Rule Based Semi-Supervised Morphological Analyzer for Extending the Range of Existing System

Thesis submitted in partial fulfillment of the requirements for the award of degree of

Master of Engineering
in
Software Engineering

Thapar University, Patiala

By:
Teena Bajaj
(80631024)

Under the supervision of:
Mr. Parteek Bhatia
Senior Lecturer
CSED, Thapar University, Patiala

JUNE 2008

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT
THAPAR UNIVERSITY
PATIALA – 147004

Certificate

I hereby certify that the work which is being presented in the thesis entitled, **"Rule Based Semi-Supervised Morphological Analyzer for Extending the Range of Existing System"**, in partial fulfillment of the requirements for the award of degree of Master of Engineering in Software Engineering submitted in Computer Science and Engineering Department of Thapar University, Patiala, is an authentic record of my own work carried out under the supervision of Mr. Parteek Bhatia.

The matter presented in this thesis has not been submitted for the award of any other degree of this or any other university.

(**Teena Bajaj***)*

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

(Mr. Parteek Bhatia)
Senior Lecturer
Computer Science and Engineering Department
Thapar University
Patiala

Countersigned by:

(Dr. (Mrs.) Seema Bawa)
Professor & Head
Computer Science & Engineering. Department
Thapar University
Patiala

(Dr. R. K. Sharma)
Dean(Academic Affairs)
Thapar University
Patiala.

Acknowledgement

I wish to express my deep gratitude to Mr. Parteek Bhatia, Senior Lecturer, Computer Science & Engineering Department for providing his uncanny guidance and support throughout the preparation of this thesis report. He always provides a motivating and enthusiastic atmosphere to work with. It was a great pleasure to do this thesis work under his supervision.

I am also thankful to Dr. (Mrs.) Seema Bawa, Head, Computer Science & Engineering Department, for the motivation and inspiration that triggered me for this thesis.

I would also like to thank all the staff members and my co-students who were always there at the need of the hour and provided with all the help and facilities, which I required for the completion of the thesis.

I am very thankful to my respectable family for their fortitude. Nothing would have ever been possible without my parents' love and encouragement. Last but not the least I would like to thank God for not letting me down at all time.

Teena Bajaj

(80631024)

Abstract

The Internet today has to face the complexity of dealing with multilinguality. People speak different languages and the number of natural languages along with their dialects is estimated to be close to 4000. Among the top 100 languages in the world, Hindi occupies the fifth position with the number of speakers being close to 200 million. The information need of this large section of humanity will place its unique demand on the web calling for knowledge processing of Hindi documents on the web.

Morphological analyzer is an essential and basic tool for building any language processing application for a natural language. There are two main approaches of learning the morphology *i.e.* Supervised and Unsupervised.

The existing morph analyzer, freely downloadable at http://www.iiit.net/ltrc/morph/, has a coverage of around 50%. The thesis focuses on how strength of existing morph analyzer can be improved by merging it with a semi-supervised approach for learning of Morphology. In the process of working towards morphological analysis for Hindi language, we have referred the algorithm implemented by Utpal Sharma, Jugal Kalita and Rajib Das in their paper 'Unsupervised learning of Morphology for Building Lexicon for a Highly Inflectional language' in our system and merged it with the existing morph analyzer in order to increase the strength of existing morph analyzer. Further, we tested our system on some new text files and discussed the consequences of algorithm implemented by which the coverage of existing morph analyzer is improved. System has around 20% more coverage than the existing system. The coverage of the system can further be improved with the help of implementing the system on new text files and algorithm being performed in iterative manner.

Table of Contents

List of Tables and Figures

Chapter 1

A Brief Review of Morphology

Morphology literally means the study of shape. An awareness of morphology begins in early childhood through adolescence. While younger children learn to add an "s" in order to make a word plural, elder children may decipher the meaning of words by identifying their common roots with other words.

The object of study in morphology is the structure of words and the ways in which their structure reflects their relation to other words. This relation can be within some larger construction such as a sentence and across the total vocabulary of the language. The importance of morphology is in the context of Machine Translation, Information Retrieval, Information Extraction and many such applications.

In every language in the world, whether it be written, spoken or signed, morphology is fundamentally involved in both the production of language, as well as its understanding. Morphology is what makes a painter someone who paints, what makes inedible something that is not edible, what makes dogs more than a single dog, and why he jumps but they jump.

In order for this process to be effective, the listeners, readers and observers of language must be able to take the inflected word (actresses) and find the underlying root (actor) as well as the set of conveyed syntactic features (feminine, plural). This decoding process is called morphological analysis [12].

1.1 History of Morphology

The term **morphology** comes from classical Greek (*morphe*) and means the study of **shape** or **form** [16]. It is concerned with structure and arrangement of parts of an object, and how these "conform" to create a whole object. The "objects" in question can be physical objects (*e.g.* an organism, an anatomy or an ecology) or mental objects (*e.g.* linguistic forms, concepts or systems of ideas).

1.1.1 Morphology in Linguistics

Morphology is the field of linguistics that studies the internal structure of words. While words are generally accepted as being the smallest units of syntax, it is clear that in most (if not all) languages, words can be related to other words by rules. For example, Hindi speakers recognize that the words नगर, नगरों and न □□□□□□ are closely related. Hindi speakers recognize these relations from their tacit knowledge of the rules of word-formation in Hindi. They intuit that नगर is to नगरों as पत्र is to पत्रों; similarly, नगर is to नगरवासी as भारत is to □□□□□य .

The rules understood by the speaker reflect specific patterns (or regularities) in the way words are formed from smaller units and how those smaller units interact in speech. In this way, morphology is the branch of linguistics that studies patterns of word-formation within and across languages, and attempts to formulate rules that model the knowledge of the speakers of those languages.

1.2 Lexemes and Word Forms

The distinction between two senses of "word" is arguably the most important one in morphology. The first sense of "word," the one in which नगर and नगरों are "the same word," is called **lexeme**.

The second sense ,where नगर and न □□□□□□ refer to two different kinds of entities, is called **word-form.**

1.3 Inflection *vs.* Word-Formation

Given the notion of a lexeme, it is possible to distinguish two kinds of morphological rules. Some morphological rules relate different forms of the same lexeme; while other rules relate two different lexemes. Rules of the first kind are called **inflectional rules**, while those of the second kind are called **word-formation**. The Hindi plural, as illustrated by नगर and नगरों, is an inflectional rule; compounds like न □□□□□□ or □□□□□□ provide an example of a word-formation rule. Informally, word-formation rules form "new words" (that is, new lexemes), while inflection rules yield variant forms of the "same" word (lexeme).

There is a further distinction between two kinds of word-formation: derivation and compounding. Compounding is a process of word-formation that involves combining complete word-forms into a single **compound** form; न □□□□□□ is therefore a compound, because both नगर and □□□□ are complete word-forms in their own right before the compounding process has been applied, and are subsequently treated as one form. Derivation involves affixing bound (non-independent) forms to existing lexemes, whereby the addition of the affix **derives** a new lexeme. One example of derivation is clear in this case: the word □□□□□□□ is derived from the word □□□□□□ by prefixing it with the derivational prefix अ-, while □□□□□□ itself is derived from the verb □□□य .

1.4 Paradigms and Morphosyntax

A **paradigm** is the complete set of related word-forms associated with a given lexeme. The familiar examples of paradigms are the conjugations of verbs, and the declensions of nouns. Accordingly, the word-forms of a lexeme may be arranged by classifying them according to shared inflectional categories such as tense, aspect, mood, number, gender or case. For example, the personal pronouns can be organized into the categories of person (1st., 2nd., 3rd.), number (singular *vs.* plural), gender (masculine, feminine, neuter), and case (subjective, objective, and possessive).

The inflectional categories used to group word-forms into paradigms cannot be chosen arbitrarily; they must be categories that are relevant to stating the syntactic rules of the language. For example, person and number are categories that can be used to define paradigms in Hindi, because Hindi has grammatical agreement rules that require the verb in a sentence to appear in an inflectional form that matches the person and number of the subject. In other words, the syntactic rules of Hindi care about the difference between नगर and नगरों, because the choice between these two forms determines which form of the verb is to be used. In contrast, however, no syntactic rule of Hindi cares about the difference between नगर and न □□□□□□, or □□□□□□ and □□□□□□□. The first two are just nouns, and the second two just adjectives, and they generally behave like any other noun or adjective behaves.

An important difference between inflection and word-formation is that inflected word-forms of lexemes are organized into paradigms, which are defined by the requirements of syntactic rules, whereas the rules of word-formation are not restricted by any corresponding requirements of syntax. Inflection is therefore said to be relevant to syntax, and word-formation is not. The part of morphology that covers the relationship between syntax and morphology is called Morphosyntax, and it concerns itself with inflection and paradigms, but not with word-formation or compounding.

1.5 Need to study the Morphology of a Language

Following are some reasons which tell why it is necessary to study the morphology of a language.

- The Internet today has to face the complexity of dealing with *multilinguality.* People speak different languages and the number of natural languages along with their dialects is estimated to be close to 4000. Among the top 100 languages in the world, Hindi occupies the fifth position with the number of speakers being close to 200 million [13]. The information need of this large section of humanity will place its unique demand on the web calling for knowledge processing of Hindi documents on the web.
- The theme of the research is to justify the stand that if morphology is strong and harnessable, then lack of training corpora is not unbearable.
- Our specific and primary focus is on *morphology*, and on how knowledge of morphology can be a useful step towards a more complete knowledge of a language's linguistic structure.
- Learners who understand how words are formed, by combining prefixes, suffixes, and roots, tend to have larger vocabularies and better reading comprehension. Morphology can become an instructional tool for all learners.
- Those learners who take unfamiliar words and break them down into smaller parts, or morphemes, have increased success in deciphering unfamiliar vocabulary [10].
- Morphological analysis provides a new light on a vital reading skill.
- Morphological analyzers are using lexicon/thesaurus, keep/stop lists, and indexing engines for their process.

1.6 Three Kinds of Morphology

There are three kinds of morphology. These are:

- **Concatenative Morphology** – Words are composed of a number of morphemes concatenated together; morphemes include stem plus prefixes and suffixes [11]. For example, □□□□□□ is composed of □□, लभ and □□.
- **Non-concatenative Morphology** – In this type, morphemes are combined in more complex ways. For example, □□□□□□□ is composed of □□□□ and □□□.
- **Template Morphology** – This is another type of non-concatenative morphology which is found very common in languages such as Arabic, Hebrew and other Semitic languages.

1.7 Models of Morphology

There are three principal approaches to morphology, which each try to capture the distinctions above in different ways. These are:

- Morpheme-based morphology, which makes use of an Item-and-Arrangement approach.
- Lexeme-based morphology, which normally makes use of an Item-and-Process approach.
- Word-based morphology, which normally makes use of a Word-and-Paradigm approach.

1.7.1 Morpheme-based Morphology

In morpheme-based morphology, word-forms are analyzed as arrangements of morphemes. A **morpheme** is defined as the minimal meaningful unit of a language. In a word like □□□□□□□□□, we say that the morphemes are अ, □□□□, ईय and □□; □□□□ is the root and the other morphemes are, in this case, derivational affixes. In a word like नगरों, we say that नगर is the root, and that □□ is an inflectional morpheme. This way of analyzing word-forms as if they were made of morphemes put after each other like beads on a string, is called Item-and-Arrangement.

The fundamental idea of morphology is that the words of a language are related to each other by different kinds of rules. Analyzing words as sequences of morphemes is a way of describing these relations, but is not the only way. In actual academic linguistics, morpheme-based morphology certainly has many adherents, but is by no means the dominant approach.

1.7.2 Lexeme-based Morphology

Lexeme-based morphology is (usually) an Item-and-Process approach. Instead of analyzing a word-form as a set of morphemes arranged in sequence, a word-form is said to be the result of applying rules that *alter* a word-form or stem in order to produce a new one.

- An inflectional rule takes a stem, changes it as is required by the rule, and outputs a word-form.
- A derivational rule takes a stem, changes it as per its own requirements, and outputs a derived stem.
- A compounding rule takes word-forms, and similarly outputs a compound stem.

1.7.3 Word-based Morphology

Word-based morphology is a (usually) Word-and-Paradigm approach. This theory takes paradigms as a central notion. Instead of stating rules to combine morphemes into word-forms, or to generate word-forms from stems, word-based morphology states generalizations that hold between the forms of inflectional paradigms.

Words can be categorized based on the pattern they fit into. This applies both to existing words and to new ones. Application of a pattern different than the one that has been used historically can give rise to a new word, such as *older* replacing *elder* (where *older* follows the normal pattern of adjectival superlatives) and *cows* replacing *kine* (where *cows* fits the regular pattern of plural formation).

1.8 Morphological Analyzer

A morphological analyzer is a program for analyzing the morphology of an input word, the analyzer including a recognition engine, identifying suffixes and finding a stem within the input word algorithms.

Morphological analysis is an integral part of larger language processing projects such as text-to-speech synthesis, information extraction, syllable identification or machine translation [8]. Traditionally, morphological analyzers are composed of three parts:

- a morpheme lexicon.
- a set of rules governing the spelling and composition of morphologically complex words.
- and a decision algorithm to choose from a set of possible analyses.

Morphological analyzer and morphological generator are two essential and basic tools for building any language processing application for a natural language[7]. Morphological analysis means to study the internal structure of the words of a language. A Morphological analyzer gives the morph analysis of a word *i.e.* for a given word a morphological analyzer will return its root word and word class along with other grammatical information depending upon its word class. Like for nouns it will provide gender, number, and case information and for verbs it will provide tense, phase *etc.* Morphological generator does exactly the reverse of it, *i.e.* given a root word and grammatical information it will generate the word form of that root word.

Google uses morphological analysis across all its products. Computational linguistic activities in India are being carried out at many institutions. Morph analyzers in Indian languages (Telugu, Hindi, Marathi, Kannada, Punjabi) have been developed. These are freely downloadable [5]. These are developed by Akshara Bharathi Group at Indian Institute of Technology, Kanpur, India and University of Hyderabad, Hyderabad, India. The morph analyzer for Punjabi language has been developed by Dr. Gurpreet singh Lehal, Mandeep Singh Gill, Dr. S. S.Joshi, Punjabi University, Patiala [4]. A snapshot of the morph analyzer is as follows:

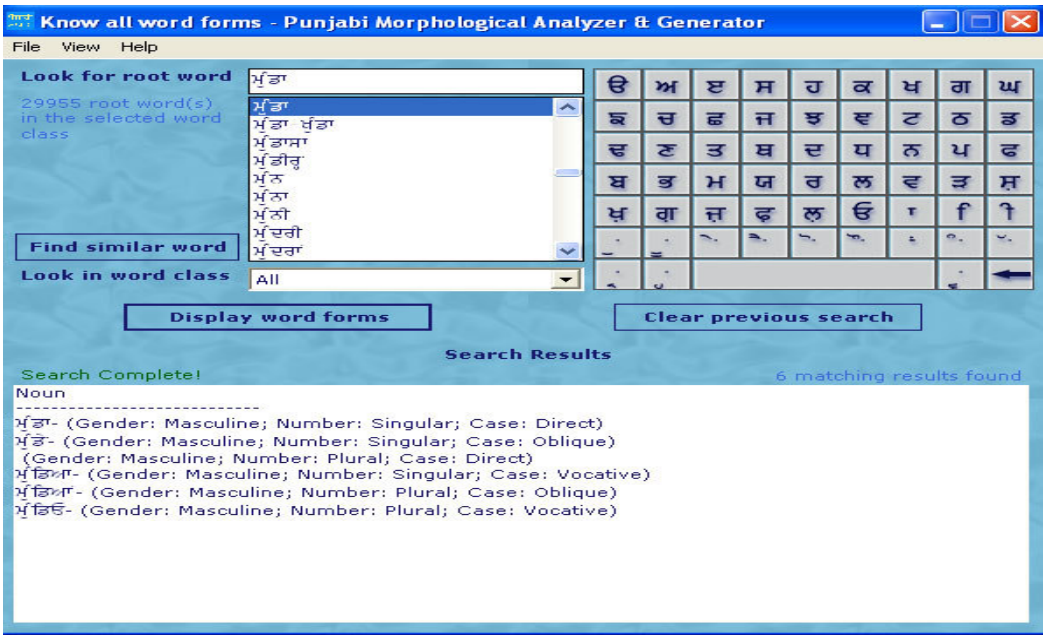


Figure 1.1: Punjabi Morph Analyzer

The database used in the software consists of more than 1.72 lakh Punjabi words, grouped into 22 word classes such as noun, personal pronoun, reflexive pronoun, verb, inflected and uninflected adverb, inflected and uninflected adjective, conjunction, interjection etc.

The morphological analysis is the process of providing grammatical information about the word on the basis of properties of the morpheme it contains [8].

Let us take an example for an English word: stayed. Morphological analysis will be as follows:

Input word=stayed

Category=Verb

Root=Stay

Suffix=-ed

Tense=past, presence of –ed

Now, let us consider a Hindi word, सुनेगा Morphological analysis for this word is:

Input word= सुनेगा

Category=Verb

Root= सुन

Suffix= ेगा

Person=3rd person, presence of े

Tense=Future, presence of ग

Gender=Male, presence of आ

In the above manner, the words are categorized and the grammatical attributes are stored in the database. Our thesis works on the basis of the word and paradigm based model of Morphology. The existing morph analyzer has a coverage of around 50% which means if a new chunk of text is given as input to existing morph analyzer, around 50% of the words are categorized and it returns the grammatical information of those words. Unsupervised learning approach is merged with the rule based approach in order to extend the coverage of existing morph analyzer. The system takes remaining undecomposed words and grammatical information of some new words is retrieved. New words which are decomposed by this system are stored in the new database and their grammatical information is stored. In this way, some new words are stored and the coverage is extended. This system is able to extend the coverage by around 20%. Therefore, the thesis work is an edge over the existing system.

Chapter 2

Morphology Learning Approaches

The first part of the thesis contains a comprehensive survey of work done on the different morphological learning approaches. All the minor and major lines of work are mentioned with a reference and a very brief characterization. Different approaches that have been prevalent in the field as a whole are highlighted and critically discussed. The general picture resulting from the survey is that much work has been repeated over and over, with little exchange and evolution of techniques.

2.1 Approaches for Learning of Morphology

There are three main types of learning morphological grammars.

- **Supervised Learning of Morphology**, which means that they have access to other sources of knowledge [12]. Supervised learning of morphological rules is based on the annotated data as the alignment of orthographic words with their morpheme representations (for example, the morpheme representation *box^s* is aligned with orthographic word *boxes* where ^*s* and *es* represent the plural noun suffix). In order to apply supervised learning methods, the data should further be extended with information about inflectional classes and features (part-of-speech tags), thus making the output of the training mechanism compatible with the input for morphological analyzers.
- **Unsupervised Learning of Morphology**, which means they can only learn from the data they are executed on [12]. The advantage of unsupervised morphological learning is that it requires only the set of orthographic words (completely "raw data") without any stem/affix lists or grammatical annotation. However, most of the unsupervised methods put restrictions on the number of morphemes per word, on rule complexity *etc.* The goal of these methods is merely splitting a word into stem and suffix without the capability of assigning any grammatical information. A number of unsupervised learning techniques have been applied: genetic algorithms, a minimal description length approach based on spelling of words and the set of suffixes that appear with each stem, and the quasi-roots algorithm.

- There is another approach that can be classified in between supervised and unsupervised learning where the input for training are the following pairs of words: word in its basic form and word in its grammatical form. Several Inductive Logic Programming (ILP) techniques as well as statistical approaches were applied in the framework of this learning approach. This approach can be considered as a compromise between the quality of the obtained rules and the amount of effort required to produce the training set.

2.2 Supervised Learning of Morphology (SLM)

The SLM approach consists of the available stem/affix lists with grammatical features, morphotactic rules governing their concatenation, and orthographic rules that change the shape of word constituents

2.2.1 Advantages of SLM Approach

Supervised learning of morphology has following advantages:

- Building the rule database for morphological analyzers by hand (stem/affix lists, morphotactic and orthographic rules) is extremely time consuming and can take several years for a single language.
- Automated techniques for generating morphological rules for morphological analysis can lead to great savings in time and resources for the languages of interest.

2.2.2 Algorithms for Supervised Learning of Morphology

Several algorithms for SLM approach have been developed. Some of the approaches are described in the following section.

- **Hyphenation System for SLM**

The task of hyphenation is to find the positions at which a hyphen may be inserted in a word. In some languages these positions are purely phonologically determined while in others the morphological structure of a word is used.

- **Word Lists**

  The simplest hyphenation method is to use a word list with hyphens derived from a dictionary. While this method will introduce only valid hyphenation points for known words it will also only cover those words that are in the dictionary.

- **Rule-based Hyphenation**

  Primary hyphenation points are introduced between compound word boundaries. Secondary points are inserted after prefixes. The remaining parts of a word are hyphenated according to phonological principles.

- **The Probabilistic Paradigm**

The probabilistic paradigm model consists of three matrices: the data matrix *D*, the morphological probabilities matrix *M*, and the lexical probabilities matrix *L*. Let *m* be the number of stems, *n* the number of stems, and *p* the number of paradigms [2].

The *D* matrix encodes the joint distribution of lexical and morphological information in a corpus. It is of size *m* x *n*, and each cell contains the frequency of the word formed by concatenating the appropriate stem and suffix. The *M* matrix is of size *m* x *p*, and each column contains the conditional probabilities of each suffix given a paradigm.

The *L* matrix is of size *p* x *n*, and contains the conditional probabilities of each paradigm given a stem. Each suffix should belong to exactly one paradigm, and the suffixes of a particular paradigm should be conditionally independent. Each column of the *M* matrix defines a *canonical paradigm*, a set of suffixes that attach to stems associated with that paradigm. A *lexical paradigm* is the full set of word forms for a particular stem, and is an instantiation of the canonical paradigm for a particular stem. The probabilistic paradigm is not well developed as the usual notion of "paradigm" in linguistics. First, the system employs no labels such as "noun", "plural", "past", *etc.* Second, probabilistic paradigms have only a top-level categorization; induced “verb” paradigms, for example, are not substructured into different tenses or conjugation.

- **Stochastic Transducers**

Stochastic transducers is another technique for the supervised learning of morphology. Stochastic transducers are trained using the Expectation-Maximization (EM) algorithm [1]. A particular strength is their ability to model both general rules and specific exceptions in a single framework. It is possible to apply the EM algorithm to learn the parameters of stochastic transducers. This approach could be used to learn morphology by starting with a randomly initialized model and using the EM algorithm to find a local maximum of the joint probabilities over the pairs of inflected and uninflected words.

2.3 Unsupervised Learning of Morphology (ULM)

Unsupervised Learning of Morphology (ULM) can be referred as "Given a large collection of written text in a given natural language, can a computer, without any explicit knowledge about the language, extract a description of how words are conjugated in that language [12]? "

2.3.1 Advantages of Unsupervised Learning of Morphology

Unsupervised learning of morphology has following advantages:

- The final suffix of a word is typically the strongest single indicator of its syntactic category.
- Analysis of a word into a stem *T* plus suffix *F* allows us (given our knowledge that the suffix *F* is a stronger indicator of category than the stem *T*) to collapse many distinct stems into a single cover symbol for purposes of analysis, simplifying our task [18].
- Unsupervised learning constitutes a (partial) linguistic theory, producing a completely explicit relationship between data and analysis of that data.

2.3.2 ULM Applications

Unsupervised learning of grammar is a problem that can be important in many areas ranging from text preprocessing for information retrieval and classification to machine translation [18].

Unsupervised Learning of Morphology has a wide variety of Language Technology applications, including Machine Translation, Document Categorization and Information Retrieval. The ULM problem is also relevant for linguistic theory, and can serve to boost empirical investigations in subfields as Quantitative Linguistics and Linguistic Typology [3]

2.3.3 Algorithms for Unsupervised Learning of Morphology

During the last decade several minimally unsupervised algorithms that address the problem have been developed [9]. These algorithms are explained in following section.

- **Hub Automaton Technique**

It describes a simple unsupervised technique for learning morphology by identifying hubs in an automaton. In this, a hub is a node in a graph with in-degree greater than one and out-degree greater than one. It creates a word-trie, transform it into a minimal DFA, then identifies hubs. Those hubs mark the boundary between root and suffix, achieving similar performance to more complex mixtures of techniques [6].

To recognize a morpheme boundary, for example between a root and a suffix, a learner must have seen at least two roots with that suffix and at least two suffixes with that root. For instance, 'helpful', 'helpless', 'harmful', and 'harmless' would be enough evidence to guess that those words could be divided as 'help/ful', 'help/less', 'harm/ful', and 'harm/less'. Without seeing varying roots and varying suffixes, there is no reason to prefer one division to another.

The simplest way to build a graph from a raw corpus of words is to construct a *trie*. A trie is a tree representation of the distinct words with a character label on each branch. The trie can be transformed into a minimal, acyclic DFA (Deterministic Finite Automaton), sharing nodes that have identical continuations. For example, suppose that, in a given corpus, the prefix ‘friend’ occurs only with the suffixes ‘NULL’, ‘s’, and ‘ly’ and the word ‘kind’ occurs only with the same suffixes. The minimal DFA has merged the nodes that represent those suffixes, and as a result has fewer links and fewer nodes than the original trie. In this DFA, some hubs will be obvious, such as for

the previous example. These are morpheme boundaries. There will be other nodes that are not obvious hubs. Some may have high out-degree but an in-degree of one; others will have high in-degree but an out-degree of one. Other candidate hubs are those nodes with high out-degree that are direct descendants, along a single path, of a node with high in-degree. In essence, these are stretched hubs. Figure 2.1 shows an idealized view of a hub and a stretched hub.

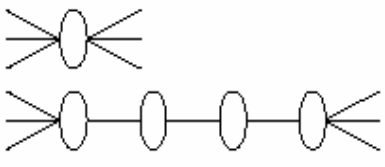


Figure 2.1: An idealized view of a hub and a stretched hub.

The lines are links in the automaton and each would be labeled with a character. The ovals are nodes and are only branching points.

In a minimized DFA of the words in a corpus, we can identify hubs and the last node in stretched hubs as morpheme boundaries. The above-mentioned technique for hub searching misses boundaries if a particular signature only appears once in a corpus. For instance, the signature for 'help' might be 'ed', 's', 'less', 'lessly', and NULL; and suppose there is no other word in the corpus with the same signature. The morpheme boundaries 'help-less' and 'help-ed' will not be found.

- **Whole Word Morphology**

Whole word morphology is capable on the one hand of identifying morphological relations within a list of words from any one of a wide variety of languages and, on the other, of putting that knowledge to use in creating previously unseen word forms [15].

This approach is set against the literature on computational morphology as an entirely different way of doing things which has the potential to be generalized to all known varieties of morphology in the world's languages, a feature not shared by previous methods.

Whole Word Morphologizer (WWM) is the first implementation of the theory of Whole Word Morphology. The theory seeks to account for morphological relations in a minimalist fashion. The central mechanism of the theory, the Word Formation Strategy (WFS), is a sort of non-decomposable morphological transformation that relates full words with full words (or helps one fashion a full word from another full word) and parses any complex word into a variable and a non-variable component.

- **Morfessor**

This method discovers the likely locations of the morpheme boundaries in words of any language. The method proposed, called *Morfessor*, learns a simple model of concatenative morphology (word forming) in an unsupervised manner from plain text [15]. Morfessor is formulated as a Bayesian, probabilistic model. That is, it does not rely on predefined grammatical rules of the language, but makes use of statistical properties of the input text.

Morfessor situates itself between two types of existing unsupervised methods: morphology learning *vs.* word segmentation algorithms. In contrast to existing morphology learning algorithms, Morfessor can handle words consisting of a varying and possibly high number of morphemes. This is a requirement for coping with highly-inflecting and compounding languages, such as Finnish. In contrast to existing word segmentation methods, Morfessor learns a simple grammar that takes into account sequential dependencies, which improves the quality of the proposed segmentations.

Morfessor is evaluated in two complementary ways in this work: directly by comparing to linguistic reference morpheme segmentations of Finnish and English words and indirectly as a component of a large vocabulary Finnish speech recognition system. In both cases, Morfessor is shown to outperform state-of-the-art solutions.

Much work has been done in the area of unsupervised learning of morphology. A table showing the studies and what is learnt from these studies is given as follows:

	Model	Supervision	Experimentation	Learns what?
Harris (1955)+	C	T	English	Analysis
Andreev (1965)*	C	T	E-type (I)	Unclear
Gammon (1969)	C	T	E-type	Analysis
Lehmann (1973)	C	T	German	Analysis
Hafer and Weiss (1974)	C	T	English	Analysis
de Kock and Bossaert (1978)+	C	T	French/Spanish	Analysis
Klenk (1992)+	C	T	E-type	Analysis
Wothke and Schmidt (1992)+	C	T	German	Analysis
Klenk (1994)	NC	T	Arabic + E-type	Analysis
Langer (1991)	C	T	German	Analysis
Flenner (1995)+	C	T	Spanish	Analysis
Brent et al. (1995)	C	T	English	Analysis
Džeroski and Erjavec (2000)+	C	T	Slovene	Analysis
Kazakov and Manandhar (2001)+	C	T	French/English	Transducer
Gaussier (1999)	C	T + AP	English (I)	Paradigms
Lepage (1998)	NC	AP	Chinese to Arabic	New word
Goldsmith (2006)+	C	T	E-type (I)	Paradigms+Lexicon
Baroni (2003)+	C	T	E-type	Analysis
Clark (2001b)+	NC	# states	German/Arabic/English	Transducer
Déjean (1998a)+	C	T	E-type	Analysis
Schone (2001)+	C	T	E-type	Related pairs of words
Neuvel and Fulop (2002)	C	T	E-type (I)	Related pairs of words
Jacquemin (1997)	C	T	E-type	Related pairs of words
Sharma et al. (2002)	C	T	Assamese	Paradigms+lexicon
Baroni et al. (2002)	NC	T	English/German (I)	Ranked list of related word pairs
Creutz (2006)+	C	T	Finnish/Turkish/English	Analysis
Kontorovich et al. (2003)	C	T	English	Analysis
Snover and Brent (2003)+	C	T	English/Polish	Related pairs of words
Johnson and Martin (2003)	C	T	Inuktitut	Unclear
Wicentowski (2004)+	NC	AP	30-ish E-type	Transducers
Gelbukh et al. (2004)+	C	T	E-type	Analysis
Ćavar et al. (2004)+	C	T	Unclear	Paradigms
Argamon et al. (2004)	C	T	English	Analysis
Goldsmith et al. (2005)+	NC	T	Unclear	Unclear
Bacchin et al. (2005)+	C	T	E-type	Stemming
Oliver (2004, Ch. 4-5)	C	T	Catalan	Paradigms
Bordag (2005)	C	T	English/German	Analysis
Hathout (2005)	C	AP	English/French	Analysis
Kurimo et al. (2005)*	C	T	Finnish/Turkish/English	Analysis
Medina-Urrea (2006)+	C	T	Chuj/Ralámuri/Spanish	Analysis
Hammarström (2006b)+	C	-	Maori to Warlpiri	Same-stem
Arabsorkhi and Shamsfard (2006)	C	T	Persian	Analysis
Monson et al. (2007)	C	T	English/German	Paradigms+Lexicon
Demberg (2007)	NC	T	E-type	Analysis
Dasgupta and Ng (2007)	C	T	Finnish/Turkish/English	Analysis
Bernhard (2006)+	C	T	Finnish/Turkish/English	Analysis+Related sets of words
Xanthos (2007)+	NC	T	Arabic	Paradigms+Lexicon

Table 2.1: Very brief roadmap of earlier studies [3]

Abbreviations used in the Table:
C = Concatenative, NC = Also non-concatenative, T = Thresholds and Parameters to be set by a human, AP = Aligned pairs of words, E-type = European Indo-European type languages, I = Impressionistic evaluation. * = this citation covers work by several different authors. + = entry also covers earlier work by the same author(s).

2.4 Comparison among Different Approaches

When morphological training data is available, the supervised methods are able to produce highly accurate lemmatization of inflections to their respective roots.

When training data is not available, the unsupervised similarity methods are capable of producing noisy alignments which can be used to bootstrap parameters of the supervised methods. Together, the supervised and unsupervised learners can be iteratively retrained to create models of lemmatization which can outperform the supervised methods trained on clean training exemplars for some sets of languages.

A morph analyzer has been developed by Akshara Bharathi Group at Indian Institute of Technology, Kanpur, India and University of Hyderabad, Hyderabad, India [5]. This analyzer follows the supervised approach for learning the morphology. And an algorithm based on unsupervised learning approach has been developed by Utpal Sharma, Jugal Kalita and Rajib Das [17]. We learnt after this thesis that if these two approaches are combined then the resulting system has a more coverage than the two individual approaches.

Chapter 3

Morphological Analysis for Hindi

3.1 Morphological Structure of Hindi

In Hindi, Nouns inflect for number and case. To capture their morphological variations, they can be categorized into various paradigms based on their vowel ending, gender, number and case information. A paradigm systematically arranges and identifies the uninflected forms of the words that share similar inflectional patterns. Looking at the morphological patterns of the words in a paradigm, suffix-replacement rules have been developed. These rules help in separating out a valid suffix from an inflected word to output the correct stem and consequently, get the correct root. For example, □□□ inflect for feminine (Gender), direct (Case), singular (Number).

Hindi Adjectives may be inflected or uninflected, *e.g.* `□□□□', `□□□□□, '□□□□□□□', '□□□□□' *etc.* do not inflect.

Hindi Verbs inflect for the following grammatical properties (GNPTAM):

1. Gender: Masculine, Feminine, Nonspecific
2. Number: Singular, Plural, Non-specific
3. Person: 1st, 2nd and 3rd
4. Tense: Past, Present, Future
5. Aspect: Perfective, Completive, Frequentative, Habitual, Durative, Inceptive, Stative
6. Modality: Imperative, Probabilitive, Subjunctive, Conditional, Deontic, Abilitive, Permissive

The morphemes attached to a verb along with their corresponding analyses help identify values for GNPTAM features for a given verb form. For example, □□□□□□□ inflect for feminine (Gender), singular (Number), future (Tense), 2nd person (Person).

3.2 Features of Existing Morph Analyzer

Existing morph analyzer is freely downloadable at http://ltrc.iiit.ac.in/downloads/. It takes 20,000 nouns which are classified in 20 paradigms [5]. It is developed at Language Technologies Research Centre, IIIT, Hyderabad, India.

The morphological analysis helps in the formation of Suffix-Replacement(S-R) rules for Hindi language [8]. This helps in reducing a lot of ambiguities in the process of stemming. For example,

Root : नदी

Plural direct form: नदियाँ

Suffix: ियाँ

Paradigm: नदी

Suffix-Replacement: ियाँ/ ी

And,

Root : मछली

Plural direct form: मछलियाँ

Suffix: ियाँ

Paradigm: मछली

Suffix-Replacement: ीयाँ/ ी

Therefore, if a word ends with the suffix ियाँ then the suffix ियाँ is replaced with ी and matches with the appropriate paradigm.

Thus, the analysis using the paradigms helps in increasing the accuracy by returning only the correct root. The paradigm analysis is also used by Morphological analyzer to correctly analyze suffixes.

3.2.1 Categorization of Words

The existing system categorizes nouns and verbs as follows.

Noun (N)

Gender

- Masculine (M, *e.g.* – ल □□□□)
- Feminine (F, eg, लड़की)

- **Number**
  - Nouns that always come in the singular form and agree only with the verb with SG attribute (*e.g.* □□□□, □□□, □□□□□)
  - Nouns that always come in the plural form and agree only with the verb with PL attribute. (*e.g.* □□□)
- **Vowel Nouns endings:**
  - For nouns ending with 'अ ': [□□□□]
  - For nouns ending with 'ा':[□□□□□□]
  - For nouns ending with small 'ि':[□□□□]
  - For nouns ending with 'ी' :[नदी]
  - For nouns ending with 'ु' :[धातु]
- **Special Nouns endings:**
  - Nouns ending with 'ें' (*e.g.* □□□□□)
  - Nouns ending with 'ँ' (*e.g.* □□□□ँ)
  - Nouns ending with 'ै' (*e.g.* लौ, जौ)
  - Nouns ending with 'िया' (*e.g.* □□□□□□, □□□□□□□, □□□□□□□)

Verb

- **Gender**
  - Masculine (M, *e.g.*, हार करना)
  - Feminine (F, *e.g.*, सहायता करना)

- **Number**
    - Singular (*e.g.* चलेगा)
    - Plural (*e.g.* चलेंगें)
- **Verbs endings:**
    - For verbs ending with ‘ता’: [चलता]
    - For verbs ending with 'ना ':[दौडना]
    - For verbs ending with small 'या ':[खाया]
    - For verbs ending with ‘कर ’ [पढकर]
    - For verbs ending with 'ा ' :[भूला]
    - For verbs ending with ‘ते ’:[जागते]

Adjective

- **Gender**
    - Masculine (M, *e.g.*, अनमना)
    - Feminine (F, *e.g.*, □□□□□□□□□□)
- **Number**
    - Singular (*e.g.* □□□□□)
    - Plural (*e.g.* □□□छे)

Sharisthi_Pronoun

- **Gender**
    - Masculine(*e.g.* जनिका)
    - Feminine(*e.g.* □□□□□)
- **Number**

- Singular(*e.g.* □□□□□)
- Plural (*e.g.* □□□□□)

Some words have been categorized by existing system as follows:

Word	Root	Grammatical Category	Gender	Case	Number	TAM
मधुभक्षिकाओं	मधुभक्षिका	Noun	feminine	oblique	plural	X
रचाओगे	रचा	Verb	masculine	X	plural	Future
रजस्वले	रजस्वला	Adjective	masculine	direct	plural	X
रखाओगी	रखा	Verb	feminine	X	plural	Future
रक्षिते	रक्षिता	Adjective	masculine	direct	plural	X
रामलीलाओं	रामलीला	Noun	feminine	oblique	plural	X
उनकी	वह	Sharisthi Pronoun	feminine	direct	plural	X
धुलाओगी	धुला	Verb	feminine	X	plural	Future
शाहंशाहीयों	शाहंशाही	Noun	feminine	oblique	plural	X
परदेशीयों	परदेशी	Noun	masculine	oblique	plural	X
पगले	पगला	Adjective	masculine	oblique	plural	X
जिसकी	जो	Sharisthi Pronoun	feminine	direct	singular	X

Table 3.1: Categorization of Words by Existing System

3.3 Porting of Existing Morph Analyzer from Linux to Windows based system

The existing system had been designed in Linux version. The Perl language was used to design the system and encoding was WX which is very uncommon. We tried to migrate the system from Linux/Perl Based version to Windows based with Graphical User Interface.

3.3.1 Motivation behind Migration

Following are the reasons why the migration from Linux/Perl Based version to Windows based version was done.

- There was no GUI for analyzer and generator.
- Encoding used in the system is WX which is very uncommon.
- It is based on traditional file based system database concept.
- It follows Data Dependence Approach.
- Even the mapping of devanagri words is different for some characters *e.g.*

  - w to न
  - W to थ

  is difficult for a layman to operate.

- Data Files (.p files, Ca, Ce, root) are very much dependent upon each other's format.

3.3.2 Features of New Morphological Analyzer

New morphological analyzer has some better features which are given as below:

- It is developed for Microsoft Windows Operating System using Visual Basic.
- It supports Unicode Format.
- It provides Graphical User Interface.
- It is easy to use.
- It follows Data Independent Approach.
- There is no complex Programming required.

3.3.3 Structure of database used in Morphological analyzer

Morph Analyzer uses six tables to store the grammatical attributes of words, paradigm information, suffix information. The tables are named as follows:

- Categories
- ParadigmRootDetails
- Cat_Features
- Root

- WordForms
- Suffix_info

Following snapshot shows the Relationship among different tables of Morph Analyzer:

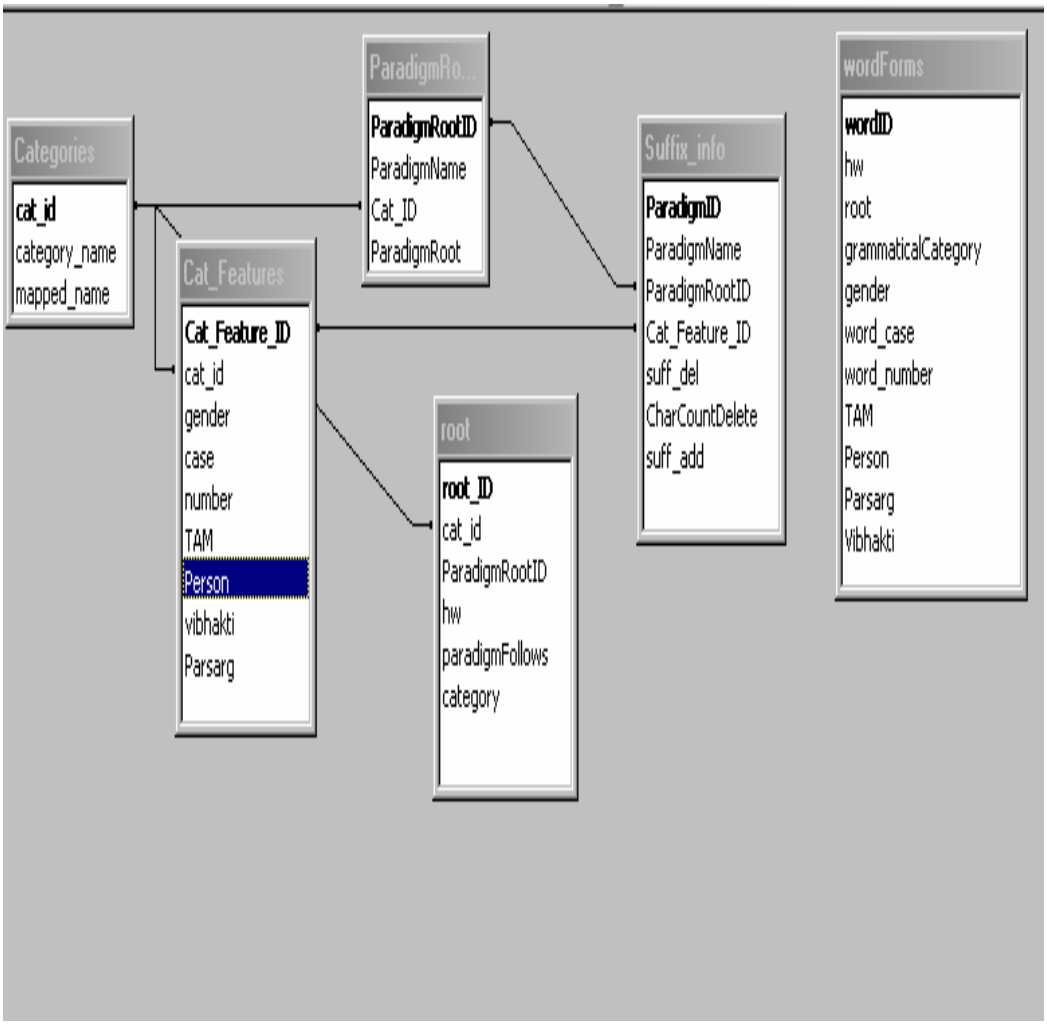


Figure 3.1: Relationship among Tables

Description of each table is as follows:

a. Categories Table

There can be 34 kinds of categories which a word may have. This table stores the category information of the words. This table is related to three tables *i.e.* Cat_features, Root, Paradigmrootdetails. Its fields are explained as follows:

- **cat_id**: This is a number which is a unique number assigned to each category. This number will be referred in all the three tables to which this table is related. This field is used as primary key.
- **category_name**: The string in this field is an acronym for the category under which the word falls.
- **Mapped_name**: The string in this field tells the actual category of the word *e.g.* noun, verb, adjective *etc.*

Categories : Table

cat_id	category_name	mapped_name
1	Noun_m	Noun
2	Noun_f	Noun
3	subj	Verb
4	Future	Verb
5	yA	Verb
6	hE	Verb
7	WA	Verb
8	yA1	Verb
9	kara	Verb
10	wA_nA	Verb
11	Adj_const	Adjective
12	Adj_m	Adjective
13	Adjtv_f	Adjective
14	Adj_m_s	Adjective
15	Adj_f_s	Adjective
16	Adj_case	Adjective
17	Advrb	Adverb
18	Post_ptn	PostPosition
19	Conjctn	Conjunction
20	Intrjctn	Interjunction
21	Parsarg	Parsarg

Figure 3.2: Table Category

For example, if the category name is Future then it falls under the category Verb because only the verbs may have tenses. Similarly, if the category name is Noun_m then the category of the word is Noun.

b. ParadigmRootDetails Table

The words are categorized according the pattern they fit into. These patterns are called as paradigms. This table stores the paradigm information. This table is related to table named suffix_info. There can be around 100 paradigms. Its fields are explained as follows:

- **ParadigmRootID**: It stores the unique number which is assigned to each paradigm. This field is used as primary key.
- **ParadigmName**: This is just a string by which the paradigm followed will be referred in other tables *e.g.* if the word follows the paradigm as राजा then it will be referred as N5 and paradigm लड़का will be referred as N6.
- **Cat_ID**: This is a number representing the category of the paradigm. This field acts as foreign key referring to table Category.
- **ParadigmRoot**: This field stores the paradigm being followed.

ParadigmRootDetails : Table

ParadigmRootID	ParadigmName	Cat_ID	ParadigmRoot
1	N1	1	घर
2	N2	1	खर्च
3	N3	1	क्रोध
4	N4	1	लडका
5	N5	1	राजा
6	N6	1	लोहा
7	N7	1	विधाता
8	N8	1	कवि
9	N9	1	आदमी
10	N10	1	पानी
11	N11	1	शत्रु
12	N12	1	कटु
13	N13	1	आलू
14	N14	1	लहू
15	N15	1	कुआँ
16	N16	1	रेडिओ
17	N17	1	गेहूँ
18	N18	1	भाषाविद्
19	N19	1	रात
20	N20	2	भौंड

Figure 3.3: Table ParadigmRootDetails

c. Cat_Features Table

This table stores the attributes information of categories of the words. The table is related to category table and suffix info. Its fields are explained as follows:

- **Cat_Feature_ID**: This is a unique number. given to every suffix. This field acts as primary key.
- **cat_id**: This field stores the category under which the suffix follows. This field acts as foreign key referring to table Category.
- Other fields stores the information about gender, case, number *etc.*

Cat_Features : Table

Cat_Feature_ID	cat_id	gender	case	number	TAM	Person	vibhakti	Parsarg
1	1	masculine	direct	singular	X	X	X	X
2	1	masculine	direct	plural	X	X	X	X
3	1	masculine	oblique	singular	X	X	X	X
4	1	masculine	oblique	plural	X	X	X	X
5	2	feminine	direct	singular	X	X	X	X
6	2	feminine	direct	plural	X	X	X	X
7	2	feminine	oblique	singular	X	X	X	X
8	2	feminine	oblique	plural	X	X	X	X
9	3	any	X	singular	subj	u	X	X
10	3	any	X	singular	subj	m	X	X
11	3	any	X	singular	subj	m_h	X	X
12	3	any	X	singular	subj	a	X	X
13	3	any	X	plural	subj	u	X	X
14	3	any	X	plural	subj	m	X	X
15	3	any	X	plural	subj	m_h	X	X
16	3	any	X	plural	subj	a	X	X
17	4	masculine	X	singular	Future	u	X	X
18	4	masculine	X	singular	Future	m	X	X
19	4	masculine	X	singular	Future	m_h	X	X
20	4	masculine	X	singular	Future	a	X	X
21	4	masculine	X	plural	Future	u	X	X

Figure 3.4: Table Category Features

For example, it is known that cat_id 1 is assigned to Noun_m. So, different categories which are noun and used as masculine gender may have different combinations for case number and other aspects.

d. Root Table

This table stores the information of root words of the words which are found in existing database. Its fields are explained as follows:

- **Root_ID**: This is a unique number given to each root word found. This field acts as primary key.
- **Cat_ID**: It stores the category ID of the root word *i.e.* root falls under which category referring to table Category. This field acts as foreign key referring to table Category.
- **ParadigmRootID**: It stores the paradigm root ID information *i.e.* which paradigm is followed by the root word. This field acts as foreign key referring to table ParadigmRootDetails.
- **Hw**: This field stores the word to be searched for its grammatical attributes.
- **ParadigmFollowed**: It stores the paradigm followed by the root word.
- **Category**: This field tells the category of the root word.

root : Table

root_ID	cat_id	ParadigmRootID	hw	paradigmFollow	category
2778	16	54	अप्रभु	अनेक	Adj_case
7223	16	54	ब्रह्मयोनि	अनेक	Adj_case
1159	16	54	अलगाऊ	अनेक	Adj_case
2163	16	54	अनुहारि	अनेक	Adj_case
1277	16	54	अल्पबुद्धि	अनेक	Adj_case
1606	16	54	अमूर्ति	अनेक	Adj_case
1539	16	54	अमोघदृष्टि	अनेक	Adj_case
3781	16	54	अवस्तु	अनेक	Adj_case
1565	16	54	अमृत्यु	अनेक	Adj_case
2721	16	54	अपति	अनेक	Adj_case
3468	16	54	अश्वारुढ	अनेक	Adj_case
2972	16	54	अरेणु	अनेक	Adj_case
5242	16	54	बाजरु	अनेक	Adj_case
3008	16	54	अर्जुनच्छवि	अनेक	Adj_case
7903	16	54	छमछम	अनेक	Adj_case
1301	16	54	अल्पायु	अनेक	Adj_case
4715	16	54	अयोनि	अनेक	Adj_case

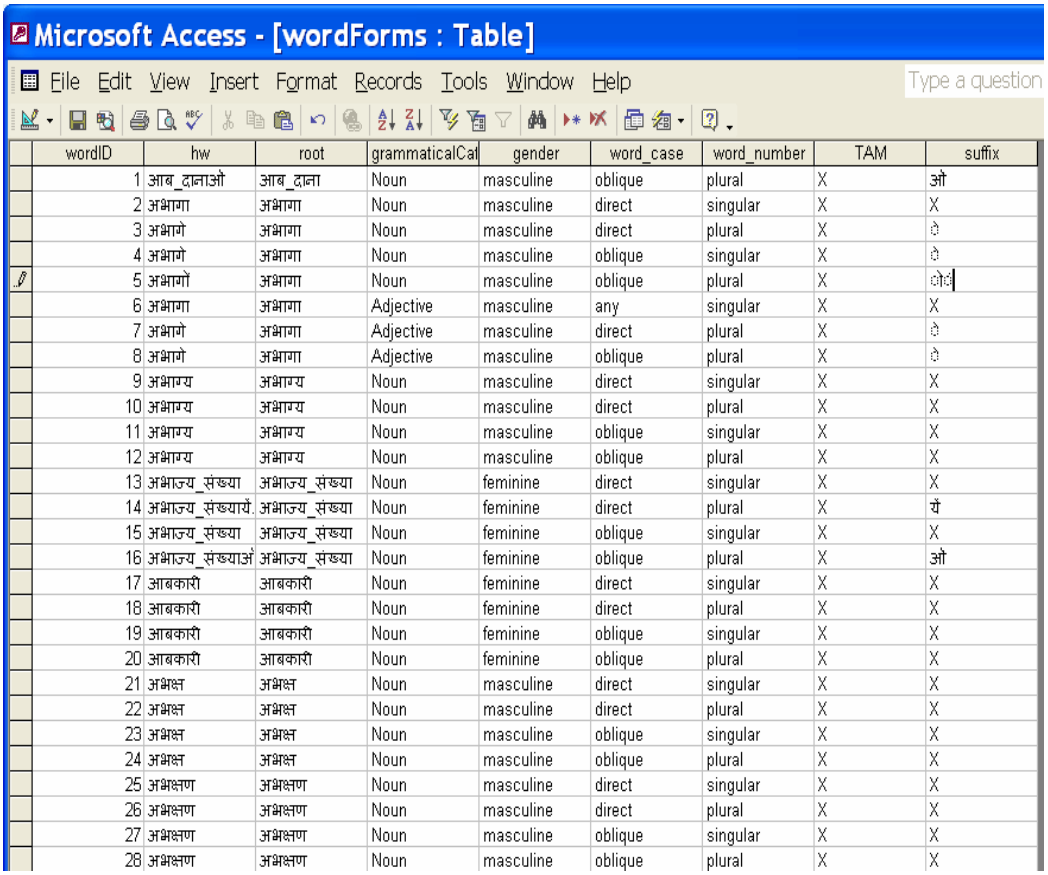
Figure3.5: Table Root

For example, the adjective words □□□□□, □□□□□□, □□□□□ follow the same paradigm □□□□.

e. WordForms

This table stores all the grammatical attributes of the words found. Its fields are explained as follows:

- **WordID**: It stores the ID information assigned to the word. This field acts as primary key.
- **hw**: This field stores the word to be searched for its grammatical attributes.
- **root**: It provides the root word of the Head word.
- **grammaticalCategory**: This field tells the category of the word *e.g.* Verb, Noun or adjective *etc.*
- **gender**: It tells the gender information of the word.
- **word_case**: It stores the case information of the word *i.e.* direct or oblique.
- **word_number**: It tells whether the word is used as singular or plural or any.
- **TAM** : This field stores the Tense information.



Microsoft Access - [wordForms : Table]

File Edit View Insert Format Records Tools Window Help Type a question

wordID	hw	root	grammaticalCat	gender	word_case	word_number	TAM	suffix
1	आब_टानाओं	आब_टाना	Noun	masculine	oblique	plural	X	ओं
2	अभागा	अभागा	Noun	masculine	direct	singular	X	X
3	अभागे	अभागा	Noun	masculine	direct	plural	X	े
4	अभागे	अभागा	Noun	masculine	oblique	singular	X	े
5	अभागों	अभागा	Noun	masculine	oblique	plural	X	ोंं
6	अभागा	अभागा	Adjective	masculine	any	singular	X	X
7	अभागे	अभागा	Adjective	masculine	direct	plural	X	े
8	अभागे	अभागा	Adjective	masculine	oblique	plural	X	े
9	अभाग्य	अभाग्य	Noun	masculine	direct	singular	X	X
10	अभाग्य	अभाग्य	Noun	masculine	direct	plural	X	X
11	अभाग्य	अभाग्य	Noun	masculine	oblique	singular	X	X
12	अभाग्य	अभाग्य	Noun	masculine	oblique	plural	X	X
13	अभाज्य_संख्या	अभाज्य_संख्या	Noun	feminine	direct	singular	X	X
14	अभाज्य_संख्यायें	अभाज्य_संख्या	Noun	feminine	direct	plural	X	यें
15	अभाज्य_संख्या	अभाज्य_संख्या	Noun	feminine	oblique	singular	X	X
16	अभाज्य_संख्याओं	अभाज्य_संख्या	Noun	feminine	oblique	plural	X	ओं
17	आबकारी	आबकारी	Noun	feminine	direct	singular	X	X
18	आबकारी	आबकारी	Noun	feminine	direct	plural	X	X
19	आबकारी	आबकारी	Noun	feminine	oblique	singular	X	X
20	आबकारी	आबकारी	Noun	feminine	oblique	plural	X	X
21	अभक्ष	अभक्ष	Noun	masculine	direct	singular	X	X
22	अभक्ष	अभक्ष	Noun	masculine	direct	plural	X	X
23	अभक्ष	अभक्ष	Noun	masculine	oblique	singular	X	X
24	अभक्ष	अभक्ष	Noun	masculine	oblique	plural	X	X
25	अभक्षण	अभक्षण	Noun	masculine	direct	singular	X	X
26	अभक्षण	अभक्षण	Noun	masculine	direct	plural	X	X
27	अभक्षण	अभक्षण	Noun	masculine	oblique	singular	X	X
28	अभक्षण	अभक्षण	Noun	masculine	oblique	plural	X	X

Figure3.6: Table Word Forms

For example, the word □□□□□ is a verb whose root is आ. The gender of the word is masculine, it is used in Future tense and it is a singular word.

To determine the grammatical attributes of the words which are found from this system, the suffix field of this table is referred.

3.4 Extending the Coverage of Existing Morph Analyzer with Semisupervised Learning Approach

The coverage of this migrated morph analyzer is approximately 50%. So, we tried to improve the coverage of morph analyzer. An algorithm based on the unsupervised approach for learning of morphology has been implemented by Utpal Sharma, Jugal Kalita and Rajib Das in their paper 'Unsupervised learning of Morphology for Building Lexicon for a Highly Inflectional language'. They have implemented the algorithm for Assamese language[17]. We have referred their algorithm in this system and merged it with the existing morph analyzer in order to increase the strength of existing morph analyzer.

Some new words are decomposed into root and suffixes with the unsupervised approach. The suffix table of new database is matched with the suffix of migrated morph analyzer and on that basis grammatical attributes of new words are determined. Our system works as semi-supervised approach for learning of Morphology.

Chapter 4

Problem Statement

A Morphological analyzer gives the morph analysis of a word *i.e.* for a given word a morphological analyzer will return its root word and word class along with other grammatical information depending upon its word class [7]. Morphological analysis means to study the internal structure of the words of a language. The morphological strength of Indian Languages warrant the use of thorough morphological analysis.

4.1 Gap analysis between the Existing work and the Desired work

The ultimate goal of research on morphological analysis is to parse and understand the text and determine the grammatical attributes of the words. For this reason, morph analyzer has been developed at IIIT Hyderabad, freely downloadable at http://ltrc.iiit.ac.in/downloads/. This morph analyzer takes 20,000 nouns which are classified in 20 paradigms[5]. The primary step towards development of Rule Based Semisupervised Morphological Analyzer for any language demands an in-depth understanding and analysis of that language. Analysis of the language then helps in developing the computational model to handle data. From the initial analysis and results of Brute-force approaches, we felt the need to extend the strength of existing morph analyzer. From earlier studies it was found that:

1. Indian Languages are morphologically rich and the best way to build the morph analyzer is to first analyze the data.
2. The linguistic information should be encoded in a way that it is independent of the code and is easy to change for a linguist.

At this stage on linguistic front, we are ready with Noun and Verb analyses that are very detailed. On computational front, we have tried to extend the coverage of existing morph analyzer. If text is given as input to existing morph analyzer, the system provides grammatical attributes of many words. But some words are not covered by this system.

4.2 Concise Statement of the Problem

The coverage of existing morph analyzer is around 50%. The target is to improve the strength of existing morph analyzer.

A new algorithm should be implemented so that coverage can be increased. The new words which remained as uncategorized by existing system should be categorized and their grammatical attributes can also be determined. Therefore, rule based semisupervised morph analyzer has been implemented to extend the coverage. The system takes those words which are not covered by existing system as input and decomposes the words into root words and suffixes. On the basis of suffixes information referred from the wordforms table of existing morph analyzer, grammatical attributes of the new words is determined.

With the use of this approach some new words can be categorized and therefore this system is an edge over the existing system.

Chapter 5

Design and Implementation of the Research Problem

This system has been designed in JAVA Netbeans platform. Backend used is MS-ACCESS which stores the database of this system.

5.1 Working of System

This system works has three important steps whose detail is as follows:

1. The words of new input Hindi text are stored into the table named as *undecomposed*. These words are searched in migrated morph analyzer. If the words exist in existing morph analyzer, it gives entire information of the words *i.e.* Gender, Case, Tense, Aspect, Modal *etc.* The information of these words are stored into the table named as *alreadyexistwords* and these words are deleted from the table *undecomposed.*

2. The remaining words of the *undecomposed* table which can not be categorized in step 1 (after searching through the existing morph analyzer) are searched through the algorithm being implemented in section 5.3.

It provides the following functionality:

- **Suffix and Root information for a word**

Undecomposed words will be entered as input into the algorithm. The system will process the text and will display the suffix and root information of the words of the input text. According to the algorithm the words are decomposed into root words and suffixes. These words are stored into the table named as *decomposed* along with its suffix and root information and other attributes *e.g.* word length *etc.* The algorithm works in phases. After phase 1, words which could be decomposed by this system are deleted from the table undecomposed. This process is repeated till the system finds no new words that can be decomposed. For example, □□□□□□□□□□□□□□ is decomposed into □□□□□□□□□□ and □□□. Word length is 14.

A table named as *suffix* stores the information of suffixes found from the new input text. One table named as *base* stores the root words found from the input text. All the words which remained as undecomposed are stored in the table named as *undecomposed*.

- **Grammatical attributes of the decomposed word**

On the basis of suffix information of decomposed words its grammatical attributes are determined. Existing Morph analyzer is referred and on the basis of suffix field of existing *wordForm* table suffix field of *decomposed* table of this system is matched and where match is found, its grammatical attributes are stored into the table *decomposed* and in this way, table *decomposed* is updated and now the table *decomposed* provides the root, suffix, Gender, Case, Number information. For example, on the basis of □□□ suffix, the attributes of the word □□□□□□□□□□□□□□□ are determined.

3. All new words, which are found after applying the algorithm, are appended in the existing morph analyzer so that strength of existing system can be improved. The word □□□□□□□□□□□□□□□ is now added into the existing database.

5.2 General Structure of How the System Works

Flow of the work in graphical manner is represented by following figure:

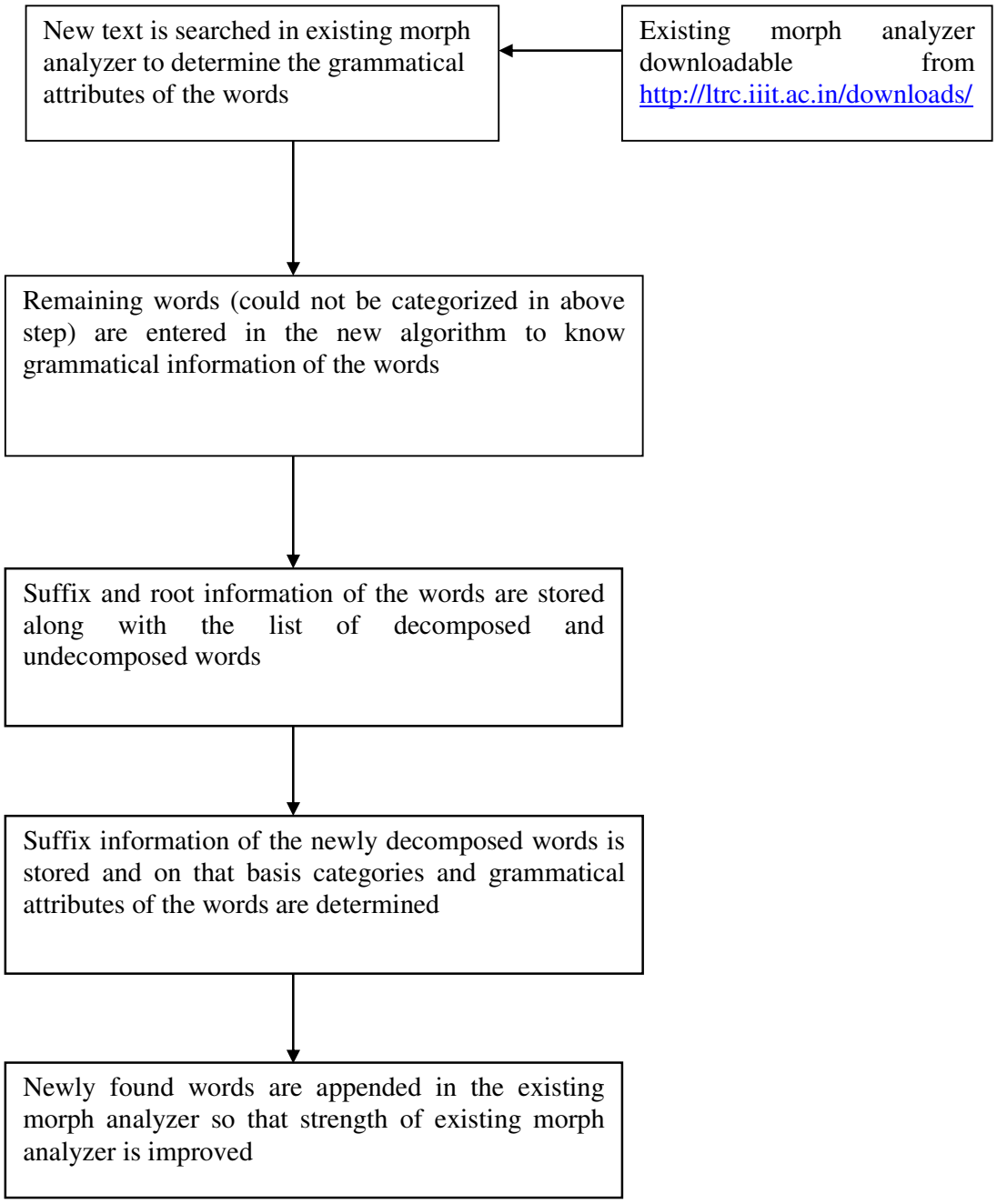


Figure 5.1: Structure describing the Working of System

5.3 Decomposing Words

A decomposition can be referred as where one word can be derived from another word by addition of some suffix in that set[17], the former word a ”derivative”, the latter a “base”, and the suffix a “rule”. *i.e.*,

Decomposition: Derivative = base + rule

For example,

सुनता = सुन + ता

जीतना= जीत + ना

We may be able to find more than one derivative for a single word from the same base by adding different rules. A base may be found to be a derivative with respect to another base. For example, suppose we have the words, a, ab, ac, abd (a,b,c and d are strings possibly longer than a single character). We can represent these as the following.

---

ab= a+b

ac= a+c

abd= a+bd,

or, abd= ab + d

Example :

मानवीय = मानव + ीय

मानवता = मानव + ता

मानवीयता = मानव + ीयता

मानवीयता = मानवीय + ता

In the case of abd, we would like to record the decomposition ab + d, since it reflects more decomposition information than the other alternative, when considered along with ab= a+ b.

The algorithm which is being implemented along with the new input text and the result of each step is summarized below.

Preprocessing

1.Take an input text *X*.

Let us take an input chunk of text as:

- मानव, मानवीय, सह, सुनना, जीत, मानव, सुनता, जीतना, लिखेगा, सुनेगा, सहता, खेलेगा, मानवीयता, मानवता.

2. Form a sorted list, *Y*, of distinct words in *X*.

- खेलेगा, जीत, जीतना, मानव, मानवीय, मानवीयता, मानवता, लिखेगा, सुनता, सुनना, सुनेगा, सह, सहता, सहना.

Phase 1

3. For each word *w* in *Y*, identify another word *b* in *Y* such that *w* can be obtained by appending some suffix *s* to *b*. If there are multiple candidates for *b*, select the longest among them and record the decomposition

w= b + s

- जीतना = जीत + ना
- मानवीय = मानव + ीय
- मानवीयता = मानवीय + ता
- मानवता = मानव + ता
- सहता = सह + ता
- सहना = सह + ना

If no *b* can be identified for a *w, w* is "undecomposed"**.**

- खेलेगा, जीत, मानव, लिखेगा, सुनता, सुनना, सुनेगा, सह

4. From the decompositions identified above,

- For each base *b* count the number of decompositions where it is a base. Name it the base count of *b*, bc_b.
  - सह: 2, मानव : 2, मानवीय: 1, जीत :1
  - सुनता, सुनना, सुनेगा, सहता, सहना, खेलेगा, जीतना, मानवीयता, लिखेगा, मानवता: 0
- For each suffix (rule), *s*, count the number of decompositions where it is a suffix. Name it the rule frequency of *s*, rf_s.
  - ता: 3, ना: 2, ीय:1

Phase 2

5. For each word that could not be decomposed in phase 1 (some of which may have come out as base in some decomposition), try each of the rules identified in that phase. That is, see if a rule (suffix) is the ending part of an undecomposed word. Record such decompositions, and the base forms.

- सुनता = सुन + ता
- सुनना = सुन + ना

Phase 3

6. For each word *w* in *Y*, identify the set of suffixes (rules) that appear with it and call it the characteristic of that word, C_w. Words are classified using their characteristics.

- सुन : ता, ना
- सह : ता, ना
- मानव : ता, ीय
- मानवीय : ता

- जीतः ना

In the first phase, we extract rules by using base forms that are known, *i.e.*, words that are present in the list of words. After this analysis, there may be words which could not be decomposed. It is possible that some of these words are actually root words and cannot be decomposed. But there may be words that could not be decomposed because no base form exists in the list of words provided. So, in the second phase, we consider the "undecomposed" words and see if any "rule" identified so far can be applied to decompose such a word, *i.e.*, if ab is word that could not be decomposed so far, is there a rule b so that we can decompose ab as a+b? In fact multiple rules might be applicable for a single word. In such a case, in a language where suffix sequences are a common phenomenon, we should give preference to rules that are longer and can be obtained as a suffix sequence. Otherwise suffixes with higher frequencies may be given preference. With more base forms known following the second phase, we can repeat the first phase analysis, then the second phase analysis, and theoretically, so on.

In the example given above, if we were to iterate the steps after the first pass, due to the word सुन in the list now, we can decompose the hitherto undecomposed word सुनेगा as सुन + ेगा. This gives us the new suffix ेगा. We can decompose the word लिखेगा as लिख + ेगा and खेलेगा as खेल + ेगा, obtaining the new word लिख, खेल. This process may be stopped when in a particular pass we do not detect any new rule or base. Phase 3 can then be undertaken to determine the nature of the words based on the suffixes that each takes. Reaching such a stage, however, does not imply that all possible decompositions have been detected. Since our method works by comparing related words, the presence or absence of certain words makes significant difference. For instance, if the input list of words in the example given above did not contain the word सुनता, the base सुन would not be obtained, and consequently the suffix ेगा would not be obtained from सुनेगा, and further लिखेगा, खेलेगा would not be decomposed.

In general, providing a large number of words for analysis will cause more rules to be detected in phase 1, and with the large number of rules so obtained, more base forms can be detected through phase 2. In practice, it is typical to expect texts in chunks in successive analysis runs. So, during implementation of the algorithm, we refine it so that while a new chunk of text is processed, the words in that text are analyzed along with the words existing in the lexicon. The focus is put on the new words and their effects.

5.4 Database Design

The system accumulates the information extracted by the analysis process in the MS-ACCESS tables. In database five tables are used. The detail of each table is as follows:

- *Alreadyexistwords:* It stores all the words which are found in the existing morph analyzer. Its fields are explained below:
    - *word*- it stores the word.
    - *Base*- it specifies the root word of the word in the word field.
    - *Suffix*- it specifies the suffix of the word in the word field.
    - *Word_length*- it stores the number of characters in the word.
    - *Gc*- it stores the grammatical category of the word *e.g.* noun, verb *etc.*
    - *Gender*- it stores the gender information of the word.
    - *Number*- it specifies the number *e.g.* singular or plural.
    - *TAM*- it specifies the Tense, aspect and Modal information.
- *Base:* It stores all root words along with their counts. Its fields are explained below:
    - *Base_str*-it stores the root word.
    - *Base_count*- it stores the number of times the word appears as root in the text.
    - *Word_length*- it stores the number of characters in the root word.

- *Suffix:* It stores all the suffixes along with their counts. Its fields are explained below:

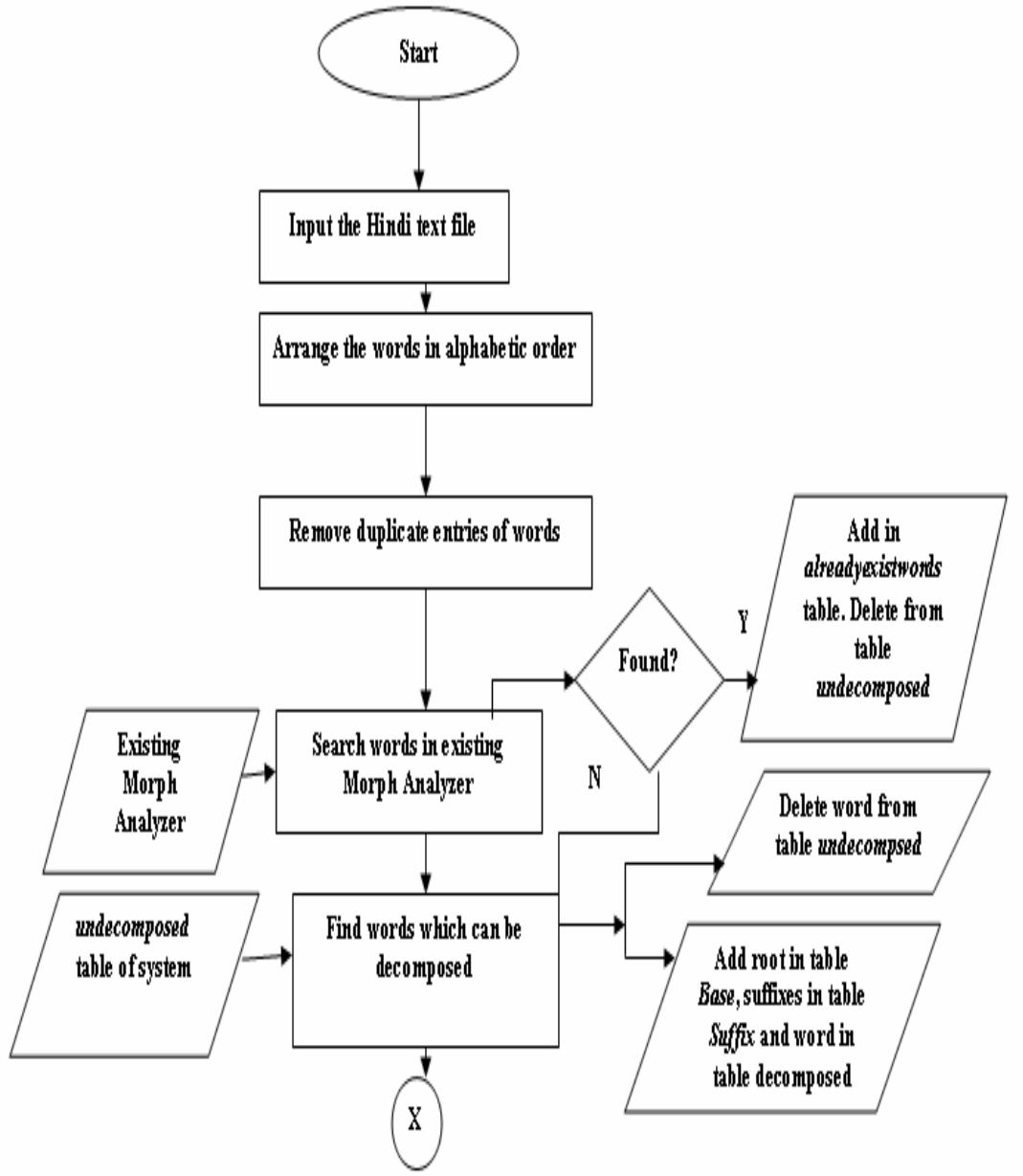
    - suffix_str-it stores the suffix.
    - suffix_count- it stores the number of times the word appears as suffix in the text.
    - Word_length- it stores the number of characters in the suffix.

- *decomposed:* It stores all the words which can be decomposed after processing through the algorithm. Its fields are explained below:

    - *word*- it stores the word.
    - *Base*- it specifies the root word of the word in the word field.
    - *Suffix*- it specifies the suffix of the word in the word field.
    - Word_*length*- it stores the number of characters in the word.
    - *Gc*- it stores the grammatical category of the word *e.g.* noun, verb *etc.*
    - *Gender*- it stores the gender information of the word.
    - *Number*- it specifies the number.
    - *TAM*- it specifies the Tense, aspect and Modal information.

- *undecomposed:* It stores all the words which remains undecomposed. Its fields are explained below:
    - *word*- it stores the word.
    - *Word_length*- it specifies the number of characters in the word.

Flowchart describing the working of algorithm is as follows:



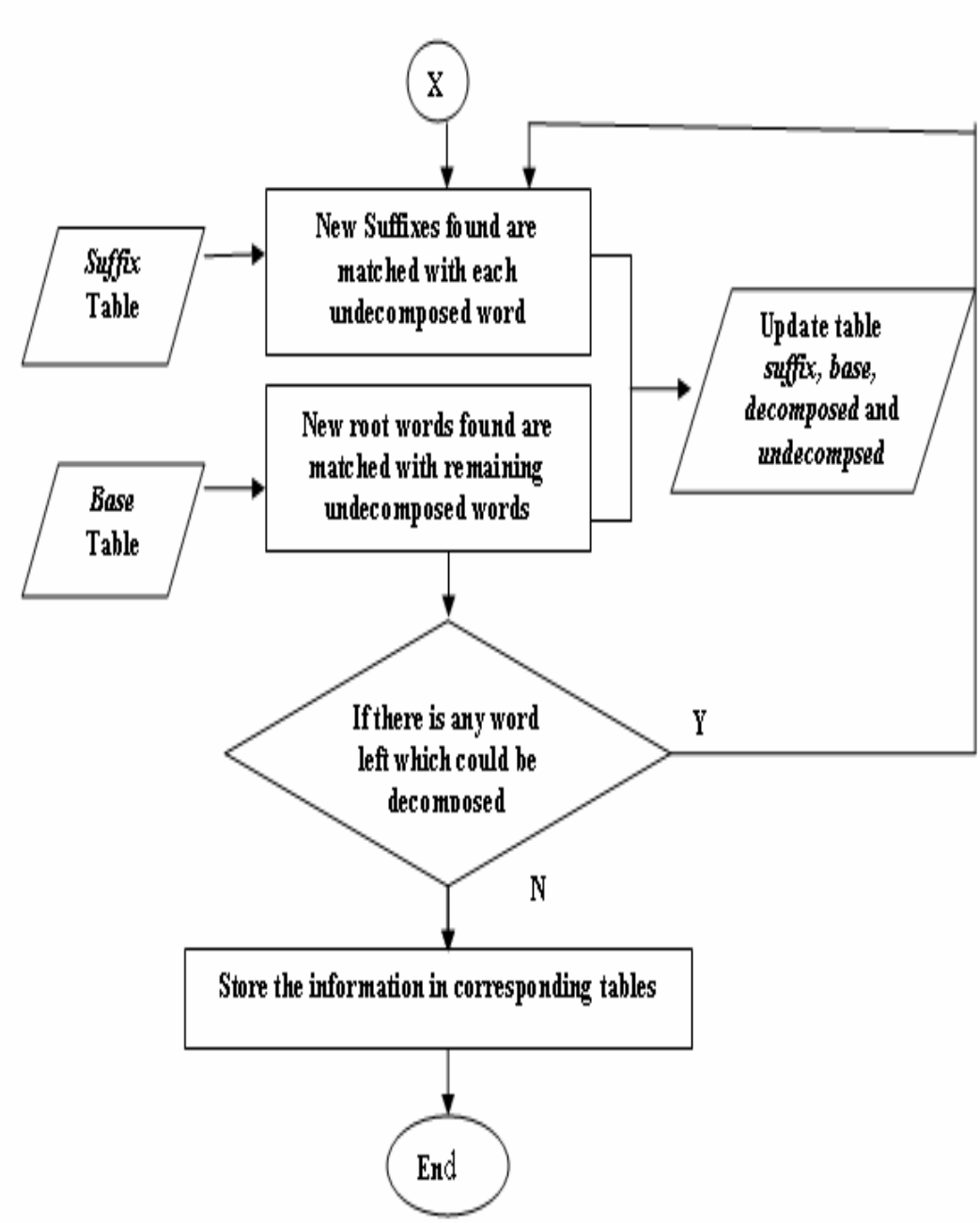


Figure 5.2: Flowchart describing the working of algorithm

Chapter 6
Experimental Results

We have tested the system on some new text files. We are giving the result of this system after applying it on a text file which contained the following words:

Text File 1

भारतीय अधिकतर रूढिवादी और कुसंस्कारों से ग्रस्त हैं इन सब जटिल समस्याओं के समाधान के लिए चलचित्रों का उपयोग होता है चलचित्रों से समाज में कुप्रवृत्तियाँ भी फैली हैं चोरी और कुप्रवृत्ति संबंधी अनेक अपराधियों ने इनसे पडे दुष्प्रभाव को स्वीकार किया है इन दुष्प्रभावों से कुप्रवृत्तियों को बढावा मिलता है

Duplicate entries from the file have been removed by the algorithm. The file contained 41 actual words.

19 words were found in existing morph analyzer. These words are stored in table named *alreadyexistwords*.

6.1 Result after 1st Phase

Remaining 22 words were entered into the system and 7 words (इन, □□□□□□□□□□□□□□□, □□□□□□□□□□□□□□□, □□□□□□□□□□□□□, □□□□, □□□□□□□□□□□, □□□□□□□□□□□□□) were found after the 1st phase analysis.

Base forms along with counts are as follows:

इन 1

□□□□□□□□□□□□ 2

□□□□□□□□□□□ 1

Suffixes information along with counts are as follows:

□□ 1

□□□ 1

□□□ 1

□□ 1

Then remaining words are again tried. On the basis of 2nd phase 2 new words (□□□□□□□□□□□, □□□□□□□□□□) which could be decomposed are found. The information of 9 words are stored in table named *decomposed.*

6.2 Result after 2nd phase of algorithm

Base forms along with counts are as follows:

इन 1
□□□□□□□□□□□ 2
□□□□□□□□□□ 1
□□□□□□□□□ 1
□□□□□□□ 1

Suffixes information along with counts are as follows:

□□ 1

□□□ 1

□□□ 1

□□ 3

Remaining 13 words are stored in table named *undecomposed.*

Snapshots of tables after testing of system on text file 1 are as follows:

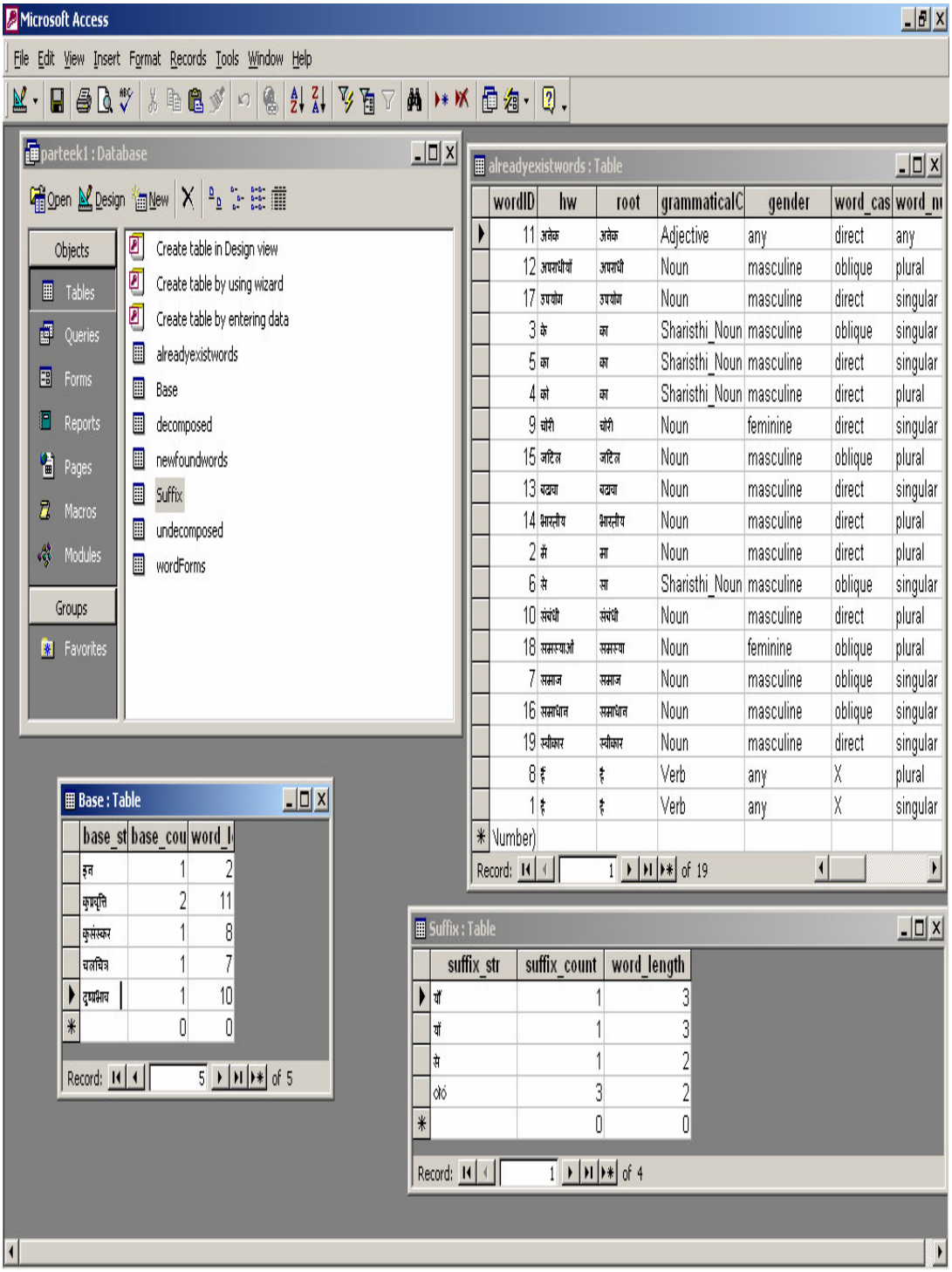


Figure 6.1: MS-ACCESS Tables

Above snapshot shows the result of three tables *alreadyexistwords, Base, Suffix*.

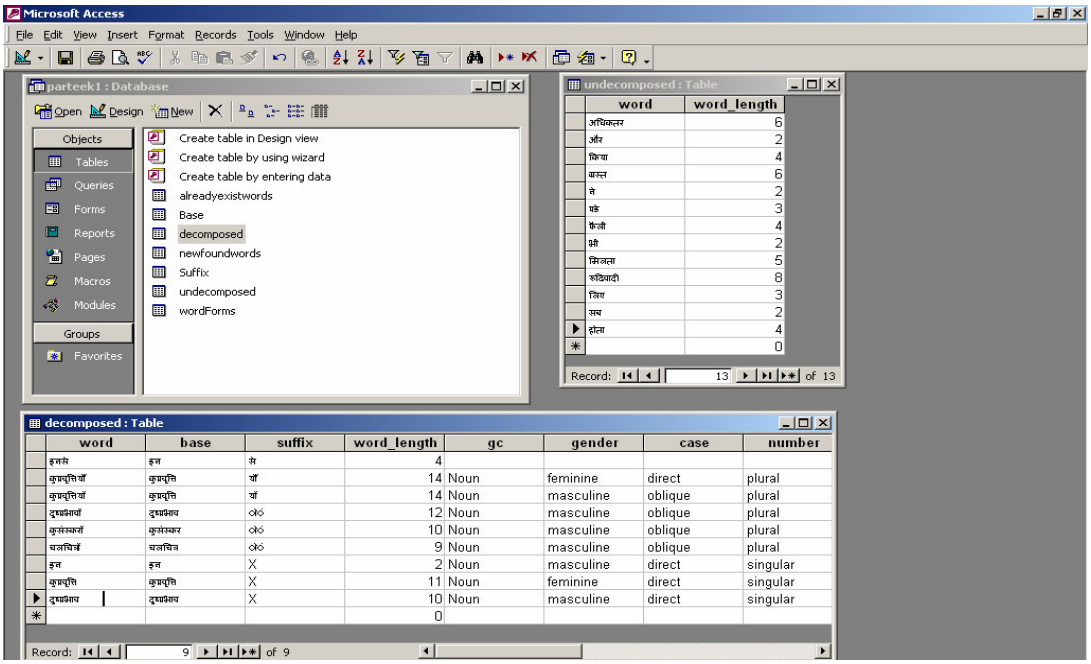


Figure 6.2 : MS-ACCESS Tables

Above snapshot shows the result of two tables *decomposed, undecomposed.*

The system was tested upon some more text files. Files and the statistics are given as follows:

Text File 2

दीवाली की उस मध्य रात्रि में सबके आशीषों का केन्द्र मैं था घर के सभी बुजुर्ग उपस्थित थे मैं बुजुर्गों का आशीष पाना चाहता था

Text File 3

अब तक मैंने कहानी और काव्य की अनेक पुस्तकें पढी हैं परन्तु औरों की अपेक्षा मेरे मन पर जिस ग्रन्थ ने सबसे अधिक प्रभाव डाला वह है तुलसीदास का संसारप्रसिद्ध रामचरितमानस बचपन में अपनी पाठ्य पुस्तकों में संगृहीत रामचरितमानस के कुछ पद्धों को पढा था इन सभी पाठ्यों में विषय की विविधता पर उत्तरोतर विकास है

Text File 4

चलचित्र एक दृश्य माध्यम है और इसका पाठ्य एवं श्रव्य माध्यमों की अपेक्षा बडा गहरा प्रभाव पडता है प्राचीन काल में जो स्थान नाटकों के लिए था आज वही चलचित्रों का है

A table showing the statistics of the results before applying this system and after this system

	Total number of words	**Words found in existing morph analyzer**	**Percent coverage by existing system**	**New words found in this system**	**Total words found after applying this system**	**Percent coverage by this system**
Text file 1	41	19	46.34	9	28	68.29
Text file 2	22	9	40.9	4	13	59.09
Text file 3	49	22	44.89	8	30	61.22
Text file 4	30	18	60	4	22	73.33

Table 6.1: Statistics of Results

A graph showing how much %coverage is extended after applying the new algorithm

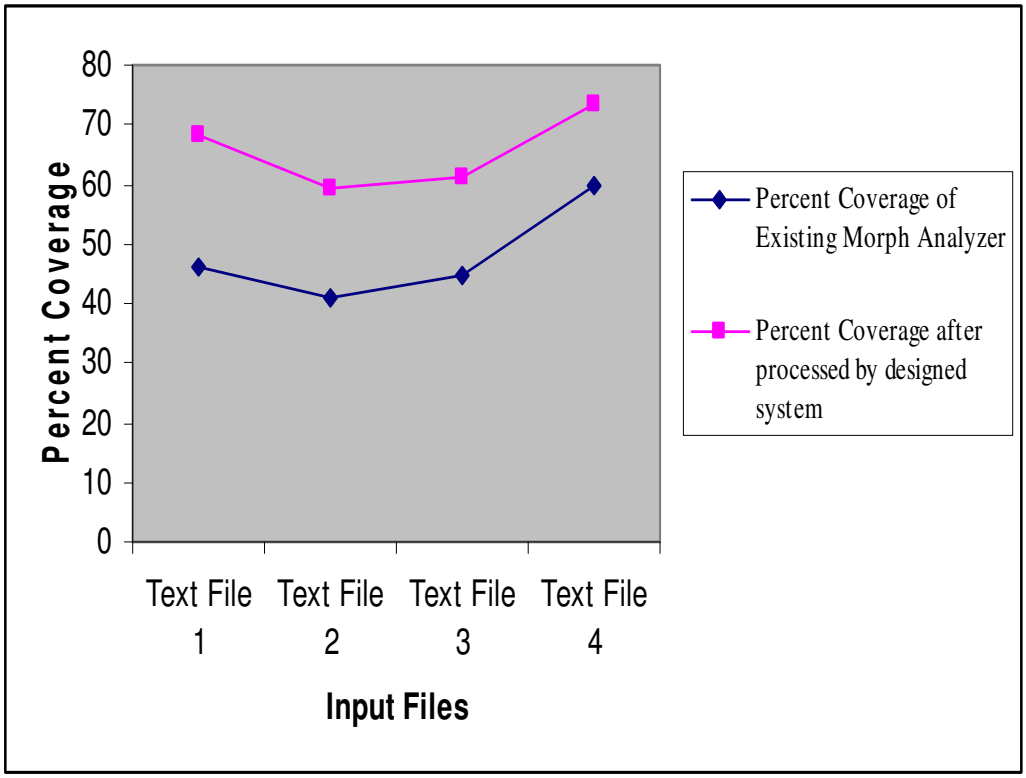


Figure 6.3: Graph of Statistics

Following snapshot shows the coding part done in JAVA Netbeans platform.

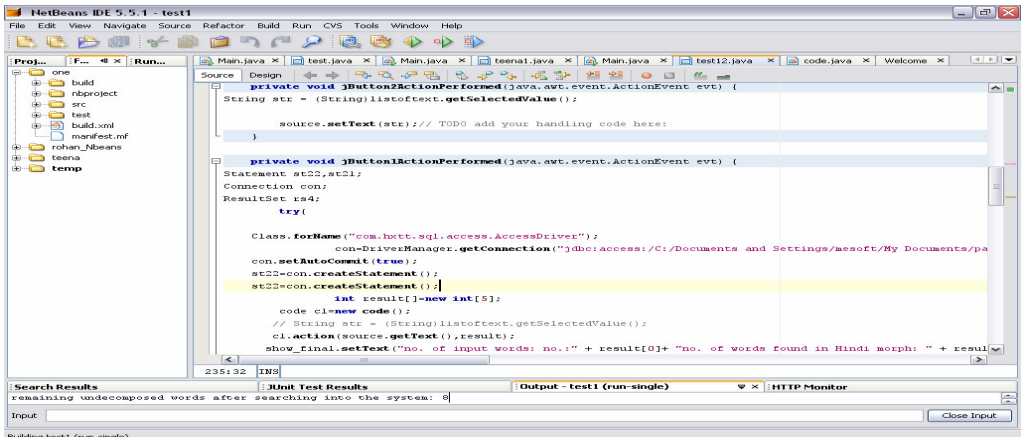


Figure 6.4: Running the program in Netbeans 5.1

Graphical User Interface for the system has been made in JAVA Netbeans platform. Following snapshot shows the interface of the system.

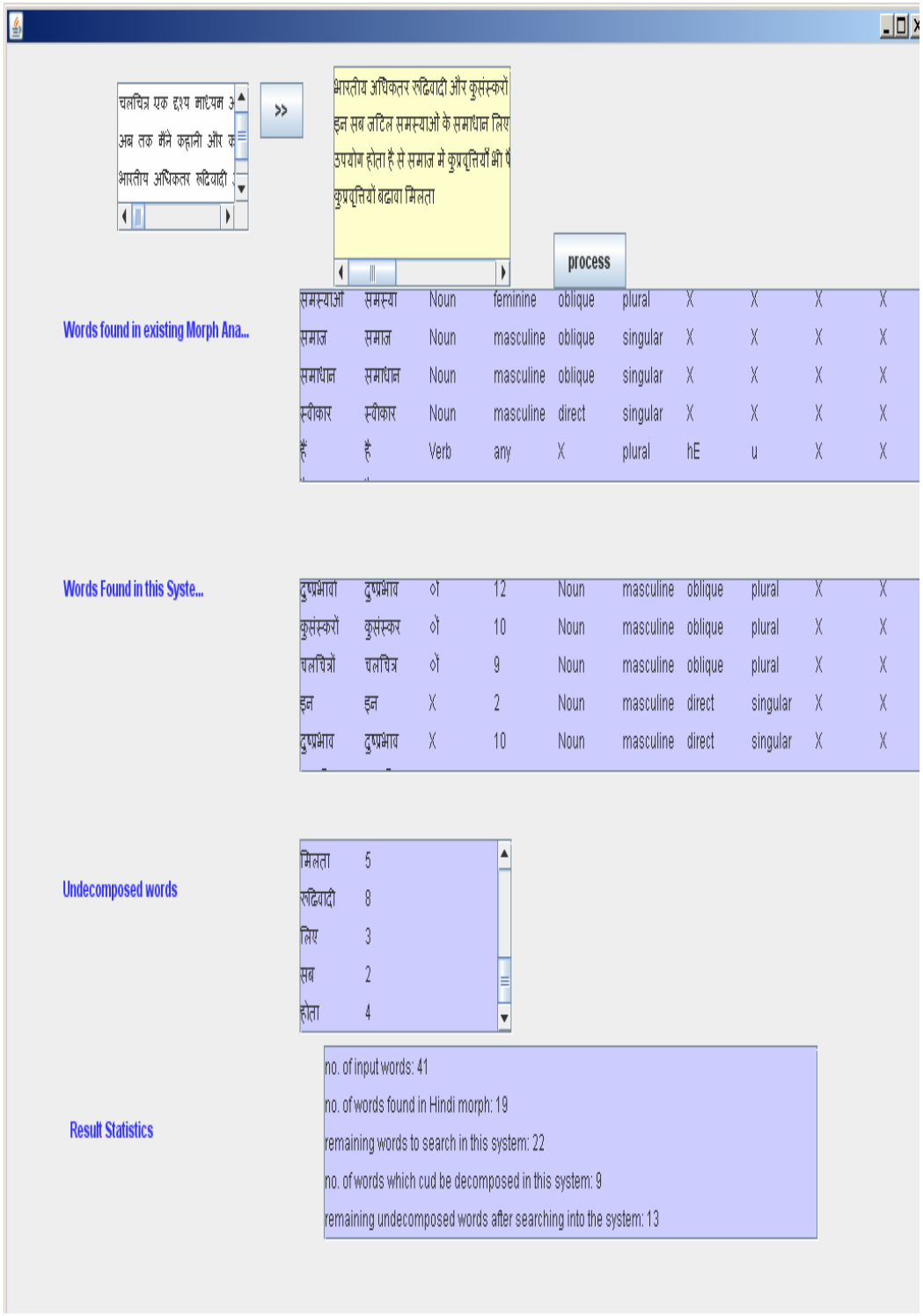


Figure 6.5: Interface of the System

Chapter 7

Conclusions and Future Scope

We have taken the existing morph analyzer and referred the algorithm which is based upon the unsupervised approach for learning of morphology and merged the two.

7.1 Conclusions

Following conclusions are drawn from the implementation:

- Existing morph analyzer has around 50% coverage. With the use of this approach some new words can be categorized. System has around 20% more coverage than the existing system and therefore this system is an edge over the existing system.
- The Rule Based Semi-Supervised Morphological Analyzer for Extending the Range of Existing System added another feather to its cap in the technical development of Hindi by extending the range of existing morph analyzer.
- This approach prefers time and accuracy to memory space. With the memory space not being a problem these days, neither in terms of cost nor in terms of storage requirements, this approach will perform better than the other approaches in which only root words and paradigms are stored in database. Though the algorithm stores the extraneous information *e.g.* word length *etc.* but it is easy in searching of the wordforms.
- The coverage of the system can further be improved with the help of implementing the system on some new text files and algorithm being performed in iterative manner.

7.2 Summary of Contributions

Some of the contributions by this system are:

- The software can be used to search for any Hindi word in it to know its root and other grammatical information.
- The common man can also get an in-depth information about the Hindi words from the software.

- The software will be a big boon for the researchers working on Natural Language Processing of Hindi language as well as the common man.
- Knowing the grammatical information of a word helps in its proper and error free use in sentences.
- This tool also helps in knowing all the words that can be derived from a single root.
- The software will help the beginners to learn new words and the specialists to create new terminology.

7.3 Future Scope

This system can be proved as an important tool for NLP applications. There is a need to test the system for other Indian languages of similar family. For this, the morph analyzer of the language should be used and designed.

Web based solution can be designed for better usage of system and broader testing of system. As we discussed earlier, system finds new words with successive testing on new inputs and will strengthen the system.

References

[1] Alexander cleark, 2001, 'Memory-Based Learning of morphology with stochastic transducers', Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, pp. 513-520.

[2] Erwin Chan, 2006, 'Learning Probabilistic Paradigms for Morphology in a Latent Class Model', Proceedings of the Eighth Meeting of the ACL Special Interest Group on Computational Phonology at HLT-NAACL, pp. 69–78.

[3] Harald Hammarstrom, 2007, 'Unsupervised Learning of Morphology: Survey, Model, Algorithm and Experiments', Technical Report no. 46L [ISSN 1652-876X].

http://publications.lib.chalmers.se/cpl/record/index.xsql?pubid=61878

[4] http://www.advancedcentrepunjabi.org/punjabi_mor_ana.asp for Punjabi Morph analyzer.

[5] http://www.iiit.net/ltrc/morph/ for Hindi Morph Analyzer.

[6] Johnson, H. and Martin J., 2003, 'Unsupervised Learning of Morphology for English and Inuktitut', Proceedings of the Human Language Technology Conference / Conference of the North American Chapter of the Association for Computation Linguistics 2003 (HLT/NAACL-03).

http://citeseer.ist.psu.edu/johnson03unsupervised.html

[7] Mandeep Singh Gill, Gurpreet Singh Lehal, S. S. Joshi, 2007, 'A Full-Form Lexicon based Morphological Analysis and Generation Tool for Punjabi', International Journal of Systematics, Cybernatics and Informatics, Hyderabad, pp. 38-47.

[8] Manish Shrivastava, Nitin Agrawal, Bibhuti Mohapatra, Smriti Singh, Pushpak Bhattacharya, 'Morphology Based Natural Language Processing tools for Indian Languages',

http://www.cse.iitk.ac.in/users/iriss05/m_shrivastava.pdf

[9] Matthew G. Snover, Gaja E. Jarosz, Michael R. Brent, 2002, 'Unsupervised Learning of Morphology Using a Novel Directed Search Algorithm: Taking the First Step', Proceedings of the 6th Workshop of the ACL Special Interest Group in Computational Phonology (SIGPHON), Philadelphia, pp. 11-20.

[10] Nonie Lesaux, 'Morphological analysis: New light on a vital reading skill'.

http://www.uknow.gse.harvard.edu/teaching/TC102-407.html

[11] Rajat Mohanty**,** 3-17 May 2007**, '**MRI Summer School in NLP: Morph Analysis', IIT-Bombay.

[12] Richard Wicentowski, 2002, 'Modeling and Learning Multilingual Inflectional Morphology in a Minimally Supervised Framework', Thesis Report.

http://www.cs.swarthmore.edu/~richardw/pubs/thesis.pdf

[13] Shachi Dave Pushpak Bhattacharyya, 2001, 'Knowledge Extraction from Hindi Texts', Journal of Institution of Electronic and telecommunication engineers, 18(4).

http://www.cse.iitb.ac.in/~pb/papers/iete.pdf

[14**]** Smriti Singh, Kuhoo Gupta, Manish Shrivastava and Pushpak Bhattacharyya, 2006, **'**Morphological richness offsets resource demand- experiences in constructing a POS tagger for Hindi', Proceedings of the COLING/ACL, Sydney, Australia, pp. 779-786.

[15] Sylvain Neuvel and Sean A. Fulop, 2002, 'Unsupervised Learning of Morphology Without Morphemes', Proceedings of the 6th Workshop of the ACL Special Interest Group in Computational Phonology (SIGPHON), Philadelphia, pp. 31-40.

[16] Tom Ritchey, 1998, 'General Morphological Analysis', Adapted from the paper "Fritz Zwicky, Morphologie and Policy Analysis", presented at the 16th EURO Conference on Operational Analysis, Brussels.

[17] Utpal Sharma, Jugal Kalita, Rajib Das, 2002, 'Unsupervised Learning of Morphology for Building Lexicon for a Highly Inflectional Language', Proceedings of the 6th

Workshop of the ACL Special Interest Group in Computational Phonology (SIGPHON), Philadelphia, pp. 1-10.

[18] Yu Hu, Irina Matveeva,John Goldsmith, Colin Sprague, 2005, 'Using Morphology and Syntax Together in Unsupervised Learning', Proceedings of the Second Workshop on Psychocomputational Models of Human Language Acquisition, Chicago, pp.20-28.

List of Papers Published

[1] Teena Bajaj, Parteek Bhatia, "Semisupervised Learning of Hindi Morphology" at 2nd National Conference on Challenges & Opportunities in Information Technology (COIT-2008) held at RIMT-IET, Mandi Gobindgarh on March 29, 2008.

[2] Teena Bajaj, Parteek Bhatia, "Semisupervised Learning approach of Hindi Morphology" at All India Conference on Advances in Communication, Computers, Controls and Knowledge Management (AICACCC-KM) held at PDM College of Engineering, Bahadurgarh on 19-20 February, 2008.

[3] Teena Bajaj, Parteek Bhatia, "Rule Based Semi-Supervised Morphological Analyzer for Extending the Range of Existing System" at 4th International Conference on Challenges and developments in IT (ICCDIT_2008) held at Punjab College of Technical Education, Ludhiana on 30th May, 2008.