ**						

क्	ख्	ग्	घ्	ङ्
k	**kh**	**g**	**gh**	**ṅ**
च्	छ्	ज्	झ्	ञ्
c	**ch**	**j**	**jh**	**ñ**
ट्	ठ्	ड्	ढ्	ण्
ṭ	**ṭh**	**ḍ**	**ḍh**	**ṇ**
त्	थ्	द्	ध्	न्
t	**th**	**d**	**dh**	**n**
प्	फ्	ब्	भ्	म्
p	**ph**	**b**	**bh**	**m**
य्	र्	ळ	ल्	व्
y	**r**	**ḷ**	**l**	**v**
श्	ष	स	ह	
Ś	**ṣ**	**s**	**h**	

CONTENTS

CHAPTER - I

1. Introduction

One of the branches of artificial intelligence is natural language processing (NLP). It deals with an automated synthesis and understanding of any given natural (Human) language justifiable to the availability of the corpus on different semantic domains of the language. Not all languages (7000) of the world have a corpus except a few, like English, Major European, and major South Asian languages. In the present scenario, the field is designated and considered the border discipline of Applied Linguistics. The domain is also called 'Natural Language Processing (NLP). From the applied linguistics perspective, language technology and computational linguistics have two significant goals viz. study of the theoretical aspects of language and the application of theoretical aspects to develop computational tools for natural languages that are in demand and need.

To build computational tools for natural (human) languages, one has to take the help of it and its mechanisms. For a theoretical grounding it makes use of disciplines like linguistics, philosophy, psychology, and other subjects to understand how humans use language for communication.

From the implementation or execution point of view, Natural Language Processing (NLP) divides into two broad areas: speech processing and text processing. While speech processing involves the analysis of speech patterns of natural language, text processing deals with text patterns of natural language. Given the interdisciplinary nature of the subject, either while doing speech processing, or text processing, the theoretical knowledge of linguistics and computational linguistics have to be combined. Apart from creating the tools, it can be applied to build language interfaces to the database, and computers, also for all language-related applications like question-answering systems, story comprehension and understanding, and machine translation.

In NLP research, especially in the areas of Language Technology and Computational Linguistics, linguistic resources play a significant role. The resources may come in different forms, viz. raw corpus, annotated corpus tagged for Morphological Analysis, POS categories, Chunking, Parsing, etc. Various types of these resources provide insight into the linguistic features of a particular language. These linguistic features may be in the form of phonological patterns, word patterns, syntactic patterns, or lexical-semantic features of the specific language. Whatever the nature and whichever level they may be at, NLP heavily depends on linguistic resources such as raw corpus annotated corpus to develop various kinds of computational tools such as speech recognizers, grammar checkers, morphological analyzers and synthesizers, automatic POS taggers, syntactic parsers, and discourse analyzers. The present study focuses on providing linguistic inputs in building a syntactic dependency parser for Marathi based on the Paninian grammar. A syntactic parser has a significant

role to play not only in analyzing the syntactic structure of a language but also in resolving ambiguity, an inherent feature of human languages.

1.2 Syntactic parser

Syntactic Parser is an important NLP tool that helps in providing a complete and exhaustive analysis of the language structure at the sentential level. The term 'Parsing'at the syntactic level involves "the process of analyzing a string of symbols either in natural language, computer languages, or data structures conforming to the rules of a specific grammatical formalism". The term '*parsing'* comes from the Latin meaning part of speech.

NLP involves analyzing natural languages at various levels, beginning from the word level, followed by recognition of parts of speech, and phrases (chunks), and a syntactic parser comes to the next level. Hence it appears in the third phase of the analysis. The major goal of this stage is to match the sentence meaning with syntactic structure by considering the rules of the language. The primary aim of the third phase is to analyze the syntactic features and generate the syntactic trees of a particular language using a specific grammar framework. The tree captures information at various levels: that of grammatical categories viz. nouns, verbs, and adverbs, followed by phrasal categories like NP, and VP, as well as grammatical relations like subject, object, and indirect object. Given these categories, it should be possible to generate syntactic trees.

1.3 Approaches to building a syntactic parser

There can be many approaches to building a syntactic parser. Significant among them are: Rule-based or machine learning based. Based on these approaches we have either Rule-based parsers, Statistical-Parsers, Generalized Parsers, or Hybrid parsers, MALT, and MST.

Developing a syntactic parser is a very challenging task. The main reason is the complexity of natural languages, and inherent ambiguity, and added to these there is also the fact that adequate linguistic resources are not available. Despite these challenges, efforts are being made to develop parsers in various Indian languages, Hindi, Urdu, and Telugu are a few languages to cite. In the present work, an attempt is made to build a Rule-Based Dependency parser using a hybrid approach viz. the dependency grammar.

1.3.1 Rule-based syntactic parser

A Rule-Based parser employs a set of hand-crafted rules based on the grammar of the language that enables the translation and encoding of natural language into machine language. Such systems are rather flexible and allow updating by extending the existing rules whenever new functions and data types arise. However, it requires experts in language and linguists to formulate the rules.

In computational linguistics, a rule-based parser for a given specific language, rules are formulated to recognize the best parse tree for a given language or given grammar. But, in this approach, there is a possibility of overlapping the rules in the course of production, and it is a recursive process. This problem is solved by using Dynamic programming (DP), also called the DP technique. This program eliminates the process of overlapping in the course of production. The cache or reserve in the DP technique is called a 'chart' and subsequently, the DP-based parsers are designated as 'chart parsers'. The Cocke Younger Kasami (CYK) algorithm is an early version of rule-based parsers. If a sentence is given to the rule-based parser, it provides the following order of the sentence:

S → NP VP

NP →N

VP → V NP

N → John

V → plays

NP →N

N → cricket

In the above rules, 'S' stands for sentence, and the arrow symbol '→' is used to represent the rewritten rule. According to phrase structure grammar, the syntactic parser first takes a sentence as input. For example, "John plays cricket". In the second step, the sentence is divided into two base boundaries such as NP and VP, In the third step, the parser recognizes parts of speech like N, V, N. In the fourth step, it generates the lexical items such as cricket, plays/plays, and John. In the final step, it generates the complete syntactic tree of a sentence.

1.3.2 Statistical-based syntactic parser

After the rule-based parser, the statistical-based parser is the more popular syntactic parser. In this method, algorithms play a significant role. Using this concept, it searches all candidate parses. For this, it needs a large number of annotated corporaare to analyze and synthesize the sentences given from a particular language. It is also one of the crucial linguistics resources, which is the backbone of a syntactic parser. For example, 'Old men and women play cricket. In this sentence, one probability is 'old men and women can be treated as one NP. In addition, in another way 'old men' and 'old women' both can be treated as two different NPs with a conjunct, which can be treated as a second possibility. However, the statistical parser uses both probabilities and explains vividly which probability occurs more in the text.

1.3.3 Hybrid-based syntactic parser

A hybrid model is a mixture of both rule-based and statistical-based parsers. Using the hybrid-based syntactic parser any one of the grammars can be tested irrespective of its complexity and genetic affiliation. It can also use any one of the grammatical frameworks (constraints) for the syntactic analysis. Using this approach, any type of ambiguity can also be solved in the analysis of a language. One can also easily find out the difference between the hybrid-based parser and the other parsers.

1.4 Research Problem

In the globalized world, language studies have become a central point, especially in natural language processing. Across the globe, university researchers, and dependent scholars are trying to build many apps in the area of Natural Language Processing. In such attempts, many grammatical frameworks have come up in the Western world. In all these attempts, the choice of a particular grammatical framework and linguistics resources play a major factor in implementing a syntactic parser. In the present scenario, various grammatical frameworks and linguistics resources are available for global languages like English and other few European languages. Among such works, Context Free Grammar (CFG), Lexical Functional Grammar (LFG), and Generalized Phrase Structure Grammar (GPSG) are a few. From the linguistic resource point of view, there are PEN treebanks and dependency treebanks available for those languages. As for the Indian languages are concerned, there are very few such attempts. Apart from this, there are very grammatical frameworks and linguistic resources such as Paninian Grammatical Framework (PGF), and Tree Adjoins Grammar (TAG). In this context, there is an emergency to develop a dependency syntactic parser for Indian languages. Among Indian languages, there are few linguistic resources like Ann-corpora for Hindi, Telugu, and Bengali. However, it is lacking in the case of other major Indian languages. Among such languages, Marathi also falls in this category. Therefore, a modest attempt is made to develop a dependency tree bank for the Marathi language, which becomes a base forfurther research on the language.

1.5 Aim and objectives

The main object of this research is to build a Paninian Dependency Parser for Marathi**.** For this, a good size of the corpus is selected from various electronic and non-electronic sources. Apart from selecting the corpus and its various cleaning processes, a hybrid-based syntactic parser is used to develop tree banks without many errors. Based on the procedures in the grammatical framework, a good size of Marathi treebank is created. In the process, many ambiguities and errors have been eliminated, thereby bringing out a good tree bank set. Later, the automatically produced data is studied manually to see the problems encountered while generating the treebanks. Finally, the study

makes hand-crafted rules for the Marathi parser. This becomes a base for further research and helps in automatic machine translation, specifically for Marathi and Indian languages in General.

1.6. Scope and Limitation of the Study

The present study is a modest attempt to build a Paninian Dependency Parser for Marathi and looks at the problems faced during the creation of a dependency parser for Marathi. The study is limited to creating tree-banks and formulating certain hand-crafted rules, by using annotated linguistic resources. This work does not claim to be a divergence study or aims at automatic machine translation. There are various studies of the Marathi language from the grammatical point of view, but very few definitive works from the Natural Language Processing (NLP) Point of View. Despite the number of dialects available within the Marathi language, this work is confined to the standard variety.

1.7. Organization of Thesis

The thesis is structured in the following way:

Chapter I Introduction

This chapter describes Natural Language Processing (NLP), its importance, and its significance. Apart from this, it also discusses the various approaches of syntactic parsers and their implementation. Finally, the chapters highlight the research problem, objectives, scope, and organization of the thesis.

Chapter II Review of literature

The chapter discusses works related to Building Panini Dependency Parser for Marathi, Treebanks, and syntactic parsing. Along with this, the chapter also discusses the categorization of Treebanks viz. Constituency Treebanks (CT), Dependency Treebanks (DT), and Treebanks for Indian Languages (TIL). In addition, various other works were reviewed related to syntactic parsing developed especially for Indian languages.

Chapter 3 Methodology

The chapter discusses in detail the methodology which includes issues such as how the data is collected, the domains of the corpus, and the size of the corpus. Later, it discussesthe implementation of the dependency annotation scheme, which is developed based on the Paninian grammar framework. Finally, the chapter concludes by discussing, how 'sanitary' tools are used to annotate Marathi data and automatically convert it into Shakti Standard Format (SSF).

Chapter 4 Parser and Parsing: A Theoretical background

It deals with the theoretical aspects of a parser and parsing techniques. Subsequently, it also discussed the significance of the Paninian grammar framework and the importance of Treebank.

Chapter 5 The Dependency annotation scheme and hand-crafted rules for Marathi

It demonstrates the application of the dependency annotation scheme for Marathi corpus. The dependency annotation scheme comprises of a two set of relations: the ... relations (dependency relations), non-... relations (other (than ...) dependency relations), and non-dependency relations. In addition, it also discusses in detail hand-crafted linguistic rules, which help in disambiguation and resolution of complexities.

Chapter 6 Conclusion and futuer work

The chapter deals on the problems faced while building a syntactic parser and solutions for their resolution. It also sets out to explain the limitations of the present work while laying directions for future study.

CHAPTER II

REVIEW OF LITERATURE

2. Introduction

In the present chapter, the researcher discusses theoretical foundations related to the research topic, i.e. 'Linguistic Inputs in building a dependency parser for Marathi: a panianian approach'. As a part of the exercise, existing literature and related works on the topic, especially on Treebank (linguistically annotated corpus, which carries or contain part-of speech, grammatical relations and represent as syntactic trees) are discussed. Historically speaking, there are two main turns in the study of a Treebank. In the first turn, Leech 1983 explained the meaning of a Treebank. It was mentioned by Sampson (2003: 23-41). According to him, the Treebank is a 'structurally analyzed sample of natural language'. In the initial phase of the Treebank research, it took one decade to draw syntactic trees, which subsequently produced a 25-page typescript, listing a set of grammatical category symbols. The work was published in 1995 as a book titled *English for the computer*. Later in the second turn, Nivre (2002: 123-138) mentioned the MAMBA annotation scheme for grammatical description of spoken and written Swedish described in Teleman. The scheme was developed at Lund University in the 1970s and used for annotating a corpus of 250000 words, containing both spoken and written material. These two turns in the history of Treebanking were starting points. However, in the present chapter, the researcher has classified the related works of Treebank based on annotation scheme viz. Constituent annotation scheme, Dependency annotation scheme and Hybrid annotation scheme.

2.1. Constituent Annotation Scheme

Constituent analysis was first used for the analysis of syntactic structure in the linguistic analysis of a language. Later the concept was adopted in the field of corpus linguistics (CL) and natural language processing (NLP). In the literature of the Treebank, the PENN Treebank used a constituent annotation scheme for corpus analysis.

2.1.1. PENN Treebank

For the first time, Marcus et al. (1993: 313-330) used a constituent annotation scheme for PENN Treebank on English in which parts of speech tagging (POS) is the main task along with syntactic bracketing (SB) and disfluency annotation (DA). The project went on for almost eight years in which corpuses were chosen from various sources viz. IBM computer manuals, nursing notes, editorials from Wall Street news paper. Simultaneously, articles, manuals, notes and transcription of telephone conversations are also taken into consideration. Seven million words from all the areas mentioned above were annotated as Parts of speech tagging (POST) among which three million were bracketed and two million were text parsed for predicate-argument structure and 1.6 million words were transcribed and annotated for speech disluency. The annotation process was done manually and with a

few automatic tools (switchboard) at three levels, viz. Part-of-speech (POS) tagging, syntactic bracketing and Disfluency annotation. Thirty six parts of speech tag sets were used at word level. Later, based on the requirements of the syntactic structure, the study also added another twelve other tag sets used for skeletal context-free bracketing with limited empty categories and a Predicate-argument scheme to label each argument with an appropriate semantic label to identify its role concerning that predicate. Finally, the project also undertook disflyency annotation. Consequently, the study had yielded excellent results. Te following is one of the examples of PENN Treebank in the bracketing format, which reflects the constituency annotation scheme.

(s (NP-SUB boy)
(VP throws
(NP-OBJ the ball)))

2.1.2. French Treebank

In the later period, building a tree-bank for the French was proposed in 1997. As a result, a project came into existence named French Treebank. For this purpose, various domains of the language were extracted from the largest circulated daily newspaper at the time titled *Le Monde*ranging (the Monderanging). The process of extracting took almost four years from 1989 to 1993. The major domains covered in extraction were Economy, Literature and Politics representing contemporary spoken and written language resulting in one million tokens. According to Abeillé et al. (2003: 165-187) the project was divided into two phases namely morphosyntactic tagging and syntactic tagging. Morphosyntactic tagging covered a few linguistic aspects of the language, viz. part of speech for determiner, sub-categorization for possessive and cardinal, inflection for masculine singular, lemma for canonical form and part with similar morphosyntactic tags for compound. In the second phase, especially in syntactic tagging, the other linguistic aspects like main category (S, PP, NP….), possible subcategory (relative clause), surface function (sub, obj for Nps), opening and closing boundaries (<></>) and diathesis (passive for verbal nuclei). However, in the first phase, they completed 218 tags with the average correction speed of 500 words per hour. The process was much slower than that of the Penn Tree-bank, which completed 2000 words per hour. The reason was compounds in the language and richer tag-sets. In the second phase too, they found that average number of words per sentence was twenty-seven and average number of internal constituents was twenty.

2.1.3. Spanish Treebank

Subsequently, another project started in 1997 for Spanish. The name of the project was Syntactically Annotated Corpus (SAC). According to Moreno et al. (2003: 149-164) mentioned in the project, which consists of 1,500 sentences, are taken from newspapers and magazines of the time. Penn Tree-bank style had been adopted annotating the sentences. In the process, they tried to exhibit four layers of information, namely syntactic categories (Noun, Adjective….), syntactic functions (SUBJ, OBJ..),

morpho-syntactic features (number, gender, tense, etc.) and some other semantic features like (Human vs Non-Human, Time and space, etc.). During the analysis, the major problem faced by the team was lexical ambiguity, especially in the case of the pronoun "se" (he/she). To handle the pronoun "se" they had established five different lexical and syntactic meanings namely reflexive or reciprocal, pronominal or intrinsic, impersonal and as an intransitive marker. However, annotating 1500 sentences, they managed to run the experiment, which was used to train a statistical parser. Since the corpus was very small, it resulted in 73.6% recall and 74.1% in the evaluation.

2.1.4. Arabic Tree-bank

Arabic Treebank (ATB) was also built on similar grounds used for English. Maamouri et al. (2004: 102-109) state that the project was executed in three phases viz. corpus collection from written standard Arabic words comprising 166k, collection from mass media corpus containing 144k words and finally 350k words from print media. According to (c.f Maamouri et al. 2004), the project is divided into two main stages. The first stage deals with mophonimmic feature of the language like POS tag, morphological segmentation, vocalization including case and mood endings, with English gloss were analyzed. And the second level dealt with syntactic features of the language like an interpretation of the sentence, constituent boundaries, functional categories added to verb phrases and empty categories. Though the project was dealt on similar lines to English PENN Treebank, it could not yield similar results in English PENN Treebank.

2.1.5. Chinese Treebank

Xue et al. (2005: 207-238) described and discussed the Chinese Treebank (CTB) in their thought-provoking paper titled 'The Penn Chinese TreeBank: Phrase Structure Annotation of a Large Corpus'. According to them, the project was started in 1998. The project was executed in two segments. As a part of the first segment, 325 articles and editorials were collected from Xinhuva newswire covering economic development at the time occurred during 1994 to 1998. In the second segment, 337 articles were collected from various newswires like Sinorama, Hong Kong news and from Xinhuva newswire, from which the first segment was also collected. As part of CTB exercises, in the first segment, the analyzed sentence with an average length was 28.7 words. In the second segment, 130,000 words from Xinhuva newswire, 15000 words from Hong Kong news and 6000 words from Sinorama with an average sentence length of 8.9. After the corpus collection, annotation scheme was executed at three levels viz. words segmentation, part of speech (POST) and syntactic bracketing. Except words segmentation, the other two levels like part of speech (POS) and syntactic bracketing were similar to English PENN Treebank. Since written Chinese is pictographic, identification of words in a particular written text became a very difficult task during the project. To handle the issues, especially the word segmentation, they adopted the notion of 'syntactic words', which is very famous in Chinese linguistics and literature. For the purpose, they had decided to take syntactic word rather than

morphological word. Subsequently, as part of the exercise, parts of speech (POS) tagging scheme established 33 tag sets based on the semantic and syntactic features. In syntactic bracketing, they also chose phrase structure constructions for labelling brackets with three grammatical relations such as complementation, Adjunct and coordination. Though, argument/adjunct distinction was not made in the PENN English tree-bank, but it was commonly agreed stated that this distinction can be made in Chinese Treebank.

2.2. Dependency Annotation Scheme

The dependency annotation scheme differs compared to the constituent annotation scheme. In dependency relations, the main goal is connecting link between the words in a given sentence of a particular language. The connecting link establishes certain relations for example the head-modifier or mother-daughter relations. For the first time, it was used for the Czech with Prague Dependency annotation scheme. The following diagram shows the dependency annotation scheme, which is referred as mother-daughter or head-modifier relations between a governor and its dependent node:

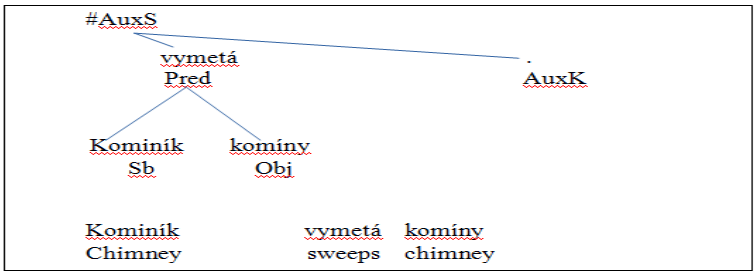


Figure 2.1 Dependency relations used for Czech Language

2.2.1. Prague Dependency Treebank

Hajičová. et al. (1998: 103-126) state in their informative paper that the Prague Dependency Tree-bank (PDT) was a long time project which lasted between 1996-2000 and later extended to up to 2004. According to them, the corpus for the project was selected from Czech National Corpus Institute (CNCI), which covered the domain like politics, economics and contemporary issues of the time from newspapers, magazines and mass media. In the execution of the project, i.e. Dependency framework, they had implemented three levels for annotating the corpus, namely morphological level, analytical level and Tectogrammatical level. Within the morphological level, they annotated words from the lemma and tagged them as particular tokens. Further, in the analytical level, they used dependency trees, which were referred to as mother/daughter or head/modifier relations between a governor and its dependent node. At the analytical level, they also handled different linguistics issues, which were confronted while analyzing especially non-determinative cases such as coordination, relative clause, apposition etc. As part of the exercise of dependency relations, they made use of the

Collins lexicalized stochastic parser was adapted to suit the data structure and format; after the training, they also analyzed 1926 annotated Czech sentences, which resulted in 80% of the dependency relations being correct. Finally,in the third level, i.e. Tectogrammatical level, they described the sentences with linguistic meaning by using the functional generative perspective module. The tectogrammatical level concentrated on the following points:

1. Addition of auto-semantic relations
2. Nodes are added in place of deletions at the surface level.
3. Not allowed the Non-projectivity
4. Analytic functions are substituted by Tectogrammatical
5. Basic features of the structure of the sentences, like Topic-Focus Articulations, are added.

Overall, the project yielded good results in dependency annotations, ultimately natural language processing.

2.2.2 Russian Dependency Treebank

In the subsequent years the Russian Dependency Treebank project was initiated in 2000. According to Boguslavsky et al. (2000:187-191) the project aimed to 'develop first annotated corpus of Russian texts'. They also stated that until then there was no annotated corpus for Russian. As consequence, there were no particular Corpora for Russian. To develop a best corpus for Russian, Russian Treebank was initiated to fill the gap. For this purpose, large corpus had been chosen from the Uppsala University. The corpus mainly consists of contemporary Russian prose, as well editorials and articles from the newspapers and magazines of recent decades. From the prose as well as articles and editorials, they extracted almost one-lakh words as a primary source belonging to various domains. The annotation scheme of the Russian Treebank had three stages viz. *lemmatized text,* which indicated words and part of speech*; annotating morphologically tagged text* that described the complete set of morphological attributes; and *syntactically tagged text,* which contained syntactic structures. However, for the syntactic structures, they used dependency formalism with 78 syntactic relations, which resulted in the generatation of 4000 sentences containing 55,000 words constituted 305 of the total corpus. By the end of the project, a Russian annotated corpus was developed, which had been useful since 2000 in the area of Natural language Processing and computational linguistics and purposes. The project also was very similar to Prague Dependency Tree-bank (PDT).

2.2.3. Dependency Treebank for English

In the subsequent years, Dependency Treebank for English was initiated at Pennsylvania University. The university was granted the project from ISLE. According to Rambow et al. (2002: 857-863), the project undertook a small corpus of 13000 words of dialogues with multiple layers of annotation. Those multiple layers of annotation deal with orthography, intonation, syntax, NP co-reference, dialogue structure and discourse structure. Most of the dialogues were conversations between travel

agents and customers. After collecting the dialogues, they transcribed them at Carnegie-Mellon University as part of the DARPA communicator project. For the project, they adopted the Prague Dependency Tree-bank (PTB) with an augmented mono-stratal analysis. In the mono-stratal analysis, thy covered morpho-syntactic features such as the word forms, the root forms of the word and parts of speech (POS) tag for morphological features. They also had implemented SRole- surface-syntactic roles (Subj, Obj, Obj2, PObj, PObj2, Adj), DRole- deep-syntactic role and FRR- functional relation reassignments. Along with the above three, they also had dealt with syntactic features like Pass(ive), Erg(ative), Pred(icative) and nonprojective features like long-distance *wh*-movement, subject-to-subject raising, control and VP-ellipsis and a few linguistics issues. Since the corpus was small, SRole had achieved the agreement of 93–94%, DRole up to 95 while other features achieved the agreement above 99%. Some of te issues discussed in the paper (c.f. Rambow et al. 2002) also can be found in the analysis and annotation of Marathi.

2.2.4. Alpino Dependency Treebank

In the same year, Dependency Treebank was the range of importance in the country. Van der Beek et al. (2002) state that the project was intended to design and analyze sentences that are unrestricted in the Dutch text. The aim behind project was creating the Alpino Dependency Treebank based on the dependency relations of the syntactically annotated corpus in order to train the grammar and finally evaluate its performance. The Treebank consists of 6000 sentences, which had been extracted from the newspaper and annotated with dependency structures. To train the grammar and to evaluate the performance of the Dependency Treebank they used a probability model, which was also called as stastical model. Since the annotation is a time-consuming process, they developed several tools like interactive lexical analysis and constituent marking, which reduces the set of parser. The innovation in the project is the introduction of the stastical method, which was not executed in the earlier studies. The study also gives inputs in the analysis of Marathi where one can also introduce the stastical method.

2.2.5. Basque Dependency Treebank

In the subsequent year, i.e. in 2003, another project was initiated for Basque. The main aim of the project was to build annotated corpora with linguistic annotation at syntactic, semantic and pragmatic levels. Aduriz et al. (2003: 201-204) state that standard written Basque articles were selected, which contains 25,000 word-forms from EPEC and 25,000 words coming from other sources like newspapers. As part of the project, the team had annotated the corpus, which was named as Eus3LB. The Eus3LB was annotated based on dependency formalism. While annotating the corpus, they identified that in syntactic argument, Basque does not appear to be lexicalized in grammatical relations. Another, identification was the tag set showed grammatical structures such as relative clauses, causative sentences, coordination, discontinuous elements, and elliptic elements. In the final

phase of annotation, the team also identified structurally case-marked complements, modifiers, negation, linking- words, auxiliaries and others semantic relations. Finally, as the tagging process went on, new solutions were found for new arising problems. Aduriz. et al. (2003: XX) also stated that at the end of the project they defined tag set of the Bosque got gradually improved in accuracy and robustness.

2.2.6. Turin University Treebank

As the research was high in computational linguistics, many projects had been started coming up various languages across the globe. So far, in the earlier projects, people made use of Treebanking, bracketing, dependency annotation etc. For the first time, argument relation structure (ARS) in the Italian language is spoken in the Turin province of Italy. Though ARS existed earlier, use of ARS was the new beginning in the area of computational linguistics, especially in the Italian language. Bosco et al. (2004: 462-472) stated that the Turin University Treebank (TUT) collected 60% of the corpus from the civil code, about 30% from legal language and about 10% from other miscellaneous sources. The new thing that came to pass in the project was the application of Augmented Relational Structure (ARS) on dependency relations, in which 1,200 sentences were annotated, which consisted of 35,000 words. As a result, the current number of values, like morpho-syntactic component includes 40 items, the functional-syntactic component includes 71 items, and the semantic-syntactic component includes 102 items. Later, they combined 213 values with 343 other valid feature structures, which are also known as grammatical relations. Finally, the project became the basis for other projects especially application of ARS on Dependency annotation scheme.

2.2.7. Greek and Latin Dependency Treebanks

Bamman et al. (2011: 79-98) mentioned in their informative paper that more than 200 researchers across the globe worked on the ancient languages Latin and Greek. They also stated that, so far people applied Treebanking, dependency scheme and argumented relational structure on contemporary and new modern languages but not classical languages. For the first time, Bamman and their team applied on Greek and Latin. Ancient Greek Dependency Treebank (v. 1.2) includes the works of Homer's Iliad and Odyssey, Sophocles' Ajax, and works of Hesiod and Aeschylus were included for a total number of 309,096 words. Later for Latin, they included eight contemporary works comprising prose, poetry and drama. The entire corpus for Latin dependency Treebank was 53,143 words extracted from the eight texts. For the Latin and Greek, they also did morphological analysis manually that extracted nine features viz. part of speech, person, number, tense, mood, voice, gender, case and degree to both languages. Subsequently they assigned a syntactic structure to all the sentences in the more extensive textual collection. Later, they also applied automatic extraction, which listed 12 syntactic features for every word in the sentence. Those features were arranged with a combination of word-level representation, the length of the syntactic tree, and the presence or absence of an edge label. To

conclude, dependency treebanking was applied not only to contemporary and new modern languages and also applied on classical languages like Greek and Latin, which yielded good results.

2.2.8. Universal Dependencies Treebank

In the foregoing studies one can observe that the application of annotation, Treebanking, dependency annotation had been applied only to a certain time-period. There is no consistency. Later in the subsequent developments in computational linguistics, universal dependencies emerged. The main goal of UD is the consistent annotation of grammar, i.e. parts of speech, morphological features, and syntactic dependencies. Using this framework researcher's across the globe worked on more than 90 languages in which Marathi also included. According, to McDonald. et al. (2013: 1659-1666), the UD framework is an open community effort which creates cross-linguistic consistent tree-bank annotation for many languages within a dependency-based lexical-list framework. Within this frame, several tools were used. Google universal tag set was used for the morphological layer, Stanford dependencies was used for the syntactic layer, and *Interset*was used for the conversion of morpho-syntactic tag sets of multiple languages. As a part of the exercise, they combined Google Universal Dependency Tree-bank (GUDT) with Stanford dependencies for results in 2013 for languages viz. English, French, German, Spanish, Swedish and Korean and 11 languages viz. Brazilian Portuguese, English, Finnish, French, German, Italian, Indonesian, Japanese, Korean, Spanish and Swedish in 2014. In this framework, analytical method was executed at tree levels namely word segmentation, morphology and syntax. At word segmentation, they used lexicalist approach for multiple word expression. At the morphological level of three sub categories like lemma, a part-of-speech tag and their set of features, which encode lexical and grammatical properties associated with the word form. At the syntactic level, they described content words, functional words and 40 grammatical relations. As result, the UD aimed to develop multilingual NLP systems and comparative linguistic studies of world languages. Though there was an earlier work on Marathi in the UD framework, the present research follows the Paninian dependency framework, which differs in theoretical framework.

2.2.9. Hamle DT Treebank

Zeman et al. (2014: 601-637) report, that HamleDT was a compilation of existing dependency tree banks. Since all the Treebanks harmonized, it was expected to improve the interpretability and comparability of parsing with accurate results, and therefore help to drive the development of dependency parsers towards multilingual robustness. They also stated that, the HamleDT has identified over 30 languages to which tree-banks were developed and made available for the researchers. The main aim of HamleDT was to identify all syntactic constructions annotated differently in different treebanks. Along with identification, it was also intended to design algorithms to normalize the annotations of many linguistic phenomena with a single style, which can referred to as the HamleDT v1.5 style. However, different treebanks have executed different annotation schemes

for their linguistic units, such as coordination, preposition, subordinate clause, verbal group etc. However, all the linguistic units were merged into one style with an automatic nominalization. Subsequently, transformation rules were applied on the linguistic units in order to get the original structural annotation, dependency labels and morphosyntactic tags. Along with they had also proposed unification tag sets such as parts of speech, morphosyntactic features, and dependency relations. Later in the subsequent works, the method of unification was implemented in NLP and computational linguistics.

2.2.10. A Gold Standard Dependency Treebnak for English

Gold Standard Dependency Corpus for English Silveira et al. (2014: 2897-2904) discuss and describe Gold Standard Dependency Corpus for English in their thought-provoking paper titled *AGold Standard Dependency Corpus for English*. In the article they clearly mentioned about two things. The first one is to examine the English Web Treebank corpus (EWT), and the second one is collecting corpus from Wall Street Journal (WSJ) consisting of 254,830-word tokens (16,624 sentences) of web text which was analyzed earlier using Penn Treebank. Later both texts were compared to examine the ratio of the annotations. Later from which they discovered that both the annotations in WET and WSJ were similar. Subsequently, they also tokenized the words into various parts of speech tags. The tokenized parts of speech tags were converted through the Stanford Dependencies. As a result, all the Treebanks have resulted in Penn Treebank style. Finally, they found that Stanford Parser had produced high-quality dependency annotations. Later they were used to train across the genres in order to improve parser performances.

2.2.11. Multi-view Chinese Treebank

Qiu et al. (2014: 257-268) explained the A multi-view Chinese Treebank. For the project, the spoken corpus was collected from the citizens working at Peking University (PKU). Since the corpus was collected from Peking University (PKU), it was named as Peking University (PKU) corpus, which was also known as Multi-view Chinese Treebank (PMT). The corpus consisted of 14,463 sentences and 336K words. The schema for annotation framework was discussed at two corners, viz. part-of-speech (POS) and Dependency categories tag-set. In the first corner, PKU POS tags were 100 that were simplified into 30 tags and mapped with PMT and in the second corner dependency, category tagsets were combined by using the combination strategy. In the process, they combined 32 dependency categories into eight classes. Finally, with the combination strategy and DS-based multi-view annotation framework, they built a Chinese Treebank. Finally, they concluded that their parser was arc-standard transition-based dependency with the combination strategy that could be applicable in NLP and computational linguistics.

2.2.12. Persian Universal Dependency Treebank

The Persian Universal Dependencies was one of the initiatives done for the tire morphological language like Persian. According to Seraji (2015: 2361-2365), the project had developed its Treebank on the original annotation scheme, which was based on Stanford Typed Dependencies. This was done with the modified version Uppsala Persian Dependency Treebank (UPDT) to the UD framework. The tree-bank consists of nearly 6000 sentences from written text comprising various semantic domains of the language written in various genres, especially from newspaper articles, fiction, technical descriptions, and documents about culture and art. When Uppsala Persian Dependency Tree-bank (UPDT) was converted to the UD framework, they identified that the UPDT was annotated with 29 part-of-speech tags with all the morphological information. Later the tags were directly mapped to 15 part-of-speech tags of the total 17 tags in the UD. In the similar manner, UPDT consisted 96 dependency relations, of which 48 were used for basic relations. Further, they also converted syntactic representation UPTD to UD. As a result, the total number of dependency relations in the Persian UD was 44 consisting of 37 universal dependencies and seven language-specific relations. They are relative clause modifier acl:relcl, predeterminer det:predet, light verb construction compound:lvc, phrasal particle compound:prt, pre-conjunction conj:preconj, the genitive modifier nmod:poss and non-canonical subject nsubj:nc. Except nsubj:nc relation, the rest of the language-specific relations presented in the Persian UD. This was a major advancement in the area of computational linguistics.

2.3. Hybrid Annotation Scheme

The hybrid annotation scheme is a combination of two annotation schemes viz. Phrase/Constituent structure and Dependency structure. The following is the work done in the hybrid framework:

2.3.1. The TIGER Treebank

According, to Brants. et al. (1999: 69-79), the project TIGER had been financed by the Deutsche Forschungsgemeinschaft since 1999 at the University of Saarbücken, Stuttgant and Potsdam. The project used the hybrid method, which is a mixture of Phrase/Constituent structure and Dependency structure. For the first time, a larger tree-bank (> 50,000 sentences) with an extended annotation scheme and improved search facilities was used for the TIGER project. As a part of the project, they abandoned the idea of a pure dependency-based annotation scheme and instead adopted a hybrid framework that combined phrase structure and dependency grammar. The corpus includes part-of-speech, syntactic structure and labelling of nodes and edges. Apart from the hybrid annotation scheme, they used statistical methods (Markov models) for the first time in the framework. As a result, the scheme highlighted the flat structures well suited to human annotation. The time spent on the corpus annotation amounts to approximately 10 minutes per sentence. Finally, the scheme yielded good results for the TIGER project.

2.3.2. The Quranic Arabic Dependency Tree-bank

The Quranic Arabic Dependency Tree-bank (QADT) was part of the Quranic Arabic Corpus. It was an online linguistic resource project organized by the University of Leeds. According to Dukes. et al. (2010: 1822-1827), the corpus was collected from the Quran, a holy book to millions of Muslims across the globe. As a part of the corpus collection, 77,430 words of the Quran were collected and manually verified for morphological and syntactic analysis. As part of a gold standard, tree-bank of 11,000 words of Quranic Arabic had been syntactically annotated. Later, for the syntactic representation, they adopted a hybrid scheme. Since the language has VSO word order, the project team discussed the application of the hybrid method in VSO language, i.e. Arabic. Apart from the word order, they also discussed issues like hidden empty nodes and prepositional phrase attachment and a few other linguistics issues were highlighted. Finally, the project was completed with good results using the hybrid method.

2.4. Dependency Treebank in Indian Languages

In the dependency Treebank framework, especially for Indian languages, The Paninian framework was introduced in the 19t century. For the following languages of India, the frame was used at various institutions of India.

2.4.1. AnnCorra : Building Tree-banks in Indian Languages

AnnCorra was one of the first initiatives in the area of computational linguistics, especially for Indian languages. A group of people from computational research initiated the AnnCorr project at Hyderabad. IIIT was the venue for this new project. The main aim of the project was to generalize the linear syntactico-semantic tag scheme for all Indian languages. To fulfil the aim, they used the Paninian (great Indian sage and Grammarian) grammatical model for the project. According to Bharati. et al. (2002) Paninian grammar would be more suitable for Indian languages than other frameworks discussed above. As a part of the project, they intended to develop formal notation to identify various kāraka relations like k1:kartā (subject or agent), k2:karma (object), k4: sampradāna (beneficiary), v: kriyā (verb) in a sentence. Apart from the development of notations, they also developed other rules to identify complex sentences, compound sentences and multiple verb sentences. To conclude, the paper by (c.e.f. Bharati. et al. 2002) was an excellent informative paper which explains identification of notations and development of various rules for complex sentences. Since Marathi is also one of the Indian languages, the research is more benefitted from the paper in creating a parser for Marathi.

2.4.2. Dependency Annotation Scheme for Indian Languages.

Based on the idea by (c. e. f. Bharati. et al. 2002), the same group of people extended the concept and stared implementing it on Hindi. According to Begum. et al. (2008: 721-726), the schemes developed

in AnnCorra were adopted and started implemented on Hindi under the project name Hyderabad Treebank. For this purpose, they collected the corpus from the Central Institute for Indian Languages (CIIL) situated in Mysore, one of the southern cities of India Karnataka. The corpus was very large and covered many semantic domains of the language viz. newspaper, literature, government reports, etc. As part of the annotations exercise, they made use of 21 relations (kāraka relations/non kāraka relations) with special tag-sets (other than kāraka and non kāraka relations) such as POF (part of relation), ccof (conjuncts relation), NULL (Null_NP, Null_VG, Null_CCP) to mark different kinds of ellipses.

Along with the annotation exercise, they had done a few preliminary experiments on a corpus of 1403 sentences with a particular method called as heuristic method (hand crafted rule H1, H2, H3….). When they executed the hand-crafted rule on the corpus, it identified 1801 instances of k1, which appeared in the nominative case. Within the 1806 occurrences H1 identified 1461 instances correctly and did not identify the rest of the occurrences due to the morphological and syntactic complexity of the language. H2 and H3 of the hand-crafted rule did not work as H1 worked. On similar lines to Dependency Annotation Scheme for Indian Languages, the researcher also intended to apply to Marathi. Under the framework, it would be more accessible to establish a good parser for Marathi.

2.4.3. Telugu Treebank

Subsequently, the Dependency Annotation Scheme for Indian Languages was applied to Telugu. According to Vempaty. et al. (2010: 50-59) similar model used for Hindi was implemented. As part of the project, they annotated 1487 sentences using the Paninian grammatical framework in order to build Telugu Treebank. By using the Paninian Grammatical Framework (c. e. f. Vempaty. et al. 2010) had divided the kāraka relations into six types viz. adhikarana 'location (k7)', apādāna 'source (k5)', sampradāṇa 'recipient (k4)', karana 'instrument (k3)', karma 'theme (k2)', kartā 'agent (k1)' and other relations like vmod and nmod, and other relations. Apart from dividing the Karaka relation they had also discussed about simple constructions like, 'go', 'cut' and 'give' and their usages in various contexts. While annotating the corpus, they identified the following issues viz. Genitives (in Telugu the genitive marker is often dropped), Conjuncts (different constructions where a conjunct presence is explicit/implicit), Copula (missing verbs i.e. verbs are dropped) and "ani" constructions and its various ways of using in various contexts. Though they encountered various morpho-syntactic problems, the project yielded good results using the Paninian grammatical model. Finally, the paper concludes that the framework is suitable for generating the possible parses and is appropriate or relevant for Marathi.

2.4.4. Kashmiri Dependency Treebank

The Kashmiri Dependency Treebank was initiated at the Luknow University as a part PhD research. According to Bhat (2012: 53-60) the corpus for the study was collected from short stories,

contemporary news paper (Sangarmaal) and a few other political domains. Five hundred sentences from short stories and the rest of the 1361 sentences were from the "Sangarmaal" newspaper and other political domains. The sentences from the short stories were shorter than the sentences of news paper and political domains. Though he used the Paninian framework as a model, he made use of Sanchya tools for syntactic annotation purposes However, in this Dependency treebank, he discussed various issues encountered during the annotation of the corpus. The main issues encountered were V2 phenomenon and discontinuous verb group, complex predicates and their discontinuity, Coordinating and Subordinating conjuncts and pronominal clitics of the language. Though the language behaves in 99% of the cases as SVO language, the Paninian framework worked perfectly and yielded good results.

2.4.5. Tamil Dependency Treebank

The Tamil Dependency Tree-bank (TamilTB) was initiated by Ramasamy and his team in 2012 According to Ramasamy et al. (2014: 143-148) the project was executed based on the dependency grammatical framework. The corpus of the project contains 600 sentences. All 600 sentences were annotated using Prague Dependency Tree-bank (PDT). In generally, the annotation process includes two layers viz. m-layer (corresponds to POS tagging using a heuristic method) and a-layer (dependency annotation). As part of the project, they tagged 217 tagsets on Tamil in m-layer. Among the 217 tag sets, were shown as unambiguous, which can be converted to 90%, and only 3% of the words showed ambiguity. The reason for the 3% ambiguity was the intangibility of the tagsets. In a-layer, they defined 21 dependency relations or analytical functions (a-fun) for labelling the edges. In evaluating m-layer annotation, they evaluated with Morce and TnT tagger. The result was good for parts of speech tagging (POST). For a-layer annotation, they considered with Malt (projective) and MST parsers. Around 80% of the data was correct in a-layer. Tamil is the first language for applying Prague Dependency Tree-bank annotation (PDT). Though the project used the foreign grammatical framework, it yielded fair results.

2.4.6. Bengali Treebank

The project was started at IIT Kadagpur in 2009. Chatterji et al. (2014) stated that the project had collected the corpus from various domains of contemporary literature and the spoken language. Four hundred and sixty-seven sentences were extracted from the corpus, which were later annotated by the dependency annotation scheme. Annotation was executed at three levels, viz. intra-chunk relations, inter-chunk relations and inter-clause relations. Since they made us of the dependency relations, they also had described 63 syntactic and semantic features in the Bengali tree-bank. Overall the Eastern Indo-Aryan language, i.e. Bengali, had good results at the end of the project.

2.4.7. Developing Hindi Treebank Based on Computational Paninian Grammar

Another work in the similar frame was initiated with the title '*The pilot Hyderabad Dependency Treebank (HyDT) for Hindi*'. The work was carried out as part PhD research at IIIT Hyderabad. Begum (2017), the researcher had annotated the Hindi corpus, which consists of 2230 sentences with the Panianian grammatical framework and guidelines. During the annotation period, the researcher encountered several issues related to causatives, complex predicates, relative clauses, etc. All the problems were analyzed within the framework of the Panianian grammatical model, i.e. syntactico and semantico analysis. Apart from the above issues, the researcher added another issue of verb-frame, which captured various syntactic distributions expected to occur in a language. After a thorough analysis, the researcher improved corpus, which yielded good results. At the end of the research, the research was productive with his corpus.

2.4.8. A Universal Dependencies Treebank for Marathi

As discussed in the foregoing part of 2.2.8, Universal Dependencies (UD), which was developed with the aim of applying to all the languages of the world, was for the first time applied to Marathi language. Ravishankar (2018: 190-200) states that the corpus was collected from stories from Wikisource. Further, to develop a Treebank for Marathi with the application of UD, the researcher collected a good corpus with 486 sentences. Later the sentences were tokenized. The number of the tokens was 3,506. As a part of the annotation Universal dependencies (UD) was implemented. During the annotation process, the researcher encountered many linguistic issues such as word segmentation, subjective case, objective case, light verbs, compound verbs, passive voice, and dislocation in Marathi. As part of the evaluation, first, the UDPipe was used for tokenization and tagging. The second BIST parser used allowed them to use custom word embeddings for POS tagging. Third Malt Parser was used for dependency relations. As a result, UDPipe, BIST Parser and Malt Parser evaluated both labelled (LAS) and unlabelled (UAS) attachment scores. Apart from the evaluation of LAS and UAS, the researcher also evaluated the weighted LAS, which underweights the contribution of correct labelling relations (like case and punct) to the final score.

2.4.9. The Treebank of Vedic Sanskrit

Hellwig et al. (2020: 5137-5146) explained that, for the first time, UD is applied to Sanskrit, the oldest classical language of India. As part of the Treebank of Vedic Sanskrit, they collected the corpus from the RIgveda which was compiled between 1300–1000 BCE from the Śaunaka Sam.hitā of the AtharvaNaveda. As a part of the project, the researchers used Universal Dependencies standard v.2.0 for dependency annotation. In the process of analysis, the researchers made use of the syntactic labeller that greatly speeded up the annotation process. As a result, the Treebank provided clues useful for setting up a complete syntactic parser of (Vedic) Sanskrit.

2.5. Syntactic Parsing

Parsing is a Latin origin word 'pars' from which the word parsing was developed. In general, the meaning of the parser is to break down the sentences into different parts of speech and assigning a particular grammatical category. Apart from breaking and assigning the grammatical relation, the parser also looks into the grammatical relations between the words that were identified. Syntactic parsing or analysis is one of the processes of natural language processing or computational linguistics. In the process, researchers analyze a string of syntactic structures/words to draw the exact meaning of the structure. This process is mostly used for conforming to the rules of a formal grammar or grammatical model of a language.

2.5.1. Designing a Constraint-Based Parser for Sanskrit

Kulkarni et al. (2010) developed a rule-based parse for Sanskrit. For this purpose, they used the concepts of Śābdabodha available in Sanskrit literature. Śābdabodha has three main aspects, namely Ākāṅks.ā (Expectancy), Yogyatā (Compatibility), and Sannidhi (Proximity). Using those three components, they developed hand-crafted linguistic rules for Sanskrit. As a result, Sanskrit Parser was developed. Corpus was collected from padaccheda-sahita-eka-tiṅ-gadya-vākyam. The corpus consisted of 113 sentences with finite verbs. Subsequently, the sentences were tagged manually, showing the relation of each word in the context, viz. Ākāṅks.ā (Expectancy), Yogyatā (Compatibility), and Sannidhi (Proximity). As a result, the parser evaluated 97 sentences which can be considered as 86% of accuracy. The remaining 16 sentences were wrongly analyzed by the parser due to the complex relations which can be converted to 14% inaccuracy. The reason for the wrong attachments was grammatical relations. The schemas and analysis techniques used in developing the Sanskrit parser would help in developing the Marathi parser because Marathi shares most of the vocabulary and the syntactic structure with the Sanskrit language.

2.5.2. Hindi Dependency Parsing and Treebank

Ambati. (2011: XX) in his Msc. dissertation, he mentions Hindi Dependency Parsing and Treebank Validation. Using Begum (2008: 721-726) Hindi Treebanks, he had done several experiments using data driven parsers viz. Malt and MST. The experiments were run with 1500 sentences, which were already trained. As a result, he was able to build a dependency parser with state-of-the-art. The accuracy of the Hindi Dependency Parsing and Treebank Validation was 74.5% with labeled attachments score (LAS) and 90.1% with unlabeled attachment score (UAS). Finally, the study also explained in detail about the problems and errors encountered and remedies for those problems and errors. The key points (statistical method) in the dissertation are more useful in developing the Marathi parser.

2.5.3. Hindi - Parsing Based On Computational Paninian Grammar

Husain. (2011), in his thesis, discusses a generalized parsing framework based on computational Paninian grammar**.** As a part of his research, he collected data from Hindi Hyderabad Treebank, where he trained the data in the frameworks of Malt and MST. Later he compared the output with the Paninian grammar parsing framework, i.e. constraint-based grammar. As a result, he discovered very crucial point's viz. incorporation of targeted features during the training, introducing linguistically constrained modularity during the parsing, and exploring linguistically rich graph-based parsing. Finally, he discussed the error patterns in the process and concluded with irregular observations. The thesis was more statistical rater than informative.

2.5.4. Dative Case in Telugu: A Parsing Perspective

Uma Maheshwar Rao. et al. (2012: 123-132) highlighted the ambiguity of the Dative Case, i.e. ki/ku occurrence in Telugu, in their thought-provoking paper. The main aim of the study was to handle the ambiguous occurrences of dative case marker in Telugu, especially using a parser in computational research. As part of the study, they discovered dative case -ki/ku exhibited as many as 16 semantic relations. In order to handle this highest ambiguous problem, they made us of the ontological features like [+/-animate], [+/-human], [+/-abstract], [+/-NST], [+/-Nouns of Cognition], [+ /- Nouns of Psych. State], [+/ - Nouns of Phys. state] etc. The study is very much helpful in handling the ambiguity of case markers in any of the Indian languages, especially from the parsing point of view in which ontological feature played a vital role in unambiguate the ambiguity.

2.5.5. Parsing strategy for major Indian languages

Tandon. et al. (2017: 255-265) in their titled ‘Unity in Diversity: A unified parsing strategy for major Indian languages' highlighted unification of parsing strategies. For this purpose, the researchers collected Treebanks from five Indian languages, namely Bengali, Marathi, Kannada, Telugu and Malayalam. The reason for selecting the Treebanks from the above five languages was they were analyzed in the Panianian grammatical framework. As part of execution, they implemented a non-linear neural network greedy transition-based strategy for the parser expecting the five best features, namely Part of Speech Tags, Word, Vibhakti (Suffix and Postpositions), Chunk Tag and Gender, Number, Person. Finally, they discovered in their comparative study that suffix information is more useful for parsing Dravidian languages and postpositions for Indo-Aryan languages, with the exception of the Marathi, because there is no much work on Marathi in perspective of parsing.

2.5.7. Summary

As a part of the research, the researcher has reviewed many works related to five areas, viz. constituent annotation, dependency annotation, hybrid annotation, dependency treebanks used in Indian Languages and syntactic parsing implemented for major Indian languages belonging to various

families of languages. Apart from the above main works, the researcher also consulted some of the works produced and produced from Indian universities for general reading. However, they were not taken into consideration on the grounds of improper methods, data and publication. By studying the above works, the researcher got many insights from the studies, especially from the Indian languages Treebank point of view. The researcher also observed that many Western languages and Middle East languages followed the Prague dependency annotation scheme, and most of the Indian languages had followed the Panianian dependency annotation scheme except Tamil. Apart from that, the researcher also observed that very few works are there on Marathi from the parsing and Treebank point of view. So, the researcher took the challenge of creating a full pledged Treebank for Marathi, which will subsequently help in developing a good parser.

CHAPTER III

METHODOLOGY

3. INTRODUCTION

The present chapter deals with the procedures and methods executed for the design of the entire research program. The methodology includes corpus collection, selecting the annotation scheme, and selecting annotation tools for the implementation of the research. In choosing a corpus, the researcher takes care mainly of the size of the corpus, the type of the corpus, and whether the corpus is balancing or domain-specific. In selecting an annotation scheme, what type of theoretical framework is used for the implementation and annotation of the sentences are discussed. Finally, at the third level, the researcher highlights the selection of annotation tools used for marking the syntactic structures in the data are discussed.

3.1. The choice of corpus

Corpus is a collection of electronically written text, which consists of various semantic domains of a natural language. Such kind of collected corpus can be used for various purposes like corpus research, natural language processing, language teaching material etc. Though there are many applications of a corpus in various fields, it plays a very important role in NLP and computational linguistics. However, in the Indian context, a corpus is available or accessible only for major Indian languages, especially the scheduled languages. Among such scheduled languages, Marathi is one of them. The corpus available for the Marathi is less in size and limited to a few domains. Since the corpus is less in size and limited to a few domains, the researcher collected the data from other sources like grammar books and short stories, articles and other domains from online sources that are available free of cost. Within the grammar books, the researcher selected Valanbe (2012) and Bagavath (2010) and extracted 500 sentences for the purpose. In a similar manner, 700 sentences from short stories, 400 sentences from articles and 600 sentences from other domains are extracted for the purpose. By including all the above-selected domains, the size of the corpus increased to more than 2000 sentences, which can be considered as a balancing corpus for Marathi.

3.2. Clearing the corpus

After selecting the corpus, the main task of the researcher in NLP or computation linguistics is cleaning the selected corpus. In the process of cleaning, one has to arrange the sentences collected from various domains in a proper manner or in a proper sequence. For this purpose, the researcher first identified the sentence boundaries. As an ordinary clue for the identification of sentences, full stops and verbs with past forms at the end of the sentences are used. Wherever ordinary clues like full stops and verbs with past forms were not working, the researcher used manual segregation for the

arrangement of the sentence in a proper shape. After the segregation and arrangement of the sentences, special characters like @, (-), -, _, +, =, $, ^,! in the arranged data are removed wherever necessary and sometimes retained depending upon the context.

3.3. Conversion of the Data into WX Notation

Most of the modern Indian languages that are scheduled in the constitution have their scripts. These languages follow Devanagari, Bramhi, Gurumukhi and Roman scripts. These scripts are being used for various purposes, viz. education, mass media, notice circulations and government administrative purposes. In the areas of computational linguistics and natural language processing, the above-mentioned scripts do not work much due to their script complexity and various technological reasons. For this purpose, unified codes, which work for all languages and are understood by computer programming, are developed. Among such codes, UTF-8 font, UTF-16, roman, international phonetics alphabet (IPA), ISCII font, ISFOCB font and WX notation are a few in number. WX notation converter tool, one of the above-mentioned tools, is freely available on the CALTS website. The researcher used the WX notation conversion tool to convert the Marathi 2k corpus, which was in the Devnagari script. The following example can explain the phenomenon.

Marathi text – शिक्षकांनीविद्यार्थ्यालाबक्षीसदिले.

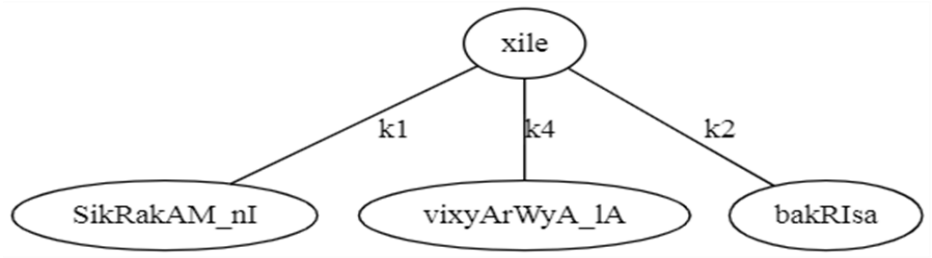
WX notation – SikRakAMnI vixyArWyAlA bakRIsa xile

3.4. The Annotation Scheme

The annotation scheme is one of the breakthroughs in the field of NLP. Though there are a number of grammar frameworks in the field of NLP and CL, Annotation Scheme is theoretically referred to as the motivated model or linguistically defined grammatical framework. The schemes are executed in two types, either as a constituency/phrase annotation scheme or as a dependency annotation scheme. For the present research purpose, the researcher follows a dependency annotation scheme, which is built based on the Paninian framework. The following examples demonstrate in detail how a sentence can be marked using dependency relations based on the Paninian framework. The following Marathi example illustrates the phenomenon.

Marathi text – शिक्षकांनी विद्यार्थ्याला बक्षीस दिले.

WX notation – SikRakAMnI vixyArWyAlA bakRIsa xile



The above diagram demonstrates dependency relations based on the Paninian grammatical framework in which relations are marked as kaaraka relations for Marathi like k1, k2, and k4, which are further labelled as kartaa, (Subject) karma (Object) and sampradana (recipient) with the main verb.

3.5. The Annotation Tool

After cleaning the corpus and selecting the annotation scheme, the third level, real annotation, plays a vital role in building the Treebanks. Annotation can be executed in two ways: manually (where the human annotator plays a role) and automatically (where the machine annotates). As already discussed in the foregoing discussion, the task can be executed to encode the linguistic information on a plain corpus (data). To encode the linguistic information, a researcher may choose a manual mode with the help of various tools available in the field of NLP as well as CL. For the present research, the researcher annotated the data with the help of Sanchay, a tool, on a plain corpus of Marathi (2K).

3.5.1 Sanchay Tool

Sanchay Tool was developed in 2007 with the aim of encoding linguistic information on the plain corpus (unannotated data). The tool was developed by Sing in 2007. According to Sing (2007: XX), it is an open-source platform which can be used for South Asian languages in text processing applications. The tool has the features of text editing, encoding, annotation interface, syntactic annotation interface, tree components, SSF (Shakti Standard Format), API (Application Programming Interfaces), parallel corpus markup interface, customizable language, encoding support, file splitter, format converter and task setup generator. For the annotation process on the plain corpus of 2k Marathi sentences, the researcher used the Sanchay tool, which has more features than other tools. Along with the above features, the tool also has various components, viz. Sanchay editor (SE), rich text editor (RE), Sanchay shell (SS), table editor (TE), tree creator (TC), resource accessors (RA), word list builder (WB), word list visualizer (WV), dictionary editor (DE), language encoding identifier (LI), n-gram language model compiler (LM), syntactic annotation (SA), prop-bank annotation (PA), frame editor (FD), parallel syntactic annotation (PS), sentence alignment interface (SI), word alignment interface (WI), parallel corpus markup (PA), discourse marker (DM), automatic annotation (AA), file splitter (FS) and Sanchya charmap (CM).

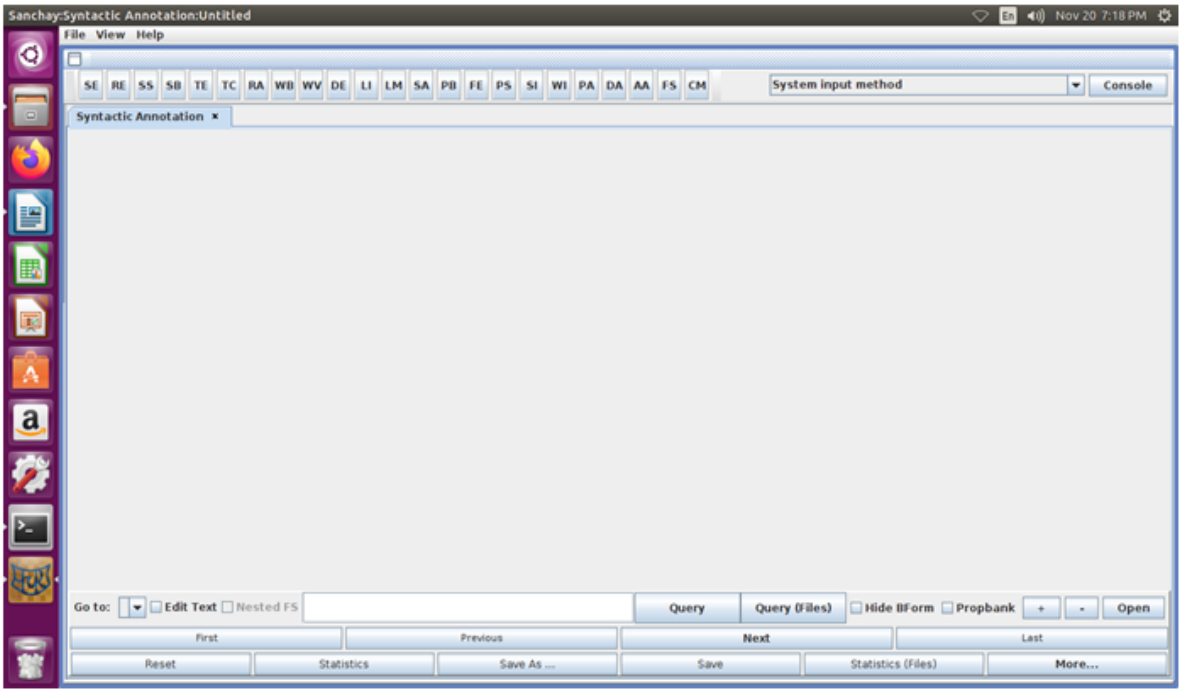


Figure 1 Sanchay Tool with its Features and Components

However, among all the components, syntactic annotation (SA) is more useful for dependency annotation, which has various other linguistic components like morph features, part of speech (POS) features and chuck features are also included. The above picture will describe the implementation of the Sanchay tool with its feature and components.

3.5.2. Annotation Procedure

After selecting the syntactic annotation scheme, the researcher uploaded the file to the Sanchay tool, which consists of 100 Marathi sentences. As a result, the tool helped the researcher to annotate the data at four levels, viz. morph features, part of speech (POS) features, chuck features and dependency relations.

3.5.2.1 Morph feature

Among the four features discussed above, the morph feature highlights the inflectional as well as derivational morphemes of the words in the sentences. They (inflectional, derivational morphemes) are much useful in identifying various syntactic components. The following picture will demonstrate how to annotate the morph feature in Marathi, which is one of the components of the Sanchay tool.

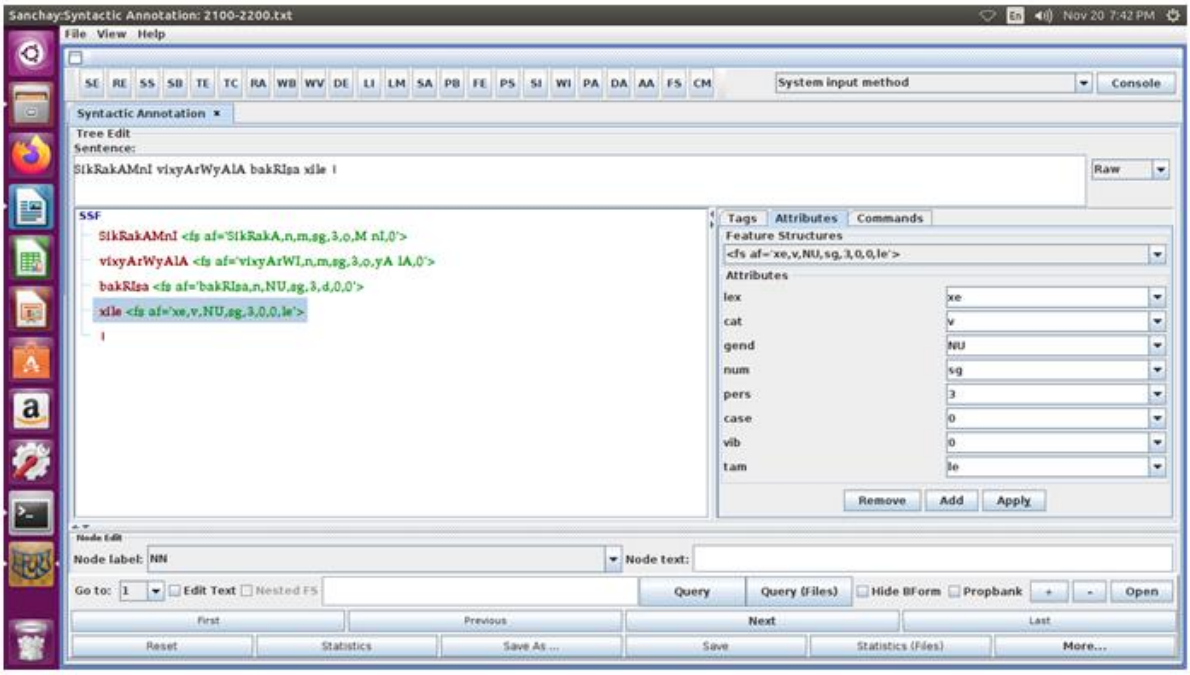


Figure 1 Annotation of Morph Feature

A file was uploaded to the tool as part of the procedure. It categorized the sentence into various words in a vertical format. These words were then categorized manually into nominal and verbal groups. The nominal group is displayed 'SikRakAMnI' in computation format as shown in the brackets (*<fs af='SikRaka,n,m,sg,3,o,M nI,0'>)*. The categorization shown in the bracket was manually analyzed, with the root word as -*SikRaka,* category as n (noun), gender as - m (masculine), number as - sg (singular), person as 3, case as – o (oblique), and the other case marker – M as adposition, _nI as nominative case marker and finally the TAM as -0. In a similar manner, the verbal group also was analyzed which is there in the bracket format ('xile' *<fs af='xe,v,NU,sg,3, 0,0, le'>)* which again marked the root word as –xe, category as -v (verb), gender as –NU (neuter), number as -3, case as - 0, case marker as – 0, and TAM feature as - _le.

3.5.2.2 Part of speech (POS) feature

One of the four features discussed in the annotation procedure, parts of speech tagging (POS), is the second level. The feature highlights grammatical categories of words viz. noun, verb, adjective, adverb, preposition and pronoun. As part of the procedure, the researcher uploaded a file which consisted of 100 sentences. These sentences are divided into various words in a vertical manner on the right side of the tool. When clicked on the word in the right side box, it leads to the tag sets, which are displayed, on the taskbar of the left side of the tool. Further, the researcher used the tag sets to mark the grammatical categories manually as a common noun -NN, main verb- VM, adjective- JJ, proper noun- NNP, auxiliary verb- VAUX etc. the following picture displays the phenomenon of POS features of Marathi.

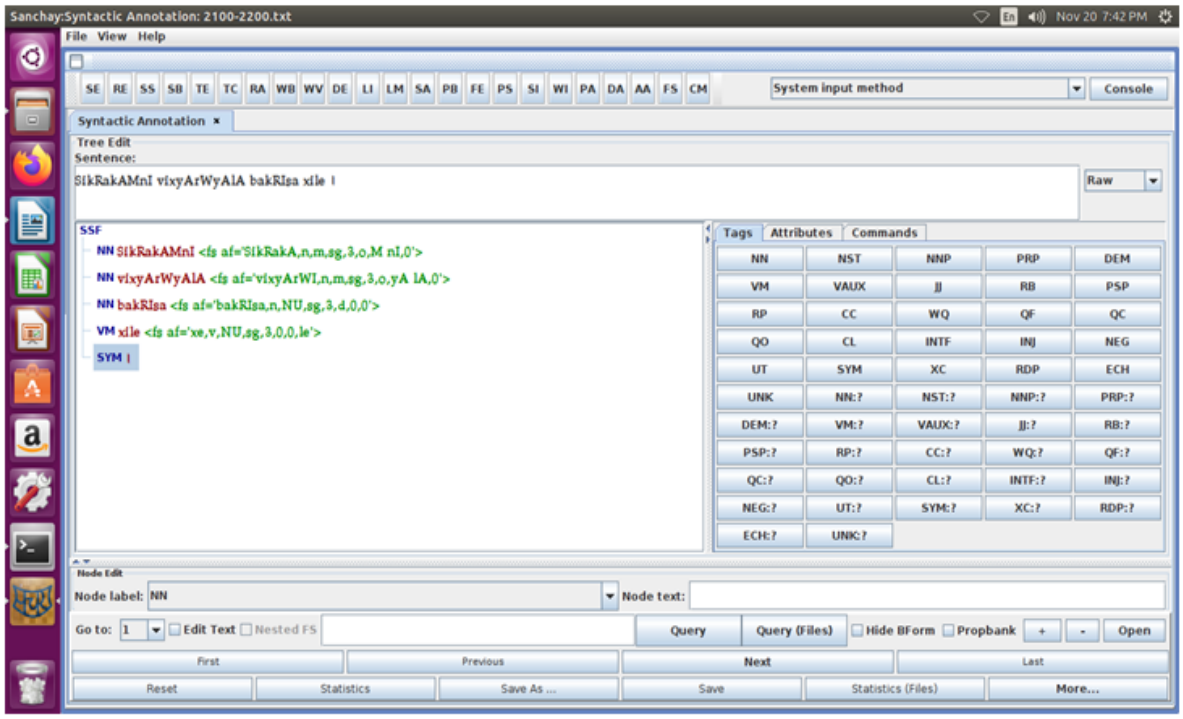


Figure 1 POS Tagging Procedure

The above picture displays SikRakAMnI as a NN, vixyArWyAlA as NN, bakRIsa as NN and xile as VM.

3.5.2.3 Chunk feature

The third feature in the procedure, chunking, groups the words in a given sentence into a combination of words. The procedure of grouping words in combinations in linguistics is referred to as a phrase or constituency grouping. These combinations of words which are considered constituency groups were displayed on the left side of the tool in a similar manner as displayed in Morph and POS tagging. Further, these combinations were highlighted, after which they lead to the additional layer in which the chunked phrases or constituency group appear on the right side of the tool the noun chunk as - NP, finite verb chunk as - VGF, non-finite verb chunk as - VGNF, infinitive verb chunk as- VGINF and adjective chunk as –JJP. The following picture illustrates the phenomenon of Marathi data.

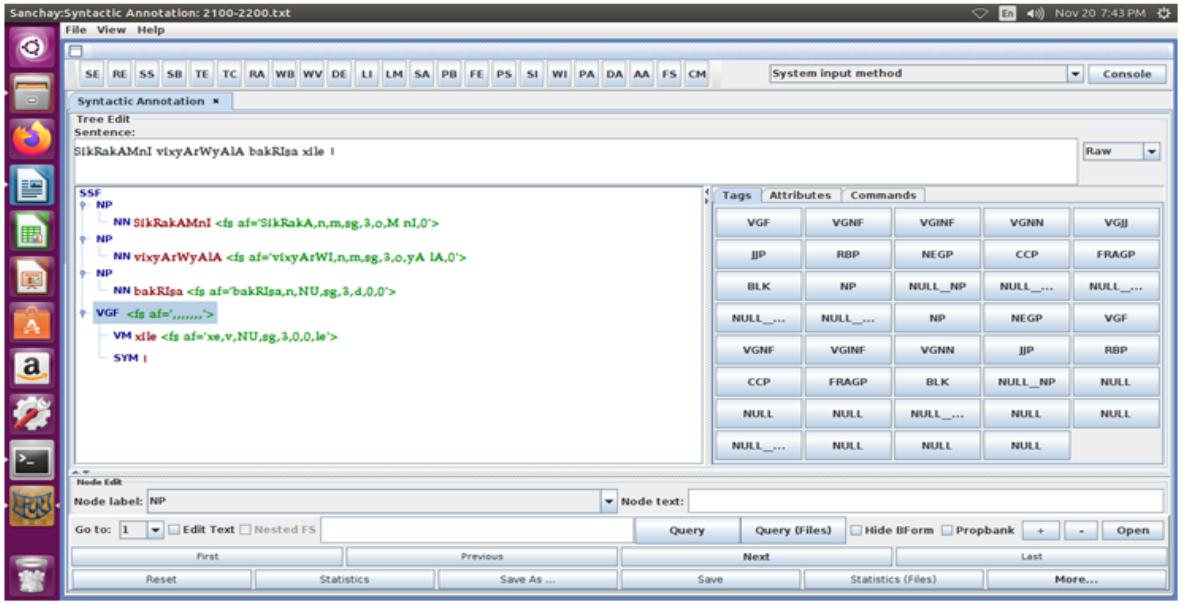


Figure 1 Chunk Annotation Procedure

The above picture demonstrates the chunk annotation process of a Marathi sentence with probable chunks. They are SikRakAMnI as NP – NN, vixyArWyAlA as –NP- NN, bakRIsa as NP- NN and xile as VGF- VM. These chunks encode the chunk features, POS tagging and also morph features.

3.5.2.4 Dependency relations

The final level in the annotation procedure is sowing the dependency relations. This shows the relation between a modifier and a modified at the syntactic level. The modifiers are nouns, and the modified are verbs. This concept of modifier and modified is referred to as karaka relations in the Paninian grammar framework. The dependency annotation scheme also was built on the Paninian grammatical framework. In the dependency annotation scheme, the dependency relations were divided into three types' viz. kaaraka relations, other than kaaraka and non-dependency relations. The following picture depicts the process of annotating karaka relations in the Sanchay tool.

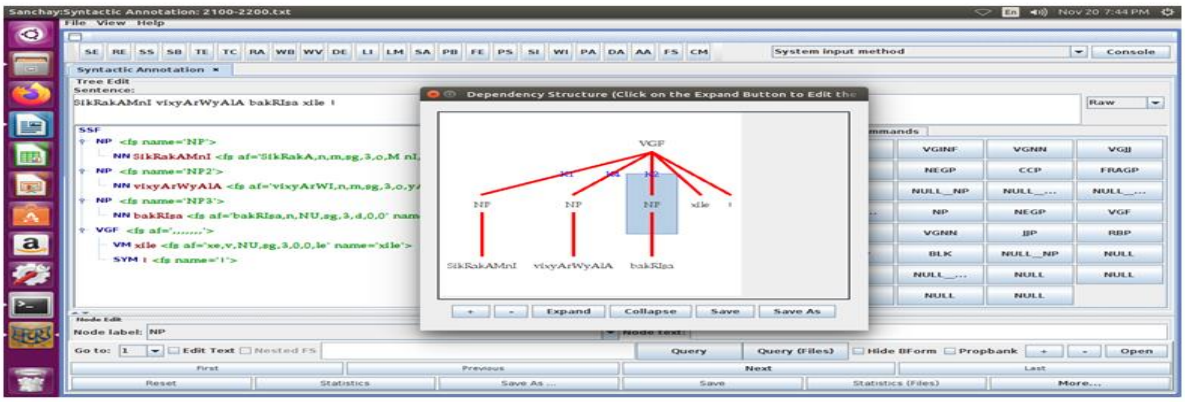


Figure 1 Dependency Annotation Procedure

The above picture describes in detail the annotating process of dependency relations in the Sanchay tool. The relations were marked in the form of tree diagrams. In the tree diagram, the verb (xile) is shown as modified, and nouns SikRakAMnI, vixyArWyAlA, bakRIsa are shown as modifiers. Along with modifier-modified relations, the tree diagram also exhibited the connecting tag sets as k1, k4, and k2 with VGF.

3.6. SSF format

After the procedure was completed in which morph features, POS tag features, chunk features, and dependency relations were implemented step by step, the annotated text was saved automatically in Shakti Standard Format (SSF), which has readable representation properties for storing the language analysis. The format was developed by Bharathi et al.; in 2014. According to Bharathi et al. (2014: 66–76), the representation is specially designed to allow various types of linguistic analyses to be stored. The following encoding format illustrates the SSF format.

```
<Sentence id=''>
1       ((      NP      <fs name='NP' drel='k1:VGF'>
1.1     SikRakAMnI  NN      <fs af='SikRaka,n,m,sg,3,o,MnI,0' name='SikRakAMnI'>
        ))
2       ((      NP      <fs name='NP2' drel='k4:VGF'>
2.1     vixyArWyAlANN     <fs af='vixyArWI,n,m,sg,3,o,yAlA,0'  name='vixyArWyAlA'>
        ))
3       ((      NP      <fs name='NP3' drel='k2:VGF'>
3.1     bakRIsa         NN      <fs af='bakRIsa,n,NU,sg,3,d,0,0' name='bakRIsa'>
        ))
4       ((      VGF     <fs af=',,,,,,' name='VGF'>
4.1     xile    VM      <fs af='xe,v,NU,sg,3,0,0,le'>
4.2     |       SYM     <fs name='|'>
        ))
</Sentence>
```

Figure 1 Illustration of Shakti Standard Format (SSF)

The above encoding format is a structural markup format, which starts with <Sentence id> and closes with </sentence>. The numbers 1.1, 2, 2.1, 3, and 3.1 that are in the Shakti Standard Format represent attribute-value sets with open bracketing and closed bracketing. In sentences 1, 1.1, 2, 2.1 and 3, 3.1, K1, K4, and K2 show dependency relations with VGF. The NP, NP, NP, and VGF, which are positioned before starting of the brackets, represent chunk features. Further, NN, NN, NN, and VM exhibit POS features. Finally, morph features are indicated in the following manner.

<fs af='SikRaka,n,m,sg,3,o,M nI,0' name='SikRakAMnI'>,

<fs af='vixyArWI,n,m,sg,3,o,yA lA,0' name='vixyArWyAlA'>

<fs af='bakRIsa,n,NU,sg,3,d,0,0' name='bakRIsa'>

<fs af='xe,v,NU,sg,3,0,0,le'>

3.7 Summary

To sum up, the present chapter depicts the entire research design used for building the Marathi Treebank. The research design follows systematic procedures, viz., choice of a corpus, annotation scheme, annotation tools and format. At the corpus choice level, how the data was collected from various domains was discussed. Subsequently, how the corpus was cleaned and refined in a user-friendly format and later how it was converted to WX notation, which is very much useful in the CL and NLP, are explained in detail. In the second level, i.e. annotation level, how the dependency annotation scheme was executed, which is built on the Panianian grammatical framework, was discussed at length. In the third level, the execution of the Sanchay tool and manual annotation was discussed in layers especially morph, POS, Chunk and dependency relations. Finally, the chapter ends with how the analyzed data was stored in Shakti Standard Format (SSF) was discussed.

CHAPTER IV

PARSER AND PARSING: A THEORETICAL BACKGROUND

4. Introduction

Parser and parsing are nucleus to natural language processing. As the name implies, the field is closely associated with many other disciplines, viz. natural language processing (NLP), theoretical and computational linguistics (T&CL), and theoretical and practical psychology (T&PP). As the field was good and suitable for NLP, the field has been growing with its own literature and specialists within the field itself. The researcher concentrates only on NLP and T&CL but not on T&PP. The reason for excluding T&PP is that it is more into psychology than NLP.

4.1. Parser and Parsing

The parser is one of the tools in natural language processing which analyzes and tokenizes by verifying lexical items in a given language. Along with tokenizing, it also detects syntax errors and creates a parse tree from which an intermediate system can be generated. According to Hindle (1994: 103-151) "parser is a computer program that can provide a syntactic description of natural language text". Further, he also states that the program depends on the formal grammar of the given language, which tries to recognize the word patterns, phrase patterns, syntactic patterns and syntactic relations in any given sentence. Along with recognition, the program also tries to disambiguate parts of speech, tries to identify the roots from the words and resolves lexical as well structural ambiguity. However, a parser is a combination various software tools, which are used to implement the task of parsing. As the field was growing, another term came into the picture, which is known as parsing. The word parsing is derived from the Latin word 'pars', which means 'part'. According to Pereira (1997: 111-121) parsing is one of the processes of determining and analyzing sentences which have clear relations between constituents that are judged to hold in a given sentence. He also states that constituents are typically defined especially characterized by relationships that hold between their parts. In other words, parsing is simply defined as the process of analyzing the strings of symbols in natural language confirming to the rules of its formal grammar.

4.2. Objectives of the Parser

According to Hindle (1994: 103-151) parser has four main objectives, viz. to report syntactic errors, to recognize part of speech in text, to identify constituency or phrase structure according to the formal grammar and to generate a syntactic tree of sentences. Among the four formal grammar plays a vital and important role in building a parser for a given language. As the parser is built on formal grammar, it tries to judge syntactic structures, parts of speech and whether syntactic trees are correct or

incorrect. So, the formal grammar of a given language is most important in building a parser for a given language.

4.3. Formal Grammar

Formal grammar refers to a set of rules which produces a formal set of rules of any given natural language. The set of produced formal rules will have the properties of mathematical expressions. In the development of NLP and CL many formal grammars have come into existence, viz. context free grammar (CFG), transition network grammar(TNG), functional unification grammar(FUG), lexical functional grammar (LFG), generalized phrase structure grammar(GPSG), head driven phrase structure grammar(HDPSG), and tree adjoining grammar (TAG) etc. For understanding the concepts of formal grammar, the researcher chosen context fee grammar (CFG) to explain. CFG was one of the first formal grammars that set for the production of formal rules in the form of mathematical expression. Backus (1956) had explained the formal production of rules through a particular formula. The formula is executed in the following manner: <ab> : == (or [or <ab> (or <ab>) <d) with a meta-linguistic formula for analyzing any given string of a language. In the subsequent years, Backus (1969) again reformulated it as a formal notation as G = (N,Σ ,R ,S), which can be explained in the following manner.

G (Grammar)

N a set of non-terminal symbols (or variables)

Σ a set of terminal symbols (disjoint from N)

R a set of rules or productions, each of the form A → β ,where A is a non-terminal,

β is a string of symbols from the infinite set of strings (Σ ∪N)∗

S a designated start symbol and a member of N

In the above formal notation of Backus further elaborates the formal notation. In the formal notation 'S' is considered as starting point, '→' is signifies for rewriting the rules, 'Σ' is used for a set of terminal symbols for the lexicon, 'N' is used for non-terminals which are constituent or phrase structure. Finally, 'R' is referred to be a rule production in the formal notation. As part of 'R' rule production derivation can be seen very often. In the derivation, one string generates next string with a '→' rewrite rule. As the rewrite rule applies on the given string of sentences, it generates a systematic process of developing syntactic structure of any given natural language. Though CFG plays a very important and vital role in NLP and CL, it is more affiliated with derivation modelling, which is more

mathematical and covers less linguistic aspects. Since it is more mathematical and covering, less linguistic aspect the concept of the lexicon has come into existence. Subsequently, in the field of NLP and CL, the models like lexicon and other have included the derivation model, which is more mathematical covering linguistic aspects.

4.4. Constraint Base Formal Grammar

In the subsequent years, researchers in the field of NLP and CL tried to combine both the derivation model as well as a lexicon. As a result, a new model has emerged, which is called as constraint based formal grammar (CBFG). Researchers who are working in constraint based formal grammar (CBFG) might use any formal grammars like Functional Unification Grammar(FUG), Lexical Functional Grammar(LFG), Generalized Phrase Structure Grammar(GPSG), Head Driven Phrase Structure Grammar(HDPSG), and Tree Adjoing Grammar (TAG) etc for the syntactic analysis. However, the ultimate aim of the researchers was to analyze syntactic structure of parser under CBFG, which is a mixed model. Since each of the formal grammars mentioned above have their own pros and cons in the analysis of syntactic structures. However in the field of parser and parsing, a new method for analyzing syntactic structures has emerged which is named the machine-learning approach with a stastical method. For the purpose, one has to gather huge number of annotated language texts, which is referred as corpora in corpus linguistics. As the machine-learning and statistical method needs annotated language texts, researchers started collecting language data and annotating them manually and simultaneously with the help few computational tools. As a result, the process of manual annotation of corpora has given rise to the concept of Treebank.

4.5. Treebank

In general, Treebank refers to a corpus, which is manually and linguistically annotated data of a given language. The linguistic analysis takes place at four levels, viz. word, chunking, syntactic and semantic level. At word level parts of speech (POST) tagging takes place. Along with POST, grammatical features like inflection and derivation features of a particular word are studied. At the chunking level, phrases are analyzed, which can be termed as shallow parsing, in which a group of phrases are annotated manually. Further sentence are analyzed by using syntactic properties like subject, direct object, indirect objects, adjunct, complement etc. Finally, at the semantic level, semantic roles of a text are annotated, such as agent, patient, theme and location etc. Abeille (2003: xiii-xix) states that Treebank refers to a linguistically annotated corpus, which provides information about constituent boundaries (clause, NP...), grammatical functions of words or constituents and dependency relation between words and constituents. Further, Nivre (2006: 225-241) defined Treebank as a linguistically annotated corpus that includes some grammatical analysis which goes beyond the part-of-speech level. Subsequently, Hajičová et al. (2010: 167-182) in their informative paper, defined Treebank as structurally annotated corpora that represent syntactic, semantic, and

sometimes even inter sentential relations. Further, they also stated that the word tree refers to the typical structure of the annotation, which corresponds to the notion of "tree" as defined in the formal graph theory. Finally, they stated that the interpretations of edges in the tree differ substantially from other Treebanks, which are used to represent syntactic relations as defined in the annotation design. However, a basic assumption of a Treebank is considered as linguistically annotated corpus that includes part of speech, morphological information, syntactic and semantic representation of a natural language.

4.6. Creation or Design of a Treebank

Creation of a Treebank always depends upon the purpose of the researcher to which he is going to create or design Treebank. Abeille (2003: xiii-xix) states that in designing a Treebank one must consider need follow three steps viz. choosing the corpus, annotation scheme, and annotation tools. Among three, annotation scheme is the pivot and a very important step in the creation or design of a Treebank. Marvelous, it works as backbone in the creation of a Treebank, which also takes into consideration of grammar framework with syntactic structures. In the literature of the Treebanks, there are mainly two types grammar frameworks are available. One is the constituent grammar framework in which constituents and phrase structures are taken into consideration. The other one is the dependency grammar framework in which dependency structures are taken into consideration. Both frameworks helps in building a Treebank for a given language.

4.7. Constituent grammar framework with constituency/phrase structure

For the first time, the term 'constituent' was introduced by Bloomfield 1933. Later the term was used in many linguistic and applied linguistic research areas, especially in syntax. According to Bloomfield (1933: 35), a sentence like '*Poor John ran way*' can be split into two Immediate constituents as (*Poor John) and (ran way*). Further, he also mentions the constituents again can be split into another four individual constituents like (*Poor) (John) (ran) (way)* by representing a sequence of strings in the sentence. Taking a clue from Bloomfield, later researchers in the area proposed the same concept in a different way by using their own notation. In the later stages, Chomsky (1957: 26-35) mentioned in one of the chapters of his book titled "syntactic structures" which highlights majorly about phrase structures. In the chapter, he elaborates about finite state grammar in, which states that constituent structure is similar to that of phrase structure. According to the finite state grammar model, the sentence can be analyzed in the following manner with a particular notation.

S → NP + VP

NP → D + N

VP → Verb + NP

D → The

N → man, ball etc

Verb → took, hit etc

The above notation almost seems to be equal to the notation of Backus reformulated of G = (N,Σ , R ,S). A few researchers in the area states that, Backus borrowed the concept from Chomsky and made it in the form of mathematical formula. Since there is no written evidence that who has borrowed from whom or they have developed these notations independently, one has to find out. According to Chomsky, there are two types of rules, viz. left hand and right hand, which generate syntactic patterns of a given language. In the left-hand rule he mentioned about terminal nodes like, S, NP, VP, D, N and Verb. In the right-hand rule he mentions about the non-terminal notations which also called ad lexicon like the, man, ball, took and hit of a given language. The arrow symbol in the notation indicates the rewriting of the rules in the terminal notation. This notation was also used in context free grammar (CFG) for the syntactic analysis. Chomsky's notation can be seen in the form of constituent/phrase structure format given below with the example, 'The *boy drinks milk* '.

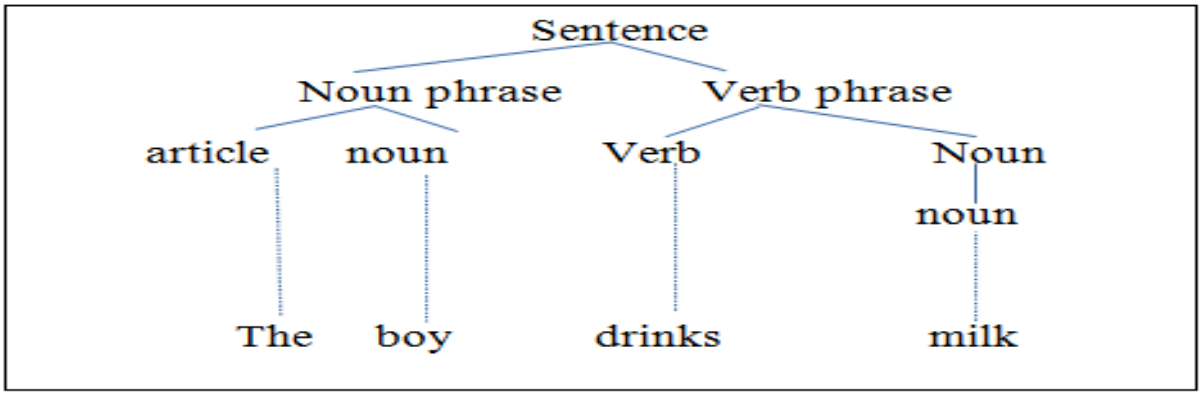


Figure 4.1 Constituent analysis of an English Sentence

In the above constituent analysis of an English sentence, the solid line indicates (edge or branches), and the dotted line indicates (projection). Along with that, the solid line describes as part of whole relations NP, VP and NP. The dotted lines show as many mapping of words like article + noun = NP, V = VP, Noun= NP) and binary divisions like Subject + Predicate. As a result, the constituent grammar framework allows indirect links with syntactic units of the sentences. However, it was Penn Treebank which first utilized a constituent grammar framework with constituency structure. The tree format in Penn Treebank was bracketing representation. The following format illustrates the phenomenon.

```
(S  (NP-SUB The boy)
    (VP drinks
        (NP-OBJ milk))))
```

Figure 4.2 Penn Treebank Bracketing Representation

The above Penn Treebank is in the format of representing bracketing in the constituent grammar framework with constituency structure.

4.8. Dependency Grammar framework and dependency structure

In modern linguistics, Tesnière (1959) discusses about dependency structure from the linguistics point of view. According to him, there will be a direct link between elements of syntactic units, i.e. between words. Further, he also mentions that each connection unites a superior term and an inferior term. The superior term is called as the governor, and the inferior term termed as subordinate'. Hence, the governor is considered as head word, and the subordinate is considered as the dependent word. Later Hudson (1984: 130-131) in one of his books states that the term 'dependency' has a reputable heritage and it has the advantage of attracting the asymmetrical relationship between super-ordinate and subordinate words. After a considerable period of time, Bharati et al. (1995) state that the dependency tradition much older and has its roots in Paninian grammatical framework. According to them, the sentence is considered as a series of modifier-modified relations in which the modifier is dependent and modified is a root word or headword.

Further, the headword is referred to as the main verb. The element, which modifies the verb, is an argument which participates in the action specified by the main verb. In the last decade, i.e. in 2005, other researchers in the area, like Nivre (2005), states that syntactic structures consist of lexical elements linked by binary and asymmetrical relations which are called as dependencies. In the following diagram, one can observe the direct link between elements of syntactic units, i.e. between words where one is head and the other is dependent. The following example, '*The boy drinks milk* 'from English, explains the phenomenon.

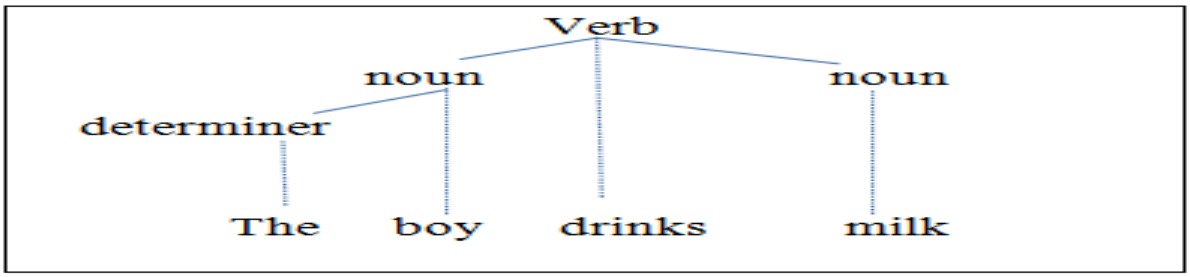


Figure 4.3 Dependency Analysis of an English Sentence

The above diagram represents the dependency relations between words of syntactic units. The solid line indicates (edge or branches), which have a direct link with the words, which is also considered as superior (governor/Head). The dotted lines indicate the inferior (subordinate/dependent) and their relations. In the subsequent research, especially in the field of Treebank, Prague Dependency Treebank applied the dependency grammar framework for the annotation scheme. The new thing observed in the implementation was the use of head-dependent, mother-daughter relations in the sentence. The phenomenon can be observed through the following diagram by taking an English sentence 'The boy drinks milk'.

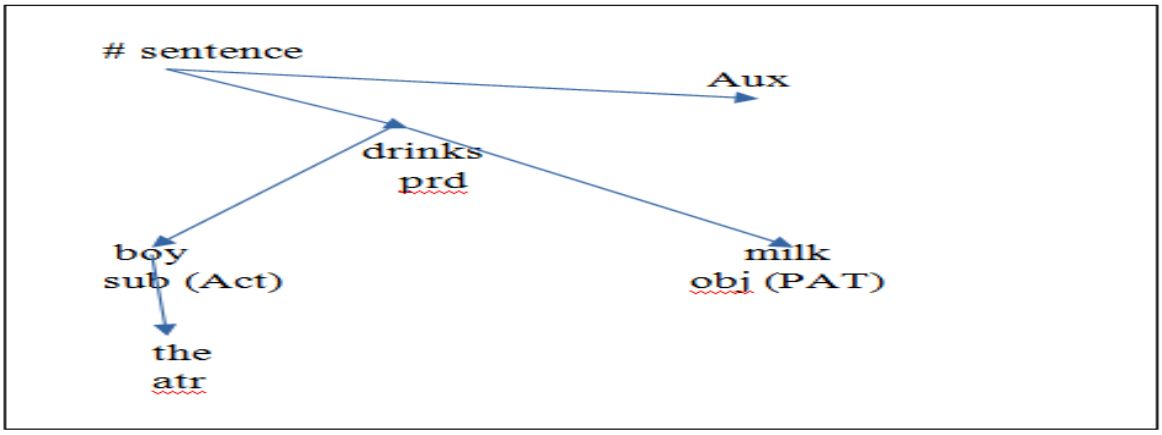


Figure 4.4 Annotation according to Prague Dependency Treebank

The above diagram clearly shows the dependency relations between the headword and the dependent word. The verb acted as a headword and other nouns are seemed to have acted as the dependent of the headword. As it was mentioned, Prague dependency Treebank has three levels in its annotation scheme viz. morphological, analytical and tecto-grammatical. At the analytical level, they annotated the syntactic properties like subject, indirect object, direct object, adjunct etc. At the tecto-grammatical level, they annotated the semantic properties like agent, theme, patient, goal and location etc.

Though there are various types of grammar frameworks are available for annotation schemes to design a Treebank for a given language, the researcher chosen dependency annotation schemefor Marathi Treebank which was constructed based on Paninan grammar framework. As mentioned in the (c.e.f. Bharati. et al. 1995) Paninian grammar would be more suitable for free word order languages. The reason for the application Paninian grammar framework is those languages exhibits rich inflections patterns which will be help in identifying syntactic components like subject, object, complement, adjunct etc. Since Marathi shows rich inflection patterns, the researcher is implementing the Paninian grammar frame work for Marathi.

4.9. A Paninian Grammar framework

Though there are many grammars before Panini, his model has good fame and name. The reason for the name was it was constructed based on syntactic analysis. The work had been and has been a long-standing work which is written in Sanskrit between (350-250 B.C); this framework has given rise to many western and Indian grammar frameworks. According to Kiparsky (2002) Paninian grammar has four components, viz. *Ashtadhyayi:* which discusses 4000 grammatical rules. *Sivasutras:* discusses the inventory of phonological segments, *Dhatupatha:* discusses a list of 2000 verbal roots, with their sub-classifications and diacritic markers which encoded their morphological and syntactic properties and *Ganapatha:* discusses about a list of 261 lexical items which are idiosyncratically subject to certain rules. But, (c.e.f. Bharati. et al. 1995) states that the main goal of Paninian grammar is to explain the utterances of what the speaker decides to say and what the listener extracts the meaning from the speaker's utterances in the speech act participation or communication. However, Panini's model differs from Chomsky's model of grammar framework. Tough Panini's work focuses from the syntactic point of view(c.e.f. Bharati. et al. 1995) focused on the understanding of semantic levels with the help of syntactic information like karaka and vibhikati levels. An excerpt from the Bharati. et al. (1995: XX) illustrates the phenomenon.

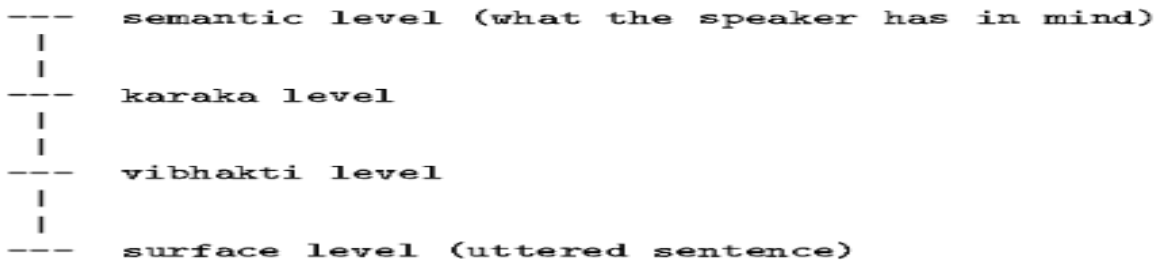


Figure 4.5 Levels in the Panini Model

The above diagram clearly explains the four levels of PGF, viz. surface level, vibhakti level, karaka level and the semantic level. The sequence of the four levels follows as the surface level represents a written or uttered sentence. The vibhakti level exhibits nominal suffixes, and verbal suffixes. Nominal suffixes include case endings, prepositions, postpositions and verbal suffixes include tense, aspect and modality. The third level is karaka level which discusses syntactic units in one hand and the semantics relationship in another hand. In combination of both syntactic and semantic components, they are called with a blend name called syntactico-semantic relations. The fourth level, which is a Semantic level, carries semantic components such as relations between *a vyaapaara* (an activity) and *a phala* (a result). When *a phala* (a result) does not yield good results, with help *vivaksha* can get good results from to speaker's point view. The following examples will give a clear explanation of the Panianian Grammar Framework levels.

a) The boy opened the lock

b) The key opened the lock

c) The lock opened.

In the above examples, example (a) illustrates that the person inserts a key in the lock and turns it. In the example (b) the key on its part presses the levers and moves them and in example (c), the latch, in turn, moves; as a result the lock opens. In the above examples, the boy, the key, and the lock are karta karaka. As discussed in the foregoing discussion, using *a vivaksha* concepts one can see the results respectively. In (a) the speaker decides to give importance to the role of the boy, while in (b) the speaker wishes to emphasize the role of the key, in (c) the speaker emphasizes the fact that the lock has been opened. In the last example, one should not care which key opened it and who opened it is unimportant as far as the utterance is concerned. Subsequently, the concepts of aakaankshaa (Expectation/Demand) and yogyataa (Eligibility) also come under the semantic level in PGF. As mentioned in the foregoing discussion the karaka level has six types of karaka relations, which are described in the following table:

Karaka-relations	Semantic notion	Tag-sets	Description
karata	Subject/Agent/doer/experiencer/force	k1	'the most independent participant in the action'
karma	object/patient/theme/goal/result(of creation).	k2	'most desired to be attained by the karta'
karana	instrument	k3	'instrument which helps in accomplishing the action'
sampradana	beneficiary/recipient	k4	'intended recipient of the object'
apadana	source	k5	'fixed point of departure (or) moving away from a source'
adhikarana	Location in place/ time /other	k7p/ k7t/k 7	

Figure 4.6 Six types of Karaka Relations under PGF Karaka level

The above table represents six types of karaka relations with semantic descriptions. Later (c.e.f. Bharati. et al. 1995) categorized all six types of karaka relations under the noun-verb modification (karaka relations). Along with the noun-verb modification, other types of relations like nominal structure with adjectives-noun modification, verbal structure with the verb as argument of the head verb, nominal structure with participle verb as a modifier of a noun, verbal structure with verb-verb modification and nominal structure with verbal nouns are also discussed.

4.10. Dependency annotation scheme

As the research went on in the Treebank annotation scheme, especially from the dependency perspective, Begum et al (2008: 721-726) discussed the design of a dependency annotation scheme for Indian languages based on the Paninian framework. According to them, there are three types of dependency annotations schemes are there viz. karaka relations, other dependency relations and non-dependency relations. In the subsequent year, Bharati et al. proposed other types full pledged tag-sets for Indian languages like underspecified tags (vmod and nmod), some other tag-sets that indicate relations which does not suit under dependency relations but requires to represent for the syntactic structures (ccoff and prof). Further in 2009 they purposed guideline for dependency annotation scheme for Indian languages. The (DG) guideline carries forty tag-sets depending on various types of sentences. The following picture illustrates the phenomenon of the full pledged dependency annotation scheme.

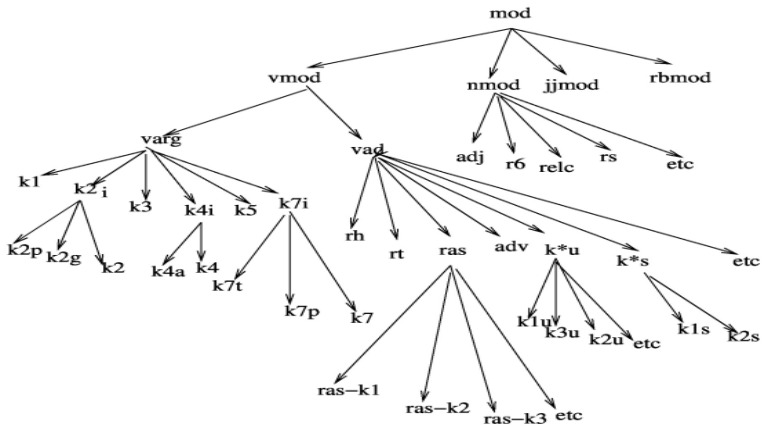


Figure 4.7 Dependency Relation Types Taken from Barathi et.al (2009)

the above diagram demonstrates full pledge dependency relations based on the Paninian framework. The dependency relations can be classified into three categories viz. kāraka relations, other dependency relations and non-dependency relations. In kāraka relations, there are six types of karakas were identified. They are labeled as ‘k’ followed by a numerical like, k1, k2, k3, k4, k5, and k7. Within ‘k’ labelled finer types of kāraka also marked, such as k2p, k1s, k2g, k4a etc. Other dependency relations were labelled as 'r' and subsequently as rh, rt, ras and r6. Under the non-dependency, relations indicate with vmod, nmod, jjmod, ccof, prof etc.

The researcher used a dependency annotation scheme built on the Paninian grammatical framework for annotating Marathi. The following example and diagram represent te dependency annotation scheme for Marathi.

Śikṣakāṃ-nī vidyārthyā-lā bakṣīsa dilē.
Teacher -Erg. Student -Dat. prize gave
The teacher gave a prize to the student

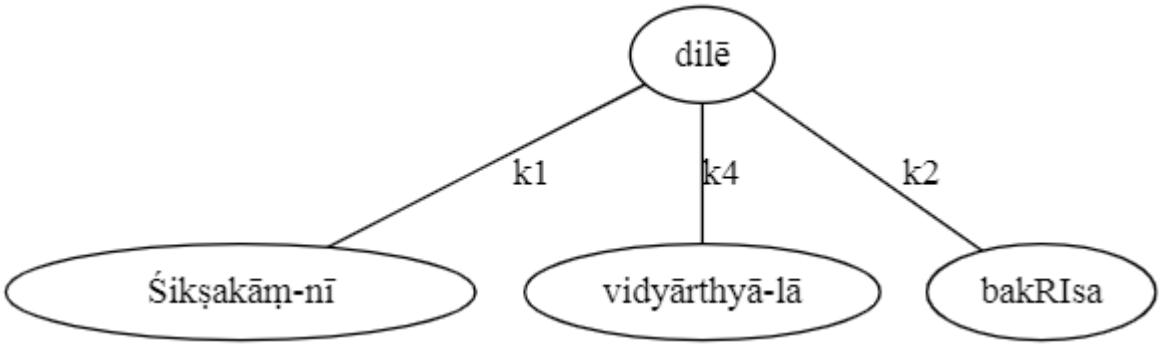


Figure 4.8 Dependency Annotation Scheme for Marathi has taken form Bharati et al. (2009)

The above diagram demonstrates kaakara relations marked by dependency relations for Marathi. The k1 is represented as kartā, (Subject) attached to the verb *dilē*, k2is represented as karma (Object) attached to the verb and finally, k4 as sampradāna (recipient/indirect object) attached to the *verb dilē*.

4.11. Summary

The chapter mainly deals with the development of the parser from various perspectives like formal grammar, constraint-based grammar framework and linguistic point of view. Further, it was discussed about how researchers in the area of formal grammar made use of derivational formulas in developing parser, which were suitable for Positional languages, especially like English was discussed at length. In addition, how researchers used the derivational formulas along with inserting lexicon in developing a parser for a particular language also talked about. Subsequently, from the Treebank perspective, how statistical methods were introduced, which were implemented for building the parser, was argued. Finally, though there are various types of Treebanks available, how the researcher adopted the dependency annotation scheme for Marathi, which was built on the Paninian grammar framework was conversed.

CHAPTER V

THE DEPENDENCY ANNOTATION SCHEME AND HAND-CRAFTED RULES FOR MARATHI

5. Introduction

The present chapter discusses the various issues involved in developing a dependency annotation scheme for building a Marathi parser. The data is analyzed based on the Paninian dependency grammar. This framework is chosen as the working model because a lot of earlier works attest to the fact that Indian languages being morphologically complex with a relatively free word order are best accounted for by the Paninian dependency grammar. (Bharati, et.al, 1993; 1995b; 2002; Rafiya Begum et.al, 2012). While constituent structure grammar is suited for languages like English where relations are hierarchically organized and positions determine the syntactic and semantic relations between the constituents in a sentence it cannot account for Indian languages where it's not position but morphological mapping that determines the syntactic functions. Flexibility in word order yields flat structures that are best accounted for by the Paninian dependency model.

Paninian Dependency Grammar is characterized by the following (Mishra, Dipti 2008):

The formalism involves two levels of analysis:

- A deep level of semantic relations that comprise: Direct participants of the action denoted by a verb (karaka) and other relations such as purpose, genitive, reason, etc.
- A surface syntactic level that has Relation Markers (vibhakti).

The model works in the following manner:

- It treats a sentence as a series of modifier-modified relations
- A sentence has invariably a primary modified element (generally a verb)
- Provides a blueprint to identify these relations
- Syntactic cues help in identifying the relation types.

In the following sections, we discuss the mapping of the various karaka relations in Marathi.

5.1 kāraka/Dependency relations:

karaka relations can broadly be defined as the relations that hold between verbs and the various arguments in a sentence. Every verb root denotes an 'action' carried by certain participants. The direct participants in an action denoted by the verb are the agent, the one who performs the act, the Karta, and the other, the affected or the karma. And then there are other participants who/which are indirectly involved in the verb's action. Based on this there are direct and indirect participants in a verb's action.

Since the grammatical model captures syntactico-semantic relations between the different elements in a sentence, these relations (karaka-relations) are represented by various tag labels, which can broadly be divided under two different heads: k-Relations and non-k-Relations or R-relations. (Rafiya Begum et.al. 2012). The data has been annotated using a tag set of nearly 28 tags based on the bifurcation into k and non-k-Relations. And there are other relations like genitive, purpose, reason, goal or destination, time, etc which need to be tagged. These are tagged under the non-K -Relations. Though they do not directly participate in a verb's action their semantic contribution is nonetheless very significant to the sentence's meaning.

The following is a list of the tags used for annotation purposes in the present work:

k1	***kartā kāraka* (doer/agent/subject)**
k2	***karma (object/patient)***
k3	***karaṇa kāraka (instrument)***
k4	***sampradāna kāraka (recipient)***
k4a	***anubhav kartā (Dative/Experience subject)***
k5	***apādāna kāraka (source or separation)***
k7t	***kālādhikaraṇa (time)***

k7p	***deśādhikarana (space)***
k7	***visayādhikarana (location elsewhere)***
rh	***hētu (reason)***
rt	***tādarthya (purpose)***
r6	***shashthi relations (genitive/possessive)***
k1s	***kartā samānadhikaraṇa (noun complement of kartā)***
k2s	***karma samānadhikaraṇa (object complement)***
k2p	***(Goal or Destination)***
pk1	***prayojaka kartā (causer)***
jk1	***prayojya kartā (causee)***
mk1	***madhyastha kartā (mediator causer)***
ccof	***co-ordination and sub-ordination conjuncts***
pof	***Conjunct Verbs***
nmod	***Participles etc modifying noun***
vmod	***verb modifier***

Table 5.1 Annotations tags

Based on the models proposed by Akshar Bharati (1996), Dipti Mishra et al. (2012) and Rafiya Begum et al. (2012), efforts have been made in the present study to provide guidelines for building a Marathi parser.

The following sections illustrate the designated features of a particular *kāraka* relation and the cues for identifying them with reference to Marathi.

5.1.1 Mapping k-Relations in Marathi:

5.1.1.1 -k1: *kartā kāraka* (doer/agent/ subject)

The *kartā kāraka* is characterized by the following features: (Bharati et al., 1995).

a) The '*kartā kāraka*' is the most independent' of all the *kāraka* (participants). [+independent]

b) The one who carries out the action. [+action]

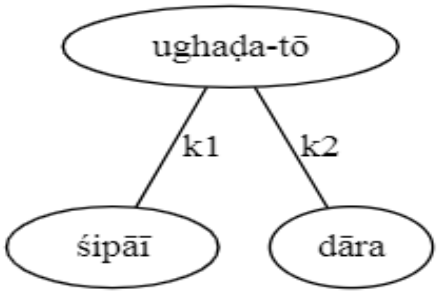
c) The locus of the activity implied by the verb root. In other words, the activity resides in or springs forth from the '*kartā*'. [+locus]

d) The '*kartā kāraka*', by not having the feature [+volitionality] in some instances, is conceptually different from the agent theta role.

The *kāraka* relations of k1 and k2 (*kartā* and *karma*) do not always correspond to the 'agent' and 'theme' roles due to the fact that the thematic roles of 'agent' and 'patient/theme' are purely semantic in nature. In contrast, the Paninian notions of '*kartā*' and '*karma*' are both semantic and syntactic in nature. (Bharti et.al.1995).

Consider the following:

1. **śipāī (k1)** dāra ughaḍa-tō.
   policeman-nom-m3sg door-acc-n3sg open-present-m3sg
   The Policeman opens the door. .



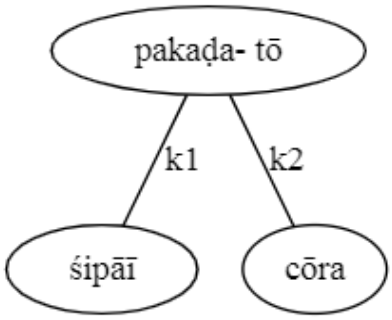
For example (1) the NP śipāī (Policeman) exhibits features (a), (b) and (c), i.e. it is the most independent of all the *kartā*, the agent of the action and the locus of activity which clearly indicates the fact that it is k1.

This identification is purely on the semantic basis and going by the syntactic cue, the NP with which the verb shows agreement for gender, number and person is the k1. In (1) the verb agrees with śipāī 'policeman', hence identified as k1.

Similarly, consider another sentence similar to the above:

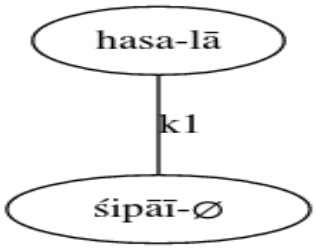
2. **śipāī (k1)** cōra pakaḍa- tō.
Policeman-nom-m3sg thief-acc-m3sg catche-present-m3sg
'The Policeman catches the thief.'

The k-relations are evident from the following tree diagram:



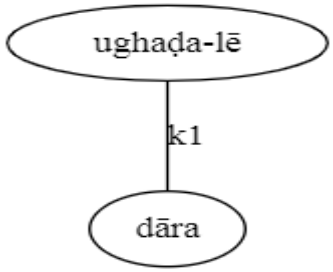
In both (1) and (2) the verb is a transitive verb. Even in the case of an intransitive verb, k1 is mapped in a similar fashion. As long as the NP to be considered as a candidate for k1 satisfies conditions (a), (b), and (c) it is deemed to be k1. Consider the following example:

3. **śipāī (k1)** hasa-lā
policeman-nom-m3sg laugh-past-m3sg
'The Policeman laughed'.



However, it is not always that all three conditions (a b & c) are satisfied. Sometimes the NP tagged k1 may not possess a feature like [+volitional] but still may be the locus of action, as in the following:

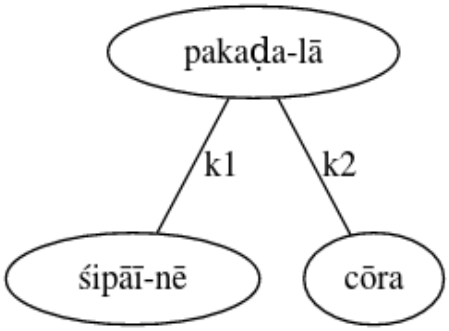
4. **dāra (k1)** ughaḍa-lē.
door-nom-n3sg open-past-n3sg
'The door opened.'



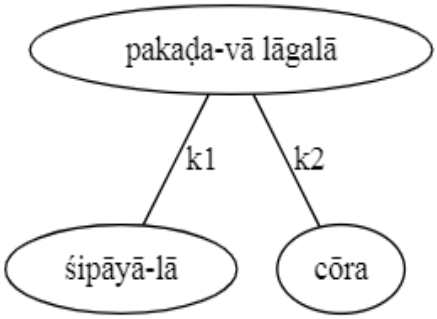
In the above sentence, the NP dāra ‘door’ by being [-animate] is also [-volitional] but still functions as k1. This is due to the verb semantics. In the case of unaccusative verbs, the grammatical subject of the sentence is not a semantic agent. Such kartā are designated as secondary kartā. (cf. Rafia Begum et.al. 2012).

k1 is not marked morphologically in most cases, but consider the cases where k1 is marked:

5. **śipāyā-nē (k1)** cōra pakaḍa-lā.
policeman-erg-m3sg thief-acc-m3sg catch-past-m3sg
‘The Policeman caught the thief.’

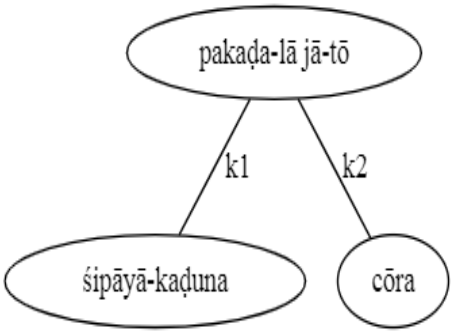


6. **śipāyā-lā (k1)** cōra pakaḍa-vā lāgalā.
policeman-dat-m3sg thief-acc-m3sg catch-inf. had
‘The Policeman had to catch the thief.’



7. **śipāyā-kaḍuna (k1)** cōra pakaḍa-lā jā-tō
policeman-by thief-m3sg catch-past-m3sg go-pass-m3sg

‘The thief was caught by the Policeman.’



In the sentences above (5, 6 and 7) NPs (*kartā kaarka*) have appeared with various kinds of *vibhikati* (case markers). However, for this, we can make the following syntactic clues:

a) When the verb is in the past tense, the k1 is marked by -ne, the ergative marker. The ergative marker blocks the agreement of the verb with the subject NP.

b) The verb in cases where Karta is blocked for agreement with the verb because of the intervening ergative case marker does not show agreement with the object NP either. The verb root shows a neutralized agreement.

c) When the verb is in infinitive past marker, the NP carries -lā marker.

d) When the verb is in passive, the subject is marked with kaḍuna ‘by’ the passive morphology.

e) In either case the verb does not agree with the Karta for Gender, number, and person.

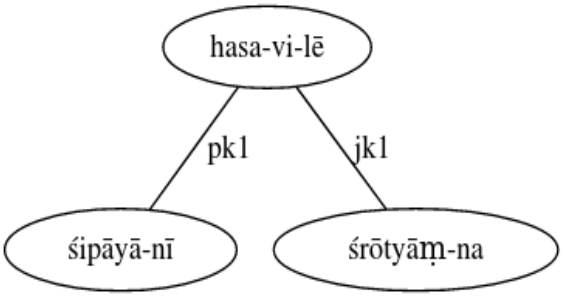
Using ontological features such as [+/- volitional], and syntactic properties of the verb such as transitivity, valency, voice, etc. the morphological marking on k1 varies.

5.1.1.1.1 prayojaka kartā (causer) [pk1]

A causative is a valency-changing operation that indicates that the subject either causes someone else or something to do or brings about a change in the state of a non-volitional event. In Marathi, like in most other Indian languages causative is marked morphologically on the verb as in *hasa-vi-lē* ‘to cause to laugh’ (to make someone laugh’. The verb argument structure is reconfigured by adding a causer, cause, and effect.

This is illustrated in the following:

8. **śipāyā-nī (pk1)** śrōtyāṃ-na hasa-vi-lē.
Policeman-cause-m3sg audience-causee-n3pl laugh-cause-past-n3pl
'The Policeman made/caused the audience to laugh.'



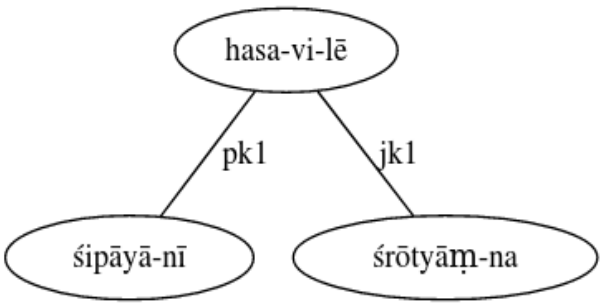
In the above example (8), 'hasa-vi-lē' (laugh) is a causative verb that needs two arguments, namely causer and causee. They are NPs 'śipāyā-nī' (policemen) and śrōtyāṃ-na (audience), respectively. Further, NP 'śipāyā-nī' is a causer that takes _nī postpositions in the Marathi. Using the Paninian terminology the causer is termed as prayojaka karta pk1.

5.1.1.1.2 prayojya kartā (causee) [jk1]

As mentioned in the foregoing discussion, a causative verb requires causer and causee relations to fulfil the meaning of the sentence.

The following example illustrates the marking of causee:

9. śipāyā-nī **śrōtyāṃ-na (jk1)** has a-vi-lē.
policeman-(causer)-m3sg audience-(causee)-n3pl laugh-(cause)-past-n3pl
'The Policeman made/caused the audience to laugh'.

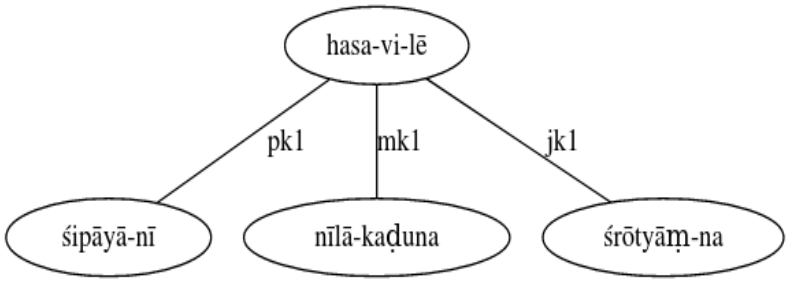


In example (9) NP 'śrōtyāṃ-na'(audience) syntactically occurs in the object position and semantically appears as causee of the causative verb. Due to this fact, we mapped NP 'śrōtyāṃ-na'(audience) with jk1 tagset, which carries _na postpositions.

5.1.1.1.3 madhyastha kartā (mediator causer) [mk1]

It is possible that causative verb can be carrying two causers in a sentence. For the second causer, the scheme has developed (mk1) tagset to mediator causer. See the following example:

10. śipāyā-nī **nīlā-kaḍuna (mk1)** śrōtyāṃ-na hasa-vi-lē.
policeman-causer-m3sg neela-by-f3sg audience-causee-n3pl laugh-cause-n3pl
'The Policeman caused Neela to make the audience laugh.'



In the above example (10) śipāyā-nī (Policeman) śrōtyāṃ-na (audience) NPs are causer and causee of causative verb hasa-vi-lē (laughed). Further, nīlā-kaḍuna is the second causer of the causative verb hasa-vi-lē (laughed). It takes _kaḍuna and _dvārē postpositions.

Syntactic clues for identifying pk-1, jk-1 and mk-1:
A causative verb can be identified by using the morphological marker -vi affixed to the verb stem.

a) The causer pk1 carries -nī postposition.
b) The causee jk1 will take -na postposition.
c) The Second causer mk1 will take -kaḍuna and -dvārē postpositions.
d) The causative verb stem always carries the suffix -vi.

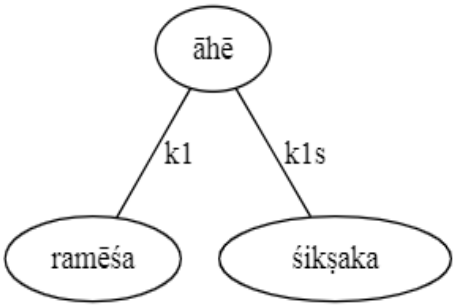
5.1.1.1.4 kartā samānadhikaraṇa (noun complement of kartā)[k1s]

The noun complement of k1 is marked as k1s kartā samānadhikaraṇa. It indicates having the same locus as the kartā. Generally, the verb in such constructions functions as a copula.

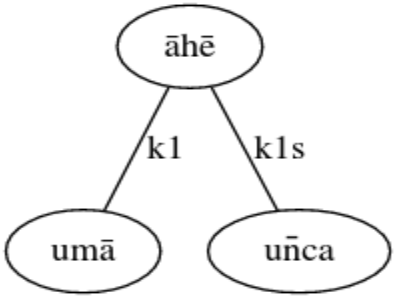
See the following examples:

11. ramēśa **śikṣaka (k1s)** āhē

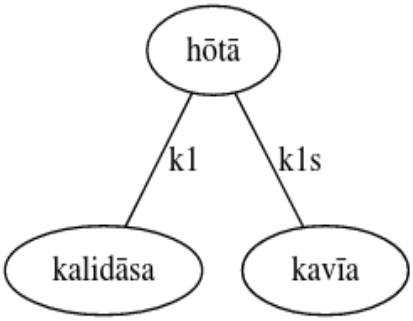
ramesh teacher be-present
'Ramesh is a Teacher.'



12. umā **uñca (k1s)** āhē
uma taller be-present
'Uma is taller.'



13. kalidāsa **kavī (k1s)** hōtā.
kalidas poet be-past
'Kalidas was a poet'



In the preceding examples (11,12 and 13), śikṣaka, uñca, and kavī express the same locus as that of nouns ramēśa, umā, and kalidāsa, respectively.

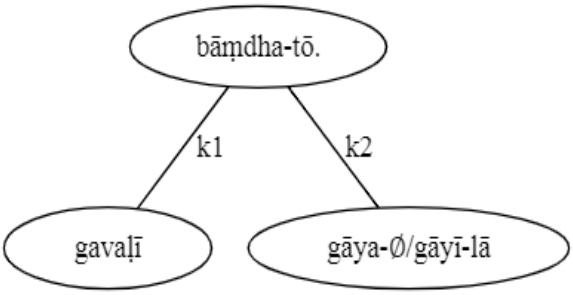
k1s can be either another noun or an adjective qualifying k1.

5.1.1.2 karma (object/patient) [k2]

The karma kāraka, k2 denotes the locus of the result implied by the verb. This can be roughly equated to the object or patient thematic roles. The following examples illustrate karma (object/patient) phenomena.

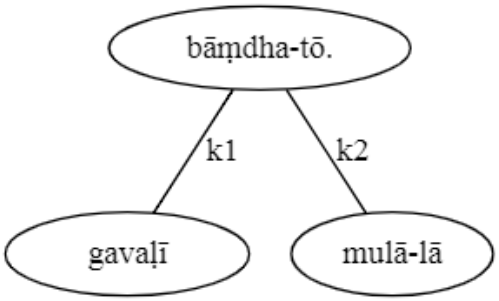
k2 -la (optional)

14. gavaḷī **gāya-Ø / gāyī-lā (k2)** bāṃdha-tō.
cowman-nom-m3sg cow-acc-f3sg tie-present-m3sg
'Cowman ties the cow'.

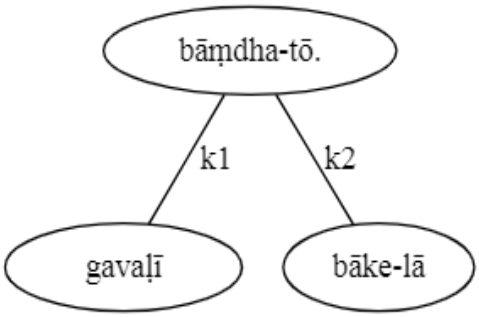


k2-la (obligatory)

15. gavaḷī mulā-lā bāṃdha-tō.
cowman-nom-m3sg boy-acc-m3sg tie-present-m3sg
'Cowman ties the boy.'



16. gavaḷī bāke-lā bāṃdha-tō.
cowman-nom-m3sg table-acc-n3sg tie-present-m3sg
'Cowman ties the table.'



In the above set of examples (14, 15 & 16) it can be observed that in the case of an NP marked for the features [+Human] as in mulā-lā and [-animate] as bāke-lā in are mandatorily marked for the case.

However, in the case of nouns that are [+animate, -Human] the case marker is optional.

Case Resolution:

17. mulagā pushtak vācha-to
    boy-nom-m3sg book-n3sg read-m3sg
    'The boy reads book'

However, an issue of case resolution arises in cases where both K1 and K2 are unmarked for the case. When both the NPS are unmarked for the case, an animate NP takes precedence over the inanimate one in being treated as K1 (karta) following the animacy scale:

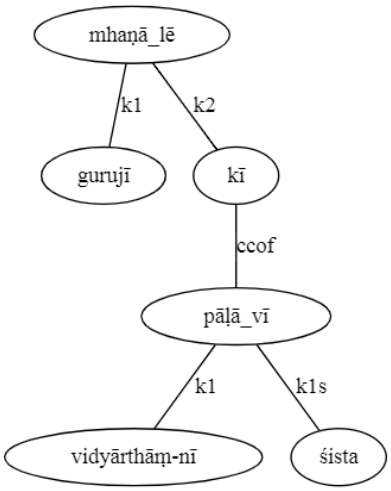
Animacy scale: human > animate > inanimate

vakya karma (sentential object):

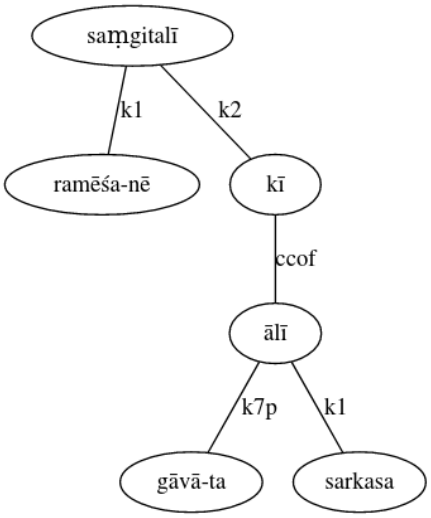
The object in a sentence can either be a phrase or a clause.

Consider the following examples:

18. gurujī mhaṇālē **kī vidyārthāṃ-nī śista pāḷāvī. (k2)**
    teacher tell-past that student-erg discipline follow-past
    ' The teacher told students to follow the discipline.'



19. ramēśa-nē saṃgitalī **kī gāvā-ta sarkasa ālī. (k2)**
    ramesh-ege inform-past that village-pp circus come-past
    'Ramesh informed that the circus has come in the village.'

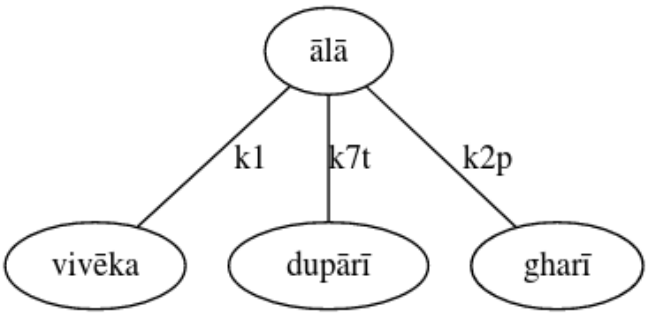


5.1.1.2.1 k2p (Goal or Destination)

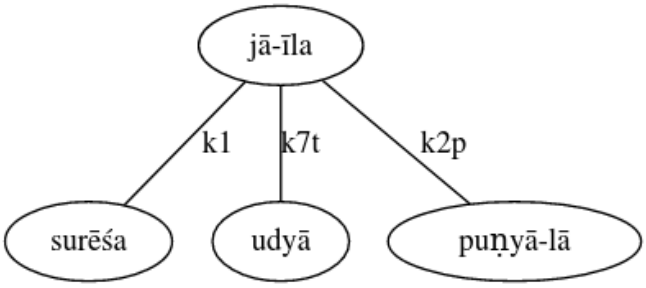
k2p is a subtype of karma (k2). The objects of motion verbs that denote the goal or destination where the action of motion ends are k2p. They also occur with other types of verbs. The syntactic behaviour of k2p is slightly different from other k2. It is also a subpart of karma. Unlike other karma, the goal/destination karma does not agree with the verb

The following example describes the goal or destination.

20. vivēka dupārī gharī **(k2p)** ālā
    vivek afternoon home come-past
    'Vivek came home in the afternoon.'



21. surēśa udyā **puṇyā-lā (k2p)** jā-īla.
    Suresh tomorrow Pune-pp-to go-present
    'Suresh will go to Pune tomorrow.'

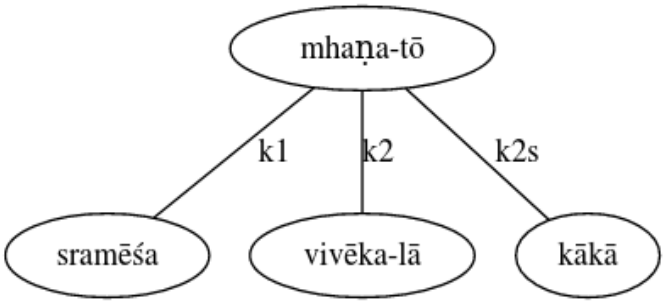


5.1.1.2.2 karma samānadhikaraṇa (object complement) [k2s]

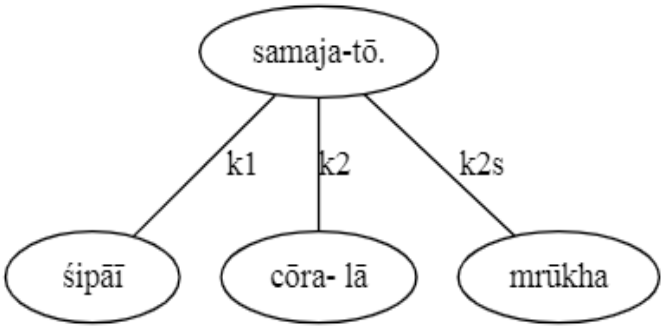
The karma samandhikaraṇa refers to an object complement in the scheme. It is considered as a finer type of karma.

Consider the following examples:

22. ramēśa vivēka-lā **kākā (k2s)** mhaṇa-tō.
ramesh-nom-m3sg viveka-acc uncle call-present-n3pl
'Ramesh calls Vivek as an uncle.'



23. śipāī (k1) cōra- lā **mrūkha (k2s)** samaja-tō.
policeman-nom-m3sg thief-acc-3sg foolish consider-present
'The Policeman considers the thief to be foolish.'

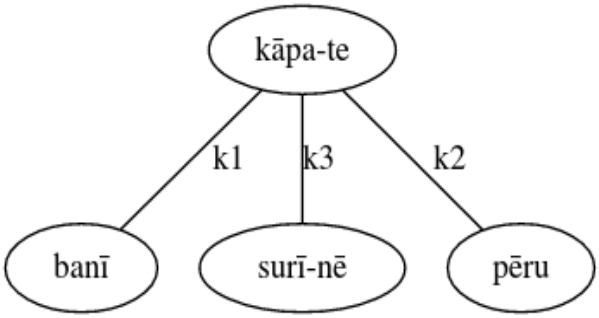


In the above examples, kākā (uncle) and mrūkha (foolish) NP function as karma samandhikaraṇa (object complement) of NP object vivēka and cōra- lā respectively. In the case of communicative and perception verbs, k2s is available.

5.1.1.3 karaṇa kāraka (instrument) [k3]

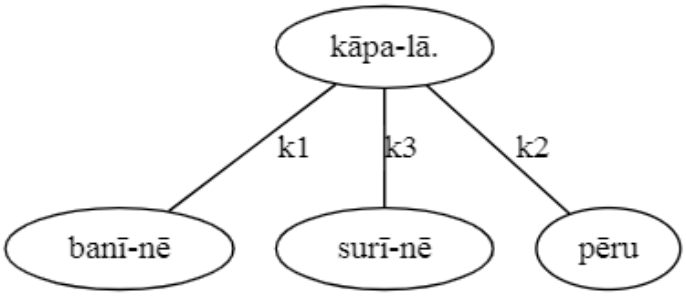
The karaṇa kāraka denotes the instrument of action expressed by a verb root. It helps in achieving the activity of the main action of the verb root. This relation is tagged as k3. The following example shows that having the karaṇa kāraka phenomenon:

24. banī **surī-nē** pēru kāpa-te
banī-f3sg knife-with guava-acc-m3sg cut-present-n3
'Bani cuts guava with a knife.'



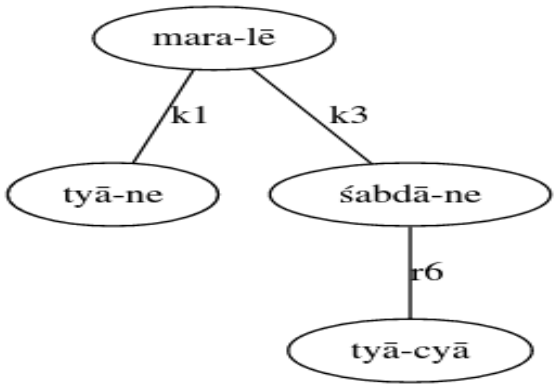
Consider yet another case where both k1 and k3 are marked alike by a case marker -ne:

25. banī-nē **surī-nē (k3)** pēru kāpa-lā.
banī-erg-f3sg knife-with guava-acc-m3sg cut-past-m3sg
'Bani cut a guava with a knife'



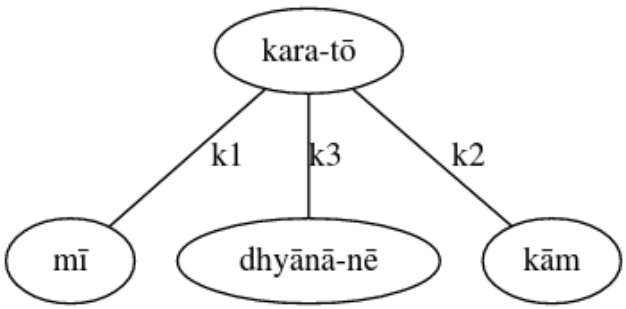
This calls for case resolution. The condition of animacy hierarchy holds here as well. The NP with the ontological features [+human] [+volitional] dominates over the one with [-animate] and marked as kartā kāraka. The other NP is the karaṇa kāraka.

26. tyā-nē tyā-cyā **śabdā-nē (k3)** mara-lē.
he-ege his-gen word-with kill-past
'He killed with his words.'



In the above sentence (śabda) 'words' is affixed with -nē karaṇa kāraka, despite not being a physical object, an instrument. But it conveys an instrument's meaning and is marked with -nē the karaṇa kāraka.

27. mī **dhyānā-nē (k3)** kām kara-tō.
I attention-with work do-present-1sgm
'I do work with attention.'



In the above sentence the NP dhyāna 'focus' is marked with -nē the karaṇa kāraka. Here -nē functions as a manner adverbial.

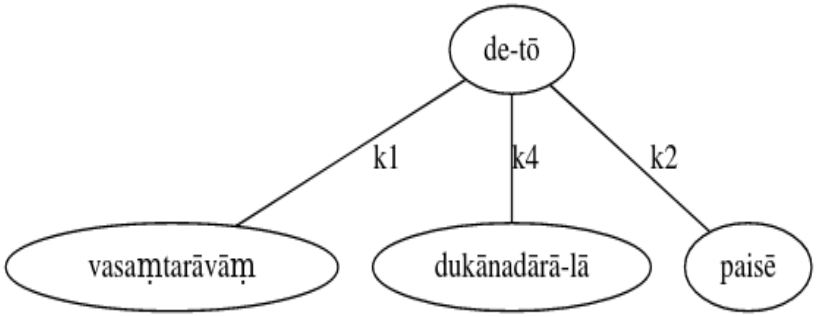
So the two examples (26, 27) cited above are instances of case syncretism.

karaṇa kāraka always takes -nē as the case marker in Marathi.

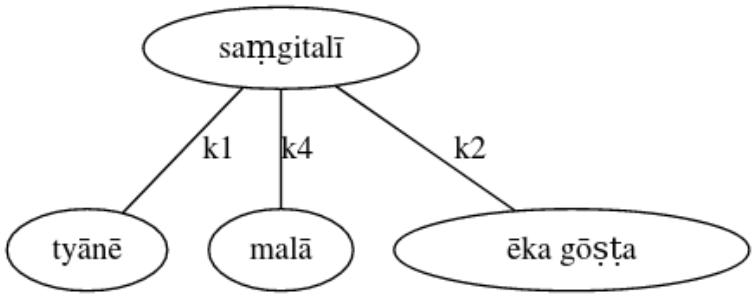
5.1.1.4 sampradāna kāraka (recipient) [k4]

The sampradāna kāraka denotes the recipient/beneficiary of an action in the sentence. sampradāna (beneficiary) is tagged k4. The suffix -lā functions as the dative marker in Marathi. Consider the following sentence:

28. vasaṃtarāvāṃ **dukānadārā-lā (k4)** paisē de-tō.
vasantarao shopkeeper-dat money-n3pl give-past-n3pl
'Vasantarao gave the money to the shopkeeper.'



29. tyānē **malā (k4)** ēka gōṣṭa saṃgitalī.
he-erg I-dat one story-f3sg tell-past-f3sg
'He told me one story.'



In the above examples, dukānadārā-lā and malā NP are the recipients and beneficiaries of the verb's action.

The following cues help identify the NP marked for sampradāna kāraka in a sentence:

a) The verb is a ditransitive verb
b) The NP usually carries _lā case marker.
c) The verbs generally belong to the semantic domains of 'give/ take' and 'communication.'

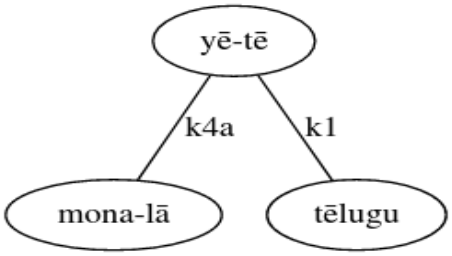
5.1.1.4.1. anubhav kartā (Dative/Experience subject) [k4a]

Here, we deal with another type of kartā or agent, viz. anubhava kartā. Dative or experiencer subject is a typical feature of South Asian languages, for which Marathi is not an exception.

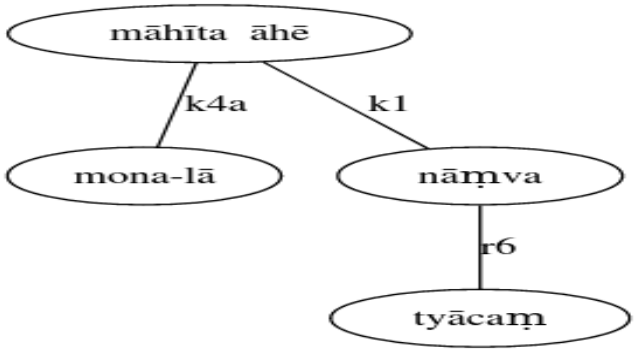
anubhava kartā refers to the passive subject i.e., the experiencer who is not making any effort but just receiving or perceiving the activity carried out by the verb. Since the passive participation of perceiving is that of a recipient, it has been placed under sampradāna here. The anubhava kartā can be equated with a dative subject. (Rafiya Begum et.al. 2012)

Consider the following examples:

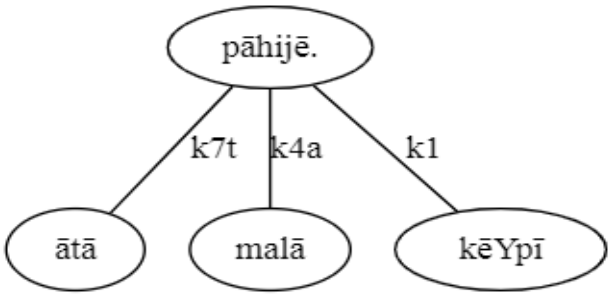
30. **mona-lā (k4a)** tēlugu yē-tē.
    I-dat telugu come-present
    'Mona knows Telugu.'



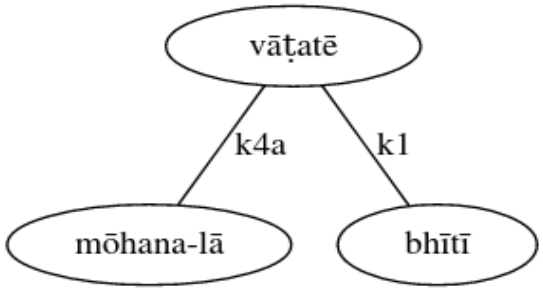
31. **mona-lā (k4a)** tyācaṃ nāṃva māhīta āhē.
    mona-dat His name know is
    'Mona knows his name.'



32. ātā **malā (k4a)** kēYpī pāhijē.
    now I-dat coffee want-present
    'Now I want coffee.'

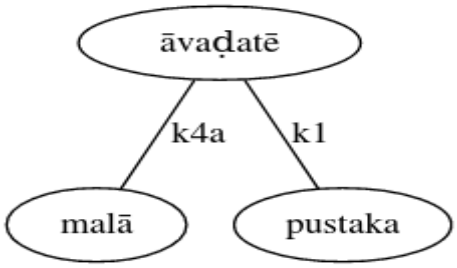


33. **mōhana-lā (k4a)** bhītī vāṭa-tē.
Mohan-dat fear-f3sg feel-present-n3sg
'Mohan feels scared.'

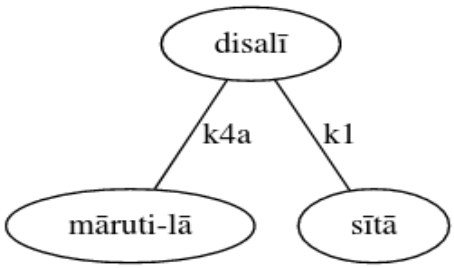


In the above examples, NPs malā and mōhana-lā function as experiencer subjects marked for dative case marker _ -lā. .

34. **malā (k4a)** pustaka āvaḍatē.
I-dat book-N3sg like-present-n3sg
'I like the book.'



35. **māruti-lā (k4a)** sītā disalī.
maruti-dat sita-nom-f3sg appear-past-f3sg
'Maruti saw Sita.'



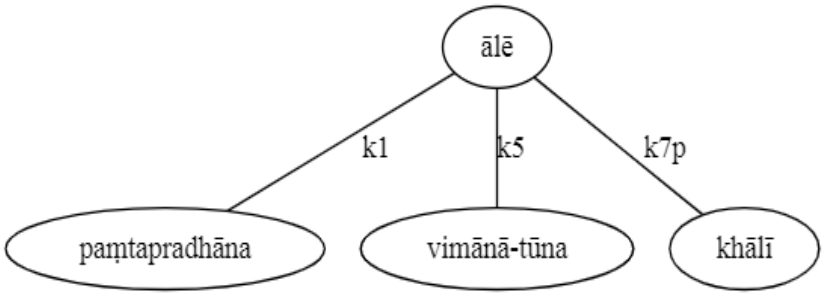
Generally in the case of verbs of cognition, and perception, and in those denoting psycho-somatic states, the subject is an experiencer subject.

5.1.1.5 apādāna kāraka (source or separation) [k5]

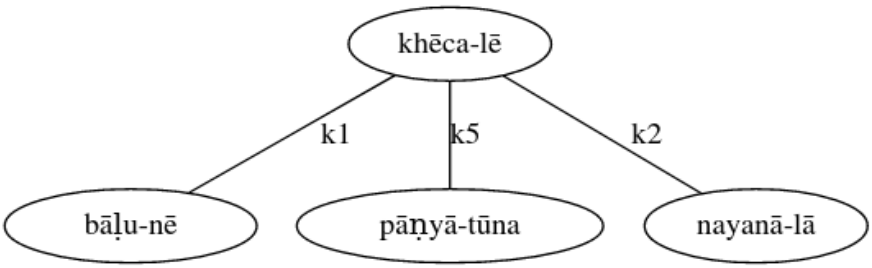
The apādāna kāraka indicates the source of action, i.e. the point of departure. In the case of a verb denoting an action that involves movement 'away from' the noun that denotes the point of separation is apādāna. In other words, the participant which remains stationary when the separation takes place is marked k5.

The following sentences illustrate this:

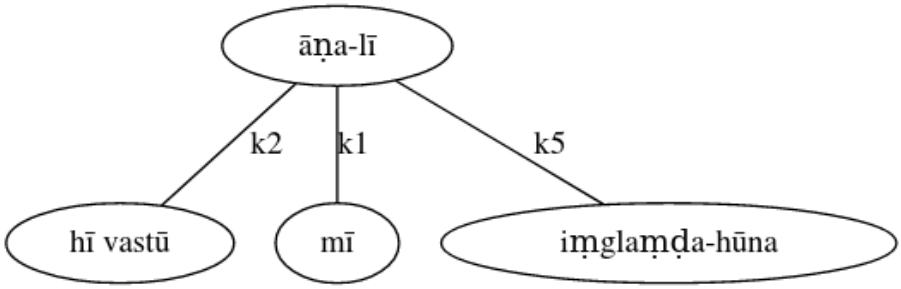
36. paṃtapradhāna **vimānā-tūna (k5)** khālī ālē.
Prime-minister aeroplane-from down come-past-m3sg
'The Prime-minister came down from an aeroplane.'



37. bāḷu-nē **pāṇyā-tūna (k5)** nayanā-lā khēca-lē
balu-erg water-from nayana-dat pull-past-n3sg
'Balu pulled Nayana from the water.'



38. hī vastū mī **iṃglaṃḍa-hūna (k5)** āṇa-lī.
This item-f3sg I England-from bring-past-f3sg
'I brought this item from England.'



The above examples demonstrate that NPs vimānā-tūna, pāṇyā-tūna and iṃglaṃḍa-hūna indicate the source of action, and are marked with respective postpositions.

5.1.6 adhikaraṇa kāraka (time/space)

Adhikaran kāraka is the locus of *kartā* or karma. It is what supports, in space or time, the *kartā* or the karma. The adhikaraṇa NP denotes time and space activity in the sentence. In other words, it is the locus of kartā and karma kāraka. This case relation can be classified under the following heads:

kālādhikaraṇa (time)

deśādhikarana (space)

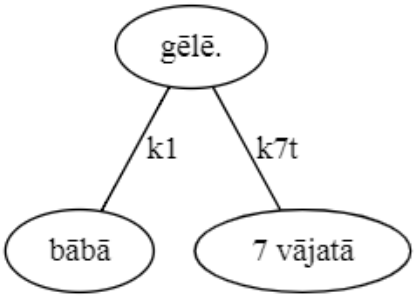
visayādhikarana (location elsewhere)

5.1.6.1 kālādhikaraṇa (time) [k7t]

kālādhikaraṇa indicates the time of action denoted by the verb root in the sentence. The participant denoting the time of action is marked as 'k7t'.

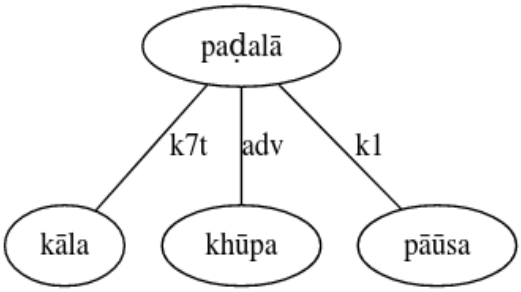
Read the following example:

39. bābā **7 vājatā (k7t)** gēlē.
Father-nom-m3sg 7 clock go-past-m3sg
'Father left at 7'0 clock.'

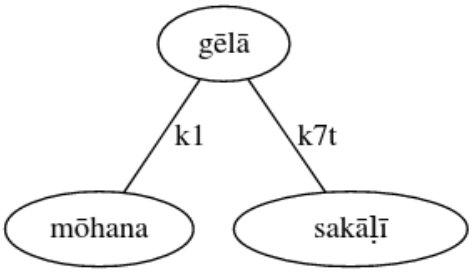


In the above example, '7 vājatā' indicates the time of action. Therefore it is marked as k7t.

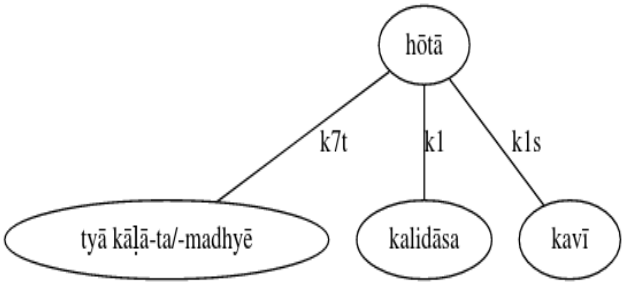
40. **kāla (k7t)** khūpa pāūsa paḍalā.
yesterday too much rain fall-past-m3sg
'Yesterday, it rained heavily.'



41. mōhana **sakāḷī (k7t)** gēlā.
Mohan morning go-past-m3sg
'Mohan went in the morning.'



42. **tyā kāḷā-ta/-madhyē(k7t)** kalidāsa kavī hōtā.
those period/day-pp kalidas-nom-m3sg poet be-past-m3sg
'Kalidas was the poet in those days.'



All the preceding examples illustrate that kāla, sakāḷī, tyā, and -ta are expressions of time. Therefore they are tagged as k7t.

The following cues help identify kālādhikaraṇ relation:

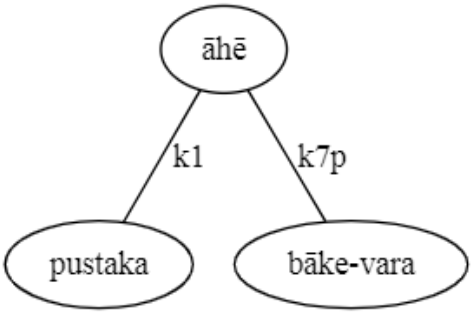
a) In the case of nouns denoting time/ temporal adverbs case marking is optional.
b) To indicate the exact 'point of time' for example. 7 vājatā '7'0 clock', at which an event took place the suffix -ta/madhyē is used.

5.1.6.2 deśādhikaraṇa (space) [k7p]

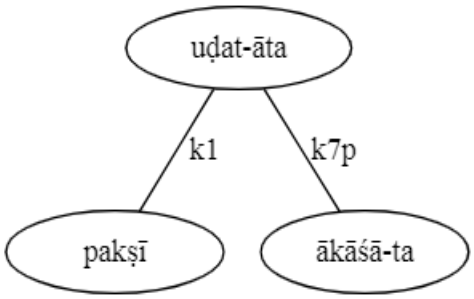
The participant denoting the location of *kartā* or karma at the time of action is called deshadhikarana. It is tagged as 'k7p'. Some examples of 'k7p' are given below.

Consider the following example:

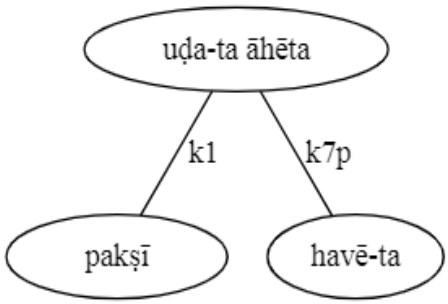
43. pustaka **bāke-vara (k7p)** āhē.
book-nom-N3sg table-pp-on be
'book is on the table'



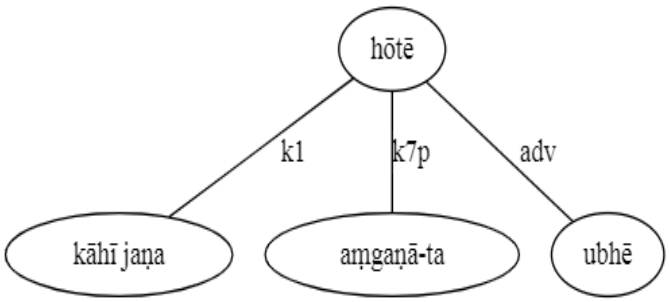
44. pakṣī ākāśā-ta (k7p) uḍat-āta.
bird-nom-N3pl sky-pp-on fly-present-n3pl
'The birds are flying on the sky'



45. pakṣī havē-ta uḍa-ta āhēta.
bird-nom-N3pl sky-pp-in fly-n3pl be-present
'The birds are flying in the air.'



46. kāhī jaṇa **aṃgaṇā-ta/madhe (k7p)** ubhē hōtē.
some people patio-pp-in stand be-past-N3pl
'Some people stood in the patio.'



The k7p can be identified on the basis of the following:

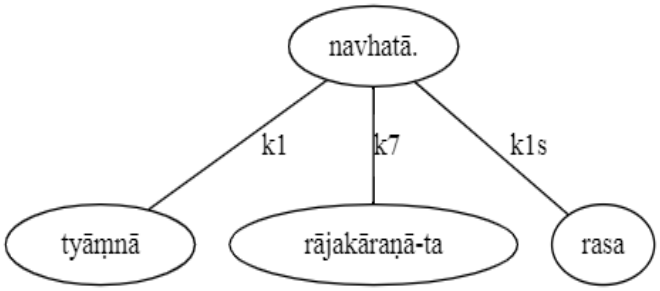
a) The suffixes _varuna, _ta and _vara generally mark an NP for desadhikaran.

b) The variation in the use of each suffix can be attributed to the noun semantics.

5.1.6.3 visayādhikaraṇa (topic and space elsewhere) [k7]

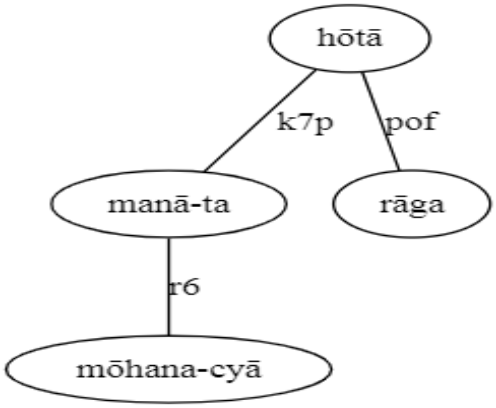
The vishayadhikarana is used to denote ‘location in a topic’ as in the following example:

47. tyāṃnā **rājakāraṇā-ta (k7)** rasa navhatā.
they-n3pl politics-pp enjoyment do not-n3pl
‘They did not have an interest in politics.’



However, it is not restricted to a ‘topic of discourse’. It also denotes a location other than time and place as in the following:

48. mōhana-cyā **manā-ta (k7)** rāga hōtā.
mohan-m3sg-pos mind-pp anger-m3sg be-past-m3sg
‘Mohan had anger in his mind.’



5.2 Other Dependency (R) Relations

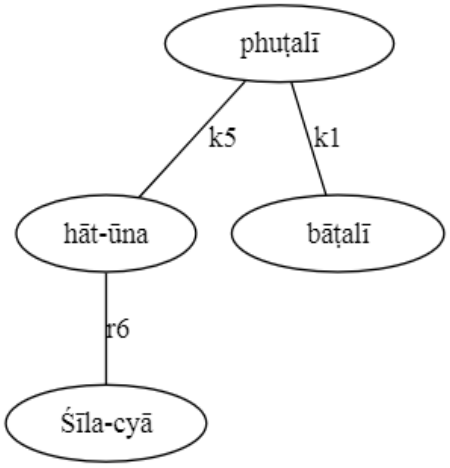
The relations are indirectly related to the verb and are labelled as r-Relations. shashthi (genitive/possessive), rh for hētu (reason), rt for tādarthya (purpose), and rd for prati (direction) are grouped under this head. We discuss these in the following sections:

5.2.1 shashthi relations (genitive/possessive) [r6]

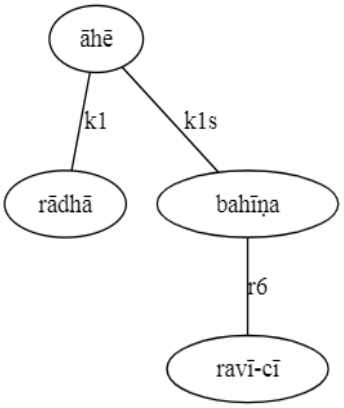
The shashthi or the genitive case denotes an adnominal relation that maps the relation between two nouns. In Marathi, _cā, _cī, _cyā and _cē are genitive or possessive markers that indicate that two nouns are related. The relationship between the two nouns is one of the possessor and possessed.

The following examples illustrate shashthi relations:

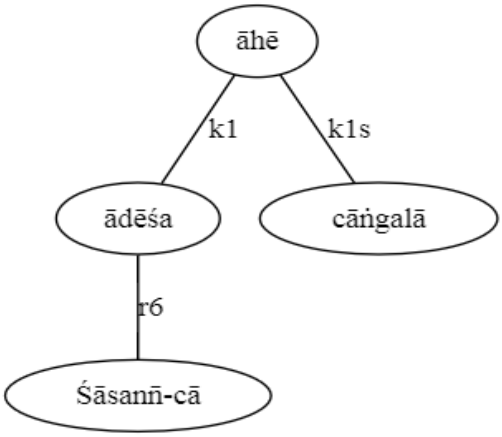
49. **Śīla-cyā (r6)** hāt-ūna bāṭalī phuṭalī.
sheela-gen. hand-from bottel-nom-f3sg burst-past-f3sg
'The bottle burst from Sheela's hand.'



50. rādhā **ravī-cī (r6)** bahīṇa āhē.
Radha-f3sg ravi-m3sg sister be-present
Radha is ravi's sister.



51. **Śāsanī-cā (r6)** ādēśa cāṅgalā āhē.
goverment-gen order good be-present
Order of the Government is good.

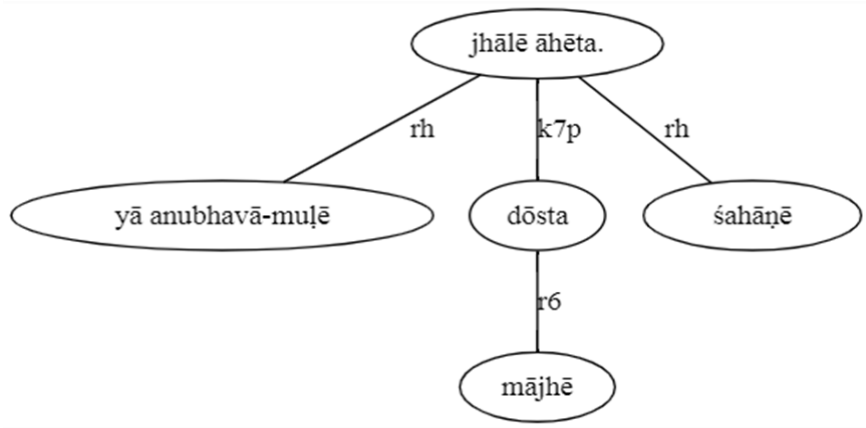


The shasthi denotes not only the possessive relationship between two objects but also part-whole, kinship, and a host of other relations.

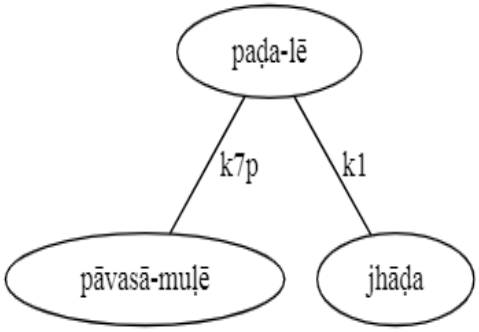
5.2.2 hētu (reason) [rh]

The rh tagset indicates the reason or the cause of action. Consider the following sentence:

52. **yā anubhavā-muḷē (rh)** mājhē dōsta śahāṇē jhālē āhēta.
this experience-because-of my friends clever be-past have
My friends have become clever because of this experience.



53. **pāvasā-muḷē** (rh) jhāḍa paḍa-lē.
rain-due-to tree fell-past
The tree fell due to rain.

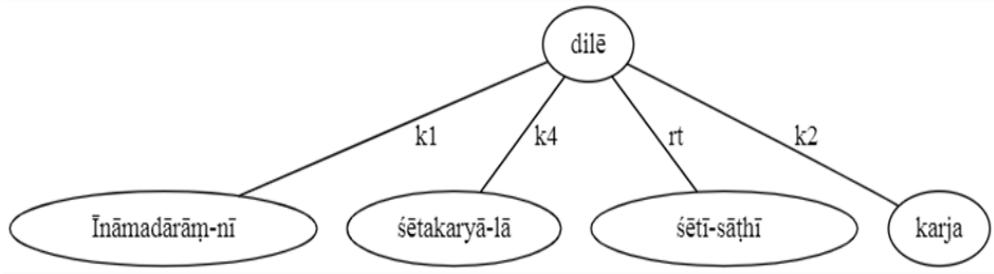


In the above examples, the postposition -muḷē marks ‘the reason’ or hetu.

5.2.3 tādarthya (purpose) [rt]

The tādarthya refers to the purpose of an action and is tagged as rt. The following example highlights the phenomenon of tādarthya.

52. Īmānadārāṃ-nī śētakaryā-lā **śētī-sāṭhī (rt)** karja dilē.
Imanadara-erg farmer-dat farming-pp laon-n3sg give-n3sg
‘Imanadar gave loans to the farmer for farming.’



In the above example, śētī-sāṭhī is tagged as rt because of its function as ‘purposive’

The following cue helps identify 'rh' relation in the sentence.

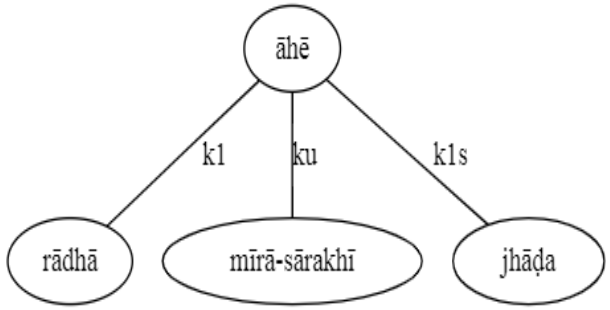
a) In most cases, the NP is marked by -sāṭhī .

5.2.4 sādrishya (similarity/comparison) [k*u]

sādrishya denotes similarity or comparison. Similarity or comparison can be established with either *kartā* or karma in a sentence:

53. rādhā **mīrā-sārakhī (k*u)** sundara āhē.
radha-f3sg mira-like beautiful be-n3sg

‘Radha is beautiful like Mira.’



54. mīrā-cyā **tulanē-ta (k*u)** rādhā jāsta sundara āhē.
mira-of comparison radha-f3sg more beautiful be-present
‘Radha is more beautiful in comparison to meera.’

In the above examples, mīrā-sārakhī and mīrā-cyā tulanē-ta NPs denote the meaning of comparison and similarity with rādhā. therefore we follow following rules

a) In most cases the NP would be carried -sārakhī, -tulanā postpositions.

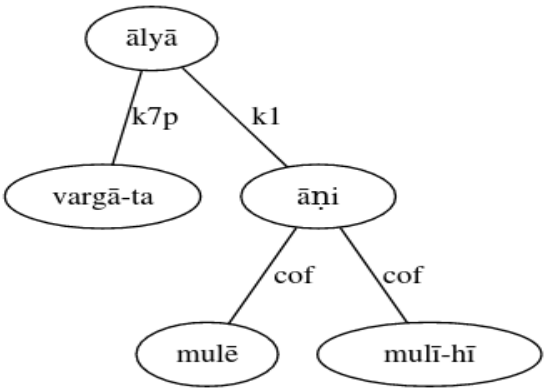
5.3 Non-Dependency relations

The non-dependency relations are not directly dependent on the verb but are essential for arranging the sequence of words in a sentence. They are tagged based on their functions: ccof, pof, nmod, jjmod, vmod etc. The following sections illustrate these:

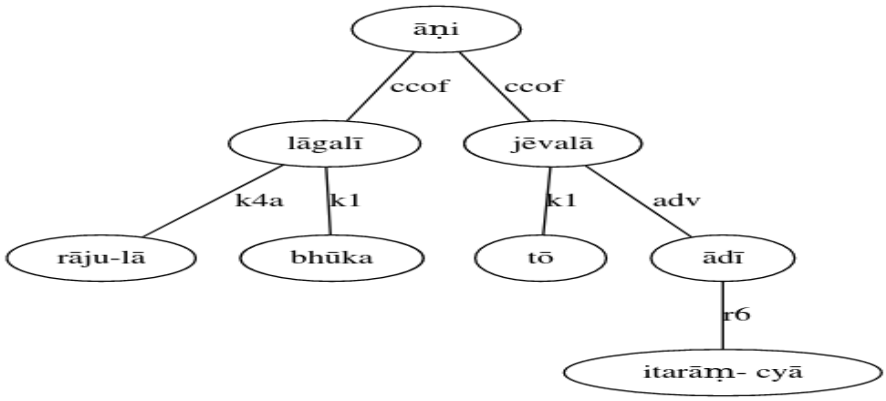
5.3.1 ccof (co-ordination and sub-ordination)

The 'ccof' tagset is a specific tagset used to label co-ordinate and sub-ordinate conjuncts in the sentence. In a dependency tree representation, a coordinate construction is represented in terms, where the conjunct is the head and other coordinating NPs are attached as child nodes to the head. In the case of a subordinating construct, the sub-ordinator is the head, and the subordinating clauses are the child nodes. The following examples can illustrate this:

52. vargā-ta mulē **āṇi (ccof)** mulī-hī ālyā.
classroom-pp boys and-conj girls-prt come-past-N3pl
Boys and girls entered the classroom.



53. rāju-lā bhūka lāgalī **āṇi (ccof)** tō itarāṃ- cyā
raju-dat hungry-f3sg get-past-f3sg and-conj he-m2sg others-gen
ādī jēvalā.
before eat-past-m2sg
Raju was hungry and he ate before others.



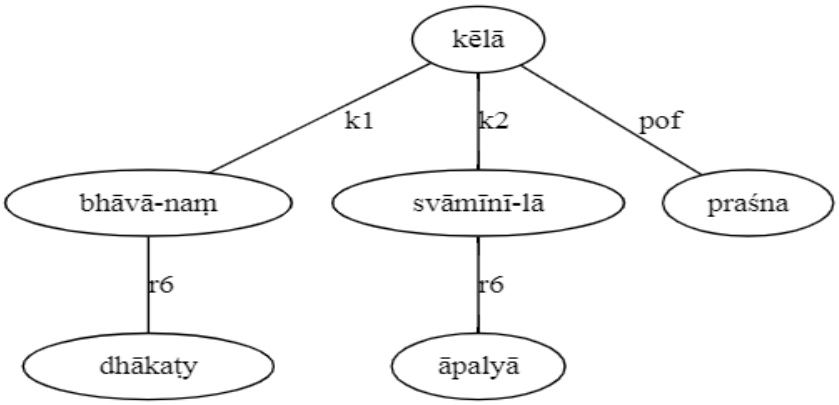
53. gurujī mhaṇ-ā-lē **kī (ccof)** vidyārthāṃ -nī śista pāḷā-vī.
teacher say-past-m3sg that students-ege discipline-f3sg follow-fut-f3sg
The teacher said that students should follow discipline.

The above examples (51, 52, and 53) illustrate coordination and conjunction.

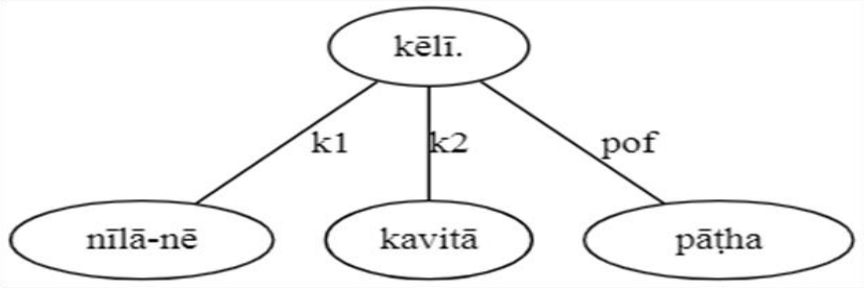
5.3.2 pof (Conjunct Verbs)

The 'pof' tagset is used to part of units of one chunk in the sentence. Conjunct verb constructions are unique to Indian languages, wherein it is possible to convert a noun or an adjective into a verb by adding a verb yielding noun+verb, adjective+verb, and adverb+verb constructs. In such constructions, the conjoined noun/adjective is linked to the verb as pof 'part of'. Consider the following sentences:

56. dhākaṭyā bhāvā-naṃ āpalyā svāmīnī-lā **praśna (pof)** kēlā.
younger brother-erg his godman -dat question-n3sg-conj do-past-N3sg
'The younger brother questioned his godman.'



57. nīlā-nē kavitā **pāṭha(pof)** kēlī.
neela-erg poem-f3sg recite-conj do-past-f3sg
'Neela recited the poem.'

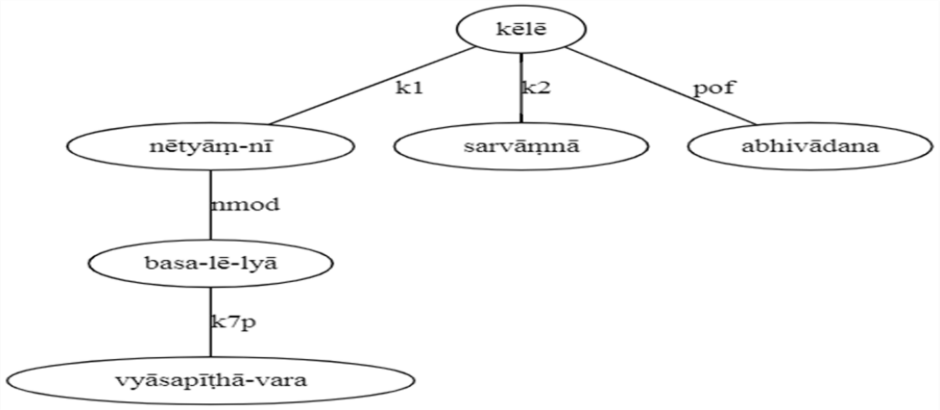


The above examples (54 and 55) demonstrate that praśna+kēlā (questioned) and pāṭha+kēlī function as conjunct verbs. Therefore, praśna and pāṭha are marked as 'pof'. The following diagram illustrates 'prof' phenomena.

5.3.3 nmod (Participles etc modifying noun)

The 'nmod' tagset denotes a verbal participial that modifies a noun.In other words, the derived verbal adjective modifies the head noun. Such participial constructions are tagged as 'nmod'. The following example illustrates this:

58. vyāsapīṭhā-vara **basa-lē-lyā (nmod)** nētyāṃ-nī sarvāṃnā abhivādana kēlē
podium-pp sit-parti-m3pl leadar-egr everyone greet- ph.v do-past
'The leaders sitting on the podium greeted everyone.'

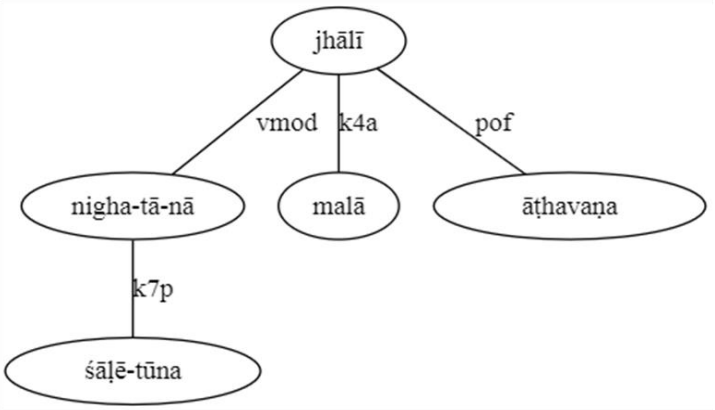


a) The non-finite verb form agrees with the modifying noun for gender and number.

5.3.4 vmod (verb modifier)

The vmod tagset is used to mark a verbal modifier. The non-finite verb acts as a modifier of the finite verb. These two clauses share the same karta. The shared karta is marked for the main clause (the clause with the finite verb) and the non-finite clauses, function as modifiers of the main clause. Consider the following diagram:

59. śāḷē-tūna **nigha-tā-nā (vmod)** malā āṭhavaṇa jhālī.
school-pp leave- parti I-dat remember be-past
'I remembered leaving to school.'



5.4. Summary:

This chapter lists the various case relations and the tags that are used to annotate the corpus. This chapter discusses the realization of the various syntactic and semantic relations in Marathi and the mechanisms for capturing these with help of Paninian Dependency grammar.

The following table illustrates this:

A total number of 23 tag sets have been identified for the purpose of annotating the corpus.

The following table illustrates the basic k-relations

Tag-sets	Dependency relations	Descriptions	Hand crafted rules
k1	kartā	Agent/doer/subject	NP carries [+independent], [+action], [+locus] [+volitionality] and [-volitionality] ontological features. When the verb is in the past tense the k1 is marked by -ne, the ergative marker. The ergative marker blocks the agreement of the verb with the subject NP. The verb in cases where kartā is blocked for agreement with the verb because of the intervening ergative case marker, does not show agreement with the object NP either. The verb root shows a neutralized agreement. When the verb is in passive, the subject is marked with kaḍuna 'by' the passive morphology. In either case the verb does not agree with the Karta for Gender, number, and person. when the verb is in infinitive past marker, the NP carries -lā marker.
pk1	prayojaka kartā	causer	a causative verb can be identified by using the morphological marker -vi affixed to the verb stem. The causer pk1 carries -nī postposition.
jk1	prayojya kartā	causee	he causee jk1 will take -na postposition.
mk1	madhyastha kartā	(mediator causer)	The Second causer mk1 will take -kaḍuna and -dvārē postpositions.
k1s	kartā samānadhikaraṇaa/ vidheya kartā	noun complement of kartā	k1s can be either another noun or an adjective qualifying k1.
k2	karma	object/patient	NP marked for the features [+Human] as in mulā-lā and [-animate] as bāke-lā in are mandatorily marked for the case. In the case of nouns that are [+animate, -

			Human] the case marker is optional. Animacy scale: human > animate > inanimate
k2p		Goal / Destination	The goal/destination karma does not agree with the verb NP carries _0 and _ lā postpositions.
k2s	karma samānadhikaraṇaa	Object complement	The k2s can only occur if there is k2 in a sentence. If a few types of verb sets (communicative and perception) are there, then k2s can occur in the sentence.
k3	karaṇa kāraka	Instrument	The NP with the ontological features [+human] [+volitional] dominates over the one with [-animate] and marked as kartā kāraka. The other NP is the karaṇa kāraka. In same cases it occur as abstract object and adverbial manner
k4	sampradāna kāraka	Recipient	The verb is a ditransitive verb The NP usually carries _lā case marker. The verbs generally belong to the semantic domains of 'give/ take' and 'communication.'
k4a	anubhav kartā	Dative/Experience subject	an experiencer subjects marked for dative case marker _ -lā a verbs of cognition, perception and psycho-somatic states are take an experiencer subject
k5	apādāna kāraka	Source/ Separation	In most cases, the NP carries _tūna and _hūna postpositions in Marathi. In general, the verb's semantic form is unclear but in some cases it depends on the context, like a point of departure or separation.
k7t	kālādhikaraṇa	Time	In the case of nouns denoting time/ temporal adverbs, case marking is optional. To indicate the exact 'point of time' for example. 7 vājatā '7'0 clock', at which an event took place the suffix -ta/madhyē is used.
k7p	deshadhikaraṇa	Space	The suffixes _varuna, _ta and _vara generally mark an NP for desadhikaran. The variation in the use of each suffix can be attributed to the noun semantics.
k7	vishayadhikaraṇa	Topic and space elsewhere	It is denoted 'location in a topic' and 'topic of discourse'.Also carry -ta

			postpositions.

The following table illustrates r-relations:

r6	shashthi relations	Genitive/possessive	NP ending with _cā, _cī, _cyā postpositions, shasthi denotes not only the possessive relationship between two objects but also part-whole, kinship, and a host of other relations.
rh	hētu	Reason	The postposition -muḷē marks ‘the reason’ or hetu.
rt	tādarthya	Purpose	In most cases, the NP is marked by -sāṭhī .

The following table illustrates Other Dependency relations:

ccof	co-ordination and sub-ordination conjuncts	A 'ccof' tagset should be attached to conjunct in Marathi (like- āṇi, va, athavā etc). A conjunct chunk should have children relations or modifying relations.
pof	Conjunct verbs	A 'pof' tagset should be attached to the conjunct verb in Marathi. A conjunct verb would follow an agreement structure with a verb.
nmod	Participles etc, modifying the noun	‘nmod’ should be attached with participle form in the sentence. The non-finite verb form such as participial are agree in gender and number with noun and it modifier.
vmod	verb modifier	‘vmod' tagset should be attached to the verb chunk. (which modifying infinitive, participle and gerund forms)

Chapter VI

CONLUSION AND FUTURE WORK

The present doctoral thesis is entitled as 'Linguistic Inputs in building a dependency parser for Marathi: A Panianian Approach'. To develop a parser for Marathi, Treebank data (annotated sentences) plays a significance role. It is a challenging task to create a Treebank for Indian languages, because they are morphologically rich and lot of linguistic information is expressed at the morphologic, syntactic and semantic level.

A corpus is available or accessible only for major Indian languages, especially the scheduled languages as a linguistic resource in the Indian context. Among the scheduled languages, Marathi is one of them. The corpus available for the Marathi is less in size and limited to a few domains.In order to build a Marathi Treebank, we collected 2k sentences form various domains, such as grammar books, short stories,articles and other domains from online sources that are available free of cost.

We followed Paninian grammatical framework (DS Guidelines) to develop Marathi Treebank data which is encoded with linguistic information (See Bharathi A. et al.1995: 2006: 2007: 2012 and See Chapter4).However,the annotation procedure has been categorized into three parts. They are k-relations, r-relations and non-dependency relations. The k-relations denotes basic six karaka relations, such as kartā (k1) ‘subject/doer’, karma (k2) ‘object’,karaṇa(k3) ‘instrument’,sampradāna(k4) ‘dative’,apādāna(k5) ‘source’ andadhikaraṇa (k7) ‘location’. These all NPsare having the direct link with the main verb in a givenMarathi sentences. The r-relations described non-kāraka (other dependency) relations in the Marathi sentences. These NP’sare having indirect link with the main verb. Such as shashthi(r6) ‘genitive’,hētu (rh) ‘reason’,tādārthya(rt) ‘purpose’ etc. In third part, Dependency Guidles (DS) takes care of non-dependency relations in the Marathi sentences, such as co-ordination/sub-ordination (ccof), conjunct verbs (pof), noun modifiers (nmod), verb modifiers (vmod) etc.

Along with that we also gave case resolutions for various case markers, The following are some of the mandatory rules:

a) -∅ zero case marker, that appeared with kartā(k1) and karma(k2) NP’s in Marathi.
b) The ‘*–lā*’ case marker occurs with karma(k2), sampradāna(k4) and anubhavkartā(k4a).
c) The ‘-nē’case marker found with kartā and karaṇa.

Subsequently, we considered developed Marathi Treebank data (linguistic resource) to interpret the Marathi syntactic units, extracted and formulated30linguistic rules to identify k-relations, r-relations and non-dependency relationsin Marathi (See Chapter 5).

Statistical frequency of the Marathi Treebank

In the following section, we described various statical frequenciesof the Marathi Treebankviz. Parts of Speech (POS) categories, Chunk (Phrasal categories), treebank labels.

POS tagging frequency of Marathi Treebank:

A POS tagging denotes grammatical categories viz. noun, verb, adjective, adverb, postposition etc in the Marathi sentences.The complete POS tag-set is given in the appendix.The frequency of the Marathi POS categories are illustratedin the following table:

POS Tag	Frequency Count
NN	7114
VM	3401
SYM	2376
RB	898
NNP	802
JJ	767
VAUX	569
DEM	404
NST	362
CC	346
PRP	301
QC	280
QF	213
NEG	182
WQ	109
QO	82
CL	56
INTF	41

Table 6.1 Frequency Count of the POS categories in Marathi Treebank

Out of 2k Marathi Treebank data NN has occurred 7114 times, VM has occurred 3401 times, RB has occurred 898 times, NNP has occurred 802 times, JJ has occurred 767 times, VAUX has occurred 569 times, DEM has occurred 404 times, NST has occurred 362 times, CC has occurred 346 times, PRP has occurred 301 times, QC has occurred 280 times, QF has occurred 213 times, NEG has occurred 182 times, WQ has occurred 109 times. QO has occurred 82 times, CL has occurred 56 times, INTF has occurred 41 times. The highest POS tag is NN 7114 times and the lowest POS tag is INTF is 41 times. The main observation is that there are 3401 main verbs and 7114 common nouns out of more than 2k Marathi Treebank data.

Chunk/phrase categories frequency of Marathi Treebank

A Chunk tag denotes a Phrasal category viz. noun phrase, verb phrase, adjectival phrase, adverbial phrase etc in the Marathi sentences.The complete Chunk tag-set is given in the appendix. The frequency of the Marathi chunk (phrasal) categories are given in the following table.

Phrase	Frequency Count
NP	8587
VGF	2187
VGNF	789
RBP	506
CCP	330
JJP	67
VGINF	34
VGNN	31

Table 6.2 Frequency Count of the Chunks in Marathi Treebank

Out of 2k Marathi Treebank data, Noun phrase has occurred 8587 times, VGF has occurred 2187 times, VGNF has occurred 789 times, RBP has occurred 506 times, CCP has occurred 330 times, JJP has occurred 67 times, VGINF has occurred 34 times, VGNN has occurred 31 times. Based on the highest frequency of POS categories of the Marathi Treebank data, chunks frequency is also reflected in the same. In POS categories NN and VM were the highest POS tags. Similarly NP and VGF are the highest phrasal categories with 8587 times and 789 times frequency. VGNN is the lowest frequent Phrase in the Marathi Treebank data.

Dependency k-relations in the Marathi Treebank

The following table shows frequency of dependency k-relations of the Marathi treebank data.

karaka Tag	Frequency count
k1	2774
pk1	8
jk1	1
mk1	2
k1s	650
k2	940

k2p	29
k2s	29
k2g	14
k3	43
k4	90
k4a	97
k5	68
k7	56
k7t	500
k7p	952

Table 6.3 Frequencies of k-labels

Out of 2k Marathi Treebank data, k1 has occurred 2774, pk1 has occurred 8 times, jk1 has occurred 1 time, mk1 has occurred 2 times,k1s has occurred 650 times, k2 has occurred 940 times, k2p has occurred 29 times, k2s has occurred 29 times, k2g has occurred 14 times, k3 has occurred 43 times, k4 has occurred 90 times, k4a has occurred 97 times, k5 has occurred 68 times, k7 has occurred 56 times, k7t has occurred 500 times, k7p has occurred 952. The main observation based on the statistical frequency of the k-labels of Marathi Treebank data, the most highest k-labels are k1, k2, k4, k4a, k7t and k7p. It means that a basic Marathi sentence consists of k1, k2 and VM as mandatory tags. Apart from them, k7t and k7p are also having the highest frequency in comparison with k1 and k2.

The r-relations in the Marathi Treebank

r-relations	**Frequency count**
r6	1170
rh	89
rt	14

Table 6.4 Frequencies of r-relations in Marathi Treebank

The non-dependency relations of Marathi Treebank

Non-dependency Relation	**Frequency Count**
ccof	779
pof	22
nmod	397
vmod	733

Table 6.5 Non-dependency frequencies

Out of 2k Marathi Treebank data, ccof has occurred 779 times, pof has occurred 22 times, nmod has occurred 397 times andvmod has occurred 733 times.

By using this linguistically annotated Marathi Treebank, we have extracted linguistic inputs. Based on that, we have hand-crafted the grammatical rules to identify various karaka , non-karaka and other dependency relations. Thirty rules are formulated by using developed Marathi Treebank as linguistics input.

Future Work

As we know that the main goal of the NLP is to develop smart expert systems, smart language applications for Indian languages. For that, we need large amount of annotated data which is encoded with linguistic knowledge to overcome the lexical, syntactic and semantic ambiguity. Since, we need huge amount of annotated data to develop Marathi parser. We use developed 2k Marathi treebank data to extract linguistic knowledge, rules and by using this linguistic knowledge, we try to annotate new Marathi sentences automatically to save time and energy. Marathi Linguistic insights will be extracted from the developed 2k Marathi treebank to develop robust Marathi parser. Our future work is to implement the developed 2k Marathi Treebank data by using the statistical parsers or data-driven parsers like MALT (Nivre et al. 2007), MST (McDonald and Pereira, 2006). Apart from that Marathi Treebank data can implemented by using the Machine learning techniques, deep learning models to develop Marathi syntactic parser.

In the near future we shall use the developed Marathi treebank data to develop Verb Frames. By doing this we can try to understand the argument structure of the Marathi verbs (transitive, intransitive, ditransitive). Ontological classification of the Marathi verbs can be evolved by using the develop Marathi treebank data. This kind of language resource will be useful to disambiguate the ambiguous nouns in Marathi. This can be useful to develop Machine translation systems from Marathi to any Indian language and vice-versa. In future, we try to convert our 2k Marathi Dependency Treebank data as Universal Dependency Treebank (UD Treebank) and will make it available for the public access.

This kind of Marathi lexical resource which is encoded with syntactic and semantic information will be helpful to develop various NLP applications like machine translation, question answering system, information extraction, information retrieval, word sense disambiguation, verb frame, question answering systems, syntactic parsing from Marathi to any Indian language.

Bibliography

1. A. Mel'cuk. 1988. *Dependency Syntax: Theory and Practice*, State University, Press of New York.
2. Abeillé, L. Clément, F. Toussenel. 2003. Building a treebank for french. In Treebanks: Building and Using Parsed Corpora, 165-187.
3. Abeille. 2003. (ed.) *Treebanks: Building and Using Syntactically Annotated Corpora*. Dordrecht, The Netherlands: Kluwer Academic Publishers, xiii-xxvi.
4. Amba Kulkarni, Sheetal Pokar, and Devanand Shukl. 2010. Designing a Constraint Based Parser for Sanskrit In Proceedings of the International Sanskrit Computational Linguistics Symposium, Springer.
5. Ambati, P. Gade, G.S.K. Chaitanya and S. Husain. 2009. Effect of Minimal Semantics on Dependency Parsing. In RANLP09 *student paper workshop*.
6. B.R. Ambati, P. Gadde and K. Jindal. 2009. Experiments in Indian Language Dependency Parsing. In:*ICON09 NLP Tools Contest:Indian Language Dependency Parsing*, pp 32-37.
7. Backus, J. W. (1996). Transcript of question and answer session. In Wexelblat, R. L. (Ed.), History of Programming Languages, p. 162. Academic Press.
8. Backus, J. W. 1959. The syntax and semantics of the proposed international algebraic language of the Zurich ACMGAMM Conference. In Information Processing: *Proceedings of the International Conference on Information Processing*, Paris, pp. 125–132. UNESCO
9. Bamman, and G. Crane. "The ancient Greek and Latin dependency treebanks." Language technology for cultural heritage. Springer Berlin Heidelberg, 2011. 79-98.
10. Begum and D. M. Sharma. 2010. A Preliminary Work on Causative Verbs in Hindi. In *Proceedings of the 8 th Workshop on Asian Language Resources* at COLING 2010.
11. Begum, K. Jindal, A. Jain, S. Husain and D. M. Sharma. 2011. Identification of Conjunct Verbs in Hindi and Its Effect on Parsing Accuracy. *In Proceedings of the 12th CICLing*, Tokyo, Japan.
12. Bhagawat,S. 2003. *Tumache Amache Marthi Vyakaran*. Vidyabhari Prakashan, Latur.
13. Bharati and R. Sangal. 1993. Parsing Free Word Order Languages in the Paninian Framework, *ACL93: Proc. of Annual Meeting of Association for Computational Linguistics.*
14. Bharati, D. M. Sharma, L. Bai and R. Sangal. 2006. AnnCorra : Annotating Corpora Guidelines For POS And Chunk Annotation For Indian Languages. *LTRC Technical Report-31*.
15. Bharati, D. M. Sharma, S. Husain, L. Bai, R. Begum and R. Sangal. 2009 AnnCorra:TreeBanks for Indian Languages, Guidelines for Annotating Hindi Treebank http://ltrc.iiit.ac.in/MachineTrans/research/tb/DS-guidelines/DS-guidelines-ver2-28-05-09.pdf

16. Bharati, M. Gupta, V. Yadav, G. Karthik and D. M. Sharma. 2009. Simple Parser for Indian Languages in a Dependency Framework. In *Proc. of the Third Linguistic Annotation Workshopat 47th ACL and 4 th IJCNLP*
17. Bharati, R. Sangal and D. M. Sharma. 2007. SSF: Shakti Standard Format Guide. *LTRC Technical Report-33,* Language Technologies Research Centre, IIIT Hyderabad, India.
18. Bharati, R. Sangal and T. P. Reddy. 2002. A Constraint Based Parser Using Integer Programming, *In Proc. of ICON-2002: International Conference on Natural Language Processing*, 2002.
19. Bharati, R. Sangal,V. Chaitanya, A. P. Kulkarni and D. M. Sharma. 2002. AnnCorra: Building Tree-banks in Indian Languages. *Published in the proceedings of COLING 2002 Post-Conference Workshops - Proceedings of the 3rd Workshop on Asia Language Resources and International Standardization (August 31, 2002)* at Taipei, Taiwan.
20. Bharati, R. Sangal,V. Chaitanya, A. P. Kulkarni and D. M. Sharma. 2002. AnnCorra: Building Tree-banks in Indian Languages. *Published in the proceedings of COLING 2002 Post-Conference Workshops - Proceedings of the 3rd Workshop on Asia Language Resources and International Standardization (August 31, 2002)* at Taipei,Taiwan.
21. Bharati, S. Husain, D. M. Sharma and R. Sangal. 2009. Two stage constraint based hybrid approach to free word order language dependency parsing. In *Proceedings of the 11th International Conference on Parsing Technologies (IWPT09)*. Paris.
22. Bharati, S. Husain, D. M. Sharma, and R. Sangal. 2008. A Two-Stage Constraint Based Dependency Parser for Free Word Order Languages. In *Proceedings of the COLIPS International Conference on Asian LanguageProcessing 2008 (IALP)*. Chiang Mai, Thailand.
23. Bharati, S. Husain, D. M. Sharma, L. Bai, and R.Sangal. 2009. AnnCorra:TreeBanks for IndianLanguages,Guidelines for Annotating Intra-chunk dependency.
24. Bharati, V. Chaitanya and R. Sangal. 1995. *Natural Language Processing: A Paninian Perspective,* Prentice-Hall of India, New Delhi.
25. Bloch,J. 1970. *The Formation of Marathi Language*. Motilal Banarsidass publication, Delhi
26. Bloomfield, L. (1933). *Language*. University of Chicago Press.
27. Bosco and V. Lombardo. 2004. Dependency and relational structure in Treebank annotation. In Proceedings of Workshop on Recent Advances in Dependency Grammar at COLING'04.
28. Brants, W. Skut, & H. Uszkoreit. 1999. Syntactic annotation of a German newspaper corpus. In Proceedings of the ATALA Treebank Workshop (pp. 69–76). Paris, France.
29. Charniak. 1996. Treebank grammars. In *National Conference on Artificial Intelligence (AAAI)*, 1996.

30. Chatterji, T.M. Sarkar, P. Dhang et al. 2014. “A dependency annotation scheme for Bangla treebank”Lang Resources & Evaluation.
31. Chaudhry and D.M. Sharma. 2011. Annotation and issues in building an English dependency treebank. In*Proceeding of: ICON-2011: 9th International Conference on Natural Language Processing*, Chennai.
32. Chomsky, N. (1957). Syntactic Structures. Mouton, The Hague.
33. Damale, M.K. 1911. *Shastriya Marathi vyakaraN* . Pune: Deshmukha and Company.
34. Dukes, E. Atwell and A. M. Sharaf. 2010. Syntactic Annotation Guidelines for the Quranic Arabic Dependency Treebank. Language Resources and Evaluation Conference (LREC). Valletta, Malta.
35. E. Hajičová. 1998. Prague Dependency Treebank: From Analytic to Tectogrammatical Annotation. In Proc. TSD’98
36. Hajic, J. Panevova, E. Hajicova, P. Sgall, P. Pajas, J. Stepanek, J. Havelka, M. Mikulova, Z. Zabokrtsky, and M. Sevcıkova-Razımova. 2006. Prague Dependency Treebank 2.0. CD-ROM, Linguistic Data Consortium, LDC Catalog No.: LDC2006T01, Philadelphia.
37. I.Aduriz, M. J. Aranzabe, J. M. Arriola, A. Atutxa, A. Díaz de Ilarraza, A. Garmendia, and M. Oronoz. 2003.Construction of a Basque dependency treebank. In Proceedings of the Second Workshop on Treebanks and Linguistic Theories (TLT), pages 201-204.
38. I.Boguslavsky, S. Grigorieva, N. Grigoriev, L. Kreidlin, and N. Frid. 2000. Dependency treebank for Russian: Concept, tools, types of information. In Proceedings of the 18th International Conference on ComputationalLinguistics (COLING), pages 987-991.
39. Iai, Romania.L. Ramasamy and Z. Žabokrtský. 2014. "Tamil Dependency Treebank v0. 1
40. J. Nivre. 2008. Treebanks. In A. Lüdeling and M. Kytö, eds., *Corpus Linguistics: An International Handbook,vol. 1, pages 225-241*. Walter de Gruyter.
41. J. Nivre. 2009. *Parsing Indian Languages with MaltParser. Proc. of ICON09 NLP Tools Contest: Indian Language Dependency Parsing, 12-18.*
42. J.Nivre, J. Hall, J. Nilsson, A. Chanev, G. Eryigit, S. Kübler, S. Marinov and E Marsi. 2007. *MaltParser: A language-independent system for data-driven dependency parsing*. NLE.
43. Juhi Tandon and Dipti Misra Sharma. 2017. Unity in diversity: A unified parsing strategy for major Indian languages. In Proceedings of the Fourth International Conference on Dependency Linguistics (Depling 2017).pages 255–265.
44. K. Singh. 2006. http://sourceforge.net/projects/nlp-sanchay.
45. Kiparsky and J. F. Staal. 1969. ‘Syntactic and Semantic Relations in Panini’, *Foundations of Language 5, 84-117.*
46. L. Ramasamy and Z. Žabokrtský. 2014. "Tamil Dependency Treebank v0. 1
47. L. Tesnière. 1959. *Eléments de Syntaxe Structurale*. Klincksiek, Paris.

48. L. Van der Beek, G. Bouma, R. Malouf, and G. van Noord. 2002. The Alpino dependency treebank. Computational Linguistics in the Netherlands.
49. M. P. Marcus, B. Santorini, and M.A. Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank, Computational Linguistics, 19, 313 – 330.
50. M. Seraji. 2015. Morphosyntactic Corpora and Tools for Persian. PhD Thesis. Studia Linguistica Upsaliensia16.
51. M.-C. de Marneffe, and C. D. Manning. 2008. The Stanford typed dependencies representation. In Workshop on Cross-framework and Cross-domain Parser Evaluation.
52. Maamouri, A. Bies, T. Buckwalter, and W. Mekki. 2004. The Penn Arabic Treebank: Building a Large-Scale Annotated Arabic Corpus. In NEMLAR Conference on Arabic Language Resources and Tools, pages102–109.
53. Moreno, S. López, F. Sánchez, R. Grishman. 2003. *Developing a Syntactic Annotation Scheme and Tools for a Spanish Treebank*. In: Abeillé A. (eds) Treebanks. Text, Speech and Language Technology, vol 20. Springer, Dordrecht.
54. Moreno, S. López, F. Sánchez, R. Grishman. 2003. Developing a Syntactic Annotation Scheme and Tools for a Spanish Treebank. In: Abeillé A. (eds) Treebanks. Text, Speech and Language Technology, vol 20. Springer, Dordrecht.
55. N. Misra. 1966. *The Descriptive Technique of Panini*. Mouton & Co.
56. N. Xue, F. Xia, F. Chiou, and M. Palmer. 2005. The Penn Chinese TreeBank: Phrase Structure Annotation of a Large Corpus. Natural Language Engineering, 11(2)207-238.
57. Navalkar, G.R. 1894. The Student Marathi Grammar. Education Society Press, Bombay.
58. Nivre, 2005. Dependency Grammar and Dependency Parsing. *MSI report 05133*. Växjö University: Schoolof Mathematics and Systems Engineering.
59. Nivre, J. (2002), What Kinds of Trees Grow in Swedish Soil? A Comparison of Four AnnotationSchemes for Swedish. In: Hinrichs/Simov 2002, 123-138.
60. Nivre. 2005. Book review of Anne Abeill´ e, editor, Treebanks: Building and using parsed corpora, Kluwer AP, 2003. Machine Translation, 18:373–376.
61. Oliver Hellwig, Salvatore Scarlata, Elia Ackermann, Paul Widmer The Treebank of Vedic Sanskrit Proceedings of the 12th Conference on Language Resources and Evaluation (LREC 2020), pages 5137–5146 Marseille, 11–16 May 2020 c European Language Resources Association (ELRA), licensed under CC-BY-NC
62. P. Kiparsky. 2002. On the Architecture of Panini's Grammar. *In Hyderabad Conference on the Architecture of Grammar*.
63. Pandaripande, R.V. 1997. Marathi Descriptive Grammar. Routledge, London and New York.
64. Petrov, H.Zhang, O. Tackstr ¨om, C. Bedini, N. Bertomeu ¨Castello, and J. Lee. 2013. Universal dependency´ annotation for multilingual parsing. In ACL.
65. Qiu et al. "Multi-view Chinese Treebanking." COLING. 2014.

66. R. Ambati. 2011. Hindi Dependency Parsing and Treebank Validation. Ph.D. thesis, IIIT-H, Hyderabad, India.
67. R. Begum, S. Husain, A. Dhwaj, D.M. Sharma, L. Bai and R. Sangal. 2008a. Dependency Annotation Scheme for Indian Languages. In *Proceedings of The Third International Joint Conference on Natural LanguageProcessing (IJCNLP)*, Hyderabad, India.
68. R. Begum, S. Husain, A. Dhwaj, D.M. Sharma, L. Bai and R. Sangal. 2008. Dependency Annotation Scheme for Indian Languages. In *Proceedings of The Third International Joint Conference on Natural LanguageProcessing (IJCNLP)*, Hyderabad, India.
69. R. Begum, S. Husain, D. M. Sharma and L. Bai. 2008. Developing Verb Frames in Hindi. In *Proceedings of The Sixth International Conference on Language Resources and Evaluation (LREC)*. Marrakech, Morocco.
70. R. Hudson. 1984. *Word Grammar*, Basil Blackwell, 108 Cowley Rd, Oxford, OX4 1JF, England.
71. R. McDonald, J. Nivre, Y. Quirmbach-Brundage, Y. Goldberg, D. Das, K. Ganchev, K. Hall, S.
72. R. McDonald, J. Nivre, Y. Quirmbach-Brundage, Y. Goldberg, D. Das, K. Ganchev, K. Hall, S. Petrov, H.Zhang, O. Tackstr ¨om, C. Bedini, N. Bertomeu ¨Castello, and J. Lee. 2013. Universal dependency ´ annotation for multilingual parsing. In ACL.
73. R.B.Ahmad, S.M.Bhat, and D.M.Sharma. 2014. *"Towards building a Kashmiri Treebank: Setting up the Annotation Pipeline." LREC.*
74. Rambow, C. Creswell, R. Szekely, H. Taber and M. Walker. 2002. A dependency treebank for English. In Proceedings of the 3rd International Conference on Language Resources and Evaluation.
75. S. Husain. 2011. *A Generalized Parsing Framework based on Computational Paninian Grammar*. Ph.D. thesis, IIIT-H, Hyderabad, India.
76. S.M. Bhat. 2012. Introducing Kashmiri dependency Treebank. In 24th International Conference on Computational Linguistics, page 53.
77. S.M. Gupta and J. Tuladhar. 1979-1980. Dative Subject Constructions in Hindi, Nepali and Marathi and Relational Grammar. *In Tibetan and Himalayan—Journals—Journals Focused on Regions—Nepal—Contributions to Nepalese Studies* -- Vol. 7, no. 1-2.
78. Sampson, G. (2003), Thoughts on Two Decades of Drawing Trees. In: Abeillé2003, 23-41.
79. Shinde. B. 2012. *ParipurN Marathi VyakaraN*. Ananad Publication, Aurangabad.
80. Silveira, et al. "A Gold Standard Dependency Corpus for English." LREC. 2014.
81. Skut, B. Krenn, T.Brants, & H. Uszkoreit. 1997. An annotation scheme for free word order languages. In Proceedings of ANLP-97. Washington, D.C.

82. Uma Maheshwar Rao G., K. Rajya Rama, A. Srinivas, 2012 Dative Case in Telugu: A Parsing Perspective in Proceedings of the Workshop on Machine Translation and Parsing in Indian Languages (MTPIL-2012), pages 123–132, COLING 2012, Mumbai, December 2012.
83. Valanbe, M.R. 2012. *Sugam Marathi vyakaraN*. Nitin Publication, Pune.
84. Vempaty, V. Naidu, S. Husain, R. Kiran, L. Bai, D. M. Sharma, and R. Sangal. 2010. Issues in analyzing Telugu sentences towards building a Telugu Treebank. In Proceedings of CICLing-2010.
85. Vinit Ravishankar, 2018 A Universal Dependencies Treebank for Marathi. In Proceedings of the 16th International Workshop on Treebanks and Linguistic Theories (TLT16), pages 190–200,Prague, Czech Republic, January 23–24, 2018. Distributed under a CC-BY 4.0 licence. 2017.
86. Wali, K. 1997. *Marathi : a study in comparative south asian languages*. Delhi: Indian institute of language studies.
87. Zeman et al. "HamleDT: Harmonized multi-language dependency treebank." Language Resources and Evaluation 48.4 (2014): 601-637.

Appendix A: POS Tagset

Sl. No.	Category	POSTag
1.1	Noun	NN
1.2	Noun of space and time	NST
2	Proper Noun	NNP
3.1	Pronoun	PRP
3.2	Demonstrative	DEM
4	Verb-finite	VM
5	Verb Auxiliary	VAUX
6	Adjective	JJ
7	Adverb (Manner)	RB
8	Post Position	PSP
9	Particles	RP
10	Conjuncts	CC
11	Question Words	WQ
12.1	Quantifiers	QF
12.2	Cardinal (Quantifier)	QC
12.3	Ordinal	QO
12.4	Classifier	CL
13	Intensifier	INTF
14	Interjection	INJ
15	Negation	NEG
16	Quotative	UT
17	Symbols	SYM
18	Compounds	*C
19	Reduplicative	RDP
20	Echo	ECH
21	Unknown	UNK

Appendix B: ChunkTagset

Sl. No.	Chunk Type	Tag name
1	Noun Chunk	NP
2	Finite Verb Chunk	VGF
3	Non-finite Verb Chunk	VGNF
4	Infinitival Verb Chunk	VGINF
5	Verb Chunk (Gerund)	VGNN
6	Adjectival Chunk	JJP
7	Adverb Chunk	RBP
8	Chunk for Negatives	NEGP
9	Conjuncts	CCP
10	Chunk Fragments	FRAGP
11	Miscellaneous	BLK

Appendix C: Dependency Tagset

Sl. No.	Tag Name	Tag Description
1.1	k1	kartā (doer/agent/subject)
1.2	pk1	prayōjakakartā (causer) The causer in a causative construction.
1.3	jk1	prayōjyakartā (causee) The causee in a causative construction.
1.4	mk1	madhyastakartā (mediator Causer) The mediator-causer in a causative construction.
1.5	k1s	vidhēyakartā (kartāsamānādhikaraṇa) Noun complements of kartā.
2.1	k2	karma (object/patient) karma is the locus of the result implied by the verb.
2.2	k2p	Goal/Destination The destination or goal is also taken as a karma. k2p is a subtype of karma k2. The goal or destination where the action of motion ends is a k2p.
2.3	k2g	gauna karma (secondary karma)
2.4	k2s	karmasamānādhikarana (object complement) The object complement is called as karma samānādhikaraṇa.
3	k3	karana (instrument) karanakāraka denotes the instrument of an action ex- pressed by a verb. The activity of karana helps in re- alizing the activity of the main action.
4.1	k4	sampradānakāraka is the recipient/beneficiary of an action. It is the person/object for whom the karma is

		intended.
4.2	k4a	anubhavakartā (Experiencer) The experiencer/perceiver in perception verbs such as seems, appear etc.
5.1	k5	apādāna (source) apādānakāraka indicates the source of the activity, i.e. the point of departure. A noun denoting the point of separation for a verb expressing an activity which in volves movement away from is apādāna.
6.1	k7t	kālādhikarana (location in time) ādhikaranakāraka is the locus of kartā or karma. It iṣ what supports, in space or time, the kartā or the karma.
6.2	k7p	dēśādhikarana (location in space)
6.3	k7	visayādhikaraṇ a (location elsewhere) Location other than time and place.
7	k*s or k*u	sādrṣya (similarity) This can be used for marking both similarity and com parison The tag is marked on the comparand in a com parative construction. Since the compared entity can compare with any kāraka, the tag includes a star. '*' in the tag name is a variable for whichever kāraka is the comparee of the comparand. Therefore, while mark ing the comparand (the compared entity), the * would be replaced by the appropriate kāraka label.
8.1	r6	saṣṭhī (possessive) The genitive/possessive relation which holds between two nouns.
8.2	r6-k1, r6-k2	kartā or karma of a conjunct verb (complex predicate)

9	adv	kriyāviśēsaṇ a (manner adverbs only) Adverbs of manner are marked as 'adv'. Note that the adverbs such as place, time, etc. are not marked as 'adv' under this scheme.
10	sent-adv	Sentential Adverbs Adverbial expressions that have entire sentence in their scope
11	rd	prati (direction) The participant indicating ‘direction’ of the activity.
12	rh	hētu (cause-effect) The reason or cause of an activity.
13	rt	tādarthya (purpose) The purpose of an action.
14	ras-k*	upapada-saha(kāra)kartā (associative) Two participants performing the same action but syn tactically one is expressed as primary and the other as its associate, the associate participant.
14.2	ras_neg	Negation in Associatives
15	rs	relation samānādhikarana (noun elaboration) Elements (normally clauses) which elaborate on a noun/pronoun
16	rsp	relation for duratives The durative expressions have two points – a point of starting and an end point. The expression as a whole may express time, place or manner etc. The tag 'rsp' shows the relation between the starting point and the end point of a durative.
17	rad	Address words In Marathi, expressions like ‘aho’, ‘he’ etc. are the address terms in Marathi.

18	nmod__relc, jjmod__relc, rbmod__relc	Relative clauses, jō-vō constructions in Hindi. (****Re-visit)
19	nmod	Noun modifier (including participles) An underspecified relation employed to show general noun modification without going into a finer type.
20	vmod	Verb Modifier Another underspecified tag. For some relations getting into finer subtypes is not yet possible. Such relations are annotated with slightly Underspecified tag.
21	jjmod	Modifiers of the adjectives
22	pof	Part of relation Part of units such as conjunct verbs.
23	ccof	Conjunct of relation Co-ordination and sub-ordination.
24	fragof	Fragment of
25	Enm	Enumerator

Appendix D: Sample of the Marathi Treebank

```
<Sentence id='1'>
1	((	NP	<fs name='NP' drel='k1:VGF'>
1.1	yA	DEM	<fsaf='yA,pn,m,sg,3,d,0,0' name='yA'>
1.2	prakArAne	NN	<fsaf='prakaran,n,NU,sg,3,o,ne,0' name='prakArAne'>
	))
2	((	NP	<fs name='NP2' drel='k7t:VGF'>
2.1	kAhI	CL	<fsaf='kAhI,nst,0,0,0,0,0,0' name='kAhI'>
2.2	velYa	NN	<fsaf='velYa,n,NU,sg,3,d,0,0' name='velYa'>
	))
3	((	NP	<fs name='NP3' drel='adv:VGF'>
3.1	KalYabalYa	RB	<fsaf='KalYabalYa,adv,0,0,0,0,0,0' name='KalYabalYa'>
	))
4	((	VGF	<fsaf=',,,,,,,' name='VGF'>
4.1	udAlI	VM	<fsaf="uda,v,f,sg,3,0,0,A' name='udAlI'>
4.2	।	SYM	<fsaf=',,,,,,,' name='।'>
	))
</Sentence>

<Sentence id='2'>
1	((	NP	<fs name='NP' drel='k1:VGF'>
1.1	jamAvAne	NN	<fsaf='jamAvAne,n,NU,pl,3,o,ne,0' name='jamAvAne'>
	))
2	((	NP	<fs name='NP2' drel='nmod:NP3'>
2.1	pANI	NN	<fsaf='pANI,n,NU,sg,3,d,0,0' name='pANI'>
2.2	puravaTA	NN	<fsaf='puravaTA,nst,0,0,0,0,0,0' name='puravaTA'>
	))
3	((	NP	<fs name='NP3' drel='k7p:VGF'>
3.1	kAryAlayAwIla	NN	<fsaf='kAryAlaya,n,NU,sg,3,d,0,0'
name='kAryAlayAwIla'>
	))
4	((	NP	<fs name='NP4' drel='nmod:NP6'>
4.1	telIPona	NN	<fsaf='telIPona,n,m,sg,3,d,0,0' name='telIPona'>
4.2	,	SYM	<fsaf=',,,,,,,' name=','>
	))
5	((	NP	<fs name='NP5' drel='nmod:NP6'>
5.1	mATa	NN	<fsaf='mATa,n,NU,sg,3,d,0,0' name='mATa'>
	))
6	((	NP	<fs name='NP6' drel='r6:NP7'>
6.1	AxIMcI	NN	<fsaf="AxI,pn,NU,pl,3,o,M,' name='AxIMcI'>
	))
7	((	NP	<fs name='NP7' drel='k2:VGF'>
7.1	wodaPoda	NN	<fsaf='wodaPoda,n,f,sg,3,d,0,0' name='wodaPoda'>
	))
8	((	VGF	<fsaf=',,,,,,,' name='VGF'>
8.1	kelI	VM	<fsaf='kara,v,f,sg,3,0,0,lI' name='kelI'>
8.2	।	SYM	<fsaf=',,,,,,,' name='।'>
	))
```

</Sentence>

<Sentence id='3'>
1 ((NP <fs name='NP' drel='k7p:VGF'>
1.1 yAmaXye RB <fsaf='yAmaXye,nst,0,0,0,0,0,0' name='yAmaXye'>
))
2 ((NP <fs name='NP2' drel='k7p:VGF'>
2.1 AsAmamaXIla NN <fsaf='AsAma,nst,0,0,0,0,maXIla,0' name='AsAmamaXIla'>
))
3 ((NP <fs name='NP3' drel='r6:NP4'>
3.1 xona QC <fsaf='xona,num,0,0,0,0,0,0' name='xona'>
3.2 mulIMcA NN <fsaf='mulIM,n,f,pl,3,o,cA,0' name='mulIMcA'>
))
4 ((NP <fs name='NP4' drel='k1:VGF'>
4.1 samAveSa NN <fsaf='samAveSa,n,m,sg,3,0,0,0' name='samAveSa'>
))
5 ((VGF <fsaf=',,,,,,,' name='VGF'>
5.1 Ahe VM <fsaf='ho,v,NU,sg,3,0,0,0' name='Ahe'>
5.2 । SYM <fsaf=',,,,,,,' name='।'>
))
</Sentence>

<Sentence id='4'>
1 ((NP <fs name='NP' drel='k1:VGF'>
1.1 halYuvAra NN <fsaf='halYuvAra,adj,0,0,0,0,0,0'>
))
2 ((VGF <fsaf=',,,,,,,' name='VGF'>
2.1 howA VM <fsaf='ho,v,m,sg,3,0,0,wA' name='howA'>
2.2 । SYM <fsaf=',,,,,,,' name='।'>
))
</Sentence>

<Sentence id='5'>
1 ((NP <fs name='NP' drel='r6:NP2'>
1.1 herewicI NN <fsaf='wi,pn,m,sg,2,d,cI,0' name='herewicI'>
))
2 ((NP <fsaf=',,,,,,,' name='NP2' drel='k1:VGF'>
2.1 kAlYajI NN <fsaf='kaYajI,n,f,sg,3,d,0,0' name='kAlYajI'>
))
3 ((NP <fs name='NP3' drel='k1s:VGF'>
3.1 GeNArA NN <fsaf='GeNArA,n,m,sg,3,d,0,0' name='GeNArA'>
))
4 ((VGF <fsaf=',,,,,,,' name='VGF'>
4.1 howA VM <fsaf='ho,v,m,sg,3,0,0,wA' name='howA'>
4.2 । SYM <fsaf=',,,,,,,' name='।'>
))
</Sentence>

<Sentence id='6'>
1 ((NP <fs name='NP' drel='adv:VGF'>
1.1 KaraMca JJ <fsaf='KaraMca,adj,0,0,0,0,0,0' name='KaraMca'>
))
2 ((NP <fs name='NP2' drel='r6:NP3'>
2.1 wyAMcaM NN <fsaf="we,pn,NU,pl,3,d,yAM,' name='wyAMcaM'>
))
3 ((NP <fs name='NP3' drel='k1:VGF'>
3.1 AyuRya NN <fsaf='AyuRya,n,NU,sg,3,d,0,0' name='AyuRya'>
))
4 ((NP <fs name='NP4' drel='k1s:VGF'>
4.1 Kupa CL <fsaf='Kupa,qutifier,0,0,0,0,0,0' name='Kupa'>
4.2 suKI JJ <fsaf='sukI,adj,0,0,0,0,0,0' name='suKI'>
))
5 ((VGF <fsaf=',,,,,,,' name='VGF'>
5.1 howaM VM <fsaf='ho,v,NU,sg,3,0,0,waM' name='howaM'>
5.2 । SYM <fsaf=',,,,,,,' name='।'>
))
</Sentence>

<Sentence id='7'>

1 ((NP <fs name='NP' drel='k4a:VGF'>

1.1 aniwalA NNP <fsaf='aniwA,n,f,sg,3,o,lA,0' name='aniwalA'>

))

2 ((NP <fsaf=',,,,,,,' name='NP2' drel='k1s:VGF'>

2.1 welugu NN <fsaf='welugu,n,f,sg,3,d,0,0' name='welugu'>

))

3 ((VGF <fs name='VGF'>

3.1 yewe VM <fsaf='ye,v,any,sg,3,d,0,we' name='yewe'>

3.2 । SYM <fsaf=',,,,,,,' name='।'>

))

</Sentence>

<Sentence id='8'>

1 ((NP <fs name='NP' drel='k1:VGF'>

1.1 vasaMwarAvAMnI NNP <fsaf='vasaMwarAva,n,m,sg,3,o,nI,0' name='vasaMwarAvAMnI'>

))

2 ((NP <fsaf=',,,,,,,' name='NP2' drel='k4:VGF'>

2.1 xukAnaxArAlA NN <fsaf='xukAnaxAra,n,m,sg,3,o,lA,0' name='xukAnaxArAlA'>

))

3 ((NP <fs name='NP3' drel='k2:VGF'>

3.1 pEse NN <fsaf='pEsA,n,m,pl,3,d,0,0' name='pEse'>

))

4 ((VGF <fs name='VGF'>

4.1 xile VM <fsaf='xe,n,NU,pl,3,0,0,le' name='xile'>

4.2 । SYM <fsaf=',,,,,,,' name='।'>

))

</Sentence>

<Sentence id='9'>

1 ((NP <fs name='NP' drel='k1:VGF'>

1.1 pujAryAne NN <fsaf='pujarI,n,m,sg,3,o,ne,0' name='pujAryAne'>

))

2 ((NP <fsaf=',,,,,,,' name='NP2' drel='r6:NP3'>

2.1 xevalYAce NN <fsaf='xevUlYa,n,m,sg,3,d,0,0' name='xevalYAce'>

))

3 ((NP <fs name='NP3' drel='k2:VGF'>

3.1 xAra NN <fsaf='xAra,n,NU,sg,3,d,0,0' name='xAra'>

))

4 ((VGF <fs name='VGF'>

4.1 uGadale VM <fsaf='uGada,v,any,sg,3,0,0,le' name='uGadale'>

4.2 । SYM <fsaf=',,,,,,,' name='।'>

))

</Sentence>

<Sentence id='10'>

1 ((NP <fs name='NP' drel='k1:VGF'>

1.1 banIne NNP <fsaf='banI,n,f,sg,3,o,ne,0' name='banIne'>

))

2 ((NP <fs name='NP2' drel='k3:VGF'>

2.1 surIne NN <fsaf='surI,n,f,sg,3,o,ne,0' name='surIne'>

))

3 ((NP <fsaf=',,,,,,,' name='NP3' drel='k2:VGF'>

3.1 peru NN <fsaf='peru,n,m,sg,3,d,0,0' name='peru'>

))

4 ((VGF <fs name='VGF'>

4.1 kApale VM <fsaf='kApa,v,f,pl,3,0,0,le' name='kApale'>

4.2 । SYM <fsaf=',,,,,,,' name='।'>

))

</Sentence>

<Sentence id='11'>

1 ((NP <fs name='NP' drel='k4a:VGF'>

1.1 ajayalA NNP <fsaf='ajaya,n,m,sg,3,o,lA,0' name='ajayalA'>

))

2 ((NP <fs name='NP2' drel='k1:VGF'>

2.1 wI DEM <fsaf='wI,pn,f,sg,3,d,0,0' name='wI'>

))

3 ((VGF <fsaf=',,,,,,,' name='VGF'>

3.1 AvadalI VM <fsaf='Avada,v,f,sg,3,0,0,lI' name='AvadalI'>

3.2 । SYM <fsaf=',,,,,,,' name='।'>

))

</Sentence>

<Sentence id='12'>

1 ((NP <fs name='NP' drel='k1:VGF'>

1.1 hA NN <fsaf='hA,pn,m,sg,3,d,0,0' name='hA'>

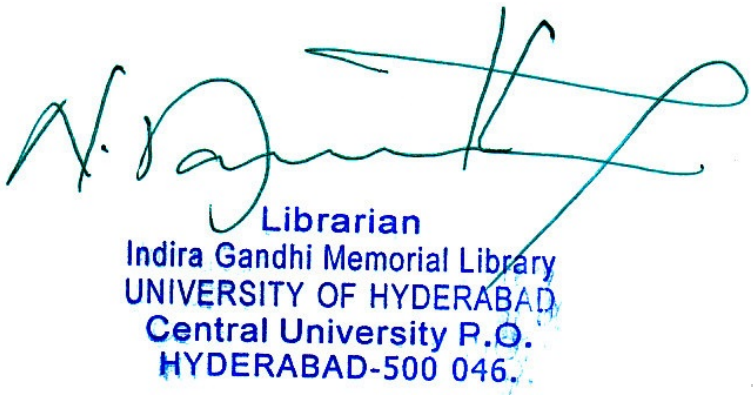
```
        ))
2       ((      NP      <fs name='NP2' drel='k1s:VGF'>
2.1     rUsUna  JJ      <fsaf='rUsUna,adj,0,0,0,0,0,0' name='rUsUna'>
        ))
3       ((      VGF     <fsaf=',,,,,,,' name='VGF'>
3.1     basalA  VM      <fsaf='basa,v,m,sg,3,0,0,lA' name='basalA'>
3.2     ।       SYM     <fsaf=',,,,,,,' name='।'>
        ))
</Sentence>

<Sentence id='13'>
1       ((      NP      <fs name='NP' drel='k7t:VGF'>
1.1     saMXyAkAlYI     NN      <fsaf='saMXyAkAlYa,n,m,sg,3,d,0,0'
name='saMXyAkAlYI'>
        ))
2       ((      NP      <fs name='NP2' drel='k7t:NP3'>
2.1     4       QC      <fsaf='4,num,0,0,0,0,0,0' name='4'>
2.2     vADUna          PRP     <fsaf='vADUna,adv,0,0,0,0,0,0' name='vADUna'>
2.3     17      QC      <fsaf='17,num,0,0,0,0,0,0' name='17'>
2.4     minitAMnI       PRP     <fsaf='minita,adv,0,0,0,0,0,0' name='minitAMnI'>
        ))
3       ((      NP      <fsaf=',,,,,,,' name='NP3' drel='k1:VGF'>
3.1     amAvasyA        NN      <fsaf='amAvasA,n,f,sg,3,d,0,0' name='amAvasyA'>
        ))
4       ((      VGF     <fsaf=',,,,,,,' name='VGF'>
4.1     suru    VM      <fsaf='suru,v,f,sg,3,0,0,0' name='suru'>
4.2     JAlI    VAUX    <fsaf="ho,v,f,sg,3,0,0,JA' name='JAlI'>
4.3     howI    VAUX    <fsaf='ho,v,f,sg,3,0,0,wI' name='howI'>
4.4     ।       SYM     <fsaf=',,,,,,,' name='।'>
        ))
</Sentence>

<Sentence id='14'>
1       ((      NP      <fs name='NP' drel='k1:VGF'>
1.1     Saraxane        NNP     <fsaf='Saraxa,n,m,sg,3,o,ne,0' name='Saraxane'>
        ))
2       ((      NP      <fs name='NP2' drel='k5:VGF'>
2.1     AmarAIwUna      NST     <fsaf='AmarAI,nst,0,0,0,0,wUna,0' name='AmarAIwUna'>
        ))
3       ((      VGNN    <fsaf=',,,,,,,' name='VGNN' drel='nmod:NP3'>
3.1     jANArYyA        VM      <fsaf='jA,v,m,sg,3,0,0,NArYyA' name='jANArYyA'>
        ))
4       ((      NP      <fs name='NP3' drel='k7p:VGF'>
```

```
4.1     raswyAvara      NN      <fsaf='raswA,n,m,sg,3,o,vara,0' name='raswyAvara'>
        ))
5       ((      NP      <fs name='NP4' drel='k7p:VGF'>
5.1     dAvIkade        NST     <fsaf='dAvIkade,nst,0,0,0,0,0,0' name='dAvIkade'>
        ))
6       ((      NP      <fs name='NP5' drel='k2:VGF'>
6.1     gAdI    NN      <fsaf='gAdI,n,f,sg,3,d,0,0' name='gAdI'>
        ))
7       ((      VGF     <fsaf=',,,,,,,' name='VGF'>
7.1     valYavalI       VM      <fsaf='valYava,v,f,sg,3,0,0,lI' name='valYavalI'>
7.2     |       SYM     <fsaf=',,,,,,,' name='|'>
        ))
</Sentence>

<Sentence id='15'>
1       ((      NP      <fs name='NP' drel='k7t:VGNF'>
1.1     woca    RB      <fsaf='woca,adv,0,0,0,0,0,0' name='woca'>
        ))
2       ((      NP      <fs name='NP2' drel='r6:NP3'>
2.1     gAdIcA          NN      <fsaf='gAdI,n,f,sg,3,o,cA,0' name='gAdIcA'>
2.2     mAgacA          NST     <fsaf='mAge,nst,0,0,0,0,cA,0' name='mAgacA'>
        ))
3       ((      NP      <fs name='NP3' drel='k1:VGNF'>
3.1     tAyara  NN      <fsaf='tAyara,n,m,sg,3,d,0,0' name='tAyara'>
        ))
4       ((      NP      <fs name='NP4' drel='k2:VGNF'>
4.1     eka     QC      <fsaf='eka,num,0,0,0,0,0,0' name='eka'>
4.2     moTA    JJ      <fsaf='moTA,adj,0,0,0,0,0,0' name='moTA'>
4.3     KilYA   NN      <fsaf='KilYA,n,m,sg,3,d,0,0' name='KilYA'>
        ))
5       ((      VGNF    <fsaf=',,,,,,,' name='VGNF' drel='vmod_rel:VGF'>
5.1     Gusuna          VM      <fsaf='Gusa,v,NU,sg,3,0,0,una' name='Gusuna'>
        ))
6       ((      NP      <fs name='NP5' drel='k1s:VGF'>
6.1     paMkcara        NN      <fsaf='paMkcara,adj,0,0,0,0,0,0' name='paMkcara'>
        ))
7       ((      VGF     <fsaf=',,,,,,,' name='VGF'>
7.1     JAlA    VM      <fsaf="ho,v,m,sg,3,0,0,JA' name='JAlA'>
7.2     |       SYM     <fsaf=',,,,,,,' name='|'>
        ))
</Sentence>

<Sentence id='16'>
1       ((      RBP     <fsaf=',,,,,,,' name='RBP' drel='adv:VGNF'>
1.1     sAXAraNa        RB      <fsaf='sAXAraNa,adv,0,0,0,0,0,0' name='sAXAraNa'>
        ))
2       ((      NP      <fs name='NP' drel='k1:VGNF'>
2.1     xona    QC      <fsaf='xona,num,0,0,0,0,0,0' name='xona'>
2.2     kilomItara      NN      <fsaf='kilomItara,num,0,0,0,0,0,0' name='kilomItara'>
```

```
        ))
3       ((      NP      <fs name='NP2' drel='k1:VGNF'>
3.1     pAra    NN      <fsaf='pAra,n,NU,sg,3,0,0,0' name='pAra'>
        ))
4       ((      VGNF    <fsaf=',,,,,,,' name='VGNF' drel='vmod_rel:VGF'>
4.1     kelyAvara       VM      <fsaf="kara,v,m,sg,3,0,0,lyA' name='kelyAvara'>
        ))
5       ((      NP      <fs name='NP3' drel='k1:VGF'>
5.1     wyAMnA          NN      <fsaf="we,v,NU,pl,3,o,yAM,' name='wyAMnA'>
        ))
6       ((      NP      <fs name='NP4' drel='k7p:VGF'>
6.1     AmarAI          NST     <fsaf='AmarAI,nst,0,0,0,0,0,0' name='AmarAI'>
        ))
7       ((      VGF     <fsaf=',,,,,,,' name='VGF'>
7.1     lAgalI  VM      <fsaf='lAga,v,f,sg,3,0,0,lI' name='lAgalI'>
7.2     ।       SYM     <fsaf=',,,,,,,' name='।'>
        ))
</Sentence>
```

Linguistic Inputs in building a Dependency Parser for Marathi: A Paninian Approach

by Yogesh Umale



Submission date: 28-Dec-2022 03:56PM (UTC+0530)
Submission ID: 1987078054
File name: Yogesh_U_Plagiarism_check_Doc_etd._2.docx (1.96M)
Word count: 21774
Character count: 120613

Linguistic Inputs in building a Dependency Parser for Marathi: A Paninian Approach

ORIGINALITY REPORT

4%	2%	2%	1%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Text Speech and Language Technology, 2003. Publication	1%
2	Submitted to University of Hyderabad, Hyderabad Student Paper	<1%
3	Submitted to Government Engineering College, Thrissur Student Paper	<1%
4	ltrc.iiit.ac.in Internet Source	<1%
5	"Handbook of Linguistic Annotation", Springer Science and Business Media LLC, 2017 Publication	<1%
6	Joakim Nivre. "Inductive Dependency Parsing", Springer Science and Business Media LLC, 2006 Publication	<1%
7	Submitted to Madan Mohan Malaviya University of Technology Student Paper	<1%

8	e-campus.iainbukittinggi.ac.id Internet Source	<1 %
9	powcoder.com Internet Source	<1 %
10	dspace.uohyd.ac.in Internet Source	<1 %
11	docplayer.net Internet Source	<1 %
12	Submitted to University of Pittsburgh Student Paper	<1 %
13	hdl.handle.net Internet Source	<1 %
14	it.scribd.com Internet Source	<1 %
15	www.duo.uio.no Internet Source	<1 %
16	"Coordinating Constructions", John Benjamins Publishing Company, 2004 Publication	<1 %
17	Amba Kulkarni, Sheetal Pokar, Devanand Shukl. "Chapter 6 Designing a Constraint Based Parser for Sanskrit", Springer Nature, 2010 Publication	<1 %

18	Giorgio Graffi. "200 Years of Syntax", John Benjamins Publishing Company, 2001 Publication	<1%
19	Submitted to University of Westminster Student Paper	<1%
20	Submitted to Chulalongkorn University Student Paper	<1%
21	Utpal Garain, Sankar De. "chapter 76 Dependency Parsing in Bangla", IGI Global, 2014 Publication	<1%

Exclude quotes	On	Exclude matches	< 14 words
Exclude bibliography	On		