

DESIGN AND DEVELOPMENT OF HINDI MORPHOLOGICAL ANALYZER

A DISSERTATION

*Submitted in partial fulfilment of the
requirements for the award of the degree*

of

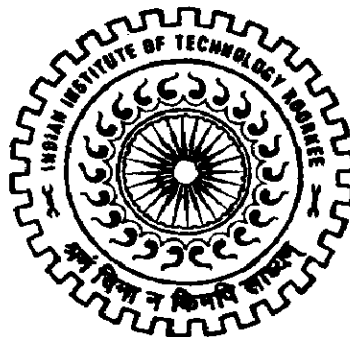
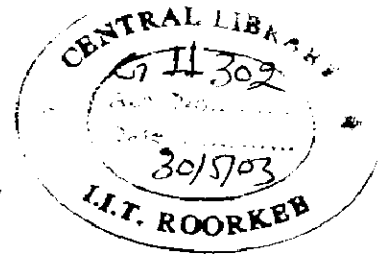
MASTER OF TECHNOLOGY

in

INFORMATION TECHNOLOGY

By

VISHAL NIJHAWAN



**ER & DCI
NOIDA**

**IIT Roorkee-ER&DCI, Noida
C-56/1, "Anusandhan Bhawan"
Sector 62, Noida-201 307**

FEBRUARY, 2003

CANDIDATE'S DECLARATION

I hereby declare that the work presented in this dissertation titled “**DESIGN AND DEVELOPMENT OF HINDI MORPHOLOGICAL ANALYZER**”, in partial fulfillment of the requirements for the award of the degree of **Master of Technology in Information Technology**, submitted in **IIT, Roorkee – ER&DCI Campus, Noida**, is an authentic record of my own work carried out during the period from August 2002 to February, 2003 under the guidance of **Dr. S.S. Agarwal**, Emeritus Scientist, CSIR; and Consultant, ER&DCI Noida.

The matter embodied in this dissertation has not been submitted by me for award of any other degree or diploma

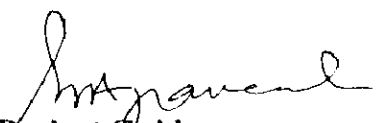
Date: 25th Feb, 2003

Place: Noida


(VISHAL NIJHAWAN)

CERTIFICATE

This is to certify that the above statement made by the candidate is correct to the best of my knowledge and belief.


Project Guide:

Dr. S. S. Agarwal

Emeritus Scientist, CSIR

Consultant, ER&DCI Noida

ACKNOWLEDGEMENT

Firstly I would like to thank **Prof. Prem Vratt**, Director, IIT Roorkee and **Sh. R. K. Verma**, Executive Director, ER&DCI, Noida for providing me the right kind of ambience to study in their institutes of great repute and providing me with this valuable opportunity to carry out this work. I also thank to **Prof. A.K. Awasthi**, Dean PGS&R, and Program Director, M.Tech.(IT), IIT Roorkee for providing the best of the facilities for the completion of this work and constant encouragement towards the goal.

My sincere thanks also go to **Prof. R.P. Agrawal**, Course Coordinator, M.Tech(IT), IIT Roorkee and **Mr. V.N. Shukla**, Course Coordinator, M.Tech.(IT), ER&DCI, Noida for giving useful suggestions and providing the adequate lab facilities and other facilities that led to the completion of this dissertation

I am highly indebted to my guide **Dr.S.S. Agrawal**, Emeritus Scientist, CSRI and Consultant, ER&DCI, Noida for his valuable suggestions, guidance and constant encouragement during the course of completion of the dissertation work. I also thank a lot to **Ms. Sunita Arora**, Sr. Project Engineer, Natural Language Processing Laboratory (ER&DCI), Noida for helping me a lot in achieving my goal.

My sincere thanks are due to **Mr. Munish Kumar**, Project Engineer, ER&DCI, Noida for the cooperation and help extended by him at every step in the course of completion of this dissertation.

I find myself short of words to thank my parents who have always been there by my side throughout my life.

Finally, a vote of thanks to all those who directly or indirectly, in some manner or the other, contributed towards the project work.



(VISHAL NIJHAWAN)

Enroll. No. -019057

ABSTRACT

Hindi Morphological Analyzer is a tool for extracting grammatical knowledge from the given Hindi text entered through the Roman Script. Morphological Analyzer splits a sentence into its constituent morphemes, which are the smallest units of the sentence carrying a meaning. These morphemes correctly describe the sentence grammatically. Thus, complete grammatical information of a sentence is obtained from the morphemes.

With Morphological Analyzer, a sentence is split into different grammatical parts of speech Patterns.

I have designed and implemented a computer application called "Hindi Morphological Analyzer". This application can be used to assist people in analyzing the grammatical knowledge in Hindi text.

The different constructs found through this Morphological Analyzer are Date, Number, Conjunction, Pronoun, Verb, Tense (whether Past, Present or Future), Adverb, and Adjective.

CONTENTS

CANDIDATE'S DECLARATION	(i)
ACKNOWLEDGEMENT	(ii)
ABSTRACT	1
1 INTRODUCTION	
1.1 Overview	3
1.2 Objective	4
1.3 Scope	4
1.4 Organization of the Thesis	4
2 LITERATURE SURVEY	7
2.1 Hindi Grammar	7
3 ANALYSIS OF THE PROBLEM	15
4 DESIGN OF THE PROPOSED SOLUTION	19
5. IMPLEMENTATION DETAILS	25
6. RESULTS AND THE DISCUSSION	37
7. CONCLUSION	45
REFERENCES	
APPENDIX A	
APPENDIX B	
APPENDIX C	

INTRODUCTION

1.1 Overview

In any language Translation Support System, there is a need of a component that could recognize the various parts of a speech. This recognizes all the words in the sentence according to their morphology. This component is called Hindi Morphological Analyzer. Morphological analysis is the basic enabling technique for many kinds of text processing. In a Translation Support System (TSS), the following steps are involved:

1. Recognition of word forms
2. Parsing
3. Translation

Morphological Analyzer does the recognition of the parts of speech in a given language text. Hindi Morphological Analyzer splits Hindi sentence entered through The Roman script into its constituent morphemes or words. A morpheme is a smallest unit of sentence conveying a meaning. These morphemes collectively describe the sentence grammatically. With morphological analyzer, a sentence is split out into different grammatical parts of Hindi text.

Here in this dissertation, I have done the work of recognition of the word forms of the Hindi text. This application uses grammatical rules, statistical rules and lexicon based techniques for determining the correct tag for Hindi text. The statistics include the common sentence patterns. This can be used to assist people in and analyzing the grammatical information in documents quickly and efficiently.

1.2 Objective of the Dissertation

The main objective is to find out the various parts of speech in a Hindi text written in Roman Script. These various parts of speech include:

1. Date
2. Number
3. Conjunction
4. Verb
5. Pronoun
6. Adjective
7. Noun
8. Adverb
9. Category of the sentence
 According to the meaning
 According to the structure
10. Tense of the sentence

1.3 Scope of the work

This application can be further extended to make a Translation support System that can translate text in one language to some other language. This can be used in assisting people in learning a language and analyzing the grammatical information. This can be a step between written text and knowledge base, because it can be used for grammatical knowledge acquisition.

1.4 Organization of the thesis

The report starts with a brief description of different parts of speech in Hindi language. The various parts of speech like Noun, Pronoun, Adjective, Adverb, Verb etc. are described according to their categories. In the chapter 3, problem has been analyzed.

The design procedure along with flow charts is described in the chapter 4. The implementation details are described in the chapter 5. The result screens are shown in chapter 6 and, finally, the last chapters contain Results and Conclusions.

LITERATURE SURVEY

Various books on Hindi grammar are studied as part of acquiring the knowledge of Hindi language. Romanization Scheme⁽¹⁾ (See Appendix) developed by IIT Kanpur has been adopted to write a sentence in Hindi language. The various papers related to the morphology analysis and sentence patterns are studied which are there in ER&DCI Noida. Various websites are surveyed in order to get more information regarding this field and also to learn the Hindi grammar.

2.1 Hindi Grammar

भाषा शब्दों की दुनिया हैं | भाषा उनमें शब्दों का
 प्रयोग अनेक रूपों में होता है | उन सभी रूपों का
 समझना -परखना व्याकरण का विषय है। अतः शब्दों का
 वर्गीकरण रण अनेक प्रकार से हो सकता है।
 निम्नलिखित चार प्रकार का वर्गीकरण महत्वपूर्ण है।
 (क) उद्गम के आधार पर
 (ख) रचना के आधार पर
 (ग) अर्थ के आधार पर
 (घ) प्रयोग के आधार पर

प्रयोग के आधार पर शब्दों का वर्गीकरण:
 (1) विकारी
 (2) अविकारी

(1) विकारी शब्द भिन्न-भिन्न परिस्थितियों में अपना रूप
 बदलते हैं इनके चार भेद होते हैं।

विकारी शब्द : विकारी शब्द चार प्रकार के होते हैं।

1. संज्ञा (Noun)
2. सर्वनाम (Pronoun)
3. विशेषण (Adjective)
4. क्रिया (Verb)

1. संज्ञा (Noun): A Noun is a word used as the name of a person, place or thing.
 किसी व्यक्ति, जीव, स्थान, वस्तु, विचार, भाव आदि के नाम को संज्ञा कहते हैं। उदाहरण : मानव, महात्मा गाँधी, घोड़ आदि।

संज्ञा के प्रकार (Types of Noun) :

- A. व्यक्तिवाचक संज्ञा (Proper Noun)
B. जातिवाचक संज्ञा (Common Noun)
C. भाववाचक संज्ञा (Abstract Noun)
D. समुदायवाचक संज्ञा (Collective Noun)
E. द्रव्यवाचक संज्ञा (Material Noun)

A. व्यक्तिवाचक संज्ञा (Proper Noun) : A proper noun is the name of some particular person or place.

जो संज्ञा किसी विशेष व्यक्ति, जीव, स्थान या वस्तु आदि का बोध कराये। उदाहरण : हिमालय, कालिदास आदि।

B. जातिवाचक संज्ञा (Common Noun) : A common noun is a name given in common to every person or thing of the same class or kind.

जो शब्द सा मान्य जाति का बोध कराये। उदाहरण : मनुष्य, हिन्दु, जैन, घोड़ा आदि।

C. भाववाचक संज्ञा (Abstract Noun) : An abstract noun is usually the name of quality, action or state considered apart from the object to which it belongs.

जो शब्द किसी विचार, भाव, गुण, दोष, स्वभाव, दशा, व्यापार को व्यक्त करें। उदाहरण : समाजवाद, प्रेम, मित्रता, चोरी आदि।

D. समुदायवाचक संज्ञा (Collective Noun) : A collective noun is the name of the number (or collection) of persons or things taken together and spoken of as one whole.

जो शब्द एक को न बताकर समूह या समुदाय को प्रकट करे। उदाहरण : भीड़, सभा, संघ, कुटुम्ब, परिवार, सेना, कक्षा आदि।

E. द्रव्यवाचक संज्ञा (Material Noun) : राशि या ढेर के रूप में पायी जाने वाली वस्तुओं को सूचित करने वाले संज्ञा शब्द द्रव्यवाचक कहलाते हैं।

उदाहरण : सोना, पानी, वायु, चाँदी, ताँबा आदि।

2. सर्वनाम (Pronoun) : A pronoun is a word used instead of a noun.

जो शब्द संज्ञा के स्थान पर आकर उसकी पुनरावृत्ति को रोकते हैं, वे सर्वनाम कहलाते हैं। उदाहरण : यह - वे, वह - वे, आप, तू - तुम

सर्वनाम के प्रकार (Types of Pronoun):

A. पुरुषवाचक (Personal Pronoun): The pronoun I and we which denote the person or persons speaking, are said to be personal pronoun of the first person.

उदाहरण : मैं , हम , वह , वे , यह , ये ।

a)उत्तम पुरुष (First Person):The pronoun I and we which denote the person or persons speaking are said to be personal pronoun of the first person.

उदाहरण : मैं , हम

b)मध्यम पुरुष (Second Person):The pronoun you which denote the person or persons spoken to are said to be personal pronoun of the second person.

उदाहरण :

c)अन्य पुरुष (Third Person):The pronoun he/she and they which denote the person or persons spoken of are said to be personal pronoun of the third person.

उदाहरण : वह , वे , यह , ये ।

B.निश्चयवाचक (Demonstrative Pronoun):It will be noticed that the pronouns in italics are used to point out the objects to which they refer, and are, therefore, called Demonstrative Pronouns.

उदाहरण : यह , ये , वह , वे ।

C.अनिश्चयवाचक (Indefinite Pronoun):All the pronouns in italics refer to persons or things in general way, but do not refer to any person or thing in particular.

उदाहरण : जो , सो , जैसा , वैसा , जिसे , वही ,जिसकी ,उसकी ।

D.प्रश्नवाचक (Interrogative Pronoun):It will be noticed that the pronouns in italics are similar in form to Relative pronouns.But the work which they do is different they are here used for asking question called interrogative Pronouns.

उदाहरण : कौन , क्या ।

E.निजवाचक (Reflexive Pronoun):When the action done by the subjectturns back(reflects)upon the subject, these noun are called reflexive pronoun.

उदाहरण : आप , स्वतः , अपने आप ।

3. विशेषण (Adjective) : A word as a word used with a noun to add something for its meaning.

जो शब्द संज्ञा अथवा सर्वनाम की विशेषण को बताते हैं , वे विशेषण कहलाते हैं ।

उदाहरण : अच्छा , बुरा , खोटा , खरा , पतला , मोटा ।

विशेषण के प्रकार (Types of Adjective) :

A.गुणवाचक (Adjective of quality)

B.संख्यावाचक (Neumaryl Adjective)

C.परिमाणवाचक (Adjective of Quantity)

D.सार्वनामिक (Interrogative Adjectives)

E.संकेतवाचक (Demonstrative Adjective)

F.व्यक्तिवाचक (Personal Adjective)

A गुणवाचक: (Adjective of quality): shows the kind or quality of a person or thing.

- a.) गुण : अच्छा , बुरा , भला , सभ्य , शिष्ट , सुन्दर
b.) दोष : खराब , बुरा , दुष्ट , अशिष्ट
c.) काल : पुराना , नया , नूतन , क्षणिक , दैनिक
d.) स्थान : मद्रासी , जयपुरी , पहाड़ी , जापानी
e.) गंध : सुगंधित , खुशबुदार , दुर्गंधित
f.) दिशा : पूर्वी , पश्चिमी
g.) दशा : गीला , सूखा , सख्त , चिकना
h.) रंग : सफेद , नीला , पीला , काला , हरा , बदांगी
i.) आकार : बौना , लम्बा , गोल , चौकोर , शूलाकार
j.) स्पर्श : कोमल , कठोर , खुरदरा , सख्त
k.) स्वाद : मधुर , कटु , तीखा , कषाय

B संख्यावाचक : (Numeral Adjective): show how many persons or things are meant, or in what order a person or thing stands.

1. निश्चितवाचक (Definite Numeral Adjectives)

उदाहरण : एक , दो , तीन , सौ , चौथे ।

2. अनिश्चितवाचक (Indefinite Numeral Adjectives)

उदाहरण : कुछ लोग , सब आदमी , थोड़े से ।

C परिमाणवाचक (Adjective of Quantity): show how much of thing is meant.

उदाहरण : दो मीटर , कम , अल्प , किंचित , जरा ।

D सार्वनामिक : What, which and whose when they are used with nouns to ask questions are called Interrogative adjectives.

जो विशेषण सर्वनाम से बने ।

उदाहरण : कौन बालक , यैसा बालक , कितना पानी , वह बालक ।

3 क्रिया (Verb) : A verb is a word used to tell or assert something about some person or thing.

जिन शब्दों से किसी काम का होना या करना प्रकट हो , उसे

क्रिया कहते हैं ।

उदाहरण : जाता है , गया , जायेगा , पढ़ रही है ।

क्रिया के प्रकार (Types of Verb):

A. अकर्मक क्रिया (Intransitive Verb)

B. सकर्मक क्रिया (Transitive Verb)

प्रयोग के आधार पर क्रिया के भेद (On the basis of use) :

1. सामान्य क्रिया (Simple Verb)
2. संयुक्त क्रिया (Compound Verb)
3. नाम धातु क्रिया (Nominal Verb)
4. प्रेरणार्थक क्रिया (Causative Verb)
5. पूर्णकालिक क्रिया (Absolute Verb)
6. अपूर्ण क्रिया (Incomplete Verb)

अकर्मक क्रिया (Intransitive Verb)
उदाहरण : दौड़ना , भागना , होना , हारना , जीतना ।

सकर्मक क्रिया (Transitive Verb)
उदाहरण : अशोक पुस्तक को पढ़ता है । (पुस्तक पढ़ी जाती है ।)

अपूर्ण क्रिया (Incomplete Verb)
उदाहरण : यह है , यह है , यह था , मैं हूँगा ।

संयुक्त क्रियाएँ (Compound Verb)

a)आरम्भबोधक
उदाहरण : पढ़ने लगा हूँ ।

b)अवकाशबोधक
उदाहरण : खेलने दो ।

c)विवशताबोधक
उदाहरण : खाना पड़ा ।

d)समाप्तिबोधक
उदाहरण : लिख चुका ।

e)शक्तिबोधक
उदाहरण : ठठ सका ।

f)नित्यताबोधक
उदाहरण : आया करता है ।

g)इच्छाबोधक
उदाहरण : बोलना चाहता है ।

h)तत्कालबोधक
उदाहरण : लिखे देता हूँ ।

i)समाप्त्यबोधक
उदाहरण : लिख रहा है ।

j)पुनरुक्तार्थक
उदाहरण : खाता -पीता है ।

k)पूर्णताबोधक
उदाहरण : कर डाला ।

l)अप्रियताबोधक
उदाहरण : आन मरा ।

(2) अविकारी शब्द वे होते हैं जिनमें प्रयोग के कारण कोई परिवर्तन नहीं आता। इनके भी चार भेद होते हैं ।

अविकारी शब्द : अविकारी शब्द चार प्रकार के होते हैं ।
1. क्रिया -विशेषण (Adverb)

- 2.सम्बन्ध बोधक (Preposition)
 3.समुच्चय बोधक (Conjunction)
 4.विस्मयादि बोधक (Interjection)

1. क्रिया -विशेषण (Adverb) : An Adverb is a word which modifies the meaning of a verb, an adjective or another adverb.

जिन शब्दों से क्रिया की विशेषता प्रकट होती है, वे क्रिया -विशेषण कहलाते हैं ।
 उदाहरण : धीरे -धीरे , जल्दी , ठीक -ठाक , आज , तुरन्त आदि ।
 क्रिया -विशेषण के प्रकार (Types of Adverb):

a)स्थानवाचक (Adverbs of place)

a1)स्थिति सूचक

उदाहरण : यहाँ , वहाँ , कहीं , जहाँ , आगे , सामने ।

a2)दिशा सूचक

उदाहरण : पूर्व की ओर , इधर , उधर ।

b)कालवाचक (Adverbs of time)

उदाहरण : आज , कल , परसों , अभी , थोड़ी देर , जब , तब ।

c)परिमाणवाचक (Adverbs of degree or quantity)

उदाहरण : बहुत , थोड़ा , अधिक , कम , अल्प , ज्यादा ।

d)रीतिवाचक (Adverbs of manner)

उदाहरण : ऐसे , कैसे , जैसे , वैसे , यों , जल्दी ।

सम्बन्ध बोधक (Preposition) : A preposition is a word placed before a noun or a pronoun to show in what relation the person or thing denoted by it stands in regard to something else.

जो शब्द संज्ञा अथवा सर्वनाम का सम्बन्ध वाक्य के अन्य शब्दों से स्थापित करें , वे सम्बन्ध बोधक कहलाते हैं ।
 उदाहरण : बिना , सिवाय , समान , नाई , तुल्य , सरीखा आदि ।

2. समुच्चय बोधक (Conjunction) : A conjunction is a word which merely joins together sentences , and sometimes words.

जो शब्द शब्दों , शब्दार्थों या वाक्यों को परस्पर जोड़ने का काम करते हैं ।
 उदाहरण : जैसे , और , पर , कि , यदि , किन्तु , परन्तु , लेकिन , अर्थात् , तथा आदि ।

3. विस्मयादि बोधक (Interjection) : An interjection is a word which expresses some sudden feeling.

जो शब्द विस्मय , अय , घृणा क्रोध आदि मनोभावों को व्यक्त करें ।

उदाहरण : हैं !, अरे !, ओ! , छि : !, घत् !।
 विस्मय : हैं !, हैं !, ओह !, ओ हो !, अहो !।
 शोक : हाय !, हा !, हा हा !, उफ !, ओह !।
 भय : हाय !, हा !, ओह !, बाप रे !।
 हर्ष : आहा !, वाह !, क्या कहने !, घत् य हैं !, आनन्द आ गया !।
 क्रोध : चुप !, हट !, परे हट !, घत् !, मर परे !, हट परे !।
 घृणा : धिक्कार !, थू: !, छि :- छि : !, छी: !।
 स्वीकृति : जी !, हाँ जी !, जी हाँ !, ठीक है !, ठीक है जी !।
 लज्जा : छि :- छि : !, हाय मरी !।

ANALYSIS OF THE PROBLEM

This Hindi Morphological Analyzer is required to recognize the various parts of speech which are there in the entered text. The various parts of speech that could be looked for in the entered sentence could be:

1. Date
2. Number
3. Conjunction
4. Verb
5. Pronoun
6. Adjective
7. Noun
8. Adverb
9. Category of the Sentence
According to the structure
According to the Meaning
10. Tense of the Sentence
Whether Past, Present or Future

1. Date Identification: Any Date, month or day which is present there in the sentence given by the user is to be identified.

Suppose there is a sentence like

“rAma 25 janavarI 2003 ko xilli jA rahA hE |”

Then in this sentence '25 janavarI' will be recognized as Date.

If the sentence is

“vaha somavAra, JanavarI 25 ko 2003 logoM ke sAWa xilli jA rahA hE |

then in this sentence somavAra, janavarI 25 will be recognized as Date and 2003 will not be included in it.

2. Number Identification: Any number present in the sentence that is given by the user, like x0, cAra will be recognized as a number by the morphological analyzer.

Suppose there is a sentence given by user like

“mohana bIsa log0M ke sAWa bIsa janavrI ko byAsa jA raha hE]”

Then in this sentence bIsa in ‘bIsa log0M’ will be recognized as a Number and bIsa janavrI will be recognized as a Date.

Any number whether entered in numerals like 25, 75, 100 or entered in text like ‘paccIsa’, ‘Pichawwara’, ‘sO’ will be recognized as a number.

3. Conjunction Identification: Any conjunction present in the sentence that is given by the user, like ‘Ora’ aWavA’, ‘yA’, ‘paranwu’ should be recognized as a conjunction by the morphological analyzer.

Suppose there is a sentence given by user like

“paras0M rAma Ora mohana KanA KA rahe We]”

Then in this sentence ‘Ora’ between ‘rAma Ora mohana’ will be recognized as a conjunction.

If the user gives a sentence like

“mohana xillI ja raha WA paranwu use bIca me hI vApisa AnA padZA]”

In the above sentence ‘paranwu’ will be recognized as a conjunction.

4. Verb Identification: Verb present in the sentence given by a user will be identified by the morphological analyzer.

Suppose there is a sentence written in Roman Script like:

“MEM jA raha huz]”

Then in this sentence, the morphological analyzer should recognize ‘jA raha huz’ as verb and ‘jAnA’ is the root verb.

5. Pronoun Identification: Any pronoun present in the sentence like ‘wuma’, ‘Apa’, ‘mEM’ should be recognized as a pronoun by the morphological analyzer. Suppose there is a sentence like “kyA Apa kala xillI jA rahe hEM ?”

Then in the above sentence 'Apa' should be recognized as a pronoun by the morphological analyzer.

6 Adjective Identification: Any adjective present in the sentence that is given by the user, like 'kaIA' 'badZA' 'CotA' should be recognized as an adjective by the morphological analyzer.

Suppose there is a sentence given by user like

"rAma badZA naWaKaWa hE |"

Then in this sentence 'badZA' should be recognized as an adjective.

7. Noun Identification: Any noun present in the sentence that is given by the user, like 'kapadZA', 'maSIna', 'purajA', 'ladZakA', 'ladZakI' should be recognized as a noun by the morphological analyzer.

Suppose there is a sentence given by user like

"rAma kala kapadZA KarIxane jAyegA |"

Then in this sentence 'rAma' and 'kapadZA' should be recognized as a noun.

If the user gives a sentence like

"mohana xilli ja rahA WA paranwu use bIca me hI vApisa AnA padZA |"

then in the above sentence 'mohana' and 'xilli' should be recognized as nouns.

8 Adverb Identification: Any adverb present in the sentence that is given by the user, like 'kala' 'Aja' 'parasoM', 'jalXI-jalXI' should be recognized as an adverb by the morphological analyzer.

Suppose there is a sentence given by user like

"rAma kala Ora parasoM se sakola nahIM AyegA |"

Then in this sentence 'kala' and 'parasoM' should be recognized as adverbs.

9. Category of Sentence: The category of the sentence that is entered by the user, should be recognized according to the structure of the sentence and the meaning of the sentence.

If the sentence is made up of two parts each separately being an individual sentence then it will be recognized as a compound sentence otherwise it will be recognized as a simple sentence.

According to the meaning the sentence could be defined in various categories like:

Sentence showing doubt 'sanxehaboXaka vAkyA'

Interrogative Sentences 'praSanaboXaka vAkyA'

Exclamatory Sentences 'vismyAxiboXaka vAkyA' etc.

Suppose if the sentence given by the user is

"rAma A rahA hE Ora mohana jA rahA hE |"

Then the sentence will be shown as a compound sentence by the morphological analyzer.

If the sentence is

"rAma ne jana WA paranwu vaha jA na saA"

The morphological analyzer should tell the above sentence as a Compound Sentence.

Suppose the sentence is

"agara wuma steSana cale jAwe wo Sayaxa wumhe gAdZI mila jAwI"

Since there is presence of 'agara' and 'Sayaxa', both of which show some doubt in some happening, the morphological analyzer should tell that the above sentence is showing some doubt.

10. Tense of the Sentence: The type of the sentence to which a sentence belongs to should be identified by the morphological analyzer. The sentence could be Past, Present or Future. The proper category of the tense to which the sentence belongs to should be shown.

Suppose if the sentence entered is

"rAma jA rahA hE"

Then the tense to which the sentence belongs to is shown as 'Present Tense'.

DESIGN OF THE PROPOSED SOLUTION

As this Morphological analyzer has to recognize the parts of speech of a Hindi sentence, then it should take care of each word in the given sentence. As the parts of speech like noun, pronoun, conjunction, adverb etc. can be anywhere in the sentence, it should check each word for its particular category. This morphological analyzer should also be capable of finding the presence of date and number in the text.

First the sentence can be checked for having a date into it. For this each word of the sentence can be separately checked whether it could be in the standard date format. If the checked word is in standard date format, then it can be tagged as date. If there is not any word in standard date format, then it may be possible that the date entered could be in the numeral format like '25 janavarI 2003' or 'paccIsa janavarI xo hajAra wIna' or 'paccIsa janavarI 2003' or '25 janavarI xo hajAra wIna'. For this we have to check each word whether it is in numerals and if numeral, is it followed by a month, year. The word can also be checked against the numbers written in text like 'paccIsa' which can be stored in a file consisting of the words in number for the checking purpose. In this way the date formed by words can be found out

Like date, each word can be checked to be a number. A word can be considered as a number if it is not a date, and either it is in numeral or in text form. This can be checked for each word of the sentence. Since date can be a number too, it is for only this purpose that date should be checked out first in a given text.

After finding the date and number in a given sentence, we have to check the verb present in the sentence. Since every language has some patterns that help to make sentences, we have to find out the most common patterns used in Hindi language that are normally used while making the sentences while writing or speaking. The patterns can be like the ones given ahead:

"_we cale jA rahe hoMge"

"_nI padaZ sakawI WI"

"_wI jA rahI hogI"

"_wI raha gayI ho"

These patterns can be stored in some file for matching against the sentences that are entered to check if they have the same pattern that is lying in the pattern database. Like patterns we will have to store the common verbs occurring in the Hindi language according to some rules. Then using these patterns and the verbs stored in our database, we can find out the verb in the given Hindi text. (This feature depends upon the vastness of the database that we have created, If for some reason we have not stored the verb that is present in the sentence then this morphological analyzer won't be able to tell the verb present in the given text.

After finding the date, number and verb, we can check out remaining parts of speech like noun, pronoun, adjective, adverb, and tense etc. present in the sentence.

Since there are a limited number of pronouns and adverbs we can store them in some file and each word of the entered text can be checked against the words present in these files. If the word in the sentence matches with the one present in these files then we can easily find out the pronoun and adverbs in the given sentences.

The same rule can be applied for finding the adjectives and nouns present in the sentence. But this like the verbs and patterns will be applicable to only those nouns and adjectives that are presently in our database. So, it should be kept in mind to make the database as large as possible to cover the most common nouns and adjectives. But there can not be all the words present in the database, so this application will be dependent upon the number of words already kept in the database or simply saying the knowledge base of the application.

The finding of these parts of speech is a mere lexicon look up in the database.

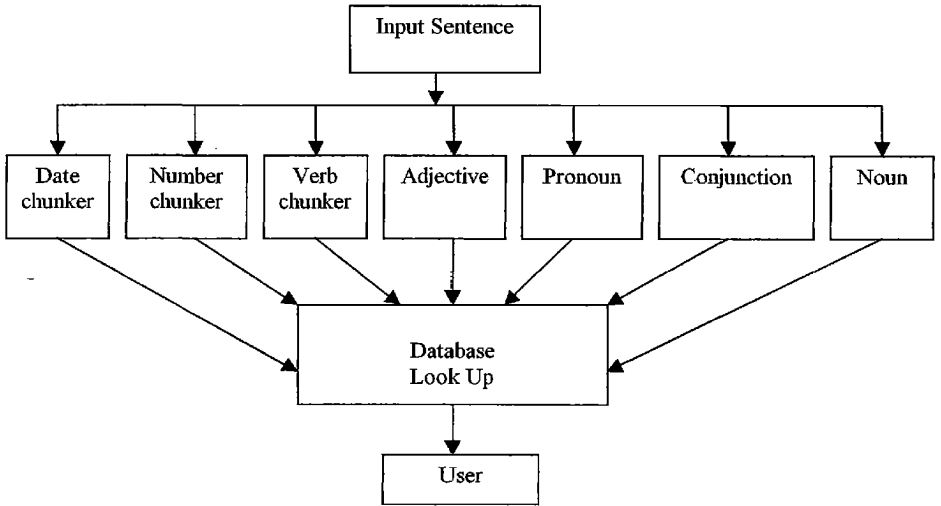


Figure 4.1 Structure Chart of Morphological analysis

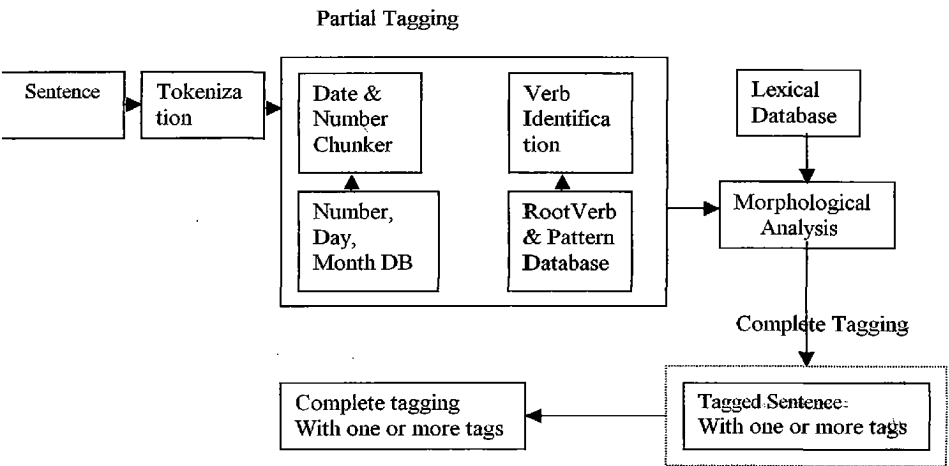


Figure 4.2 Block Diagram for Morphological analyzer

So, according to the functionality of the application, the design can be made of having the following different modules.

1. Tokenization: Tokenization process breaks input sentence (text) into words. The words that we get after splitting process are called tokens.

2. Chunking: This module will consist of the following sub modules

Chunking for Date and Number: In this module, we will pass the complete tokenized sentence as input and get output sentence tagged as Date and Number.

Suppose the sentence entered is

“ kala paccIsa janavarI 2003 ko sAwa AxamI xillI jA rahe hEM |”

Output of the Date chunker module will be

paccIsa janavarI 2003 [Date]

and the output of the Number chunker sub-module will be

sAwa [Number]

3. Verb Chunker: This module will be used to find out the verb present in the sentence

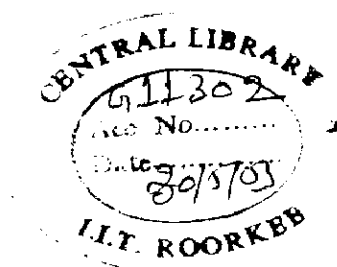
4. Modules for finding Adjective, Noun, Pronoun, Adverb and Tense: Each module will do the look up in the database for the relevant category of the word.

IMPLEMENTATION DETAILS

For implementation of this morphological analyzer we have to create some database in which we can store some knowledge base that can be used by the morphological analyzer for tagging the parts of speech.

For this the following parts/words need to be stored.

1. Numbers in text form like : eka, xo, wlna, cAra etc. This can be stored in a table in the database. We have stored this in the **Number** table. This table contains numbers in text written in Roman Script. For Date findings to take place months and days in roman script are also stored in the database as **Days** and **Months** tables.
2. Common Patterns and verbs. The patterns are stored in the **Pattern.txt** file and the verbs are stored in the database as **RootVerbEntry** table. The various forms extended from the **RootVerbEntry** are stored in a separate table called **RootVerbExts**.
The root verbs are stored along with a paradigm number which is a number that is given for a type of verb.
Like the verb 'KA' is given the paradigm number 1.
The table **RootVerbExts** containing the verb extensions has several fields like verb, root verb, gender, singular or plural and extended form.
3. Adjectives are stored in the database as a table **Adjective** having two fields: adjective and its category.
4. Like adjectives, Adverbs are also stored in a table **Adverb** in the database. The table has only a single field of adverb only.



Name
eka

Table 5.1 Number Table-It contains numbers in roman Scheme like ‘eka’.

Days
somavAra
maMgalavAra

Table 5.2 Days Table-It contains day in roman Scheme like ‘somavAra’.

Verb	Paradigm No
Uta	11
jA	3

Table 5.3(i) RootVerbEntry Table It contains the Verb and Paradigm Number of that Verb. Like verb ‘jA’ has a paradigm number 3

Verb	RootVerb	Gender	S/P	Tense	RootVerbexts

Table 5.4(ii) RootVerbExts Table-It contains all the possible generated forms and various statuses like tense, gender etc from the root word.

Adjective	Category
badZA	Quality

Table 5.4 Adjective Table-This table contains adjective and category of that adjective.

Adverb
aBI

Table 5.5 Adverb Table-This table contains adverbs like ‘aBI’.

Noun	Gender	Category
ladZaki	f	jAwIvacaka

Table 5.6 Noun Table-This table contains the nouns like ‘ladZaki’

These tables are stored in the MS-Access. The implementation of the code modules is done in the MS Visual Basic 6.0

5. Nouns are also stored in the database in a table called Noun. It has three fields called noun, gender and its category.

6. As Pronouns and Conjunctions are limited so they need not be stored in the database. They are stored in text files.

All these tables, which are stored in the database, are shown as on the left page.

For the implementation of the code module of date chunker the flow diagram in the figure 5.1 can be used.

The Pseudo-Code is given as

1. START
2. READ THE SENTENCE
3. SPLIT INTO WORDS AND STORE IN AN ARRAY
4. TAKE NEW WORD FROM THE WORDS ARRAY
5. IS END OF SENTENCE REACHED, IF YES GOTO 7 ELSE GOTO 6
6. IS WORD A DATE, A NUMBER, DAY OR MONTH
IF YES TAG IT AS WHAT IT IS, STORE INTO A TEMPORARY TABLE
AND GOTO 4
7. CHECK TEMPORARY TABLE AND CHUNK. IS DATE FOUND. IF YES
GOTO 8 ELSE GOTO 9
8. SHOW DATE
9. STOP

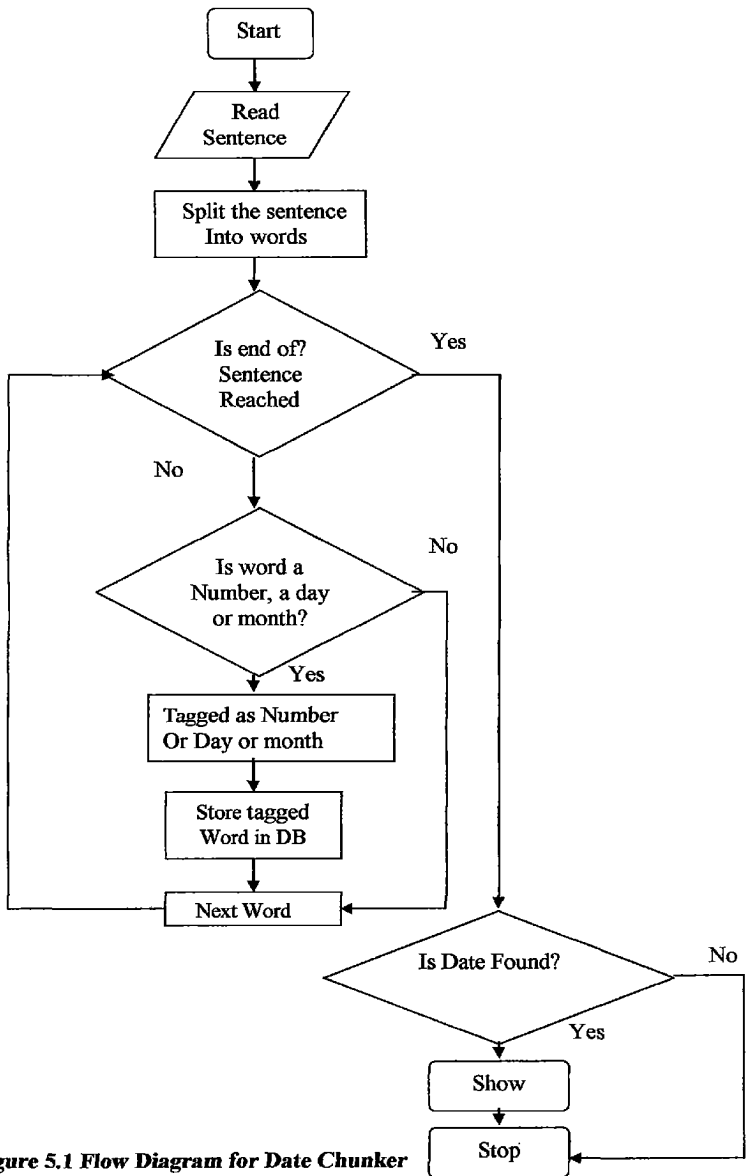


Figure 5.1 Flow Diagram for Date Chunker

Flow Chart for the Number Chunker is shown on the left page in figure 5.2

Pseudo-Code for Number Chunker is given as

1. START
2. READ THE SENTENCE
3. SPLIT INTO WORDS AND STORE IN AN ARRAY
4. TAKE NEW WORD FROM THE WORDS ARRAY
5. IS END OF SENTENCE REACHED IF YES GOTO 7 IF NO GOTO 6
6. IS WORD A NUMBER IF YES TAG AS A NUMBER AND STORE IN A TEMPORARY TABLE AND GOTO 4
7. CHECK TEMPORARY TABLE. IS NUMBER FOUND IF YES GOTO 8 ELSE GOTO 8
5. SHOW NUMBER
6. STOP

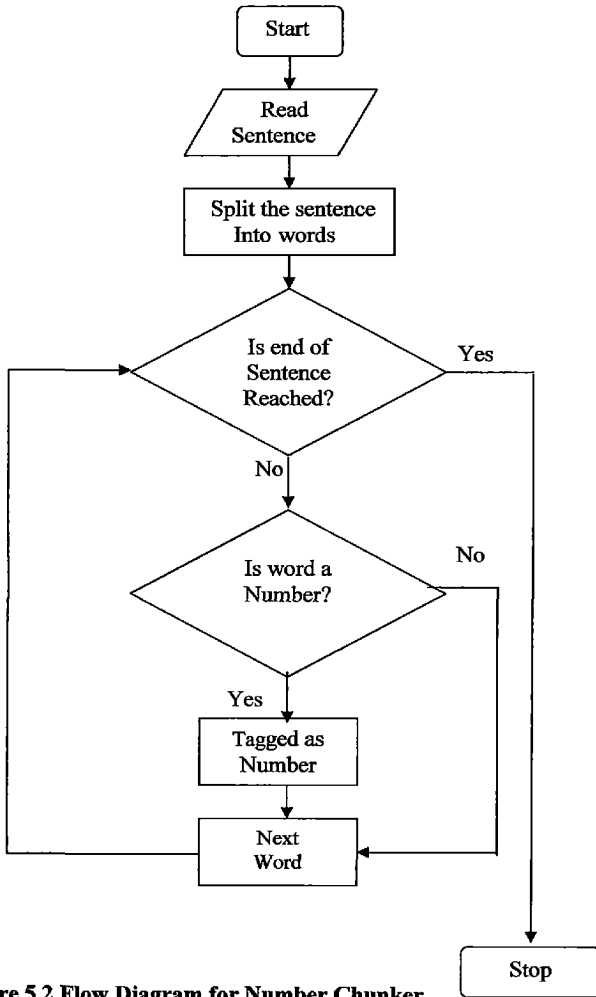


Figure 5.2 Flow Diagram for Number Chunker

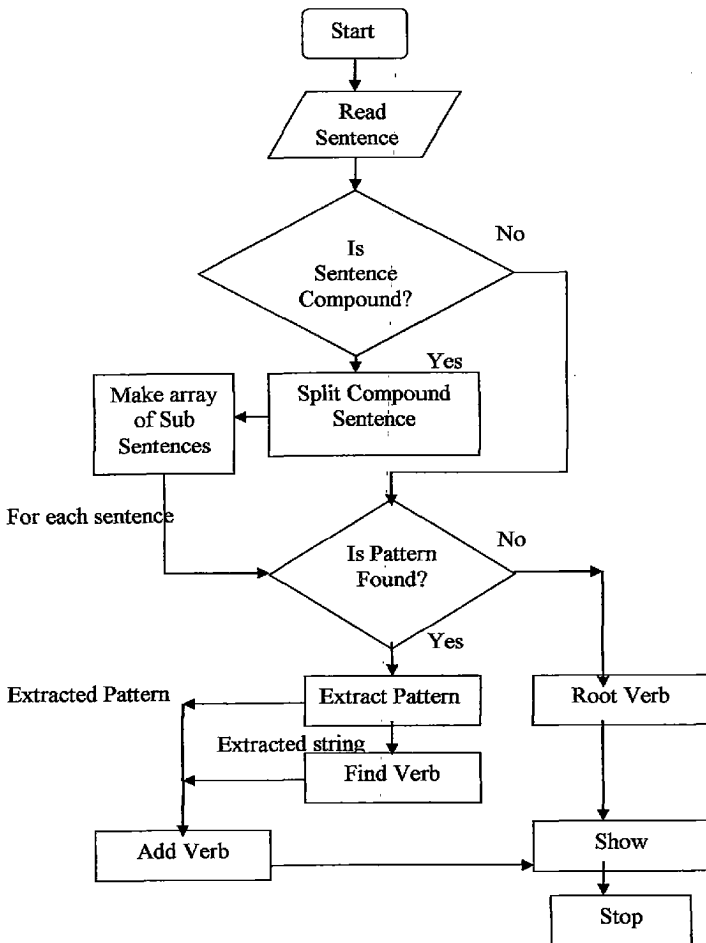


Figure 5.3 Flow Chart for Verb Identification

Pseudo Code for Verb Identification:

1. **START**
2. **READ SENTENCE**
3. **IS SENTENCE COMPUND IF YES MAKE AN ARRAY OF SUB SENTENCES IF NO GOTO 4 I**
4. **FOR EACH SUB SENTENCE**
 - I. **IS PATTERN FOUND IF YES GOTO II IF NO GOTO 5**
 - II. **EXTRACT PATTERN AND EXTRACT STRING BEFORE PATTERN**
 - III. **FIND VERB IN THE EXTRACTED STRING**
 - IV. **GIVE VERB BY ADDING PATTERN AND ROOT VERB**
 - V. **SHOW VERB**
5. **FIND ROOT VERB**
6. **SHOW VERB GOTO 7**
7. **STOP**

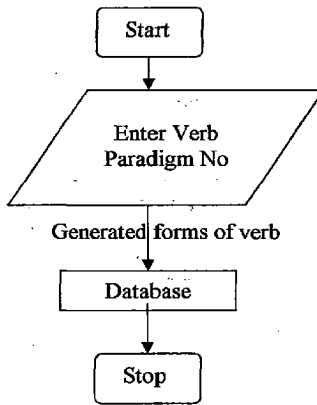


Figure 5.4 Flow Chart for Verb Entry

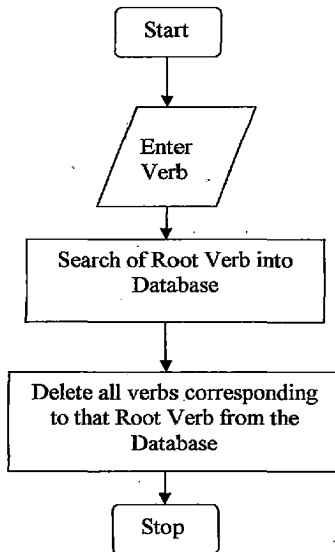


Figure 5.5 Flow chart for Verb Deletion

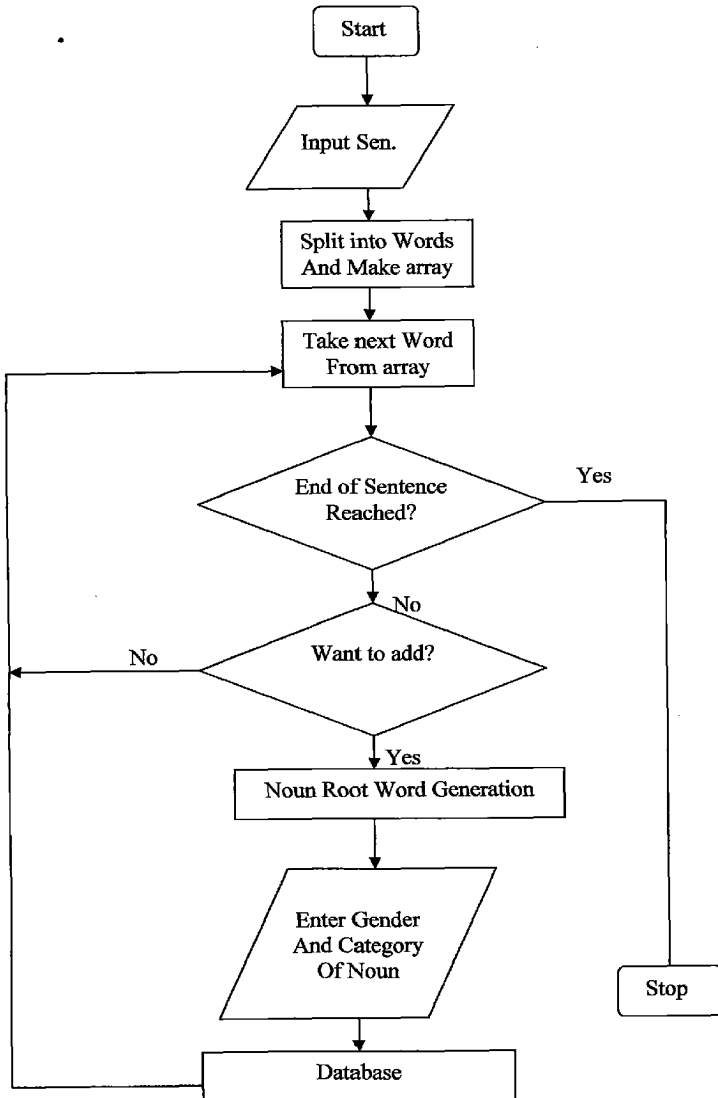


Figure 5.6 Flow Chart for Noun Entry

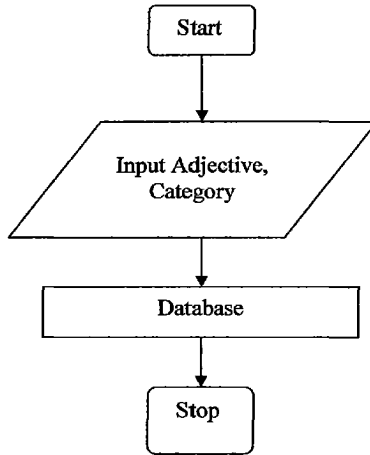


Figure 5.4 Flow Chart for Adjective Entry

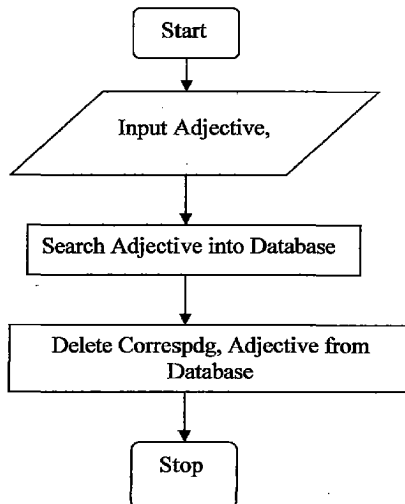


Figure 5.5 Flow Chart for Adjective deletion

RESULTS AND THEIR DISCUSSIONS

As the user starts the application, he/she is presented with a GUI of the Hindi Morphological Analyzer. This is shown in figure 6.1

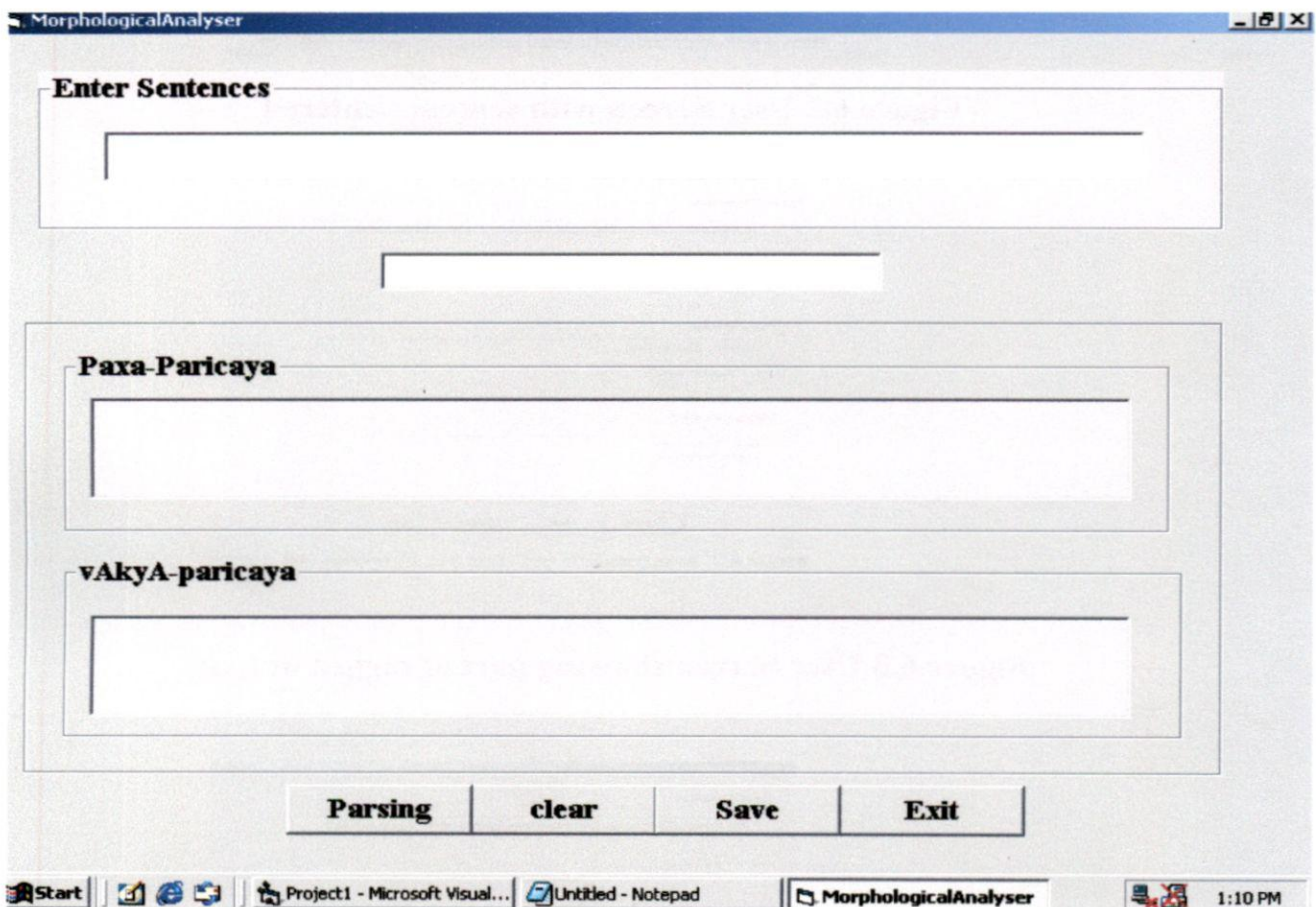


Figure 6.1 First Screen Presented to the User

The user can enter the sentence in the text box which has a label 'Enter Sentence' as shown in figure 6.2 . As the user presses the 'Parse' button, The analysis of the entered sentence starts. This is shown in figure 6.3. The tagged out is shown in the figure 6.3 and 6.4, 6.5

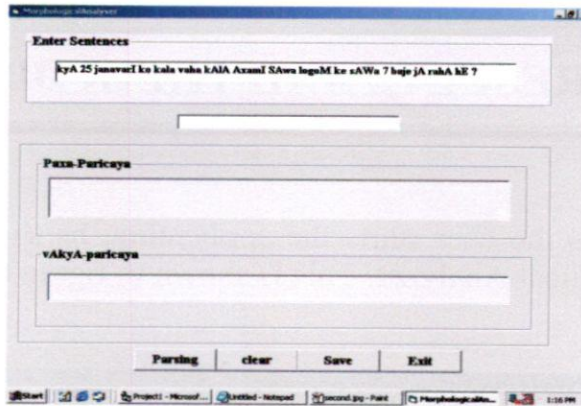


Figure 6.2 User Screen with sentence entered

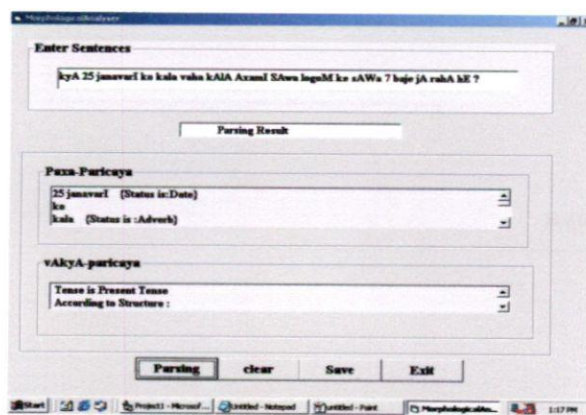


Figure 6.3 User Screen showing part of tagged output

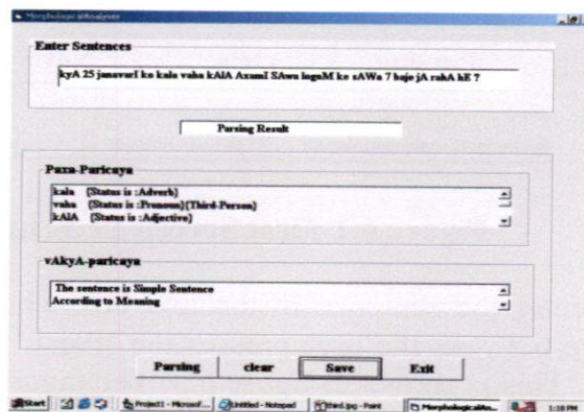


Figure 6.4 User Screen showing part of tagged output

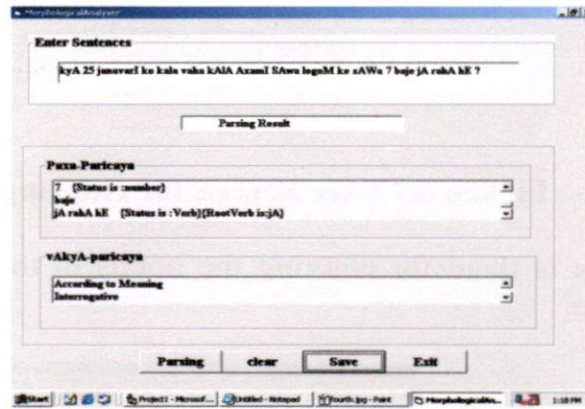


Figure 6.5 User Screen showing part of tagged output

The user can also save the sentence and its output in a text file. For this a GUI is also provided as shown in the figure 6.6

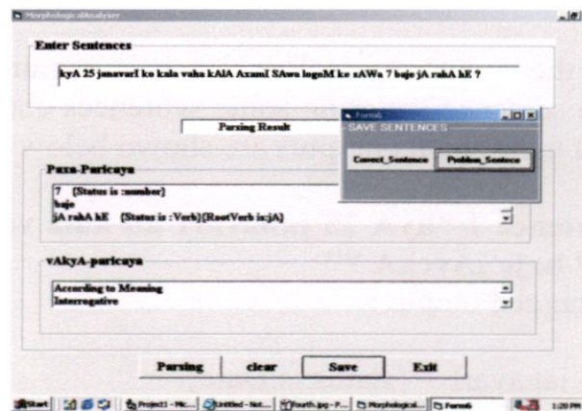


Figure 6.6 User Screen for saving the sentence and its output

These were the user screens showing the tagged output. Besides these, There are more GUI for entering the knowledge base in the database and modifying them. The user screen for entering the verb and modifying them is shown as in figure 6.7

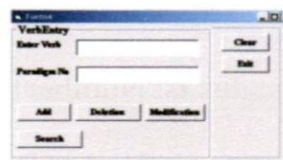


Figure 6.7 User Screen for entering verb

Similarly there is another user screen for entering the adjectives and their categories in the knowledge base as shown in the figure 6.8.

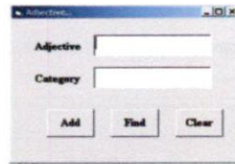


Figure 6.7 User Screen for entering adjective

Yet another screen is there for entering the nouns in the database. This interface is as shown in the figure.

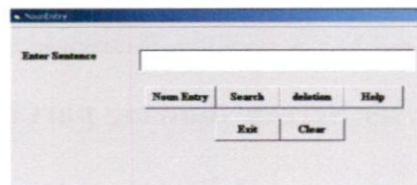


Figure 6.7 User Screen for entering nouns

As the morphological analyzer is required to tag the various parts of speech in a sentence, entering some sentences checks this. Some of the sentences entered and their results/outputs are shown below as:

Sentence 1 “kyA 25 janavarI ko kala vaha kAlA AxamI sAwa logoM ke sAWa 7 baje jAyegA ?”

Its tagged output is:

```

325 janavarI  {status is:Date}
ko
kala          {status is: Adverb}
vaha          {status is: Pronoun} {Third Person}
kAlA          {status is Adjective}
AxamI         {status is: Noun}
sAwa          {status is: Number}
sAwa
logoM
ke
sAWa
7             {status is: Number}
baje
jAyegA       {status is:Verb} {Root verb is jAnA} {m} {s} {future}

```

Tense is future

According to Structure:

The sentence is simple sentence

According to Meaning

Interrogative

Sentence 2 “mere pApA baMgalora rahawe hEM |”

Its tagged output is:

mere {status is:Pronoun}{First Person}
pApA
baMgalora
rahawe hEM {status is:Verb}{Root verb is rahanA}

Tense is Present

According to Structure:

The sentence is simple sentence

According to Meaning

Sentence is simple sentence

Sentence 3. “Aja waka use xillI se AnA cAhiye WA !”

Its tagged output is:

Aja {status is:Adverb}
waka {status is:Adverb}
use {status is:Pronoun}{Third Person}
xillI
se
AnA cAhiye WA {status is:Verb}{Root verb is AnA}

Tense is Past Tense

According to Structure:

The sentence is simple sentence

According to Meaning

Exclamatory

Sentence 4. “vo KAnA KA cuke hoMge |”

Its tagged output is:

vo {status is:PRonoun}{Third Person}
KAnA KA cuke hoMge {status is:Verb}{Root verb is: KAnA KAnA}
|

Tense is Future

According to Structure:

The sentence is simple sentence

According to Meaning

Sentence is simple sentence

Sentence 5. “vaha ladakI apne Gara jAnA cAhawI hogI .”

Its tagged output is:

vaha {status is:Pronoun} {Third Person}

ladakI

apne {status is:Pronoun} {First Person}

Gara {Status is:Noun}

jAnA cAhawI hogI {status is:Verb} {Root verb is:jAnA}

|

Tense is Future Tense

According to Structure:

The sentence is simple sentence

According to Meaning:

Interrogative

Sentence 6. “mohana ko xasa mla pExala calanA padaZ sakawA WA

.”

Its tagged output is:

mohana

ko

xasa {status is:number}

mla

pExala {status is Adjective}

calanA padaZ sakawA WA | {status is:Verb} {Root verb is:chlanA}

Tense is Past Tense

According to Structure:

The sentence is simple sentence

According to Meaning:

Sentence is simple sentence

Sentence 7. “kyA Apa kala paccIsa ParavarI 2003 ko blsa logo??M ke sAWa jA rahe hEM ?”

Its tagged output is:

kyA

Apa {status is:Pronoun} {Third Person}

kala {status is:Adverb}

paccIsa ParavarI 2003 {status is:Date}

ko

bIsa {status is:Number}
 logoM
 ke
 sAWa
 jA rahe hEM ? {status is:Verb}{Root verb is:jAnA}

Sentence 8. “xera sabera yaha wo honA hI WA |”
 Its tagged output is:

xera
 sabera
 yaha {status is:Pronoun}{Third Person}
 wo {status is:Conjunction}
 honA hI WA {status is:Verb}{Root verb is:honA}
 |

Sentence 9. “kAnUna wo nirA akRama hI sixXa howA hE |”
 Its tagged output is:

kAnUna
 wo {status is:Conjunction}
 nirA
 akRama
 hI
 sixXa
 howA hE {status is:Verb}{Root verb is:honA}

Tense is Present Tense
 According to Structure:
 The sentence is simple sentence
 According to Meaning
 Sentence Showing Doubt

Sentence 10. “mEM Aja yA kala waka ye Bexa jAna luMGA ki wuma kahAz BAgA rahe ho | “
 Its tagged output is:

mEM {status is:Pronoun}{First Person}
 Aja {status is:Adverb}
 yA {status is:Conjunction}
 kala {status is:Adverb}
 waka {status is:Adverb}

ye	{status is:Pronoun}{Third Person}
Bexa	
jAna	
luMgA	
ki	{status is:Conjunction}
wuma	{status is:Pronoun}{Second Person}
kahAz	{status is:Adverb}
BAGA rahe ho	{status is:Verb}{Root verb is:BaganA}

Tense is Present Tense
According to Structure:
The sentence is simple sentence
According to Meaning
Interrogative, Showing Doubt

We see here that in some tagged outputs that in some cases the words which are nouns have not been tagged as nouns. Why this so? This is because this morphological analyzer is dependent upon the Knowledge Base of the system i.e. it works for only those words that are already been stored in the database.

CONCLUSION

This can be seen from the results of some sentences in the previous section that we are getting mixed results for nouns and adjectives. Sometimes a noun word is shown as a noun and sometimes a noun is not shown as a noun. This is due to the limits of the Knowledge Base i.e our database. This morphological analyzer is dependent upon whether a word that is checked for tagging is already there in our Knowledge Base or not.

If the word we are checking does not happen to be there in our stored database, then nothing could be said about that word and that word is left without tagging.

Just take an example. Suppose the sentence entered is :

“kAnUna wo nirA akRama hI sixXa howA hE |”

Its tagged output is:

kAnUna

wo {status is:Conjunction}

nirA

akRama

hI

sixXa

howA hE {status is:Verb} {Root verb is:honA}

Tense is Present Tense

According to Structure:

The sentence is simple sentence

According to Meaning

Sentence Showing Doubt

Here ‘kAnUna’ is not shown as a noun although it is a noun. Also ‘akRama’ is not tagged as an adjective though it is an adjective. This is due to the

simple reason that they are not present in our Knowledge Base. Also, since we have included a limited number of patterns, it is not capable of finding all the possible patterns and thereof verbs.

So, this morphological analyzer is capable of tagging only those words which are already present in its database or Knowledge Base.

Also, it will show correct tagged output if it is entered with correct sentences. The entered sentence must be grammatically correct.

REFERENCES

- [1] Romanisation Scheme, IIT Kanpur
- [2] Hindi Vyakarana, Dr.O.P. Sharma
- [3] Mastering vb 6.0, Evangelos, Sybex Publications
- [4] Visual Basic 6.0 Programmaning, Steven Holzner
- [5] On Correction of Ill-Formed Hindi Sentences, R. M. K. Sinha
- [6] Dev Drishti: A Devanaagri Text Reader - Version 1, R. M. K. Sinha
- [7] Devedrishti: Computer Recognition of Hand Printed Devnagari Text, R. M. K. Sinha
- [8] Introduction to Hindi Morphology
Google Search at
<http://home.t-online.de/home/LINCOM.EUROPA/6806.htm>
<http://www.idsia.ch/wordmanager.html>
- [9] Hindi Dictionary
www3.aa.tufs.ac.jp/~kmach/hnd_la-e.htm
- [10] Some useful Hindi Language and Grammar Learning Links
<http://www.cs.colostate.edu/~malaiya/hindilinks.html>
<http://www.cs.colostate.edu/~malaiya/hindiint.html>

APPENDIX

(A) IIT Kanpur Romanization Scheme

अ	आ	इ	ई	उ	ऊ
a	A	i	I	u	U
े	ए	ऐ	ओ	औ	
q	e	E	o	O	
ः	ः	ः			
M	H	z			
क	ख	ग	घ	ङ	
k	K	g	G	f	Z
च	छ	ज	झ	ञ	
c	C	j	J	F	
ट	ठ	ड	ढ	ण	
t	T	d	D	N	
त	थ	द	ध	न	
w	W	x	X	n	
प	फ	ब	भ	म	
p	P	b	B	m	
य	र	ळ	व	श	
y	r	l	v	S	
ष	स	ह	क्ष	ज्ञ	श्र
R	s	h	kR	jF	Sr
त्त	प्र	क्र			
wwa	pra	kra			

SOME EXAMPLES

BaMga	भंग
sarvanASI	सर्वनाशी
gadZabadZa	गडबड
uwarA	उतरा
kRewra	क्षेत्र
hIMxi akRaramAIa	हिन्दी अक्षरमाला
kqwa	कृत
SrqMgAra	श्रृंगार
n	न
KZ	ख
PZ	प
pza	फ
UzcA	उँचा
kArya	कार्य

(B) Pattern File Containing Some Patterns

"_we cale jA rahe hoMge*2"

"_wI call jA rahi hogI*2"

"_wI call jA rahi howI*1"

"_wA calA jA rahA hogA*2"

"_wA calA jA rahA howA*1"

"_we cale jA rahe howe*1"

"_we cale jA rahe hEM*1"

"_nI padaZ sakawI WI*3"

"_nI padaZ sakawI hE*1"

"_nI padaZ sakawI hogI*2"

"_nI padaZ sakawI ho*1"

"_nI padaZ sakawI howI*1"

"_nA padaZ sakawA WA*3"

"_nA padaZ sakawA hE*1"

"_wI call jA rahi ho*1"

"_wI call jA rahi WI*3"

"_wI call jA rahi hE*1"

"_wA calA jA rahA ho*1"

"_wA calA jA rahA WA*3"

"_wA calA jA rahA hE*1"

"_we cale jA rahe We*3"

"_we cale jA rahe hE*1"

"_we cale jA rahe ho*1"

"_we cale gaye hoMge*2"

"_wA calA gayA hogA*2"

"_wA calA gayA howA*1"

"_wI call gayI hogI*2"

"_wI call gayI howI*1"

"_we cale gaye howe*1"

"_ne jA rahe hoMge*2"

"_ye jA rahe hoMge*2"
"_we jA rahe hoMge*2"
"_yI jA rahe hoMgI*2"
"_wA calA gayA WA*3"
"_wA calA gayA hE*1"
"_wA calA gayA ho*1"
"_wI callI gayI WI*3"
"_wI callI gayI hE*1"
"_wI callI gayI ho*1"
"_we cale gaye We*3"
"_we cale gaye hEM*1"
"_we cale gaye hoM*1"
"_wI jA rahI hogI*2"
"_wI jA rahI howI*1"
"_we jA rahe howe*1"
"_wA jA rahA hogA*2"
"_wA jA rahA howA*1"
"_ye jA rahe hoMge*2"
"_ye jA rahe howe*1"
"_yI jA rahI hogI*2"
"_yI jA rahI howI*1"
"_yA jA rahA hogA*2"
"_yA jA rahA howA*1"
"_I jA rahI hoMgI*2", "A"
"_ne jA rahA hogA*2"
"_ne jA rahI hogI*2"
"_ne jA rahI howI*1"
"_we jA rahe hEM*1"
"_wI jA rahI ho*1"
"_wI jA rahI WI*3"
"_wI jA rahI hE*1"

(C) Case Files for generating extended forms of Verbs

CASE1.txt (Used with Verbs having paradigm number 1 like 'KA')

"*0", "m", "s", "add", "present"
"e*0", "m", "p", "", "past"
"I*0", "f", "s", "", "past"
"IM*0", "f", "p", "", "past"
"Uz*0", "m", "s", "add", "present"
"eM*0", "m", "p", "", "present"
"yA*0", "m", "s", "", "past"
"o*0", "m", "p", "add", "present"
"UzgA*0", "m", "s", "add", "future"
"yegA*0", "m", "s", "add", "future"
"yeMge*0", "m", "p", "add", "future"
"oge*0", "m", "p", "add", "future"
"UzgL*0", "f", "s", "add", "future"
"egI*0", "f", "s", "add", "future"
"yeMgL*0", "f", "p", "add", "future"
"ogI*0", "f", "p", "add", "future"
"ezgL*0", "f", "p", "add", "future"
"wA*0", "m", "s", "add", "past"
"wI*0", "f", "s", "add", "past"
"we*0", "m", "p", "add", "past"
"nA*0", "m", "s", "add", "future"
"iyegA*0", "m", "s", "add", "future"
"iye*0", "m", "s", "add", "future"
"ie*0", "m", "s", "add", "future"
"iegA*0", "m", "s", "add", "future"

CASE2.txt (Used with Verbs having paradigm number 2 like ‘bawA’, ‘bulA’)

"*0", "m", "s", "add", "present"
"yA*0", "m", "s", "", "past"
"e*0", "m", "p", "", "past"
"Uz*0", "m", "s", "add", "present"
"I*0", "f", "s", "", "past"
"IM*0", "f", "p", "", "past"
"o*0", "m", "p", "add", "present"
"UzGI*0", "f", "s", "add", "future"
"egI*0", "f", "s", "add", "future"
"ogI*0", "f", "p", "add", "future"
"UzGA*0", "m", "s", "add", "future"
"yegA*0", "m", "s", "add", "future"
"yeMge*0", "m", "p", "add", "future"
"oge*0", "m", "p", "add", "future"
"yeMGI*0", "f", "p", "add", "future"
"ezGI*0", "f", "p", "add", "future"
"ogI*0", "f", "p", "add", "future"
"wA*0", "m", "s", "add", "past"
"wI*0", "f", "s", "add", "past"
"we*0", "m", "p", "add", "past"
"nA*0", "m", "s", "add", "future"
"iyegA*0", "m", "s", "add", "future"
"iye*0", "m", "s", "add", "future"
"ie*0", "m", "s", "add", "future"
"iegA*0", "m", "s", "add", "future"

CASE3.txt (Used with Verbs having paradigm number 3 like 'ja')

"*0", "", "", "", "pre"
"Uz*0", "m", "s", "", "pre"
"e*0", "m", "s", "", "past"
"eM*0", "m", "p", "", "pre"
"o*0", "m", "p", "add", "pre"
"UzGA*0", "m", "s", "add", "future"
"yegA*0", "m", "s", "add", "future"
"yeMge*0", "m", "p", "add", "future"
"oge*0", "m", "p", "add", "future"
"UzGI*0", "f", "s", "add", "future"
"ogI*0", "f", "p", "add", "future"
"yegI*0", "f", "s", "add", "future"
"yeMgI*0", "f", "p", "add", "future"
"wA*0", "m", "s", "add", "past"
"wI*0", "f", "s", "add", "past"
"we*0", "m", "p", "add", "past"
"nA*0", "m", "s", "add", "future"
"iyegA*0", "m", "s", "add", "future"
"iye*0", "m", "s", "add", "future"
"ie*0", "m", "s", "add", "future"
"iegA*0", "m", "s", "add", "future"

CASE4.txt (Used with Verbs having paradigm number 4 like so, 'ro')

"*0","m","s","add","present"
"yA*0","m","s","","past"
"e*0","m","p","","past"
"I*0","m","s","","past"
"IM*0","m","p","","past"
"Uz*0","m","s","add","present"
"eM*0","m""p","","present"
"UzgA*0","m","s","add","future"
"yegA*0","m","s","add","future"
"yeMge*0","m","p","add","future"
"oge*0","m","s","add","future"
"Uzgl*0","f","s","add","future"
"yegI*0","f","s","add","future"
"yeMgl*0","f","p","add","future"
"ogI*0","f","p","add","future"
"wA*0","m","s","add","past"
"wI*0","f","s","add","past"
"we*0","m","p","add","past"
"nA*0","m","s","add","future"
"iyegA*0","m","s","add","future"
"iye*0","m","s","add","future"
"ie*0","m","s","add","future"
"iegA*0","m","s","add","future"

CASE5.txt (Used with Verbs having paradigm number 5 like 'kara')

"*0","m","s","add","present"
"iyA*3","m","p","","past"
"ie*3","m","p","","past"
"I*3","f","s","","past"
"IM*3","f","p","","past"
"Uz*1","m","s","add","present"
"e*1","m","s","add","past"
"eM*1","m","p","add","present"
"o*1","m","p","add","present"
"UzgA*1","m","s","add","future"
"egA*1","m","s","add","future"
"eMge*1","m","p","add","future"
"oge*1","m","p","add","future"
"Uzgl*1","f","s","add","future"
"egl*1","f","s","add","future"
"eMgl*1","f","p","add","future"
"ogI*1","f","p","add","future"
"eMgl*1","f","p","add","future"
"wA*0","m","s","add","past"
"wI*0","f","s","add","past"
"we*0","m","p","add","past"
"nA*0","m","s","add","future"
"iyegA*1","m","s","add","future"
"iyeg*1","m","s","add","future"
"ie*1","m","s","add","future"
"iegA*1","m","s","add","future"

CASE6.txt (Used with Verbs having paradigm number 6 like le, 'xe')

"iyA*1","m","s","","past"
"ie*1","m","p","","past"
"iye*1","m","p","","past"
"I*1","f","s","","present"
"IM*1","f","p","","past"
"Uz*1","m","s","add","present"
"o*1","m","p","add","present"
"UzGA*1","m","s","add","future"
"oge*1","m","p","add","future"
"UzGI*1","f","s","add","future"
"ogI*1","f","p","add","future"
" *0","m","s","add","present"
"M*0","m","p","","present"
"gI*0","f","s","add","future"
"MgI*0","f","p","add","future"
"Mge*0","m","p","add","future"
"wA*0","m","s","add","past"
"wI*0","f","s","add","past"
"we*0","m","p","add","past"
"nA*0","m","s","add","future"
"IjiyegA*1","m","s","add","future"
"Ijiye*1","m","s","add","future"
"Ijie*1","m","s","add","future"
"IjiogA*1","m","s","add","future"

CASE7.txt (Used with Verbs having paradigm number 7 like 'ho')

"uA*1","m","s","","past"
"Uz*1","m","s","add","present"
"uc*1","m","p","","past"
"uI*1","f","s","","past"
"uIM*1","f","p","","past"
"e*0","m","s","","past"
"M*0","m","p","","past"
"eM*0","m","P","","past"
"UzgA*0","m","s","add","future"
"gA*0","m","s","add","future"
"Mge*0","m","p","add","future"
"gc*0","m","p","add","future"
"UzgL*0","f","s","add","future"
"yegI*0","f","s","add","future"
"yeMgI*0","f","P","add","future"
"MgI*0","f","p","add","future"
"gI*0","f","s","add","future"
"wA*0","m","s","add","past"
"wI*0","f","s","add","past"
"we*0","m","p","add","past"
"nA*0","m","s","add","future"
" *0","m","s","add","present"
"iyegA*0","m","s","add","future"
"iye*0","m","s","add","future"
"ie*0","m","s","add","future"
"iegA*0","m","s","add","future"

CASE8.txt (Used with Verbs having paradigm number 8 like 'sl', 'jl')

"jyA*1","m","s","", "past"
"ie*1","m","p","", "past"
"iUz*1","m","s","", "present"
"ieM*1","m","p","", "present"
"io*1","m","p","add", "present"
"iUzgl*1","m","s","add", "future"
"iegA*1","m","s","add", "future"
"ioge*1","m","p","add", "future"
"iUzga*1","m","s","add", "future"
"iUzgl*1","f","s","add", "future"
"ieMge*1","m","p","add", "future"
"iegI*1","f","s","add", "future"
"iogI*1","f","s","add", "future"
"iezgl*1","f","s","add", "future"
"wA*0","m","s","add", "past"
"wl*0","f","s","add", "past"
"we*0","m","p","add", "past"
"nA*0","m","s","add", "future"
"M*0","f","p","", "past"
"*0","m","s","add", "present"
"jiyegA*0","m","s","add", "future"
"jiye*0","m","s","add", "future"
"jie*0","m","s","add", "future"
"jiegA*0","m","s","add", "future"
"Uz*1","m","s","add", "present"

CASE9.txt(Used with Verbs having paradigm number 9 like 'CU')

"*0","m","s","add","present"
"uA*1","m","s","","past"
"ue*1","m","p","","past"
"ul*1","f","s","","past"
"uIM*1","f","p","","present"
"ueM*1","m","s","","present"
"uo*1","m","p","add","present"
"uUz*1","m","s","add","present"
"uUzGA*1","m","s","add","future"
"ueGA*1","m","s","add","future"
"uezge*1","m","p","add","future"
"uoge*1","m","p","add","future"
"uezGI*1","m","s","add","future"
"ueGI*1","f","s","add","future"
"uogI","f","p","add","future"
"wA*0","m","s","add","past"
"wI*0","f","s","add","past"
"we*0","m","p","add","past"
"nA*0","m","s","add","future"
"iyegA*0","m","s","add","future"
"iye*0","m","s","add","future"
"ie*0","m","s","add","future"
"iegA*0","m","s","add","future"

CASE11.txt (Used with Verbs having paradigm number 11 like 'cala', 'beca')

"*0","m","s","add","present"

"A*1","m","s","","past"

"e*1","m","p","","past"

"I*1","f","s","","past"

"IM*1","f","p","","past"

"o*1","m","s","add","present"

"eM*1","m","s","","present"

"Uz*1","m","s","add","present"

"UzgA*1","m","s","add","future"

"Uzgl*1","f","s","add","future"

"egA*1","m","s","add","future"

"eMge*1","m","p","add","future"

"oge*1","m","p","add","future"

"egl*1","f","s","add","future"

"eMgl*1","f","p","add","future"

"ogI*1","f","p","add","future"

"wA*0","m","s","add","past"

"wI*0","f","s","add","past"

"we*0","m","p","add","past"

"nA*0","m","s","add","future"

"iyegA*1","m","s","add","future"

"iye*1","m","s","add","future"

"ie*1","m","s","add","future"

"iegA*1","m","s","add","future"

CASE10.txt (Used with Verbs having paradigm number 10 like 'uTa')

"*0","m","s","add","present"
"A*1","m","s","","past"
"e*1","m","p","","past"
"I*1","f","s","","past"
"IM*1","f","p","","past"
"o*1","m","s","add","present"
"eM*1","m","s","","present"
"Uz*1","m","s","add","present"
"UzgA*1","m","s","add","future"
"Uzgl*1","f","s","add","future"
"egA*1","m","s","add","future"
"eMge*1","m","p","add","future"
"oge*1","m","p","add","future"
"egl*1","f","s","add","future"
"eMgl*1","f","p","add","future"
"ogl*1","f","p","add","future"
"wA*0","m","s","add","past"
"wI*0","f","s","add","past"
"we*0","m","p","add","past"
"nA*0","m","s","add","future"
"iyegA*1","m","s","add","future"
"iye*1","m","s","add","future"
"ie*1","m","s","add","future"
"iegA*1","m","s","add","future"

