Kṛdanta Recognition and Processing for Sanskrit

Dissertation submitted to Jawaharlal Nehru University
In partial fulfillment of the requirements
For the award of the
Degree of
MASTER OF PHILOSOPHY

SURJIT KUMAR SINGH

Special Centre for Sanskrit Studies
Jawaharlal Nehru University
New Delhi-110067
INDIA
2008

विशिष्ट संस्कृत अध्ययन केन्द्र
जवाहरलाल नेहरू विश्वविद्यालय
नई दिल्ली-११००६७

SPECIAL CENTRE FOR SANSKRIT STUDIES
JAWAHARLAL NEHRU UNIVERSITY
NEW DELHI-110067

July 28, 2008

DECLARATION

I declare that the dissertation entitled "**Kṛdanta Recognition and Processing for Sanskrit**" submitted by me for the award of the degree of **Master of Philosophy** is an original research work and has not been previously submitted for any other degree or diploma in any other Institution/University.

[illegible]

(Surjit Kumar Singh)

विशिष्ट संस्कृत अध्ययन केन्द्र
जवाहरलाल नेहरू विश्वविद्यालय
नई दिल्ली-११००६७

SPECIAL CENTRE FOR SANSKRIT STUDIES
JAWAHARLAL NEHRU UNIVERSITY
NEW DELHI-110067

July 28, 2008

C E R T I F I C A T E

This dissertation entitled "**Kṛdanta Recognition and Processing for Sanskrit**" submitted by **Surjit Kumar Singh** to **Special Centre for Sanskrit Studies, Jawaharlal Nehru University, New Delhi-110067**, for the award of the degree of **Master of Philosophy**, is an original work and has not been submitted so far, in part or full, for any other degree or diploma of any University. This may be placed before the examiners for evaluation.

[illegible]

Prof. Varyam Singh
(Chairperson)

Prof. Varyam Singh
Chairperson
Special Centre for Sanskirt Studies
[illegible] 67

[illegible]

Dr. Girish Nath Jha
(Supervisor)

SUPERVISOR
DR. GIRISH NATH JHA
Assistant Professor
Special Centre for Sanskrit Studies
Jawaharlal Nehru University
New Delhi - 110 067

ACKNOWLEDGMENT

*I acknowledge with deep sense of gratitude, constant help and encouragement from my supervisor **Dr. Girish Nath Jha**, which he has provided despite his busy schedule. It is his patience, generosity and excellent guidance which have molded my work in this present shape. He trained me in Java programming and developing database, creating text files and other linguistic resources. He also helped me from time to time in programming and dissertation writing and solved several difficulties.*

*I express sincere thanks and gratitude for Chairperson, **Prof. Varyam Singh** and other faculty members - **Prof. Shashiprabha Kumar, Dr. Hariram Mishra, Dr. Ram Nath Jha, Dr. Rajnish Kumar Mishra, Dr. Santosh Kumar Shukla, and Dr. C. Upendra Rao** of the **Special Center for Sanskrit Studies (SCSS), Jawaharlal Nehru University, New Delhi**, for encouraging and allowing this research to complete. I also acknowledge all the staff members for their cooperation and extending facilities to complete my work.*

I pay my sincere thanks to Amba Kulkarni Ma'am (Deptt. of Sanskrit Studies, University of Hyderabad) for providing data of kṛdanta and invaluable suggestions.

I duly acknowledge University Grants Commission (UGC) for providing me the financial assistance (JRF).

My special thanks to my friend Diwakar Mishra for helping me systematize and structure the ideas for my research and other classmates, Shashikant Tripathi, Diwakar Mani, Sureswar Meher, Anil Pratap Giri of the SCSS. I express my thanks to Sanjeet Jha, Vishvesh, Baldev, for checking data.

My special thanks to my seniors Kuldeep kumar Shukla, Dr. Sudhir Mishra, Dr. R.Chandrashekhar, Subhash Chandra, Muktanand Agrawal, Sachin, Manji Bhadra for helping me to systematize the ideas for my R & D. I extend my special appreciation to Neha, Sneha, Chandrakant, Alok and Ravi.

I am also thankful to my friends Pankaj Kumar Dwivedi, Dev Kumar Mishra, Shiv Sagar Ojha, Rajkumar who always been helpful and inspired me for this work.

With deep sense of indebtedness I express my heartfelt gratitude to ***my parents, relatives and my brothers*** *for their constant motivation and invaluable sacrifice all through my studies.*

[illegible]

SURJIT KUMAR SINGH

Contents

List of Abbreviations

AD	*Aṣṭādhyāyī*
ASR	Academy of Sanskrit Research
JNU	Jawaharlal Nehru University
JSP	Java Server Pages
KV	*Kāśikāvṛtti*
LTRC	Language Technologies Research Centre
MAT	Machine Aided Translation
MT	Machine Translation
MTS	Machine Translation System
NL	Natural Language
NLP	Natural Language Processing
POS	Part of Speech
R&D	Research and Development
RCILTS	Resource Centre for Indian Language Technology Solutions
RSV	Rashtriya Sanskrit Vidyapeetha
SCSS	Special Centre for Sanskrit Studies
SK	*Siddhāntakaumudī*
TDIL	Technology Development for Indian Languages
NLG	Natural Language Generation
OCR	Optical Character Recognition
TTS	Text to speech
WALTT	Web Assisted Learning and Teaching of Tamil
NLSI	New Linguistic Survey of India
CIIL	Central Institute of Indian Languages
LSI	Linguistic Survey of India
ISCII	Indian Standard Code for Information Interchange
CDAC	Centre for Development of Advanced Computing
GIST	Graphic based Intelligence Script Technology
MIT	Ministry of Information Technology
SaHiT	Sanskrit - Indian Languages Translator
AL	Artificial language
AS	*Akṣara-samāmnāya*

SP	*Sūtrapāṭha*
DP	*Dhātupāṭha*
GP	*Gaṇapāṭha*
MWSDD	Monier Williams Sanskrit Digital Dictionary
KRAS	*Kṛdanta* Recognizer and Analyzer for Sanskrit

List of Tables

List of Figures

Transliteration key used in the dissertation

अ	=	a
आ	=	ā
इ	=	i
ई	=	ī
उ	=	u
ऊ	=	ū
ऋ	=	ṛ
ॠ	=	ṝ
ऌ	=	ḷ
ए	=	e
ऐ	=	ai
ओ	=	o
औ	=	au
क्	=	k
ख्	=	kh
ग्	=	g
घ्	=	gh
ङ्	=	ṅ
च्	=	c
छ्	=	ch
ज्	=	j
झ्	=	jh
ञ्	=	ñ
ट्	=	ṭ
ठ्	=	ṭh
ड्	=	ḍ
ढ्	=	ḍh
ण्	=	ṇ
त्	=	t
थ्	=	th
द्	=	d
ध्	=	dh

न्	=	n
प्	=	p
फ्	=	ph
ब्	=	b
भ्	=	bh
म्	=	m
य्	=	y
र्	=	r
ल्	=	l
व्	=	v
श्	=	ś
ष्	=	ṣ
स्	=	s
ह्	=	h
क्ष्	=	kṣ
त्र्	=	tr
ज्ञ्	=	jñ
ऽ	=	’
ं (*Anusvāra*)	=	ṃ
ः (*visarga*)	=	ḥ

Devanagri Input Mechanism according to Baraha software (http://www.baraha.com)

VOWELS

a [अ],	aa/A [आ],	i [इ],	ee [ई],	u [उ],
oo [ऊ],	Ru [ऋ],	RU [ॠ],	lRu [ऌ],	lRU [ॡ],
e [ए],	ai [ऐ],	o [ओ],	au [औ],	aM [अं],
aH [अः]				

CONSONANTS

k [क्],	kh/K [ख्],	g [ग्],	gh [घ्],	~G [ङ्],
c [च्],	C [छ्],	j [ज्],	jh/J [झ्],	~J [ञ्],
T [ट्],	Th [ठ्],	D [ड्],	Dh [ढ्],	N [ण्],
t [त्] ,	th [थ्],	d [द्],	dh [ध्],	n [n],
p [प्],	ph [फ्],	b [ब्],	bh [भ्],	m [म्],
y [य्],	r [र्],	l [ल्],	v/w [व्],	sh/S [श्],
Sh;[ष्]	s [स्],	h [ह्],	kSh [क्ष्],	tra [त्र्],
j~J [ज्ञ्],				

Introduction

Introduction

Kṛdanta analyzer will be a very important component in any Machine Translation System (MTS) or analysis system that attempts to analyze and understand Sanskrit for computational purposes. Any MTS involving Sanskrit as a source language will require a Sanskrit analyzer which must have the following components –

- sandhi analysis
- subanta analysis
- kṛdanta analysis
- taddhita analysis
- samāsa analysis
- gender analysis
- tinanta analysis
- kāraka analysis

as well as many lexical resources. Kṛdanta is primary derivative from verb root and it may be analyzed after *subanta, taddhita* and *samasa* are analyzed because the kṛt suffix may not be visible at the end of word till other later suffixes are separated. Knowledge of kṛdanta is important for Sanskrit reading and speaking. Most of Sanskrit prose literature abound in kṛdanta usage, e.g. *saḥ gataḥ, saḥ gatavān, tena dṛṣṭam, saḥ dṛṣṭavān* in the place of *so 'gamat, so 'paśyat.*

The scope of this work was to develop a system to identify and process kṛdanta in Sanskrit based on Pāṇinian formulations. So far mostly theoretical work has been done in this area but a computational system which identifies the kṛdanta does not exist so far. This work is a very important component in any Natural Language (NL) system that attempts to analyze and understand Sanskrit for computational purposes. Kṛdanta analysis is a critical module and a complex task for any larger NL system for Sanskrit. A Kṛdanta Analyzer system thus is not only essential for any larger Sanskrit NL system, but is also helpful for self-reading and understanding of Sanskrit texts by those readers who do not know or want to go through the complexities of kṛdanta. It will also be helpful for interpretation and simplification of Sanskrit text. Any NL or NL like Sanskrit compiler must have kṛdanta analyzer as a necessary component.

The objectives of this Research & Development (R&D) were the following -

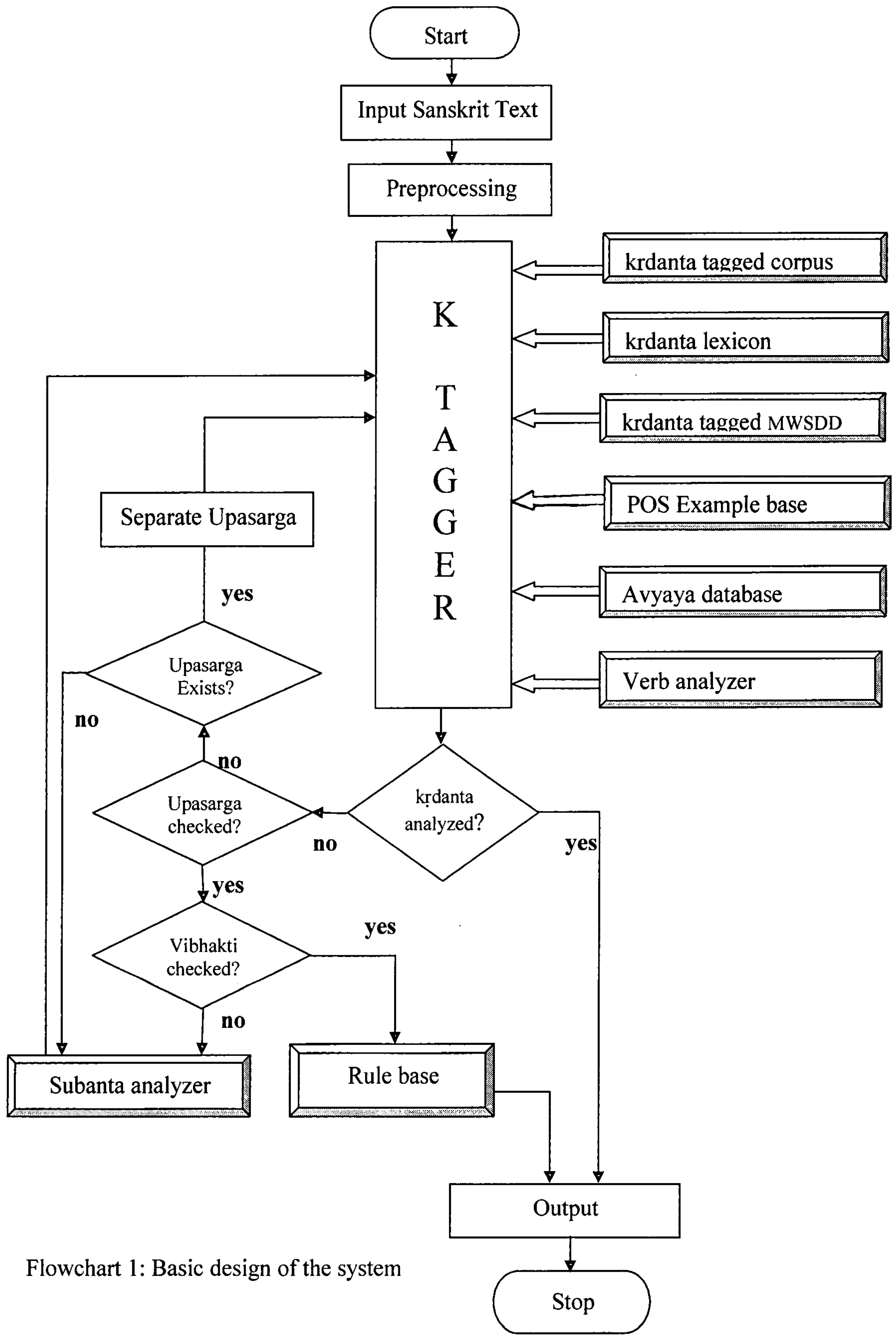
- To build a rule-base and example-base of Pāṇinian kṛdanta (Primary derived nouns) rules for kṛdanta identification and analysis.
- To adapt available e-corpora and customize them for kṛdanta (Primary derived nouns) analysis purposes.
- To build a servlet based online Java engine which will consult the rule-base and the linguistic resources to analyze kṛdanta (Primary derived nouns) of a Sanskrit text.
- To simplify Sanskrit text for self reading, understanding, and also for any Machine (Aided) Translation from Sanskrit to other languages.
- Comprehensive research on the Primary derived nouns rules of Aṣṭādhyāyī (AD), Siddhānta Kaumudī (SK) and Kaśikā vṛtti (KV).

For this R&D, the methodology of computational Sanskrit and software engineering was used. This R&D is based on a hybrid approach of rule base and example base. The study consists of a descriptive, analytical as well as application work. The study is based on the primary and secondary resources available on the topic. The primary sources include *Aṣṭādhyāyī.*, *Siddhāntakaumudī of Bhaṭṭojidīkṣita*, *Kāśikāvṛtti of Vāmana and Jayāditya*, Mahābhāṣya of Patañjali and adapted and customized e-corpora. Secondary materials include several books of grammar, published articles and web resources.

As part of the research, various linguistic resources were developed and adapted according to the need of the system. A verb (tiṅanta) and indeclinable (avyaya) database was adapted to exclude the tiṅanta and avyaya from processing. An example lexicon of kṛdanta is developed from different sources like *Kṛdantarūpamālā* (a concordance of verbal derivative) and *Siddhāntakaumudī.* In this lexicon the rūpa (prātipadika), dhātu (verb), gaṇa (class), kṛt-suffix, verbal-der-suffix, POS_gender are listed. The example-base is for analyzing those forms which are very complex to analyze through rules. Besides this upasargavikāra data, dhatuvikāra data, pratyayavikāra data is also developed for rule-base system. Rule-base system is for analyzing less complicated forms which have some simple and common structure and pattern.

For online processing of Sanskrit text, a Java based web-application has been developed.

The basic design of the system is illustrated through flowchart. That is given below-



Flowchart 1: Basic design of the system

The dissertation titled 'Kṛdanta Recognition and Processing for Sanskrit' is divided into four chapters.

- ❖ The first chapter titled 'Computational Morphology and Sanskrit' introduces NLP, its goal and the tools of NLP, Morphology and Computational Morphology, its complexities and approaches, Survey of R & D in Indian Language Technology. Also discussed Sanskrit and computation, Sanskrit morphology, especially derivational morphology and its importance and need for kṛdanta analysis.

- ❖ The second chapter titled 'Kṛdanta in Sanskrit Grammar' discusses an overview of Pāṇinian system e.g, sūtra and their types, śiva-sūtra, sūtra-pāṭha, dhātupāṭha and gaṇapāṭha. This chapter also discusses kṛdanta in Sanskrit grammar, suffixes joined to dhātus, e.g., dhātu-pratyaya, vikaraṇa-pratyaya, tiṅ-pratyaya and kṛt-pratyaya, an overview of kṛdanta system, classification of kṛt suffixes, kṛdanta tags, kṛt-suffixes table, usage of commonly used kṛt suffixes, role of anubandhas in kṛt suffixes, rules of formation of kṛt-suffixes in general, major characteristic of kṛt suffixes and kṛdanta in Hindi grammar.

- ❖ This third chapter titled 'Kṛdanta Recognition and Analysis describes the lexical resources needed for recognition and analysis of *kṛt* suffixes in a Sanskrit text according to *Pāṇinian* formalism. Strategy of example-base system and rule-base system and the algorithm for them has also been described in this chapter. Finally, the process of rule-base system is illustrated step by step through an example. The important data used in the system is in different tabular and textual forms. The sample of the same has also been given in the chapter.

- ❖ The fourth chapter titled 'Kṛdanta Recognizer and Analyzer for Sanskrit' discusses the partial implementation of the kṛdanta analysis methodology to computer program. The techniques used are Java programming language, Java Servlet Technology and JSP run on Apache Tomcat 4.0 Web-server. The system takes a running text of Sanskrit in Devanāgarī UTF-8 format as input.

The limitations of the system and its implications for future research and development have been summarized in concluding part of the dissertation. The appendices contain the sample data of linguistic resources used to develop the system, the screen shots of the interface and the debugging process of the system. A portable CD has also been enclosed with the dissertation which comprises the sample data of each linguistic resource java objects and screen-shots of the interface with the URL of the system. The system is available online at http://sanskrit.jnu.ac.in.

Chapter – I

Computational Morphology and Sanskrit

Chapter-1

Computational Morphology and Sanskrit

1.1 Introduction

Computattional Linguistics is an inter-disciplinary study of linguistics and computer science that is concerned with computer processing of human language, which includes automatic machine translation of one language into another, the analysis of texts, the use of human language in person-computer interactions, artificial intelligence, and computer modeling of human linguistic competence and performance. Artificial Intelligence is the endowing of machine with humanlike intellectual capabilities. The use of language by machines to communicate with humans is one of the most important manifestations of artificial intelligence.[1]

Natural language processing (NLP) is a subfield of artificial intelligence and computational linguistics. It studies the problems of automated generation and understanding of natural human languages. Natural-language-generation systems convert information from computer databases into normal-sounding human language. Natural-language-understanding systems convert samples of human language into more formal representations that are easier for computer programs to manipulate.[2] The goal of natural language Processing (NLP) is to build computational models of natural language for its analysis and generation.[3] The tools of work in NLP are Automatic summarization, Grammer formalisms, Algorithms and data structures, Foreign Language Reading Aid, Foreign Language Writing Aid, Information extraction, Information retrieval, Machine translation, formalism for representing world knowledge, Named entity recognition, Natural language generation, Optical Character Recognition, Question answering, Reasoning mechanisms, Speech recognition, Spoken dialogue system, Text simplification, Text to speech,Text-proofing.

[1] Fromkin & Rodman, An Introduction to Language.
[2] http://en.wikipedia.org/wiki/Natural_language_processing
[3] Bharati, Akshar, Vineet Chaitanya and Rajeev Sangal, *N L P: A Paninian Perspective*

1.2 Morphology

Morphology is the field of linguistics that studies the internal structure of words; the component of the grammer that includes the rules of word formation. The term Morphology was an adaptation of the German word 'morphologie' first used by August Schleicher in 1859. The history of morphological analysis dates back to the ancient Indian linguist Pāṇini, who formulated the 3,959 rules of Sanskrit morphology in the text *Aṣṭādhyāyī* by using a Constituency Grammar.

1.2.1 Computational morphology

Computational morphology is analysis and generation of word-forms through computational techniques.[4] This morphological information is very useful in analyzing a language because syntactic analysis requires morphological analysis. This morphological information can be used in various NL applications such as parsing, lemmatization, text-to-speech, Machine Translation (MT), spell checker, spell corrector, automatic word separator, text generation and word paradigm builder.

Morphological analysis is the process in which a word is analyzed into its root word and associated morphemes. Morphological analyzers are programs used to provide morphological analysis of a language. They include recognition engine, lexicon/thesaurus, and algorithms to find out stem within an input word and identifying its suffixes.[5] Morphological analysis is a complex task. It has various dimensions which can be described as follows[6] –

- **Complexity of word formation**

  Words are built up by joining morphemes according to the permissible patterns in a language. Typologically, languages are of Agglutinative, Isolating, Inflectional and Polysynthetic types based on how morphemes combine to form words productively

- **Morphological processes:**

  There are essentially three types of morphological processes which determine the function of morphemes. These three processes are inflectional, derivational and compounding.

[4] Jha, Girish Nath. 2007, "*Introduction to Computational Morphology*", Lecture delivered on 5 January 2007 at CDAC, Noida.

[5] *http://www.acroterion.ca/Morphological_Analysis.html*

[6] Oflazer,Kemal. http://folli.loria.fr/cds/2006/courses/Oflazer.ComputationalMorphology.pdf

- **Morpheme combination**

  Morphemes can be combined in a variety of ways to build the words such as concatenation, infixation, circumfixation, templatic combination and reduplication.

1.2.1.1 Different Approaches

For performing different kinds of works, different types of methods are used in Computational Morphology. The most common of them are rule based string processing techniques and finite state techniques.

- **Cut and Paste method**

  Cut and paste is a very popular method in computational linguistics. The canonical form is derived by removing and adding letters to the end of a string. The best known ancestor of these systems dates back to the 1960s[7].

  Another system known as MORPHOGEN (Petheroudakis, 1991) is a commercial toolkit for creating sophisticated cut and paste analyzers[8]. The MAGIC (Schuller, Zierl, 1993) is a cut and paste rule based system in which rules are applied in advance to produce the right allomorph for every allowed combination of a morpheme[9]

- **Finite State techniques**

  Finite state techniques are used in cases where large lexicons are to be checked. It also explains morphotactics better than the cut-paste method. Automatic recognition and generation of word forms was introduced early 80s. Rules of morphological alternations could be implemented using FSTs as a finite state network (Johnson 1972, Kaplan and Kay 1994)[10]. First practical application of

---

[7] Allen, J., Hunnicutt, M. S., and Klatt, D. (1987). *From text to speech---the MITalk system.* MIT Press, Cambridge,
Massachusetts.

[8] Petheroudakis, J. (1991). *MORPHOGEN automatic generator of morphological information for base form reduction.*
Technical report, Executive Communication Systems ECS, Provo, Utah.

[9] Schuller, G., Zierl, M., and Hausser, R. (1993). MAGIC. *A tutorial in computational morphology.* Technical report,
Friedrich-Alexander Universitat, Erlangen, Germany.

[10] Kaplan, Ronald M., and Kay, Martin (1981). *Phonological Rules and Finite-State Transducers.* Paper presented at
the Annual Meeting of the Linguistic Society of America. New York.

model appeared in the 90s (Koskenniemi 83, Karttunen 1993, Antworth 1990, Kartunnen and Beesley 1992, Ritchie, Russell et al, 1992, sproat 1992)[11]. These systems used linked letter trees for the lexicon and parallel FSTs encoding morphemic alternations. The FS techniques are generally used for searching large scale spellchecking wordlists. They also allow bi-directional processing (i.e. both generation and analysis can be performed)

1.3 Survey of R & D in Indian Language Technology

The Linguistic and cultural diversity of India is well known. This is often cited as India's strength. The 200 languages enumerated in the Census are a linguistic subtraction of over 1,600 mother tongues reported by the people indicating their perception of their linguistic identity and linguistic difference.

1.3.1 Indian language families

- Indo Aryan, Dravidian, Munda (Emeneu 1980)
- Indo Aryan, Dravidian, Austro Asiatic, Tibeto Burman, Andamanese (Abbi, 2001)
- Indo Aryan, Dravidian, Austro Asiatic, Tibeto Burman.[12]

1.3.2 Technology Development in Indian Languages (TDIL)

Technology Development in Indian Languages (TDIL) [13] was initiated by the Department of Information Technology, Ministry of ICT or Government of India with the objective of developing Information Processing Tools and Techniques to facilitate human-machine interaction without language barrier; creating and accessing multilingual knowledge resources; and integrating them to develop innovative user products and services. Since then, TDIL has been facilitating and supporting the development of language technology resources in all Indian languages, and promoting their dissemination. Towards this goal it has established 13 Resource Centers for

[11] Kimmo Koskenniemi. *Two-level morphology: A general computational model for word form recognition and production.* Publication No: 11, Department of General Linguistics, University of Helsinki, 1983.
[12] Jha, Girish Nath. Regional & linguistic perspective on internationalization: the case of Hindi/Sanskrit,2007.
[13] http://tdil.mit.gov.in/

Indian Language Technologies (RCILTS) in March 2000 at different Universities and Institutes.[14] These Resource centers are aimed at:

- To improve the quality of life of people of India by enabling to use Information Technology through Indian Languages.
- To develop new products and services for processing information in Indian Languages.
- To facilitate Research in technology areas such as Machine Translation (MT), Optical Character Recognition (OCR), Text-to-Speech (TTS), and Speech Recognition in Indian Languages that leads to product development.

1.3.3 TDIL Resource Centers[15]

Resource Centre	**Languages Associated With**
Indian Institute of Technology, Kanpur	Hindi, Nepali
Indian Institute of Technology, Mumbai	Marathi, Konkani
Indian Institute of Technology, Guwahati	Assamese, Manipuri
Indian Institute of Science, Bangalore	Kannada, Sanskrit (CognitiveModels)
Indian Statistical Institute, Kolkata	Bengali
Jawaharlal Nehru University, New Delhi	Foreign Languages (Japanese, Chinese) & Sanskrit (Language Learning Systems)
University of Hyderabad, Hyderabad	Telugu
Anna University, Chennai	Tamil
MS University, Baroda	Gujarati
Utkal University	Oriya
Department of Computer Science and Application Thapar Institute of Engg. & Tech., Patiala	Punjabi
ERDCI, Trivendrum	Malayalam
CDAC, Pune	Urdu, Sindhi, Kashmiri

Table 1.1: TDIL Language Resource Centre

[14] http://tdil.mit.gov.in/languagetechnologyresourcesapril03.pdf
[15] *http://tdil.mit.gov.in/*

1.3.4 The Sanskrit Heritage Site

Gerard Huet, Director, INRIA[16] has developed various computational tools for Sanskrit, which are available online. The **Declension Engine** takes a nominal base with its gender information as input and gives all the nominal inflectional forms as output. The **Conjugation Engine** is for verb generation. It takes root as input and gives all the possible forms of the verb root in its *ātmane* and/or *parasmai* terminations, in *kartṛ* and *karmaṇi/bhāve* voices in eight *lakāra*-s. **Lemmatiser** and **Sanskrit Readers** are the analyzers. While the Lemmatiser tags a given simple inflected noun or a verb (without *upasarga*-s), the Sanskrit Reader Companion does analysis of a given phrase or a simple sentence, segments it into individual words and tags each word.

1.3.5 The Sanskrit (Digital) Library

The Sanskrit Library Project, under the guidance of Peter M. Scharf, Classics Dept., Brown University, is engaged in philological research and education in Vedic and classical Sanskrit language and literature by documenting, collecting, preserving, and publishing oral, written, and printed texts in digital form, and by developing innovative research and educational tools. Current research involves linguistic issues in encoding, computational phonology and morphology, OCR for Indic scripts, and mark-up of digitized Sanskrit lexica. The Sanskrit Library currently contains two independent study Sanskrit readers, Sanskrit grammatical literature, dynamic software for nominal and verbal inflectional morphology, a digital version of W. D. Whitney's *The Roots, Verb-Forms, and Primary Derivatives of the Sanskrit Language*, and instructional materials.[17]

1.3.6 The Clay Sanskrit (Digital) Library

The **Clay Sanskrit Library** is a series of books published by New York University Press and the JJC Foundation. Each work features the text in its original language (transliterated Sanskrit) on the left-hand page, with its English translation on the right. The series was modeled on the Loeb Classical Library, and its volumes are bound in teal cloth. The JJC Foundation was founded by John P. Clay and his wife, Jennifer.

[16]http://sanskrit.inria.fr/
[17] http://sanskritlibrary.org/

John Clay.[18] The Clay Sanskrit Library has been created to introduce Classical Sanskrit literature to a wide international readership. Forty-four leading scholars from ten countries are cooperating to produce fresh new translations that combine readability and accuracy. The first fifteen titles appeared in 2005, co-published by NYU Press and the JJC Foundation, followed by nine volumes in 2006 and eight volumes in 2007. They will be followed by three volumes in March 2008 and six more in August 2008. The selection will focus on drama, poetry and novels, together with the famous epics.[19]

1.3.7 University of Pennsylvania

Vasu Renganathan, Penn Language Centre (UPENN – PLC) has been working on Tamil NLP system for quite long and has developed a Tamil Morphological Tagger. This program identifies suffixes in transliterated input Tamil word(s), labels them and separates the root. This version processes sentences from text files. He is also working on online Hindi morphological tagger, Tamil transliteration and viewer program (This program converts Tamil document written in Roman script to Tamil script) and Web Assisted Learning and Teaching of Tamil (WALTT) Projects. [20]

1.3.8 PC Kimmo

PC-KIMMO is a new implementation for microcomputers of a program dubbed KIMMO after its inventor Kimmo Koskenniemi (Koskenniemi 1983). It is of interest to computational linguists, descriptive linguists, and those developing natural language processing systems[21]. The program is designed to generate (produce) and/or recognize (parse) words using a two-level model of word structure in which a word is represented as a correspondence between its lexical level form and its surface level form. Work on PC-KIMMO began in 1985. A PC-KIMMO description of a language consists of two files provided by the user:

- A rules file, which specifies the alphabet and the phonological (or spelling) rules.

[18] http://en.wikipedia.org/wiki/Clay_Sanskrit_Library
[19] http://www.claysanskritlibrary.org/
[20] http://www.sas.upenn.edu/~vasur/project.html
[21] http://www.sil.org/pckimmo/

- A lexicon file, which lists lexical items (words and morphemes) and their glosses, and encodes morph tactic constraints.

The theoretical model of phonology embodied in PC-KIMMO is called two-level phonology. The two functional components of PC-KIMMO are the generator and the recognizer. The generator accepts as input a lexical form, applies the phonological rules, and returns the corresponding surface form. It does not use the lexicon. The recognizer accepts as input a surface form, applies the phonological rules, consults the lexicon, and returns the corresponding lexical form with its gloss.

The main components of the PC-KIMMO system[22].

```
              +-----------+         +-----------+
              |  RULES    |         |  LEXICON  |
              +----+------+         +------+----+
                   |-------+       +-------|
                           |       |
                           v       v
Surface Form:   +------------------+       Lexical Form:
 spies ------->|    Recognizer     |----> `spy+s
                +----+-------------+       [N(spy)+PLURAL]
                     |
                     v
                +------------------+
 spies <-------|    Generator      |<----- `spy+s
                +------------------+
```

Diagram 1.1: Structure of the PC-KIMMO

PC-KIMMO runs on the Windows, Macintosh and UNIX systems. There are two versions of the PC-KIMMO release, one for IBM PC compatibles and one for the Macintosh. Each contains the executable PC-KIMMO program, examples of language descriptions, and the source code library for the primitive PC-KIMMO functions. The PC-KIMMO executable program and the source code library are copyrighted but are made freely available to the general public under the condition that they not be resold or used for commercial purposes. The PC-KIMMO release contains the executable PC-KIMMO program, the function library, and examples of PC-KIMMO descriptions for various languages, including English, Finnish, Japanese, Hebrew, Kasem, Tagalog, and Turkish. These are not comprehensive linguistic descriptions; rather they cover only a selected set of data[23].

[22] http://www.sil.org/pckimmo/

[23] http://www.sil.org/pckimmo/

1.3.9 CLAWS

CLAWS (Constituent Likelihood Automatic Word-tagging System)[24] POS tagging software for English text has been continuously developed by University Centre for Computer Corpus Research on Language (UCREL) in early 1980s. The latest version of the tagger, CLAWS-4, was used to POS tag 100 million words of the British National Corpus (BNC). CLAWS, has consistently achieved 96-97% accuracy (the precise degree of accuracy varying according to the type of text). Judged in terms of major categories, the system has an error-rate of only 1.5%, with c.3.3% ambiguities unresolved, within the BNC. More detailed analysis of the error rates for the C5 tagset in the BNC can be found within the BNC Manual[25].

1.3.10 Central Institute of Indian Languages (CIIL, Mysore)

The New Linguistic Survey of India (NLSI) initiative by CIIL Mysore under the guidance of Prof. Uday Narayan Singh is commendable and is going to have far reaching impact in this field. The presentation made by Prof. U.N. Singh[26] at LSI conference at BHU gave details of the project. A massive training programme of linguists was successfully conducted at CIIL.

Besides this, CIIL has developed 45 plus million word corpora in Scheduled Languages under the scheme of TDIL.[27] These are the domain specific corpora in the area of newspaper, child language, pathological speech/language data, speech error data, and Historical/Inscriptional database of Indian language. They have POS tagged corpora and chunked corpora. This corpora was created as ISCII format. The CIIL has converted this data in Unicode in collaboration with the Lancaster University. The Institute is now developing corpora in languages like Bodo, Dogri, Maithili and Santhali. The institute is providing these data free of cost to research purpose.

1.3.11 C-DAC

DESIKA[28] (Natural Understanding System), a software package, developed by Indian Heritage Group, C-DAC, Bangalore led by P. Ramanujan, claims to generate and analyze plain and accented written Sanskrit texts using grammar rules of Pāṇini's

[24] http:// www.comp.lancs.ac.uk/ucrel/claws
[25] www.comp.lancs.ac.uk/ucrel/claws
[26] Singh, Udaya Narayan, 2006, 28th AICL, BHU.
[27] http://www.ciilcorpora.net/
[28] Deśika (Natural Language Understanding System), http://tdil.mit.gov.in/download/Desika.htm

Aṣṭādhyāyī, with an exhaustive database of Amarakośa and heuristics of semantics and contextual processing from Nyāya and Mīṃāmsā Śāstra. It also claims to analyze Vedic texts. The analysis module of the software is called a general purpose Sanskrit parser which claims to dissolve and identify the compound and combined word forms, though the present version of the software downloadable from the TDIL (Technology Development for Indian Languages) website has subanta generation module only. This module has two modes: **choose mode** declines for the *prātipadika*-s already present in the list and **edit mode** declines for the entries which are not present in the list for which suitable gender and paradigm should be selected. DESIKA is also supposed to include Vedic processing and śābdabodha[29].

Sanskrit Authoring System (VYASA)[30] claims to be a robust multilingual document editor with transliteration among the Indian Languages and Roman, sorting and searching facilities, indexing, and concordance. It also says that it provides tools for analyses at morphological, syntactic and semantic levels. Tools for searching/indexing/sorting, lexical updation, lexical tagging, extraction/indexing of quotations in commentaries/explanations, transliteration facility, word split programs for *sandhi* and *samāsa*, poetry analysis (textual/metric/statistical), statistical tools like concordance, thesauri, and electronic dictionaries are also said to be included. This system is not available anywhere to evaluate or check.

1.3.12 Academy of Sanskrit Research, Melkote

Academy of Sanskrit Research (ASR), Melkote under the Directorship of Prof. M.A. Lakshmitatachar has been working on NLP in Sanskrit and other Indian languages for more than 10 years. TDIL website hosts software *śābdabodha* developed at ASR, Melkote. It is said to be an interactive application built to analyze the semantic and syntactic structure of Sanskrit sentences. This software works on DOS 6.0 or higher with GIST (Graphic based Intelligence Script Technology) shell on Windows 95 platform. The software is said to include a user interface. This claims to process all types of sentences of Sanskrit, and can handle generation and analysis of *subanta* of

[29] C-DAC R&D Activies: Development of Desika A Natural language Understanding (NLU) system for Sanskrit. [1990 - 1994], http://www.cdac.in/html/ihg/activity.asp
[30] C-DAC, R&D Activities: Developing Sanskrit Authoring System (VYASA), http://www.cdac.in/html/ihg/activity.asp and The House Magazine of C-DAC, Pen to Paper Developing Sanskrit Authoring System – VYASA, http://www.cdac.in/html/connect/3q2000/art10a.htm

more than 26,000 stems, *tiṅanta* conjugations of roots, in two voices, ten *lakāras* and three modes viz. *kevala tiṅanta, ṇijanta* and *sananta.* It also handles the generation and analysis and identification of case inflected forms of 11 types of *kṛdanta* of 150 roots. Apart from this it is said to have a database of 690 *avyayas*, 26,000 nominal stems, 600 verbal roots, *kṛdanata* forms of 600 verbal roots, 5 *taddhita* suffixes. For handling the semantic analysis, a matrix of 52 sets of nouns with their synonyms amounting to 300 nouns, 27 actions denoted by nearly 200 verbs are said to have been prepared.[31] This institution is also working with 20 odd software tools like Bodha (Sentence disambiguation system according to *śābdabodha* of *navīna nyāya* system), Śemuṣī (*subanta* generator/analyser), Prajñā (*tiṅanta* generator/analyser), Cetanā (*kṛdanta* generator/analyser), Pāṇini (*sandhi* joiner according to *Pāṇinian* rules) etc. which are in the pipeline for release.[32]

1.3.13 Rashtriya Sanskrit Vidyapeetha (RSV), Tirupati

RSV Tirupati has been working on developing linguistic resources for NLP in Sanskrit. Prof. K.V. Ramakrishnamacharyulu, (presently V.C. of Rashtriya Sanskrit Sansthan, Jaipur) and Dr. Srinivasa Varkhedi along with Prof. Vineet Chaitanya and Amba P. Kulkarni have initiated many projects and have developed many tools like *pada-ccheda*, which segregates Sanskrit compound words into its components, which works on Sanskrit ISCII text in Linux environment. It is also working on developing Sanskrit MorphA. An initial analyzer developed in collaboration with IIIT-H is already online. Apart from this it is also concentrating on *kṛdanta* and *tiṅanta* analyzers and also generators for *subanta, tiṅanta* and *samāsa.* RSV Tirupati along with C-DAC Bangalore, Ahobila Mutt Sanskrit College Madhurantakam Tamil Nadu, Poorna Prajna Samshodhana Mandiram Bangalore, Chinmaya International Foundation Veliyanad Kerala, ASR Melkote Karnataka, IIIT-H and Dept. of Sanskrit H.S.Gour University Sagar Madhya Pradesh has combined initiative to develop a large Sanskrit Corpus.[33] RSV, Tirupati also worked on a project of Veda and śāstrārtha recording, funded by the Ford Foundation of USA.

1.3.14 RCILTS – Utkal University

[31] Language Processing Tools: TDIL website, http://tdil.mit.gov.in/nlptools/ach-nlptools.htm
[32] Academy of Sanskrit Research, Melkote, http://www.sanskritacademy.org/About.htm
[33] RSV Tirupati, http://rsvidyapeetha.ac.in and http://www.sansknet.org

RCILTS – Oriya Centre at the Department of Computer Science and Application, Utkal University[34] has been working on the various areas of NLP led by Prof. Sangamitra Mohanty and funded by Ministry of Information Technology (MIT). The institution has developed an Oriya OCR **'DIVYADRUSTI'** and text-to-speech for Oriya, Hindi and Bengali. It is also working on building Oriya Machine Translation (OMT), Ori-Net (Word-Net for Oriya), parsers, morphological analyzers and spell checkers for Oriya language. Besides these Oriya NLP tools, the centre also claims to have developed Sanskrit Word-Net (San-Net) using *Navya-Nyāya* philosophy and *Pāṇinian* Grammar. The system has 300 Sanskrit words (250 Nominal words and 50 Verbal words) and it explains synonymy, antonym, hyponymy, hypernymy, holonymy and meronymy relationship of words with their analogy, etymology, and definitions. This Centre is also developing POS tagger and Shallow Parsers for Oriya. For POS tagger they are using Kāraka theory and rule based methods. POS tagger is being used to develop tagged corpora in Oriya.[35]

1.3.15 IIIT Hyderabad

Language Technologies Research Centre (LTRC) at IIIT, Hyderabad is a leading NLP research centre. LTRC with the collaboration of Govt. of India, Carnegie Mellon University's Language Technology Institute, University of Pennsylvania, HP Labs, Google, TCS and other academic institutions aims at developing technologies related to MT among English and Indian languages, speech processing for Indian languages, search engines, information extraction and retrieval for English and Indian languages. LTRC has developed **'Shakti'**[36] system for MT from English to Indian languages. It combines rule-based approach with statistical approach and currently claims to work for three target languages: Hindi, Telgu and Marathi. Besides this, LTRC is also developing several machine readable bilingual dictionaries, tense aspect modality dictionary and multi-word expressions dictionary for language pairs of English-Hindi, English-Marathi, and English-Bengali. In addition to the above, LTRC is also working on various projects such TTS for Telugu and Hindi, Telgu to Hindi Machine Translation, morphological analyzers for Indian languages, POS tagger for Hindi and

[34] RCILTS, Utkal University, http://www.ilts-utkal.org,

[35] Sangamitra Mohanty etal., 2004, "Rule based Part-of-Speech Tagging for Oriya Tagged Corpora: Based on Karaka Theory", in the Proceedings of International Symposium on Machine Translation NLP and TSS, New Delhi, pp.160-165.

[36] Shakti, LTRC, IIIT, Hyderabad, http://www.iiit.net/ltrc/index.html

Bengali. The Search and Information Extraction Lab (SIEL) focuses on solving problems in the areas of Information Retrieval and Extraction using NLP techniques. SIEL is currently focusing on applications areas like '**Ask Buddha**' (Web based question answering system in News), General Search Engines, Indian Language Search Engines, Document Categorization, Document Summarization, Information Extraction and Ontologies.[37]

1.3.16 IIT Bombay

Resource Centre for Indian Language Technology Solutions (RCILTS), IIT-Bombay[38], led by Dr. Pushpak Bhattacharya is a happening place for NLP in India. The institution aims to offer information technology through Indian languages, and to develop resource information in Indian languages and Sanskrit in a way relevant to the present day needs. The institution has developed an online **Hindi Wordnet** which is a lexical database for nearly 60000 Hindi words. It takes input in Unicode *Devanāgarī* and gives different synonyms of the word with their example in Hindi sentence. It also uses an inbuilt keyboard to enter the input.[39] The institution is working on POS taggers for Hindi and Marathi and MT systems among multiple languages with a semantic net like representation called the Universal Networking Language (UNL) as interlingua. This interlingua is based on the concepts of language independent words, relations and attributes which are captured in lexical resources like the wordnet.

1.3.17 AU-KBC Research Centre

NLP Group at Anna University KB Chandrashekar (AU-KBC) Research Centre, Madras Institute of Technology, Chennai is working on Tamil NLP. Among many other NLP endeavors, AU-KBC has developed Tamil-Hindi Machine Aided Translation (MAT) system on the model of **Anusaraka** which is said to have a performance of 75%. This MAT system requires a MorphA, which uses the Paradigm-based approach and implemented using Finite State Mechanism. This MorphA is said have coverage of 95% on Central Institute of Indian Languages (CIIL) corpus. Both these systems have a demo and online service on their website.[40] The centre is also

[37] http://search.iiit.ac.in/
[38] IIT, Bombay, http://www.cse.iitb.ac.in
[39] http://www.cfilt.iitb.ac.in/wordnet/webhwn/wn.php
[40] AU-KBC Research Centre - http://www.au-kbc.org/frameresearch.html

working on developing MT systems between Tamil and other languages particularly English and Hindi, a Tamil Word-net in collaboration with Dr. S Rajendran of Tamil University, Thajavur and a POS tagger for Tamil.

1.3.18 Jawaharlal Nehru University (JNU)

The RCILTS – Sanskrit, Japanese, Chinese unit of JNU, under the leadership of Prof. G.V.Singh claims to have developed web based Sanskrit Learning System for the use of scholars for designing Knowledge based systems based on the Indian traditions. The unit has developed a computational module of *Aṣṭādhyāyī* of *Pāṇini*, Sanskrit-English lexicon, English-Sanskrit lexicon and a lexicon of *Nyāya* terms. It also says that it has made some efforts on the *sandhi* analysis system.[41]

Girish Nath Jha[42], has developed a Nominal Inflection Generator for Sanskrit using Prolog as part of his M.Phil, dissertation. This program generates all the inflections of *subanta* given a Sanskrit word with gender and ending letter information.

Special Center for Sanskrit Studies (SCSS), JNU

The Computational Linguistics R&D at Special Centre for Sanskrit Studies J.N.U., since 2002 under the supervision of Dr. Girish Nath Jha, doing research in several areas of language technology for Sanskrit and other languages. Currently they are focusing on developing Sanskrit analysis tools for building Sanskrit - Indian Languages Translator (SaHiT). So far, the following tools and resources have been developed - *Sandhi* Splitter (vowel *sandhi*), *Sandhi* Generator, *Subanta* analyzer, *Tiṅanta* analyzer, *Tiṅanta* generator, POS tagger, *Kāraka* analyzer, Online Multilingual Amarakośa, Andamanese verb analyzer. All these tools are running on Apache Tomcat Platform using Java Servlet and MSSQL Server 2005 as back end. The tools developed can be used live at (http://sanskrit.jnu.ac.in). Some of them can be described as follows-

Online Multilingual Amarakośa (http://sanskrit.jnu.ac.in/amara/index.jsp) - A project on Amarakośa under the guidance of Dr. Girish Nath Jha, has been built up in SCSS, JNU. It is a Multilingual Online project, funded by UGC under UPOE program.

[41] RCILTS, JNU – Achievements: http://tdil.mit.gov.in/SanskritJapaneseChinese-JNUJuly03.pdf

[42] Jha, Girish Nath, 1993, 'Morphology of Sanskrit Case Affixes: A Computational Analysis', M.Phil., submitted to JNU, New Delhi.

The Unicode based software supports seven languages- Sanskrit, Hindi, Kannada, Punjabi, Bangla, Oriya and English and allows the user to search the synonym from one language to another. The output displays the grammatical and semantic category of the word, its base word, reference and ontological information. The software also provides the facility to enter and edit the data by language experts. The software will be extended as a multilingual interface, search engine and text processing tool.

R. Chandrashekhar, as part of his Ph.D. thesis, has developed a **POS tagger** for *sandhi*-free classical Sanskrit prose text which is an online system running on Apache Tomcat platform using Java Servlet. The system will be the basic requirement for the further R&D on the Sanskrit-Indian Languages MT Systems.

Subash Chandra, as part of his M.Phil. Dissertation has developed a **Sanskrit subanta Recognizer and Analyser System** which is an online system on Apache Tomcat platform using Java Servlet. The system uses a hybrid approach of *Pāṇinian* formalism and example-based techniques and gives a comprehensive computational analysis of *subanta-padas* in a (*sandhi-rahita*) Sanskrit text of *Devanāgari* script and does basic tagging of verbs and *avyayas* too. The system can be used for larger processing of Sanskrit, text simplification and MT. The system has a reasonable accuracy on simple Sanskrit prose texts.

Sudhir Kumar Mishra, as part of his Ph.D. thesis, has developed a ***Kāraka* Analyzer** for *Laukika* Sanskrit prose text based on *Pāṇini* and *Kātyāyana Kāraka* formulations. This work can be an important component in any Sanskrit-Indian language translation system.

The following tools - **Russian-English divergence marker, gender analyzer for Sanskrit noun phrases** are partially developed so far. Other important R&D currently underway are - Sanskrit e-learning, Sanskrit derivational morphology analyzer, Mahābhārata indexer (*ādi parvan*), and *Aṣṭādhyāyī* indexer.[43]

Besides the above mentioned centres, the following institutions, organizations, companies are actively engaged in NLP R&D for Indian languages- Microsoft India,

[43] http://sanskrit.jnu.ac.in/index.jsp

IBM, HP Lab, HCL, Webdunia, Thapar Institute of Engineering and Technology, Patiala, Vanasthali Vidyapeeth, Rajasthan, Malaviya Centre for Information Technology Localization, BHU, Varanasi, Indian Statistical Institute, Kolkatta etc.

1.4 Sanskrit and Computation

Sanskrit is virtually an unexplored treasure for modern computational linguistics and AI researchers. The Sanskrit language was studied to a high degree of formalization from high antiquity by genius linguists such as *Pāṇini*, and a continuous tradition of commenting and refining his work (*Kātyāyana, Patañjali, Bhartṛhari, Nāgeśa Bhaṭṭa* etc.), still very much alive, leaves hope for the emergence of new computational models of linguistic treatment, well tuned by definition to the Sanskrit language. The Sanskrit corpus contains a wealth of knowledge and wisdom (most of which is also available in electronic media) which has not been yet properly brought to light, despite centuries of both Western and Indian scholarship. Sanskrit benefits from meticulously checked databases of verb forms, noun paradigm classes, and a host of other information necessary for building the computational tools for analysis and generation of Sanskrit texts.[44] By these reasons Sanskrit can very well be the 'lingua franca' for NLP research.

In the case of Sanskrit computation there are some theoretical issues to be kept in mind. The first thing is the difference in the nature of Natural language (NL) and Artificial language (AL) and the status of Sanskrit. To understand the need and possibilities of formalisation of Sanskrit language and grammar, the advantages and disadvantages of NL and AL are to be kept in mind. On one hand, AL is necessary for communicating with machine, on the other, NL is more expressive and capability of handling human communication needs. NL is more ambiguous and AL is less ambiguous. NL tolerates errors, but AL has zero tolerance for errors. NL is used in Human to Human communication while AL is used in Human to Machine and Machine to Machine communication. It appears that Sanskrit stads somewhere between a NL and an AL – more precise than a typical NL and less mechanical and more expressive than an AL.

[44] Jha Girish et.al., 2007, Proc. Of FISSCL, INRIA, Paris, Pg. No. iii

The second issue is the nature of Sanskrit language. In comparison to other languages, Sanskrit is easily adaptable for computing because it has a regular structure, smaller speech community with almost non existent regional variations. . Pāṇini's grammar is the only complete grammar for any natural language so far. It has been used as a NL for communicating in day to day life and also as an adapted metalanguage for writing precise shāstraic expressions.

The third point which goes in favour of Sanskrit is the fact that most NL applications need fair amount of knowledge computing for robustness and usefulness purposes. Sanskrit contains explict theories of knowledge in *Nyāya* and *Mīmāṁsā Śāstra* and also a continuous lexicographic tradition

While Sanskrit can help NLP on various fronts, it can also be helped in following ways:

- Electronic storage and publication
- Access and preservation of Sanskrit corpus- Vedic/Laukika
- Sanskrit computational lexicography
- Sanskrit-Indian languages Machine Translation
- Automatic reading of manuscripts (OCR)
- Textual interpretation
- Computational pedagogy of Sanskrit
- Building interactive indices, glossary etc.

1.5 Sanskrit morphology

Sanskrit has two types of morphology- nominal and verbal. In Sanskrit, a syntactic unit is called *pada.* **Cardona[45] (1988)** posits the formula for Sanskrit sentence **$(N\text{-}E^{n})p\ldots(V\text{-}E^{v})p$.** According to *Pāṇini*, *Pada* can be nominal (*subanta*) or verbal (*tiṅanta*). These forms are formed by inflecting the stems and hence they are part of Sanskrit inflectional morphology. *Yāska* had divided the Sanskrit word-forms into four types- *Nāma, Ākhyāta, Upasarga, Nipāta.*

[45] George Cardona, 1988 Pãnini, His Work and its Traditions, vol ... i (Delhi: MLBD, 1988)

The derivational morphology in Sanskrit includes the study of primary forms (*kṛdanta*) and secondary forms (*taddhitānta*), feminine forms (*strī pratyayānta*), derived verb-forms etc.

Sanskrit has approximately 2014 verb roots (including *kaṇdvādi),* classified in 10 *gaṇas*, the derived verb forms can have 12 derivational suffixes[46]. These can have *ātmanepadī* and *prasmaipadī.* A verb root may have approximately 2190 (tense, aspect, number etc.) morphological forms.

Sanskrit Nominal morphology of two types, Primary [*kṛdanta* (roots forms that end with *kṛt* suffixes)] and secondary [*taddhitānta* (noun forms that end with *taddhita* suffixes)]. Secondary nominal morphology may be of following types like- *samāsānta* (compound nouns), *strīpratyayānta* (feminine forms) etc. They can also include *upasargas* (prefix) and *avyayas* (indeclinables) etc. According to Pāṇini, there are 21 morphological suffixes (seven *vibhaktis* and combination of three numbers = 21)[47] which are attached to the nominal bases (*prātipadika*)[48] according to syntactic category, gender and end character of the base.

1.5.1 Derivational morphology

Derivational morphology, changes the meaning of words by applying derivations. Derivation is the combination of a word stem with a morpheme, which forms a new word, which is often of a different class, for example, *gam* becomes *gantṛ, gantum* and *gatavat.* The suffixes *tṛc, tumun* and *ktavatu* change the class and meaning of the base word. Derivational Morphology can be quite complicated.

Sanskrit derivational morphology is basically of four types but many times more than one of these together form very complex derivations. Derived nouns as the forms which are formed by inflecting *Kṛt Taddhita* or *strī* or *Samāsa* in verb root or simple nouns or derived nouns.

---

[46] सन्-क्यच्-काम्यच्-क्यङ्-क्यषोऽथाऽऽचारक्विब्-णिज्यङस्तथा ।
यगाय ईयङ् णिङ् चेति द्वादशाऽमी सनादयः ॥

[47] स्वौजसमौट्छष्टाभ्याम्भिस्ङेभ्याम्भ्यस्ङसिभ्याम्भ्यस्ङसोसांङ्योस्सुप ।४।१।२॥

[48] अर्थवधातुरप्रत्ययः प्रातिपदिकम् ।१।२।४५॥, कृत्तद्धितसमासाश्च ।१।२।४६॥

Sanskrit derived nouns are of the four types-

- *kṛdanta* (primary derived)
- *taddhitānta* (secondary derived)
- *strīpratyayānta* (feminine forms)
- *samāsānta* (compound)

1.5.1.1 kṛdanta

The primary nominal derivatives from the verb roots are *kṛdanta*. The derived *kṛdanta* fall into categories of *substantives, participles, gerunds, infinitives* and *indeclinables*. when a verb root is operated with *kṛt* suffixes then the derived form is *kṛdanta*. *kṛt suffixes* are of three kinds according to the *Siddhāntakaumudī* (SK) by *Bhaṭṭojidīkṣita-*

***kṛtya* suffixes**

kṛtya suffixes are always used in *bhāva-vācya* and *karma-vācya* and are in neuter singular. *kṛtya* suffixes are *tavyat, tavya, anīyar, kelimar, yat, kyap, ṇyat etc.* For example- *paṭhitavyam, pātavya, paṭhanīya, pacelima, jeyam, deyam,* etc.

***Pūrvakṛdanta* suffixes**

These suffixes uses in *kartṛvācya* only. These suffixes are *ṇvul, tṛc, lyu, ṇini, ac, ka, aṇ, ṭa, khaś, khac, kvanip, ḍa, kta, ktavatu, śatṛ, śānac, śākan, u, kvip, itra, etc.* For example *karttā, kumbhakāraḥ, janamejayaḥ, pāṭhakaḥ, paṭhantī* etc.

***uttarakṛdanta* suffixes**

These suffixes are *tumun, ghañ, erac, ap, ktṛ, athuc, nañ, nan, ktin, khal, yuc, ktvā, lyap, ṇamula, uṇa etc.* For example, *gantum, khāditum, svapnam, gatiḥ, gatvā, vihāya, ādāya* etc.

1.5.1.2 taddhitānta

The secondary derivatives are called *taddhita*. *taddhitānta* are words ending in *taddhita* affixes before getting sup inflectionṣ. *taddhita* affixes derive secondary nouns and change their meanings in various ways for example - *dāśarathī, gauṇ* etc. Pāṇini described many *taddhita* suffixes. Some suffixes are- *a, akañc, ac, añ, aṇ, at, iṣṭhan, īyasun, kan, ḍhak, ḍhañ, tamap, tarap, tayap, tal, tyap, tral, dvayasac, fak, matup, mātrac, yak, yat, yañ, ḍāc, kha, gha, cha, uraca, ṭhak, ṭhañ, ṭhan, na, ha, va, vatup* etc. For example - *dākṣī, kva, aśvakaḥ, viśvajanīnam, kṣatriyaḥ, mālīyaḥ, raivatikaḥ,* *daṇḍikaḥ, laghutamaḥ, gurutaraḥ, gārgyāyaṇaḥ, iha, balavān* etc. *taddhitānta* forms inflected with *sup* are called *taddhita subanta*

1.5.1.3 strīpratyayānta

Sanskrit has eight feminine suffixes *ṭāp, cāp ḍāp, ṅīṣ, ṅīn ṅīp, uṅ* and *ti* and words ending in these suffixes are called *strīpratyayānta.* For example - *ajā, gaurī, mūṣikā, indrāṇī, gopī, aṣṭādhyāyī, kurucarī, yuvatī, karabhorū* etc. *strīpratyayānta* forms inflected with *sup* are called *strīpratyayānta-subanta.*

1.5.1.4 samāsānta

Simple words (*padas*), whether substantives, adjectives, verbs or indeclinable, added with another *subanta-padas* are called *samāsa* (compound). Sanskrit *samāsas* are divided into four categories, some of which are in turn divided into sub-categories. The four main categories of compounds are as follows: (1) adverbial or *avyayībhāva,* (2) determinative or *tatpuruṣa,* (3) attributive or *bahuvrīhi* and (4) copulative or *dvandva. dvandva* and *tatpuruṣa* compounds can further be subdivided into sub-categories.[49]

For any natural language processing system for Sanskrit one has to process the derived nouns but it is difficult to identify the derived nouns in a sentence. Typically the identification of derived nouns will start after the *prātipadika* has been separated from the *sup* suffixes. Thereafter the *prātipadika* has to be identified as one of the four types- *kṛdanta, taddhitānta, strīpratyayānta* and *samāsānta.* Without doing it we can not meaningfully process or understand Sanskrit text for example-

(1) *Jagadguruḥ, Gopālakaḥ*

After Subanta processing it can be rewritten as-

(2) *Jagadguru + su [pum; pra; ek], Gopalaka + su [pum; pra; ek]*

Now the problem is to interpret the derived nouns in (2) as *kṛdanta, taddhitānta, strīpratyayānta* and *samāsānta.*

[49] "Machine Recognition and Morphological Analysis of Subanta-Padas" (M.Phil. Dissertation, Subash, J.N.U., 2006)

How many types of derived nouns can be formed we can see from following diagram-

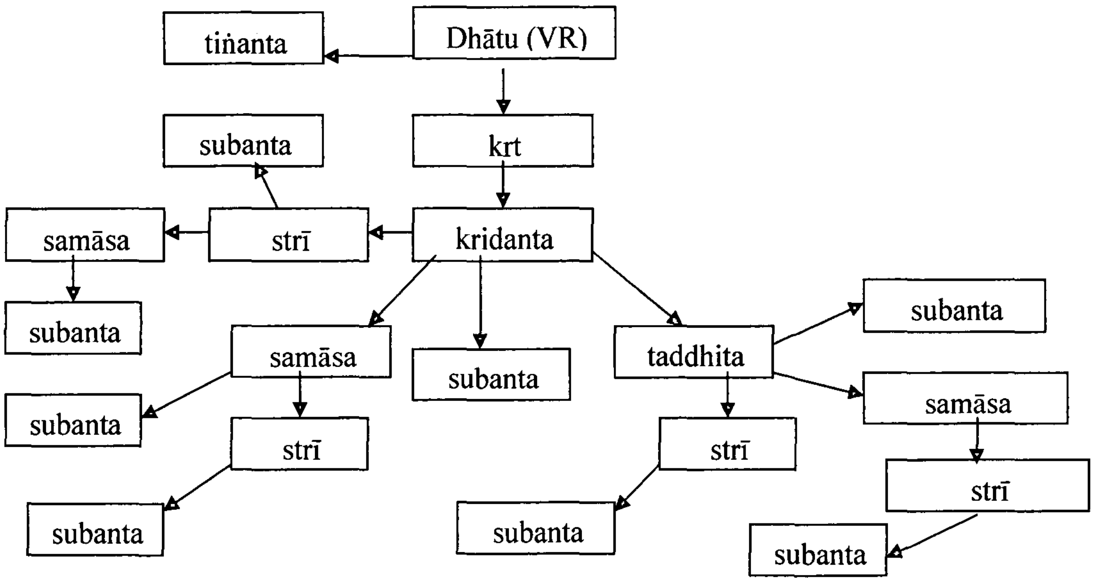


Diagram 1.2: Nominal derivation from verb root

Here we can see at least eight types of derived nouns are formed only with the verb root besides these some are derived from nominal bases.

1.5.1.5 Importance of derivational Morphology

- A new word with a new meaning is derived.
- Derivational morphemes have clear semantic content.
- When a derivational morpheme is added to a root or stem, it ads meaning.
- Semantic interpretation of the derived word is often difficult while a derivational suffix can usually be given a unique semantic meaning, may of the derived words, and may still resist compositional interpretation.
- Derivational Morphology is very useful for guessing meaning and part of speech of unknown words.
- In the context of indo-Aryan languages grapheme to phoneme mapping requires derivational morphological analysis.
- Computational modelling of derivational morphology helps us gauge the computational nature of word production

1.6 Need for the Kṛdanta analysis

Kṛdanta analyzer will be a very important tool in any Machine Translation or analysis system that attempts to analyze and understand Sanskrit for computational purposes. A good Sanskrit analyzer should have tagged lexicon, POS Tagger, *kāraka* analyzer, syntactical analyzer, *subanta* analyzer, *kṛdanta* analyzer, *taddhita* analyzer, *tiṅanta* analyzer, Gender analyzer, *sandhi* analyzer, *samāsa* analyzer etc. kṛdanta is primary derivative from verb root and it may be analyzed after *subanta, taddhita* and *samasa* are analyzed because the kṛt suffix may not be visible at the end of word till other later suffixes are separated.

Knowledge of kṛdanta is important for Sanskrit reading and speaking. Maximum part of Hitopadeśa, Pañcatantra and Sanskrit dramas and stories is full of kṛdanta usage, e.g. *saḥ gataḥ, saḥ gatavān, tena dṛṣṭam, saḥ dṛṣṭavān* in the place of *so'gamat, so'paśyat.* In the later period tendency of using kṛdanta in the place of tiṅanta has been increased. In seventh Wilson lecture, R.G. Bhandarkar has traced out the changing tendency of using kṛdanta in place of tiṅanta from Pāṇini to Patañjali. From *Mahābhāṣya Paspaśāhnika* he quotes a discussion about unused words. In the place of *ūṣa, tera, cakra, peca* currently *uṣita, tīrṇa, kṛtavat, pakvat* were being used. First ones are the forms of *vasa, tṝ, ḍukṛñ, ḍupacaṣ* respectively in *liṭ lakāra* and later ones are the forms of the same *dhātus* with *kta* and *ktavatu kṛt* suffixes.[50] Use of kṛdanta is more frequent in spoken Sanskrit than written Sanskrit. This tendency has also been seen in the other languages of Sanskrit family; and in Hindi (standard) almost verb forms are constituted by kṛdanta and the gender of verb is according to subject, or object in case of passive voice. Thus the structure of Hindi is nearer to Sanskrit kṛdanta than tiṅanta. Usage of kṛdanta is much because it is the first derivation and derivation of taddhita also depends upon it. Taddhita literally means which serves kṛdanta. Analysis of kṛdanta is also important for semantic interpretation. The meaning of kṛt is not only subject object or verb (*kartā, karma* or *bhāva*), but specific relation of verb root and the derived noun. e.g. *tṛc, lyuṭ, ṇvul* suffixes denote the agent and *lyuṭ* and *ṇvul* suffixes denote the instrument, *tumun* denotes the purpose of verb and *ktvā* suffix denote previous action and *kta, ktavatu* suffixes denote the agent as well as action of past perfect.

[50] Staal, J.F., A Reader on Sanskrit Gramarrian, p.p. 90-91

1.7 Research Methodology

For this R&D, the methodology of computational Sanskrit and software engineering has been used. This R&D is based on a Pāṇinian rule- based approach with necessary lexical interfacing).

The following steps have been followed –

- Building lexical resources for kṛdanta
  - Collecting primary texts of grammar, critical editions, translations etc for building kṛdatna lexical resources.
  - Prepared a rule base for kṛdanta analysis system according to Paṇinian formalism.
  - Adapted Monier Williams Sanskrit Digital Dictionary (MWSDD) by Louis Bontes with kṛdanta words tagged with kṛdanta information.
  - Annotated corpus of current Sanskrit prose with labelled kṛdanta information was created.
  - An avyaya database was created.
  - A lexical kṛdanta database was obtained from Amba Kulkarni and was cleaned, normalized and updated.
  - Prepared a kṛt modification table with basic affixes and their modified forms e.g.

    a. ktvā > [tvā]

    b. lyap > [ya]

    c. tumun > [tum/ṭum/ḍhum]

    d. ktavatu > [tavat]
  - prepared a verb modification table with dhātus and their kṛt modifications e.g.

    a. kṛ > [kṛ/kar/kār/kur]
  - A new categorization of verbs and suffixes has been made according to their modification behaviour.
  - Modification of verbs has gone through a process in which they are given a unique identification number that will be used to bind compatible suffixes. This helps in checking irrelevant over generation.

    - Created a prefix modification table which stores all the 22 prefixes with their modifications and combinations.

- Building the Kṛdanta analyzer - java-servlet system
    - The pre-processor object first checks the integrity and consistency of the input.
    - POS Tagger (a separate program) called identifying verb forms, avyayas and punctuations which not be processed for krdanta. This component also identifies some of the explicit kṛdantas. The remaining words are subantas which are taken to the next stage of processing.

The nominal bases (prātipadikas) are checked for solutions in lexical kṛdanta database, Monier Williams Sanskrit Digital Dictionary (MWSDD) and the kṛdanta-tagged corpus. If it is found in these three resources, it is recognized as kṛdanta as well as analyzed. If not found in the database and corpus, it will be identified and analyzed through the rule based mechanism.

- Potential complex forms are checked for prefixes (upasargas) If found, the nominal base without upasarga will again be sent to lexical databases and rule base for identification and analysis.
- Results are displayed in the dhātu- suffix pairs according to probability.
- If no analysis is found or system fails, the nominal base will be returned without analysis.

- **System evaluation**

The results have been evaluated manually against the human annotated corpus, and performance has been tabulated in the fourth chapter.

<u>Chapter – II</u>

Kṛdanta in Sanskrit Grammar

Chapter-2

Kṛdanta in Sanskrit Grammar

2.1 Overview of Paṇinian System

Pāṇini's grammar AD (approximately 7th BCE) is important for linguistic computation for two reasons. One, it provides a comprehensive and rule based account of a natural language in about 4000 rules - the only complete grammatical account of any language so far. Two, the model of a 'grammar-in-motion' that it provides seems to closely mimic a fully functional Natural Language Processing (NLP) system-

SOUND CLASSES (phonetic module)
|
RULE-BASE (parser/grammar module)
|
LEXICONS (lexical interface modules)

The possibility that a Natural Language (NL) parser based on Pāṇini can help analyze Indian languages has gained momentum in recent years.[1]

Aṣṭādhyāyī (7th BCE) is a composite text including the following components -

- *akṣara-samāmnāya* or *māheśvara-sūtra* (14) (AS)
- *śabdānuśāsana* or *sūtrapāṭha* (3965 or 3983 in *kāśikāvṛtti* (SP)
- *dhātupāṭha* (1967 verb roots - 2014 including *kaṇḍvādi* roots) (DP)
- *gaṇapāṭha* (other pertinent items like primitive nominal bases, *avyayas*) (GP)

The core of Pāṇinian grammar is a set of statements, each called *sūtra (*rule*)*. A *sūtra* is a statement in a formulaic form which is brief but unambiguous, concise but comprehensive, impersonal and objective.[2] These *sūtras* are of six types[3]: *samjña (*definitional rule), p*aribhāṣā* (metarule)*,* v*idhi* (operational rule), *niyama (*restriction rule*)* *atideśa* (extension rule) and *adhikāra* (heading rule).

---

[1] Jha, Girish Nath. '*The System of Panini*' http://www.languageinindia.com/feb2004/panini.html

[2] alpākṣaramasandhigdhaṃ sārvadviśvatomukham
astobhamanavadyaṃ ca sūtraṃ sūtravido viduḥ

[3] saṃjñā va paribhāṣā ca vidhirniyama eva ca
atideśo'dhikāraśca ṣadvidha sūtralakṣaṇam

Sūtrapāṭha **(SP)** is arranged in eight *adhyāyas* (chapters) each divided into four sub-chapters (*pādas*). The SP has approximately 3965 rules (*sūtras*) which have been arranged in x.x.x format (to be accessed in as *adhyāya . pāda . sūtra* format)[4]

The following is a summary of topics discussed in the Aṣṭādhyāyī[5] -

Chapter I

- Major definitional and interpretational rules
- Rules dealing with extension *(atideśa)*
- Rules dealing with *atmanepada-parasmaipada*
- Rules dealing with the *kāraka*

Chapter II

- Rules dealing with compounds (*samāsa*)
- Rules dealing with nominal inflection
- Rules dealing with number and gender of compounds
- Rules dealing with replacements relative to roots
- Rules dealing with deletion by *luk*

Chapter III

- Rules dealing with derivational of roots ending in affixes *san* etc.
- Rules dealing with the derivational of ending in a *kṛt*
- Rules dealing with the derivational of ending in a *tiṅ*

Chapter IV

- Rules dealing with derivation of a *pada* ending in a *sup*
- Rules dealing with feminine affixes
- Rules dealing with the derivational of nominal stems ending in an affix termed *taddhita*

[4] Jha Girish Nath 'The System of Panini' Language in India, volume 4:2 February 2004

[5] Sharma, Rama Nath, The Aṣṭādhyāyī of Pāṇini – Volume-I page-75-76

Chapter V, VI & VII

- Rules dealing with doubling
- Rules dealing with *samprasāraṇa*
- Rules dealing with the *saṁhitā*
- Rules dealing with the augment (*āgama*) *suṭ*
- Rules dealing with accents
- Rules dealing with phonological operations relatives to a pre-suffix base (*aṅga*)
- Rules dealing with operations relative to affixes augment etc.

Chapter VIII

- Rules dealing with doubling (*dvitva*) relative to a *pada*
- Rules dealing with accent relative to a *pada*
- Rules dealing with other phonological relatives to a *pada*
- Rules dealing with miscellaneous operations relative to a non-*pada*

Chapter	**Pāda I**	**Pāda II**	**Pāda III**	**Pāda IV**	**Total Rules**
1st	74	73	93	109	349
2nd	71	38	73	85	267
3rd	150	188	176	117	631
4th	176	144	166	144	630
5th	135	140	119	160	554
6th	217	198	138	175	728
7th	103	118	119	97	437
8th	74	108	119	68	369
Total Rules in Aṣṭādhyāyī					**3965**

Table 2.1: Distribution of AD sūtras

Kapoor(1992) has reduced the treatment of subject matter into four divisions[6]: Chapters 1-2 dealing with classification and enumeration of bases and categories, Chapters 3-5

[6] Kapoor, Kapil, "Text Interpretation: The Indian Tradition"

consist of *prakṛti-pratyaya* enumeration, and derivation of bases, Chapters 6-8.1 deal with the synthesis of *prakṛti-pratyaya,* and Chapters 8.2-8.4 deal with the rules of morphophonemics.

2.1.1 Śiva sūtras or pratyāhāra sūtra (PS)

Śiva sūtras or *pratyāhāra sūtra* is a set of 14 *sūtras.* Pāṇini uses these *sūtras* to generate *pratyāhāras* (abbreviatory terms). The use of these *pratyāhāras* is to build phoneme-cluster which he uses to economically specify in the domain of application of various rules. These 14 *sūtras* are-

a i u Ṇ	[अ इ उ ण्]
ṛ ḷ K	[ऋ ऌ क्]
e o Ṅ	[ए ओ ङ्,]
ai au C	[ऐ औ च्]
h y v r Ṭ	[ह य व र ट्]
la Ṇ	[ल ण्]
ñ m ṅ ṇ n Ṃ	[ञ म ङ ण न म्]
jh bh Ñ	[झ भ ञ्]
gh ḍh dh ṣ	[घ ढ ध ष्]
j b g ḍ d ś	[ज ब ग ड द श्]
kh ph ch ṭh th c ṭ t V	[ख फ छ ठ थ च ट त व्]
k p Y	[क प य्,]
ś ṣ s R	[श ष स र्,]
h L.	[ह ल्]

Of the hundreds of *pratyāhāras* that could in principle be formed from these *sūtras,* Pāṅini has used 43 (of a 44th introduced by later grammarians, *rañ*=(*r,l*)). Some *prtyāhāras* are ambiguous. For example, *ṇ* occurs twice in the list, which means that two different meanings can be assigned to *pratyāhāra* ***aṇ*** (including or excluding *ṛ* etc.)

2.1.2 Dhātupāṭha (1967 verb roots - 2014 including *kaṇḍvādi* roots) (DP).

The *dhātupāṭha* is a lexicon of Sanskrit verb roots assumed or explicitly called by the SP component There are 1967 verb roots, 2014 including *kaṇḍvādi* roots in Pāṅini *dhātupāṭha.* It is organized into ten classes as follows –

Sr.	*Class*	*Total roots*	*Modification*
1	bhvādi	1035	śap
2	adādi	71	Luk
3	juhotyādi	24	śu
4	divādi	141	śyan
5	svādi	34	śnu
6	tudādi	155	śa
7	rudhādi	25	śnam
8	tanādi	10	U
9	kryādi	62	śnā
10	curādi	410	śap
Total	**10**	**1967**	**10**

Table 2.2: Distribution of DP

2.1.3 Gaṇapāṭha (GP)

GP is a group-wise list of primitive nominal bases, each group (known as *gaṇa*) serving a common function in the description. For example: *sarvādi, ajādi, śaradādi* etc. The various classes like *kṛt, taddhita, strī, sup, tiṅ* and the 18 *upasargas* operate on these bases (including 23 pronouns)[7].

2.2 Kṛdanta in Sanskrit Grammar

[7] Jha, Girish Nath, 'The System of Panini' Language in india http://www.languageinindia.com/feb2004/panini.html

The *adhikāra* of sūtra *'pratyayaḥ' [3.1.1]* is carried to the sūtra *'niṣpravāṇiśca' [5.4.160]*. Thus the *adhikāra* of *pratyayaḥ* is in third, fourth and fifth adhyāya of *aṣṭādhyāyi*. The suffixes said in these adhyāya are of two kinds, the suffixes which are added to verb root and the suffixes which are added to nominal base.The suffix means, which being added after verb root or nominal base, extend their meaning, e.g., *kṛ* (root) means 'to do' and *kṛ* + *tṛ* = *kartā* means 'doer'

2.2.1 Suffixes joined to dhātus

The suffixes which are added to verb roots are in the third chapther of aṣṭādhyāyi. They are of four kinds-

dhātu-pratyaya (*san, kyac, kāmyac, kyaṣ* etc. 12 suffixes which are described in sūtras 3.1..5 to 3.1.32).

vikaraṇa-pratyaya (The suffix which comes in between verb root and *kṛt* or *tiṅ* suffix. These suffixes are described in sūtras 3.1.33 to 3.1.90).

tiṅ-pratyaya (The suffixes in the place of *laṭ, liṭ, luṭ* etc. *lakāras* These suffixes are described in sūtras 3.1.91 to 3.4.116, these are inflectional suffixes).

kṛt-pratyaya (*tavyat, tavya, anīyar* etc. suffixes. *tiṅ* and *kṛt* suffixes are described from *dhātoḥ(3.1.91)* to *chandasyubhayathā(3.4.117)*. The suffixes which are not *tiṅ* are known as kṛt suffixes by the sūtra *kṛdatiṅ(3.1.93)*. When the word is formed by adding kṛt suffix to verb root, it is called *kṛdanta* and by *kṛttaddhitasamāsāśca* it is declared *prātipadika*, by adding '*su*' etc. suffixes, a *subanta* form is formed. By *'suptiṅantam padam'* it becomes *pada*. To form a *kṛdanta*, we have to know with which verb root, in which meaning, which suffix is congugated and how and according to which sūtra. *Yāska* has described four types of *padas- nāma, ākhyāta, upasargas* and *nipāta*. The *kṛdanta* comes under *nāma pada*.

2.2.2 An overview of kṛdanta system

Words derived from Sanskrit verbal roots are classified into two groups: -Finite verbs, Word other than finite verbs. The difference of formation can be achieved by adding different suffixes to the verb root. Finite verbs are formed by adding *tiṅ* suffixes while all other words are derived by adding *kṛt* suffixes to the verb root. All the verbal suffixes besides *tiṅ* are called *kṛt*. *kṛt* is a technical term of *Pāṇinian* grammar that covers a vast field, both structurally as well as semantically.[8] The primary nominal derivatives from the verb roots are *kṛdanta*. The *kṛt* suffixes are added to roots or their modified forms, to form nouns, adjectives and indeclinables; e.g. *kṛ* - *kāra, kṛtṛ, karaṇa, kurvat, karuṣyat, cakṛvas, kṛtvā, kartum*. These are called *kṛdantas* or primary nominal bases.[9] The main difference between *kṛt* and *tiṅ* is that *kṛdanta* are nouns, adjectives and indeclinables not verb, but *tiṅanta* are always verb. The main difference between *kṛt* and *taddhita* is that *taddhita* comes always after noun, adjective, indeclinable or verb to form a new noun, adjective, indeclinable or verb while *kṛt* suffix comes only after verb root. The *kṛdantas* which are noun or adjective have various word forms and those which are indeclinable are unchanged. For example, *gam* + *tṛc* = *gantṛ* has various forms while *gam* + *ktvā* = *gatvā* is unchangeable. There is a peculiar class of *kṛt* or primary affixes technically designated by Sanskrit grammarians as *Uṇādi* or those beginning with the affix *uṇa* i.e. the affix *'u'* with the mute or indicatory letter *'ṇ'*, so called from the words *kāru, vāru* etc. These *uṇādi* affixes form primary nouns, like *kṛt* affixes, form verbal roots, but are classed separately because their application is limited, and because the nouns derived by their means are either formed irregularly, or the connection between their senses and the meanings of roots from which they are supposed to be derived is not so clearly discernible as in the case of other primary derivatives.[10]

Pāṇini gave a negative definition of *kṛt* by saying *kṛdatiṅ*[11], *'asmin dhātvadhikāre tiṅvarjitaḥ pratyayayaḥ kṛt saṃjño bhavati'* i.e. all that has been said under the adhikāra of *dhātoḥ*[12] besides *tiṅ* is *kṛt*. Two important features of *kṛt* which emerge from the definition are-

- kṛt suffixes are added to verbal roots
- All suffixes added to verbal roots with the exception of tiṅ are kṛt

[8] Sharma, Dipti, Structure and Meaning
[9] Kale, M.R., A Higher Sanskrit Grammar
[10] Kale, M.R., A Higher Sanskrit Grammar
[11] Pāṇini Aṣṭādhyāyī; 3.1.93
[12] Pāṇini Aṣṭādhyāyī; 3.1.91

Words formed by adding *kṛt* suffixes are known as *kṛdanta* and words formed by adding *tiṅ* suffixes are known as *tiṅanta*. But both are not identical in nature. Whereas *tiṅ* suffixes are those that denote syntactical relationship, kṛt suffixes are only word-forming suffixes. According to Pāṇini's definition a *prātipadika* (nominal stem) can be of two types, underived and derived. The derived *prātipadika* can be so formed by adding *kṛt* suffixes to verbal roots and *taddhita* suffixes to nominals. They can also be obtained by compounding two or more words. The two sūtras of Pāṇini *'arthavadadhāturapratyayaḥ prātipadikam'* and *'kṛttaddhitasamāsāśca'* lead us to the conclusion that though root and suffix are independent meaningful units under certain circumstances, when they are joined they merge their separate identity and form a meaningful unit.[13]

kṛt suffixes are of three kinds according to the *Siddhāntakaumudī* (SK) by *Bhaṭṭojidīkṣita-*

2.2.2.1 Kṛtya

From the sūtra *kṛtyāḥ*[14] to the sūtra *ṇvultṛcau*[15] all the suffixes treated will get the name of *kṛtya*. The term *kṛtya*, though constituting a subdivision of *kṛt* is very significant. It is used for the following suffixes *tavyat, tavya, anīyar, yat, ṇyat, kyap* and *kelimar*. For example- *paṭhitavyam, pātavya, paṭhanīya, pacelima, jeyam, deyam,* etc. Afterwards *Kātyāyana* has added *kelimar* in the the list of kṛtya suffixes. *kṛtya* suffixes are always used in *bhāva-vācya* and *karma-vācya* in neuter singular. *kṛtya* are also used as a noun or an adjective as opposed to other suffixes of this class, which are prescribed in specific tenses. The word formed by *kṛtya* suffixes may be called Passive Potential Participles or Future Passive Participles.

2.2.2.2 Pūrvakṛdanta suffixes

[13] Sharma, Dipti, Structure and Meaning
[14] Pāṇini Aṣṭādhyāyī; 3.1.95
[15] Pāṇini Aṣṭādhyāyī; 3.1.113

From the sūtra *ṇvultṛcau* to the sūtra *tumunṇvulau kriyāyāṃ kriyārthāyām*[16] all the suffixes treated, will get the name of *Pūrvakṛdanta.* These suffixes are used in *kartṛvācya* only. These suffixes are *ṇvul, tṛc, lyu, ṇini, ac, ka, aṇ, ṭa, khaś, khac, kvanip, ḍa, kta, ktavatu, śatṛ, śānac, śākan, u, kvip, itra, etc.* For example *karttā, kumbhakāraḥ, janamejayaḥ, pāṭhakaḥ, paṭhantī* etc.

2.2.2.3 Uttarakṛdanta suffixes

From the sūtra *tumunṇvulau kriyāyāṃ kriyārthāyām* to the sūtra *anvacyānulomye*[17] all the suffixes treated will get the name of *uttarakṛdanta.* These suffixes are *tumun, ghañ, erac, ap, ktṛ, athuc, nañ, nan, ktin, khal, yuc, ktvā, lyap, ṇamula, uṇa etc.* For example, *gantum, khāditum, svapnam, gatiḥ, gatvā, vihāya, ādāya* etc.

2.2.3 Classification of kṛt suffixes

Within the purview of *kṛt* there are a number of subordinate terms such as, *kṛtya, niṣṭha, sat, sārvadhātuka, Ārdhadhātuka*, etc. The following is the list of kṛt suffixes. In this list those kṛt suffixes are also included which have been grouped under various subheads.

Kṛtya- From the sūtra *kṛtyāḥ*[18] to the sūtra *ṇvultṛcau*[19] all the suffixes treated will get the name of *kṛtya*. These suffixes are seven in number- *tavyat, tavya, anīyar, yat, ṇyat, kyap* and *kelimar.*

Sārvadhātuka[20]- suffixes denoted by the abbreviatory term *tiṅ* (3.4.78) and suffixes marked with *ś* as an it, are termed *sārvadhātuka.* These suffixes are nine in number- *śatṛ, khaś, śānan, eśa, śadhyain, śānac, cānaś, śa, śadhyai.*

Ārdhadhātuka[21]- The suffixes other than *tiṅ* and those with an indicatory *ś* subjoined to a verbal root, are called *Ārdhadhātuka.* These suffixes are- *ṇvul, ṇvuc, ṇa, aṇ, ṇvi, ṇvin,*

[16] Pāṇini Aṣṭādhyāyī; 3.3.10
[17] Pāṇini Aṣṭādhyāyī; 3.4.64
[18] Pāṇini Aṣṭādhyāyī; 3.1.95
[19] Pāṇini Aṣṭādhyāyī; 3.1.113
[20] Pāṇini Aṣṭādhyāyī; 3.4.113

ghinuṇ, uṇ, inuṇ, vuñ, ṇini, ṇyuṭ, khukañ, ñyuṭ, ghañ, ukañ, ṇac, khamuñ, ṇyat, iñ, ka, kvin, vic, kañ, kvanip, ktavatu, kānac, gasnu, kmarac, kvarap, ki, kru, klukan, naṅ, aṅ, kse, kadhyai, taveṅ, kasun, Kenya, kelimar, tavyat, yat, lyu, ṣvun, vun, ṇmul, kyap, ṭak, viṭ, kvip, kap, kta, ḍvanip, kvasu, knu, kurac, kin, najiṅ, krukan, ktri, ktin, ktic, kasen, kadhyain, kamul, ken, ktvā, tavya, anīyar, tṛc, ath, thakan, ṭa, in, ḍa, khiṣṇuc, manin, ini, tṛn, yuc, āluc, ghurac, ūka, āru, ḍu, itra, ap, nan, ani, gha, se, ase, adhyai, tavai, tosun, khac, ḍar, atṛn, vanip, khyun, iṣṇuc, ṣākan, ru, u, ra, varac, ṣṭran, tumun, athuc, a, lyuṭ, khal, sen, asen, adhyai, taven, tvan.

Of these kṛt suffixes twenty-six, i.e. *se, sen, ase, asen, kse, kasen, adhyai, adhyain, kadhyai, kadhyain, śadhyai, śadhyain, tavai, taveṅ, taven, tosun, kasun, ṇmul, kamul, ken, kenya, tvan, ki, kin, kvasu* and *kānac* are used only in Vedic and not in the classical language. *se, sen, ase, asen, kse, kasen, adhyai, adhyain, kadhyai, kadhyain, śadhyai, śadhyain, tavai, taveṅ* and *taven* are used as an alternative of *tumun* while *tosun, kasun, ṇmul* and *kamul* are the alternatives of *ktvā, ken, kenya* and *tvan* are used in place of *kṛtya* suffixes. *tavai* which is used as an alternative form of *tumun* is also used in place of *kṛtyas*. Similarly, *tosun* and *kasun* which are alternative forms of *ktvā* are also used to denote the name of an action only. *Kvasu and kānac* are used as participles of the perfect.[22]

2.3 Kṛdanta Tags

R. Chandrashekhar, as part of his Ph.D. thesis (Part-of-speech tagging for Sanskrit) designed tagset for *sandhi*-free classical Sanskrit prose text. The designed tagset is classified according to the morphological structure of the categoris. There are two kinds of tags in this tagset. Word class main tags and feature sub-tags. This Sanskrit tagset contains 65 word class tags, 43 feature sub-tags and 25 punctuation tags and one tag UN to tag unknown words, totally 134 tags. A single tag is a combination of word class tag

[21] Pāṇini Aṣṭādhyāyī; 3.4.114
[22] Sharma, Dipti, Structure and Meaning

and feature sub-tags. The participle tags or the kṛdanta tags are representing the tense and mood information are 9 in number in this tagset.[23]

KV1	Kṛdanta Vartamāna 1 (śatṛ, with gender and declensional sub-tags) (e.g. *kurvan, gacchat*) Past Active Participle
KV2	Kṛdanta Vartamāna 2 (śānac, śatṛ, with gender and declensional sub-tags) (e.g. *labhamānah, vardhamānam*) Past Middle/Active Participle
KB1	Kṛdanta Bhūta 1 (kta with gender and declensional sub-tags) (e.g. *dṛṣṭaḥ, gatam*) Past Passive Participle
KB2	Kṛdanta Bhūta 2 (ktavat, with gender and declensional sub-tags) (e.g. *uktavat, dṛṣṭavān*) Past Active Participle
KAa	Kṛdanta Agami a (sya-śatṛ, with gender and declensional sub-tags) (e.g. *kariṣyat*) Future Active Participle
KAb	Kṛdanta Agami b (sya-śānac, with gender and declensional sub-tags) (e.g.*kariśyamāna*) Future Passive Participle
KVI1	Kṛdanta VIdhyarthaka 1 (-ya, with gender and declensional sub-tags) (e.g. *kārya*) Gerundive
KVI2	Kṛdanta VIdhyarthaka 2 (-tavya, with gender and declensional sub-tags) (e.g. *kartavya*) Gerundive
KVI3	Kṛdanta VIdhyarthaka 3 (-aniya, with gender and declensional sub-tags) (e.g. *karaṇīya*) Gerundive

Table 2.3: kṛdanta tags

2.4 Kṛt suffixes table

In the following table 136 kṛt suffixes are listed with their meaning and example of their use.

[23]"Part-Of-Speech Tagging For Sanskrit" (Ph.D., Thesis, R. Chandrashekar, J.N.U., 2007)

Suffixes prescribed by Pāṇini	**Sūtra No.**	**Suffixes in efficient form**	**Meaning**	**Example**
क्विन् (kvin)	3.1.58-60	Null	कर्तरि (kartari)	मन्त्रस्पृक् (mantrasprk)
क्विप् (kvip)	3.2.61, 3.2.87-92, 3.2.76, 3.2.177-179, 3.3.94 (Vārtika)	Null	कर्तरि, भूते-कर्तरि, भावे, अकर्तरि च कारके संज्ञायाम् (kartari, bhūte-kartari, bhāve, akartari ca kārake saṁjñāyām)	उपनिषद्, ब्रह्महा, विभ्राट्, सम्पत् (upaniṣad, brahmahā, vibhrāṭ, sampat)
विट् (viṭ)	3.2.67-69	Null	कर्तरि (kartari)	अब्जाः (abjāḥ)
ण्विन् (ṇvin)	3.2.71-72	Null	कर्तरि (kartari)	श्वेतवाः (śvetavāḥ)
ण्वि (ṇvi)	3.2.62-64	Null	कर्तरि (kartari)	अंशभाक् (aṁśabhāk)
विच् (vic)	3.2.73	Null	कर्तरि (kartari)	उपयट् (upayaṭ)
अच् (ac)	3.1.134, 3.2.9-15, 3.3.56	अ (a)	कर्तरि, भावे, अकर्तरि च कारके संज्ञायाम् (kartari, bhāve, akartari ca kārake saṁjñāyām)	पचः, चयः (pacaḥ, cayaḥ)
अण् (aṇ)	3.2.1-2, 3.3.23, 3.3.12	अ (a)	कर्तरि, भविष्यति-क्रियार्थायां क्रियायाम् (kartari, bhaviṣyati-kriyārthāyāṁ kriyāyām)	कुम्भकारः, काण्डलावः (kumbhakāra ḥ, kāṇḍalāvaḥ)

अप् (ap)	3.3.50-55, 3.3.57-87	अ (a)	भावे, अकर्तरि च कारके संज्ञायाम् (bhāve, akartari ca kārake saṁjñāyām)	आरवः, वधः (āravaḥ, vadhaḥ)
क (ka)	3.1.135-36, 3.1.144, 3.2.3-7, 3.2.55, 3.2.77	अ (a)	कर्तरि (kartari)	क्षिपः (kṣipaḥ)
कञ् (kañ)	3.2.60	अ (a)	कर्तरि (kartari)	तादृशः (tādṛśaḥ)
खच् (khac)	3.2.38-47	अ (a)	कर्तरि (kartari)	प्रियंवदः (priyaṁvada ḥ)
खल् (khal)	3.3.126-127	अ (a)	भावे, कर्मणि च, कृच्छ्र-अकृच्छ्रार्थेषु (bhāve, karmaṇi ca, kṛcchra-akṛcchrārtheṣu)	ईषद्भोजः, दुष्करः (īṣadbhojaḥ, duṣkaraḥ)
खश् (khaś)	3.2.28-37, 3.2.83	अ (a)	कर्तरि (kartari)	जनमेजयः (janamejayaḥ)
घ (gha)	3.3.118-119, 3.3.125	अ (a)	करणे, अधिकरणे च संज्ञायाम् (karaṇe, adhikaraṇe ca saṁjñāyām)	दन्तच्छदः, आखनः (dantacchada ḥ, ākhanaḥ)
घञ् (ghañ)	3.3.16-42, 3.3.45-55, 3.3.62-65 3.3.120-125	अ (a)	भावे, अकर्तरि च कारके संज्ञायाम् करणाधिकरणयोः (bhāve, akartari ca kārake saṁjñāyām karaṇādhikaraṇayoḥ)	त्यागः, रङ्गः, अवतारः (tyāhaḥ, raṅgaḥ, avatāraḥ)
ट (ṭa)	3.2.16-22	अ (a)	कर्तरि (kartari)	कुरुचरः (kurucaraḥ)
टक् (ṭak)	3.2.8, 3.2.52-55	अ (a)	कर्तरि (kartari)	सुरापः (surāpaḥ)

ड (ḍa)	3.2.48-50, 3.2.97-101, 3.3.125 (Vārtika)	अ (a)	कर्तरि, भूते-कर्तरि, करणाधिकरणयोः (kartari, bhūte-kartari, karaṇādhikaraṇayoḥ)	अन्तगः, सरसिजम्, आखः (antahaḥ, sarasijam, ākhaḥ)
णच् (ṇac)	3.3.43	अ (a)	भावे, अकर्तरि च कारके संज्ञायाम् (bhāve, akartari ca kārake saṁjñāyām)	व्यावक्रोशी (vyāvakrośī)
श (śa)	3.1.137-139, 3.3.100-101	अ (a)	कर्तरि, भावे, अकर्तरि च कारके संज्ञायाम् (kartari, bhāve, akartari ca kārake saṁjñāyām)	पिबः, क्रिया (pibaḥ, kriyā)
ण (ṇa)	3.1.140-143	अ (a)	कर्तरि (kartari)	ज्वालः (jvālaḥ)
अङ् (aṅ)	3.3.104-106	अ (a)	भावे, अकर्तरि च कारके संज्ञायाम् (bhāve, akartari ca kārake saṁjñāyām)	जरा (jarā)
अ (a)	3.3.102-103	अ (a)	भावे, अकर्तरि च कारके संज्ञायाम् (bhāve, akartari ca kārake saṁjñāyām)	चिकीर्षा (cikīrṣā)
ण्वुल् (ṇvul)	3.1.133, 3.3.10, 3.3.108-110	अक (aka)	कर्तरि, भविष्यति-क्रियार्थायां क्रियायाम्, भावे, अकर्तरि च कारके संज्ञायाम् (kartari, bhaviṣyati-kriyārthāyāṁ kriyāyām, bhāve, akartari ca kārake saṁjñāyām)	गायकः, प्रच्छर्दिका (gāyakaḥ, pracchardikā)
ण्वुच् (ṇvuc)	3.3.111	अक(aka)	भावे (bhāve)	शायिका (śāyikā)

वुञ् (vuñ)	3.2.146-147	अक(aka)	ताच्छीलिके-कर्तरि (tācchīlike-kartari)	निन्दकः (nindakaḥ)
वुन् (vun)	3.1.149-150	अक(aka)	कर्तरि (kartari)	जीवकः (jīvakaḥ)
ष्वुन् (ṣvun)	3.1.145	अक(aka)	कर्तरि (kartari)	नर्त्तकः (narttakaḥ)
ख्युन् (khyun)	3.2.56	अन (ana)	कर्तरि (kartari)	आढ्यङ्करणम् (āḍhyaṅkaraṇam)
ञ्युट् (ñyuṭ)	3.2.65-66	अन (ana)	कर्तरि (kartari)	हव्यवाहनः (havyavāhanaḥ)
ण्युट् (ṇyuṭ)	3.1.147-148	अन (ana)	कर्तरि (kartari)	गायनः (gāyanaḥ)
युच (yuc)	3.2.148-151, 3.3.107, 3.3.128-130	अन (ana)	ताच्छीलिके-कर्तरि, भावे, अकर्तरि च कारके संज्ञायाम्, कृच्छ्र-अकृच्छ्रार्थेषु (tācchīlike-kartari bhāve, akartari ca kārake saṁjñāyām kṛcchra-akṛcchrārtheṣu)	चलनः, कारणा, दुष्पानः (calanaḥ, kāraṇā, duṣpānaḥ)
ल्यु (lyu)	3.1.134	अन (ana)	कर्तरि (kartari)	नन्दनः (nandanaḥ)
ल्युट् (lyuṭ)	3.3.113, 3.3.115-117	अन (ana)	बहुल-अर्थे, नपुंसके, भावे, करणे, अधिकरणे (bahula-arthe, napuṁsake, bhāve, karaṇe, adhikaraṇe)	राजभोजनाः, हसनम्, इध्मप्रव्रश्चनः (rājabhojanāḥ, hasanam, idhmapravraścanaḥ)
अन (ana)	3.3.112	अन् (an)	भावे, अकर्तरि च कारके संज्ञायाम् (bhāve, akartari ca kārake saṁjñāyām)	अकरणिः (akaraṇiḥ)

कानच् (kānac)	3.2.106	आन (āna)	भूते-कर्तरि (bhūte-kartari)	चिक्यानः (cikyānaḥ)
चानश् (cānaś)	3.2.129	आन (āna)	वर्तमाने-कर्तरि (vartamāne-kartari)	भुञ्जानः (bhuñjānaḥ)
शानच् (śānac)	3.2.124-127, 3.3.14 (Vārtika)	आन (āna)	वर्तमाने-कर्तरि, अनद्यतने भविष्यति (vartamāne-kartari, anadyatane bhaviṣyati)	पचमानम्, पचिष्यमाणम् (pacamānam, paciṣyamāṇa m)
शानन् (śānan)	3.2.128	आन (āna)	वर्तमाने-कर्तरि (vartamāne-kartari)	पवमानः (pavamānaḥ)
इन् (in)	3.2.24-27	इ (i)	कर्तरि (kartari)	शकृत्करिः (śakṛtkariḥ)
इञ् (iñ)	3.3.110	इ (i)	भावे, अकर्तरि च कारके संज्ञायाम् (bhāve, akartari ca kārake saṁjñāyām)	कारिः (kāriḥ)
कि (ki)	3.2.171, 3.3.92-93	इ (i)	ताच्छीलिके-कर्तरि, भावे, अकर्तरि च कारके संज्ञायाम् कर्मण्यधिकरणे च (tācchīlike-kartari bhāve, akartari ca kārake saṁjñāyām, kṛcchra-akṛcchrārtheṣu, karmaṇyadhikaraṇe ca)	पपिः, उपाधिः, जलधिः (papiḥ, upādhiḥ, jaladhiḥ)
किन् (kin)	3.2.171	इ (i)	ताच्छीलिके-कर्तरि (tācchīlike-kartari)	ददिः (dadiḥ)
घिनुण् (ghinuṇ)	3.2.141-145	इन् (in)	ताच्छीलिके-कर्तरि (tācchīlike-kartari)	शमी (śamī)

इनुण् (inuṇ)	3.3.44	इन् (in)	भावे, अकर्तरि च कारके संज्ञायाम् (bhāve, akartari ca kārake saṁjñāyām)	सांराविणम् (sāṁrāviṇam)
इनि (ini)	3.2.93, 3.2.156-157	इन् (in)	भूते-कर्तरि, ताच्छीलिके-कर्तरि (bhūte-kartari, tācchīlike-kartari)	सोमविक्रयी, प्रजवी (somavikrayī, prajavī)
णिनि (ṇini)	3.1.134, 3.2.78-82, 3.2.51, 3.2.85-86, 3.3.170	इन् (in)	कर्तरि, भूते-कर्तरि, आवश्यके आधमर्ण्ये च (kartari, bhūte-kartari, āvaśyake ādhamarṇye ca)	ग्राही, सोमयाजी, अवश्यङ्कारी (grāhī, somayājī, avaśyaṅkārī)
इष्णुच् (iṣṇuc)	3.2.136-138	इष्णु (iṣṇu)	ताच्छीलिके-कर्तरि (tācchīlike-kartari)	अलङ्करिष्णुः (alaṅkariṣṇu)
खिष्णुच् (khiṣṇuc)	3.2.57	इष्णु (iṣṇu)	कर्तरि (kartari)	आढ्यम्भविष्णुः (āḍhyambhav iṣṇuḥ)
उकञ् (ukañ)	3.2.154	उक (uka)	ताच्छीलिके-कर्तरि (tācchīlike-kartari)	लाषुकः (lāṣukaḥ)
खुकञ् (khukañ)	3.2.57	उक (uka)	कर्तरि (kartari)	स्थूलम्भावुकः (sthūlambhāv ukaḥ)
कुरच्(kurac)	3.2.162	उर (ura)	ताच्छीलिके-कर्तरि (tācchīlike-kartari)	विदुरः (viduraḥ)
घुरच् (ghurac)	3.2.161	उर (ura)	ताच्छीलिके-कर्तरि (tācchīlike-kartari)	भासुरः (bhāsuraḥ)
तव्यत् (tavyat)	3.1.96	तव्य (tavya)	विधौ, भावे, कर्मणि, बहुल-अर्थे (vidhou, bhāve, karmaṇi, bahula-arthe)	गातव्यम् (gātavyam)

तव्य (tavya)	3.1.96	तव्य (tavya)	विधौ, भावे, कर्मणि, बहुल-अर्थे (vidhou, bhāve, karmaṇi, bahula-arthe)	आसितव्यम् (āsitavyam)
तृच् (tṛc)	3.1.133, 3.3.169	तृ (tṛ)	कर्तरि, अर्हे (kartari, arhe)	कर्ता, वोढा (kartā, voḍhā)
तृन् (tṛn)	3.2.135	तृ (tṛ)	ताच्छीलिके-कर्तरि (tācchīlike-kartari)	कर्ता (kartā)
क्त (kta)	3.2.102, 3.2.187-188, 3.3.114, 3.4.71-72	त (ta)	भूते-कर्मणि, वर्तमाने, संज्ञायाम्, आदिकर्मणि कर्तरि, भावे कर्मणि च, कर्तरि कर्मणि भावे च, अधिकरणे (bhūte-karmaṇi, vartamāne, saṁjñāyām, ādikarmaṇI kartari, bhāve karmaṇi ca, kartari karmaṇi bhāve ca, adhikaraṇe	पठितम्, मतः, देवदत्तः, प्रस्वेदितः, गतः आसितः (paṭhitam, mataḥ, devadattaḥ, prasveditaḥ, gataḥ)
क्तिन् (ktin)	3.3.94-97	ति (ti)	भावे, अकर्तरि च कारके संज्ञायाम् (bhāve, akartari ca kārake saṁjñāyām)	कृतिः (kṛtiḥ)
क्तिच् (ktic)	3.3.174	ति (ti)	संज्ञायाम् (saṁjñāyām)	भूतिः (bhūtiḥ)
क्त्रि (ktri)	3.3.88	त्रि (tri)	भावे, अकर्तरि च कारके संज्ञायाम् (bhāve, akartari ca kārake saṁjñāyām)	कृत्रिमम् (kṛtrimam)
नङ् (naṅ)	3.3.90	न (na)	भावे, अकर्तरि च कारके संज्ञायाम् (bhāve, akartari ca kārake saṁjñāyām)	यज्ञः (yajñāḥ)

नन् (nan)	3.3.91	न (na)	भावे, अकर्तरि च कारके संज्ञायाम् (bhāve, akartari ca kārake saṁjñāyām)	स्वप्नः (svapnaḥ)
क्नु (knu)	3.2.140	नु (nu)	ताच्छीलिके-कर्तरि (tācchīlike-kartari)	त्रस्नुः (tresnuḥ)
नजिङ् (najiṅ)	3.2.172	नज् (naj)	ताच्छीलिके-कर्तरि (tācchīlike-kartari)	स्वप्नक् (svapnak)
क्यप् (kyap)	3.1.106-121, 3.3.98-100	य (ya)	विधौ, भावे, अकर्तरि च कारके संज्ञायाम्, कर्म, बहुल (vidhou, bhāve, akartari ca kārake saṁjñāyām, karma, bahula)	खेयम्, व्रज्या, समज्या (kheyam, vrajyā, samajyā)
ल्यप् (lyap)	7.1.37	य (ya)	भावे (bhāve)	आगत्य (āgatya)
यत् (yat)	3.1.97-106	य (ya)	विधौ, भावे, कर्मणि, बहुल-अर्थे (vidhou, bhāve, karmaṇi, bahula-arthe)	गेयम् (geyam)
ण्यत् (ṇyat)	3.1.122-132	य (ya)	विधौ, भावे, कर्मणि, बहुल-अर्थे (vidhou, bhāve, karmaṇi, bahula-arthe)	कार्यम् (kāryam)
क्रु (kru)	3.2.174	रु (ru)	ताच्छीलिके-कर्तरि (tācchīlike-kartari)	भीरुः (bhīruḥ)
क्रुकन् (krukan)	3.2.174 (Vārtika)	रुक् (ruk)	ताच्छीलिके-कर्तरि (tācchīlike-kartari)	भीरुकः (bhīrukaḥ)
रु (ru)	3.2.159	रु (ru)	ताच्छीलिके-कर्तरि (tācchīlike-kartari)	दारुः (dāruḥ)
र (ra)	3.2.167	र (ra)	ताच्छीलिके-कर्तरि (tācchīlike-kartari)	नम्रः (namraḥ)

क्वनिप् (kvanip)	3.2.74-75, 3.2.94-96	वन् (van)	ताच्छीलिके-कर्तरि (tācchīlike-kartari)	सुधीवा, पारदृश्वा (sudhīvā, pāradṛśvā)
वनिप् (vanip)	3.2.74-75	वन् (van)	कर्तरि (kartari)	भूरिदावा (bhūridāvā)
ड्वनिप् (ḍvanip)	3.2.103	वन् (van)	भूते-कर्तरि (bhūte-kartari)	सुत्वा (sutvā)
क्वरप् (kvarap)	3.2.163-164	वर (vara)	ताच्छीलिके-कर्तरि (tācchīlike-kartari)	इत्वरः (itvaraḥ)
वरच् (varac)	3.2.175-76	वर (vara)	ताच्छीलिके-कर्तरि (tācchīlike-kartari)	ईश्वरः (īśvaraḥ)
शतृ (śatṛ)	3.2.124-127, 3.2.130-133, 3.3.14 (Vārtika)	अत् (at)	वर्तमाने-कर्तरि, अनद्यतने भविष्यति (vartamāne-kartari, anadyatane bhaviṣyati)	पचन्तम्, पचिष्यन्तम् (pacantam, paciṣyantam)
एश् (eś)	3.4.15	ए (e)	छन्दसि-कृत्यर्थे (विधौ) (chandasi-kṛtyarthe (vidhou))	अवचक्षे (avacakṣe)
केन् (ken)	3.4.14	ए (e)	छन्दसि-कृत्यर्थे (विधौ) (chandasi-kṛtyarthe (vidhou))	अवगाहे (avagāhe)
शध्यैन् (śadhyain)	3.4.9	अध्यै (adhyai)	छन्दसि-तुमर्थे (chandasi-tumarthe)	पिबध्यै (pibadhyai)
शध्यै (śadhyai)	3.4.9	अध्यै (adhyai)	छन्दसि-तुमर्थे (chandasi-tumarthe)	पिबध्यै (pibadhyai)
उण् (uṇ)	3.3.1	उ (u)	बहुल-अर्थे (bahula-arthe)	वायुः (vāyuḥ)

खमुञ् (khamuñ)	3.4.25	अम् (am)	समानकर्तकयोः पूर्वकाले (samānakartakayoḥ pūrvakāle)	चोरङ्कारम् (coraṅkāram)
णमुल् (ṇamul)	3.4.12, 3.4.22, 3.4.24, 3.4.26-64	अम् (am)	छन्दसि-तुमर्थे, आभीक्ष्ण्ये, समानकर्तकयोः पूर्वकाले, अयथाभिप्रेताख्याने (chandasi-tumarthe, ābhīkṣṇye, samānakartakayoḥ pūrvakāle, ayathābhipretākhyāne)	विभाजम्, स्मारं स्मारम्, स्वादुङ्कारम्, उच्चैः-कारम् (vibhājam, smāram, smāram, svāduṅkāram , uccaiḥ-kāram)
कमुल् (kamul)	3.4.12	अम् (am)	छन्दसि-तुमर्थे (chandasi-tumarthe)	अपलुपम् (apalupam)
क्तवतु (ktavatu)	3.2.102	तवत् (tavat)	भूते-कर्तरि (bhūte-kartari)	गतवान् (gatavān)
ग्स्नु (gasnu)	3.2.139	स्नु (snu)	कर्तरि (kartari)	जिष्णुः (jiṣṇuḥ)
क्मरच् (kmarac)	3.2.160	मर (mara)	ताच्छीलिके-कर्तरि (tācchīlike-kartari)	सृमरः (sṛmaraḥ)
क्लुकन् (klukan)	3.2.174	लुक (luka)	ताच्छीलिके-कर्तरि (tācchīlike-kartari)	भीलुकः (bhīlukaḥ)
क्से (kse)	3.4.9	से (se)	छन्दसि-तुमर्थे (chandasi-tumarthe)	प्रेषे (preṣe)
से (se)	3.4.9	से (se)	छन्दसि-तुमर्थे (chandasi-tumarthe)	वक्षे (vakṣe)

कध्यै (kadhyai)	3.4.9	अध्यै (adhyai)	छन्दसि–तुमर्थे (chandasi-tumarthe)	आहुवध्यै (āhuvadhyai)
कध्यैन् (kadhyain)	3.4.9	अध्यै (adhyai)	छन्दसि–तुमर्थे (chandasi-tumarthe)	श्रियध्यै (śriyadhyai)
तवेङ् (taveṅ)	3.4.9	तवे (tave)	छन्दसि–तुमर्थे (chandasi-tumarthe)	सूतवे (sūtave)
कसुन् (kasun)	3.4.13, 3.4.17	अस् (as)	छन्दसि–तुमर्थे (chandasi-tumarthe)	विलिखः (vilikhaḥ)
केन्य (kenya)	3.4.14	एन्य (enya)	छन्दसि–कृत्यर्थे (विधौ) (chandasi-kṛtyarthe (vidhou))	दिदृक्षेण्य (didṛkṣeṇya)
केलिमर् (kelimar)	3.1.96 (Vārtika)	एलिम (elima)	विधौ, भावे, कर्मणि, बहुल–अर्थे (vidhou, bhāve, karmaṇi, bahula-arthe)	पचेलिमाः (pacelimāḥ)
कप् (kap)	3.2.70	क (ka)	कर्तरि (kartari)	कामदुघा (kāmadughā)
क्वसु (kvasu)	3.2.107-109	वस् (vas)	भूते–कर्तरि (bhūte-kartari)	अघायुः (aghāyuḥ)
कसेन् (kasen)	3.4.9	असे (ase)	छन्दसि–तुमर्थे (chandasi-tumarthe)	श्रियसे (śriyase)
क्त्वा (ktvā)	3.4.18-22, 3.4.24, 3.4.59-64	त्वा (tvā)	प्रतिषेधे, व्यातीहारे, परावरयोगे, आभीक्ष्ण्ये, समानकर्तकयोः पूर्वकाले, अयथाभिप्रेताख्यान pratiṣedhe, vyātīhāre, parāvarayoge, ābhīkṣṇye, samānakartakayoḥ	अलं दत्त्वा, अपमित्य, अप्राप्य, भुक्त्वा, स्मृत्वा स्मृत्वा, उच्चैः कृत्वा (alam dattvā,

			pūrvakāle, ayathābhipretākhyāne)	apamitya, aprāpya, bhuktvā, smṛtvā smṛtvā, uccaiḥ kṛtvā)
अनीयर् (anīyar)	3.1.96	अनीय (anīya)	विधौ, भावे, कर्मणि, बहुल-अर्थे (vidhou, bhāve, karmaṇi, bahula-arthe)	करणीयम् (karaṇīyam)
थकन् (thakan)	3.1.146	थक (thaka)	कर्तरि (kartari)	गाथकः (gāthakaḥ)
मनिन् (manin)	3.2.74-75	मन् (man)	कर्तरि (kartari)	सुदामा (sudāmā)
आलुच्(āluc)	3.2.158	आलु (ālu)	ताच्छीलिके-कर्तरि (tācchīlike-kartari)	दयालुः (dayāluḥ)
ऊक (ūka)	3.2.165-166	ऊक (ūka)	ताच्छीलिके-कर्तरि (tācchīlike-kartari)	जागरूकः (jāgarūkaḥ)
आरु (āru)	3.2.173	आरु (āru)	ताच्छीलिके-कर्तरि (tācchīlike-kartari)	वन्दारुः (vandāruḥ)
डु (ḍu)	3.2.180	उ (u)	कर्तरि (kartari)	प्रभुः (prabhuḥ)
इत्र (itra)	3.2.184-186	इत्र (itra)	कर्तरि (kartari)	पवित्रम् (pavitram)
असे (ase)	3.4.9	असे (ase)	छन्दसि-तुमर्थे (chandasi-tumarthe)	जीवसे (jīvase)
अध्यै (adhyai)	3.4.9	अध्यै (adhyai)	छन्दसि-तुमर्थे (chandasi-tumarthe)	उपाचरध्यै (upācaradhya i)

तवै (tavai)	3.4.9, 3.4.14	तवै (tavai)	छन्दसि–तुमर्थे-कृत्यर्थे (विधौ) (chandasi-tumarthe-kṛtyarthe (vidhou))	परिधातवे (paridhātave)
तोसुन् (tosun)	3.4.13, 3.4.16	तोस् (tos)	छन्दसि–तुमर्थे (chandasi-tumarthe)	अभिचरितोः (abhicaritoḥ)
डर (ḍara)	3.3.125 (Vārtika)	अर (ara)	करणाधिकरणयोः (karaṇādhikaraṇayoḥ)	आखरः (ākharaḥ)
अतृन् (atṛn)	3.2.104	अत् (at)	भूते–कर्तरि (bhūte-kartari)	जरन् (jaran)
षाकन् (ṣākan)	3.2.155	आक (āka)	ताच्छीलिके–कर्तरि (tācchīlike-kartari)	जल्पाकः (jalpākaḥ)
उ (u)	3.2.168-170	उ (u)	ताच्छीलिके–कर्तरि (tācchīlike-kartari)	भिक्षुः (bhikṣuḥ)
तुमुन् (tumun)	3.3.10, 3.3.158, 3.3.167, 3.4.65-66	तुम् (tum)	भविष्यति–क्रियार्थायां क्रियायाम्, इच्छार्थेषु, शक्नोत्यादिष्वर्थेषु (bhaviṣyati-kriyārthāyāṁ kriyāyām, icchārtheṣu, śaknotyādiṣvartheṣu)	पातुम्, भोक्तुम्, शक्नोति भोक्तुम् (pātum, bhoktum, śaknoti bhoktum)
ष्ट्रन् (ṣṭran)	3.2.181-183	त्र (tra)	कर्तरि (kartari)	धात्री (dhātrī)
अथुच् (athuc)	3.3.89	अथु (athu)	भावे, अकर्तरि च कारके संज्ञायाम् (bhāve, akartari ca kārake saṁjñāyām)	वेपथुः (vepathuḥ)
सेन् (sen)	3.4.9	से (se)	छन्दसि–तुमर्थे (chandasi-tumarthe)	वक्षे (vakṣe)

असेन् (asen)	3.4.9	असे (ase)	छन्दसि-तुमर्थे (chandasi-tumarthe)	जीवसे (jīvase)
अध्यैन् (adhyain)	3.4.9	अध्यै (adhyai)	छन्दसि-तुमर्थे (chandasi-tumarthe)	उपाचरध्यै (upācaradhya i)
तवेन् (taven)	3.4.9	तवे (tave)	छन्दसि-तुमर्थे (chandasi-tumarthe)	गन्तवे (gantave)
त्वन् (tvan)	3.4.14	त्व (tva)	छन्दसि-कृत्यर्थे (विधौ) (chandasi-kṛtyarthe (vidhou))	कर्त्वम् (kartvam)
गस्नु (gasnu)	3.2.139	स्नु (snu)	ताच्छीलिके-कर्तरि (tācchīlike-kartari)	ग्लास्नुः (glāsnuḥ)
इक् (ik)	3.3.108 (Vārtika)	इ (i)	भावे, अकर्तरि च कारके संज्ञायाम् (bhāve, akartari ca kārake saṁjñāyām)	पचिः (paciḥ)
श्तिप् (śtip)	3.3.108 (Vārtika)	ति (ti)	भावे, अकर्तरि च कारके संज्ञायाम् (bhāve, akartari ca kārake saṁjñāyām)	पचतिः (pacatiḥ)
इक (ika)	3.3.125 (Vārtika)	इक (ika)	करणाधिकरणयोः (karaṇādhikaraṇayoḥ)	आखनिकः (ākhanikaḥ)
इकवक (ikavaka)	3.3.125 (Vārtika)	इकवक (ikavaka)	करणाधिकरणयोः (karaṇādhikaraṇayoḥ)	आखनिकवकः (ākhanikavak aḥ)
कै (kai)	3.4.10	ऐ (ai)	छन्दसि-तुमर्थे (chandasi-tumarthe)	प्रयै (prayai)
इष्यै (iṣyai)	3.4.10	इष्यै (iṣyai)	छन्दसि-तुमर्थे (chandasi-tumarthe)	रोहिष्यै (rohiṣyai)

के (ke)	3.4.11	ए (e)	छन्दसि-तुमर्थे (chandasi-tumarthe)	दृशे (dṛśe)

Table 2.4: kṛt suffixes

2.5 Usage of commonly used kṛt suffixes

Tavyat, tavya, anīyar, yat, ṇyat and *kyap* are the six suffixes which are used to derive verbal adjectives. The suffix *yat*[24] is added on to roots ending in a vowel in the sense of 'fit for' or 'fit to be' or 'ought to be'. When the suffix *yat* is added to roots ending in vowels the vowels undergo the following changes, final '*ā, e, ai*' and '*o*' are changed to '*e*', e.g. *deya, dheya.* Roots ending in '*ṛ*' and those ending in a consonant take the affix *ṇyat* in the same sense as *yat.* Before the affix the ending '*ch*' and '*j*' of a root are changed to '*k*' and '*g*' respectively and the final vowel and the penultimate a take *vṛddhi* substitute; any other penultimate vowel generally take *guṇa.* e.g., *kārya, dhārya.*[25] The root '*iṇ ṣtu, śāsa, vṛñ, dṛṅ*' and roots having '*ṛ*' short for their penultimate, except *kṛp* and *cṛt*, take the affix *kyap* in the same sense as *yat.* When a root ends in a short vowel, '*t*' is inserted between the final vowel and affix *kyap.* e.g., *itya, stutya, śās, vṛtya, ādṛtya, juṣya.*[26] The suffixes *tavyat* and *anīyar* are added to root or derivative bases in the sense of 'must be', 'fit to be'. Before these the ending vowel and the penultimate short of a root take their *guṇa* substitute. Before *tavyat set* roots take *'i'*, *anit* roots do not, and *vet* roots take it optionally. Before *anīyar* penultimate *'ṛ'* is always changed to *'ar'* and not to *'r'*. e.g., *dātavya, dānīya.*

The **Potential Participle** is formed by means of affixes *tavya, anīyar*[27], *yat* and rarely *kelimar*, added to a root or derivative verb. This is passive when the verb is transitive and impersonal when the verb is intransitive. The suffix *kelimar* was added later by *kātyāyana* in the list of *kṛtya* suffixes. *Kātyāyana* had probably detected in his time the use of a number of forms showing the suffix *kelimar.* In order to account for these forms

[24] Pāṇini Aṣṭādhyāyī; 3.1.97
[25] Pāṇini Aṣṭādhyāyī; 3.1.124
[26] Pāṇini Aṣṭādhyāyī; 3.1.109
[27] Pāṇini Aṣṭādhyāyī; 3.1.96

he added the suffix *kelimar. 'Pacelimā māṣāḥ'* meaning *paktavyāḥ* and *'bhidelimā saralāḥ* meaning *bhettavyāḥ* are the example provided by *patañjali* in the *Mahābhāṣya.*[28]

Words ending in *śatṛ* and *śānac* are **Present Participles**. Shorn of the indicatory letters *'ś, ṛ'* and *'c'* these suffixes are only '*at*' and '*āna*' in their efficient form. '*at*' is found with parasmaipada roots while *'āna'* is found with ātmanepada roots. At times *'m'* is inserted between the root and the suffix *āna* and therefore, it appears to be *māna* in the final stage. Usually this '*m*' is inserted after roots ending in short *'a'*. These suffixes are not added straight to the root, istead, they are added after the *vikaraṇa* have been added and a stem has been formed as in the case of present indicative, *'m'* e.g., *bhū* > *bhava* > *bhavat.* The participle of the Present Passive is also formed with the suffix *śānac* along with the augment *mum*. This suffix is added to the passive base of the present indicative e.g., *ci* > *cīya* > *cīyamaan.* The suffixes of present participle denote the agent and object of the action. Therefore, the number and gender of these words follow either that of the agent or of the object. These suffixes are used in active as well as passive voice with the verbs which may be *tiṅanta* or *kṛdanta.*

The suffix *kta* and *ktavatu* are added to the primary as well as the derived roots to form the **Past Participle.** The '*ta*' of the suffix *kta* undergoes morphophonemic change after certain roots. Hence it is found as '*na, ka, va, la'* and '*ma'* also. The change of the suffix *kta* to *na* has been described by Pāṇini in a number of situations, i.e., (a) wherever the suffix *kta* immediately follows the consonant *'d'* and '*r'*, (b) after those roots ending in *'ā, e, ai'* and *'o'* which begin with a conjuct consonant and certain a semivowel (c) after some other roots, e.g., *hā, śvi, bhuj* etc. When the suffix *kta* changes to '*na*', the final '*d*' of the root is also changed to '*na'*. e.g., *svid* + *kta* = *svinna*, *bhid* + *kta* = *bhinna* etc. There are some roots with which *kta* is optionally changed to '*na'*, such as *vid* > *vitta/vinna.* The suffixes *kta* and *ktavatu* are added to a root in three ways, with an intermediate *'i'*, without an intermediate *'i'* and with an optional intermediate *'i'*. kta is

[28] Sharma, Dipti, Structure and Meaning

added without *'i'* to *aniṭ* roots and with *'i'* to *seṭ* roots. To all derivative roots except those which already end in *'i'*, *kta* and *ktavatu* are added with intermediate *'i'*.[29]

The Participles of the Perfect, both reduplicated and periphrastic can be formed by adding different suffixes. The reduplicated perfect takes the participle suffix *kvasu.* It is generally added to the weak base of the reduplicated perfect but when the weak base consists of only one syllable, the suffix is added with an intermediate *'i'*. The suffix *kvasu* is for Perfect Parasmaipada, for Perfect Ātmanepada the suffix is *kānac*, the suffix *kānac* also like *śānac* remains only *āna* in the efficient form. Words ending in *kvasu* and *kānac* are generally used in the Vedic language and sparingly in the classical language.

The **Indeclinable Past Participles** are of the nature of gerund. The gerund is formed by adding the suffixes *ktvā* and *lyap* to the primary root as well as to the derivative root. *ktvā* is added to roots without an *upasarga* and *lyap* to those with an *upasarga.* e.g., *nītvā, āgatya.* When *ktvā* is added to roots without an intermediate *'i'* the roots undergo all the changes that they undergo when *kta* is added but when *ktvā* is added with an intermediate *'i'* the roots take place change. Generally *guṇa* is substituted for penultimate short vowels and for final vowel and *Guṇa* may optionally be substituted for penultimate *'i'* and *'u'* of roots which begin with consonants and end in any consonant except *'v'*. The indeclinable gerund formation ending in *ktvā* and *lyap* is always used to denote an action in relation to another action, i.e. is refers to an action that has been done or will be done before some other action. Hence, it can be used in conjunction with verbs of all the three tenses. Morever, the verb can be a *tiṅanta* as also a *kṛdanta.*[30]

The **infinitives** are found with the ending *tumun* suffix. This is verbal formation which does not have variation of suffixes. *Tumun* is added to all types of verbal roots to denote the infinitive. As in the case of other verbal *kṛt* formation tumun may be used with primary as well as derived roots. It may be followed by a past participle or a *tiṅanta*

[29] Sharma, Dipti, Structure and Meaning
[30] Sharma, Dipti, Structure and Meaning

formation in any of the three tenses. In order to denote the sense of *oucitya* and *sāmarthya* the infinitive may be used with words other than verbals.
khaś, ṇini, kvip, kta, ktavatu, tṛn, kvasu, ktvā and *ḍ* kṛt suffixes denote the sense of past tense. The words formed by adding these suffixes are of two types, those that denote the agent of the action and those that denote the time of the action, e.g., *sahayudhvā, gataḥ, gatavān* etc.

khyun, ṇyuṭ, lyuṭ, lyu and *ñyuṭ* are the five kṛt suffixes having yu which is ultimately replaced by *an* in all the cases. *ñyuṭ* is used in the Vedic language only, while *ṇyuṭ* and *lyuṭ* are used in the classical language. *ṇyuṭ* is added to a root to express the sense of an agent who perform an art, e.g., *gai* + *ṇyuṭ =gāyana.* The suffix *lyuṭ* is added to denote action only in neuter gender, e.g., *has* + *lyuṭ* = *hasanam.* The suffix *khyun* indicates an instrumental agent. *Khyun* is added to verb root *kṛ* to denote the sense of instrumental agent when *kṛ* follows the words *āḍhya, subhaga* etc. The words thus formed are *āḍhyaṁkaraṇam, subhagaṁkaraṇam* etc. The suffix *yuc* comes in the sense of the agent being in such a habit and after intransitive verbs denoting motion or sound. *Yuc* can be added to any transitive verb to denote a habit of the agent but it can be used with those intransitive roots only which denote motion or sound. e.g., *cal* + *yuc =calan.*

The Suffixes *ṇvul, ṇvuc, vun, vuñ* and *ṣvun* are *'aka'* in their applied form.The suffix *ṇvul* is added to all roots and denote the agent of the action, e.g., *kāraka.* With some roots the *ṇvul* suffix forms the name of the diseases, e.g., *prachardikā* (vomiting).Sometimes it is added to denote the meaning of the roots, e.g., *āsikā.* The suffix *ṇvuc* comes after a root in the sense of turn or order of succession, worthiness of respect, debt owned to another person or birth or production. The suffix *vun* is added to *pru, sru* and *lu* in the sense of skilful, e.g., *pravaka, saraka, lavaka* etc. This may be added to any root when the idea of a blessing is to be conveyed. The suffix *vuñ* is added to *nind, hiṁs, kliś, khād, naś* with *vi, kship* with *pari, pariraṭ, parivādi, vyābhāṣ* and *asūy* to denote the agent of the action being in the habit of that action, e.g., *nindaka, hiṁsaka, khādaka* etc. The suffix *ṣvun* denotes an artist. It is used after a verb root to express the agent in the sense of an artist, e.g. *nartaka, rajaka, khanaka* etc[31].

The suffix *khal* is added to any verb root with *īṣat, dur* or *su* prefixed to it when the idea of ease or difficulty is present, e.g., *īṣatkaraḥ, duṣkaraḥ, sukaraḥ* etc. The suffix *ghañ* is added to roots ending in consonants has almost a universal application and a variety of senses, before this the final *'c'* or *'j'* are changed to *'k'* or 'g'; e.g., *pac - pākaḥ, ruj - rogaḥ* etc.

Besides these common suffixes there are various kṛt suffixes by which verbal nouns are formed with various significations from roots or derivative bases.

2.6 Role of anubandhas in kṛt suffixes

Anubandhas in general are a technical device adopted by Pāṇini to describe the structure of Sanskrit and therefore, they do not figure in actual use. An *anubandha* is a code-letter which indicates a grammatical function like elision and reduplication. *Anubandhas* plays an importence role in *kṛt* suffixes. In the *kṛt* suffixes *'i', 'u', 'ṛ', 'k', 'kh', 'gh', 'ṅ', 'c', 'ñ', 'ṭ', 'ḍ', 'ṇ', 't', 'n', 'p', 'r', 'l', 'ś',* and *'ṣ'*, are used as indicatory letters. In these indicatory letters *'c', 'ñ', 'ṭ', 'n', 'p', 'r',* and *'l',* indicate the accent.[32] In *kṛt* suffixes *'i'* as an *anubandha* is found in the penultimate as well as final position. In *kvanip, vanip, ḍvanip, manin, kvin, kvip*, etc. it is in the penultimate position while in *vini, ini, ṇini, ki* etc. it is in the final position. In most of these cases, especially if it is followed by a consonant it only facilitates pronunciation. But if *'i'* is in the final position it also serves to save the consonant before *'it'* from becoming *'it'* and thus from being dropped.
The presence of *'u'* as *anubandha* in the suffix *ktavatu* and *kvasu* etc. indicates that a stem formed by adding such a suffix takes the nasal augment *'num'* before *sarvanāmasthāna* ending.

[31] Sharma, Dipti, Structure and Meaning
[32] Sharma, Dipti, Structure and Meaning

'ṛ' as an *anubandha* is found in *atṛn* and *śatṛ* suffixes. Like the anubandha *'u', 'ṛ',* also indicate that a word formed by adding these suffixes would take the augment *'num'* before *sarvanāmasthāna* ending. But if *śatṛ* comes after a reduplicated root it does not take the augment *'num'*.

Those suffixes in which *'k'* is used as a indicatory letter (*kit*), the base does not undergo any phonological change or there is neither *guṇa* nor *vṛddhi* in the base.
'kh' is an exclusive *anubandha* of *kṛt* suffixes. It always stands at the beginning and has another letter (*c, ñ, n, l* and *ś*) as a second *anubandha* along with it. The purpose of mute *kh* is to indicate the augment of *'mum'* to words compounding with those that end in *'khit'* suffixes. The presence of *'gh'* as an *anubandha* in a suffix is indicative of gutturalization of palatals, i.e. change of *'c'* to *'k'* and *'j'* to *'g'*. *'ḍ'* like *'k'* as an *anubandha* in *kṛt* suffixes prevents the possibility of *guṇa* or *vṛddhi* in the base.[33]
'c' is used as an *anubandha* with the largest number of *kṛt* suffixes and is indicative of accent. The vowels used with *'c'* are usually for the ease of pronunciation. The words formed with *'cit'* suffixes are *antodātta*. *'ñ'* as an *anubandha* is used with the eight *kṛt* suffixes. It indicates accent as well as certain grammatical processes. The word formed with a *'ñit'* suffix has the acute accent on the first syllable. The exception of this rule are *ghañnta* formations of verb root *kṛṣh* and those having *ā* in the base. Such words have the acute accent on the final syllable. *'ṭ'* as an *anubandha* with *kṛt* suffixes comes singly as well as with other *anubandhas*. It indicates that the words formed with *'ṭit'* suffixes will form the feminine by adding *ḍīp*. *'ḍ'* as an *anubandha* found with *ḍa* and *ḍu kṛt* suffixes. It indicates that the base drops its *ṭi*. *ṭu* is an exclusive *anubandha* with roots, *ḍu* is also a root *anubandha*. *'n'* is found as an anubandha with number of *kṛt* suffixes. The presense of *'n'* in a suffix is indicative of the *vṛddhi* of the penultimate *'a'* and the final vowel of the root. *'t'* as an *anubandha* is found with the *ṇyat, tavyat* and *yat kṛt* suffixes. It denotes only the accent of the words formed by adding such suffixes. The words derived with the *'tit'* suffixes have *svarita* on the final syllable. But some exceptions are these rules. The presence of *'p'* in a suffix indicates that it is *anudātta*. In most of the

[33] Sharma, Dipti, Structure and Meaning

suffixes it has no other purpose to serve. But if *'p'* appears with *'c'*, it loses its force and the word becomes *antodātta*.

'r' used as an *anubandha* with a single *kṛt* suffix besides *kelimar* also indicate the accent of a word. The words formed with *'rit'* suffixes have the acute accent on the penultimate syllable. *kamul, khal, ṇamul, lyu, lyuṭ* and *lyap* suffixes, *'l'* is used as an *anubandha*. The base to which these suffixes are added always becomes *antodātta*.[34]

2.7 Rules for formation of Kṛt Suffixes in general

The process of joining *kṛt* suffix to a verb root according to Pāṇinian rules is as follows[35]-

- *it-saṃjñā* and deletion of *anubandhas* of verb root
- *ṇatva, ṣatva, upadhādīrgha* and *numāgama* processes of verb root whichever applicable.
- *it-saṃjñā* and deletion of *anubandhas* of *kṛt* suffix.
- Substitution (*ādeśa*) of *kṛt* suffix with *ana, aka* etc. whichever applicable.
- If *kṛt* suffix is applied with a verb root in condition of some neighbouring word (*upapada*), compound of the verb root with that word[36], *prātipadika-saṃjñā* of whole[37] and deletion of the *vibhakti* of *upapada*[38].
- Recognition of suffix as *sārvadhātuka* and *ārdhadhātuka.*
- If *sārvadhātuka,* insertion of *vikaraṇa,* between verb root and *kṛt* suffix, of the *gaṇa* which the verb root belongs to. If *ārdhadhātuka,* consideration of advisability of *iṭ* insertion (*iḍāgama*).
- If there is a rule for substitution of whole verb root in presense of specific suffix then substitution of verb root.
- Consideration of extention rules about the *kṛt* suffix.
- *aṅga-saṃjñā* of the whole part before *kṛt* suffix[39].

[34] ibid.
[35] Dikshit, Pushpa, Aṣtādhyāyī Sahajabodha Vol. III
[36] Pāṇini Aṣtādhyāyī; 2.2.19
[37] Pāṇini Aṣtādhyāyī; 1.2.46
[38] Pāṇini Aṣtādhyāyī; 2.4.71
[39] Pāṇini Aṣtādhyāyī; 1.4.13

- Processing of *aṅga* according to the *kṛt* suffix.
- *Sandhi* wherever applicable.
- *ṇatva, ṣatva* etc. processes wherever applicable.
- Now *kṛdanta* base ready for as a nominal derivative.

2.8 Major Characteristic of kṛt Suffixes

Kṛt suffixes can be structurally divided into two classes-

- Those *Kṛt* Suffixes that form declinable words
- Those *Kṛt* Suffixes that form indeclinable words

There division can also be made on the basis whether they form substantives or verbals. In all these *kṛt* suffixes *yat, ṇyat, ghañ, tṛc, ktin, lyuṭ, kta, ktavatu, tumun* and *ktvā* are very generally used and the rest are added to particular roots only and that also under specific conditions.

Semantically the words formed by the *Kṛt* Suffixes can be divided into the following categories-

- Those carrying the sense of agent, e.g. *nandanaḥ, śobhanaḥ, bhūṣaṇaḥ, nartakaḥ, kārakaḥ, hārakaḥ, nāyakaḥ, niyāmakaḥ, parivrājakaḥ* etc. These words are derived by adding various suffixes. *lyu, lyuṭ, kvip, kvin, ṇvul, ṣvun* etc. are generally used to denote the agent.
- Those carrying the sense of object or impersonal action, such as *hāsaḥ, vāsaḥ, kāryam, deyaḥ, geyaḥ* etc. Words denoting object are derived from transitive roots while those denoting action only are derived from intransitive roots.
- Those carrying the sense of *karaṇa* such as *vastra, vaktra, pātra, netra, śrotra* etc. These words are used in the neuter gender.
- Those carrying the sense of locus, e.g. *āsana, śayan, yān* etc.

A notable feature of *kṛdanta* formations which are substantives is that *taddhita* suffixes can be added to them and secondary words can be derived, e.g. *kṛtitva, buddhimān,* *dhṛtimat, dharmavittara, vyaktitva* etc. *Kṛdanta* functioning as verbals do not take any secondary suffix. A chief characteristic of the primary and secondary formations, i.e. *kṛdanta* and *tadditānta* words as well as compounds has been indicated by *Kātyāyana* as being dependent on their significatory power, that is to say, on their actual usage[40]. *Patañjali* clearly brings out the reference intended by *Kātyāyana* in the term pratyaya by saying *'abhidhānalakṣaṇāḥ kṛttaddhitasamāsāḥ'*.

2.9 Kṛdanta in Hindi Grammar

Kṛdanta in Hindi is used in the form of verb, adjective, noun and adverb. Kṛdanta has a significant role in Hindi verb Morphology. Kṛdantas are mainly of two types, *'vikārī'* and *'avikārī'*. *'Vīkārī'* are those whose form changes according to Gender and Number. *'Avikārī'* are those whose form does not change. Both again are of four types each.[41]

2.9.1 Vikārī kṛdanta

1. **Present or continious kṛdanta**: This type of kṛdanta is formed by 'verb root + *t* + gender and number suffix' e.g., *cala + t + ā = calatā*, similarly, *jāte, sotā, likhatīṃ* etc. This kṛdanta is used in four forms.
   a. Noun- *maratā kya na karatā?*
   b. Adjective- *sote sāṃpa ko mata cheḍo*
   c. Adverb- rāma douḍatā ā rahā hai
   d. Verb- *rāma paḍhatā hai*

   As in Hindi, present kṛdanta is formed by '*t*' suffix, in Sanskrit, the same is formed by adding *śatṛ and śānac* suffixes. The Hindi *'t'* suffix is related to *śatṛ* suffix. e.g., sk. *cal + śatṛ = calantaḥ* > pr. *calaṁto* > *calatā, calatī*

2. **Past or perfect kṛdanta**: This type of kṛdanta is formed by 'verb root + *0 suffix* + gender and number suffix' e.g., *cala + 0 + ā = calā*, similarly, *paḍhā, mare, gaī, likhī* etc. This kṛdanta is used in four forms.

[40] *'Abhidhānalakṣaṇatvāt pratyayasya'* Kātyāyana on Pāṇini Aṣṭādhyāyī; 3.3.19
[41] Tiwari, Bholanath, Hindi Bhasha,

a. Noun- *soye ko mata jagāo*

b. Adjective- *soye vyakti ko mata jagāo*

c. Adverb- *rāma douḍā āyegā*

d. Verb- *rāma abhī nahīṃ soyā*

In Sanskrit, past kṛdanta is formed by adding '*kta*' and '*ktavatu*' suffixes. In Hindi, past kṛdanta forms are developed from Sanskrit '*kta*' suffix forms. eg., *sk. cal + kta = calitaḥ > caliyo > cal +* gender and number suffix *(ā, ī, e) = calā, calī, cale.*

3. **Imperative or verb sense kṛdanta**: This type of kṛdanta is formed by 'verb root + *n* + gender and number suffix' e.g., *cala + n + ā = calanā,* similarly, *jānā, jīnī,* etc. In imperative only *ā* comes as gender-number suffix, but in other forms, *ā, ī, e* also come. These forms are developed from Sanskrit *lyuṭ (ana)* suffix forms. e.g., *calanam > calanā, paṭhanam > paḍhanā*. It used as verb and noun.

4. **Agent denoting kṛdanta**: This type of kṛdanta is formed by 'verb root + *ne* + *vāl* + gender and number. This kṛdanta is used in three forms.

   a. Noun- *bhāgane vāloṃ ko pakaḍo*

   b. Adjective- *bhāganevāle laḍakoṃ ko pakaḍo*

   c. Verb- *rām bhāganevālā hai*

2.9.2 Avikārī kṛdanta

1. **Past participle kṛdanta**: This type of kṛdanta is formed three way-

   a. 'verb root + *kar*', e.g., *rāma khākara āyā hai*

   b. 'verb root + *ke*', e.g., *mohana kāma karake āyā hai*

   c. 'verb root + 0 suffix', e.g., *mai khā āyī hūṁ*

some times suffixes '*kar*' and '*ke*' are together forming this type of kṛdanta. e.g., *tum acchī taraha paḍhakar ke ānā.* In Sanskrit, kṛdantas of this meaning are formed from *ktvā* and *lyap* suffixes. '*kar*' and '*ke*' suffixes of Hindi are developed from Sanskrit *ktvā* suffix.

sk. *kṛtvā* > pr. *karittā* > *karia* > hi. *kari* > *kar*

sk. *kṛtvā* > pr. *karittā* > *karia* > hi. *kai* > *kai* > *ke*

2. **Continious sense kṛdanta**: This type of kṛdanta is formed by adding *'te'* suffix after verb root and it denotes the continious action. e.g., *tumhe pustaka likhate bahuta din ho gaye*. Duplication of the same becomes inter-action (*madhyakālika*) kṛdanta. e.g., *mai calate-calate tumhare bāre me socha rahā thā.*

3. **Perfect sense kṛdanta**: This type of kṛdanta is formed by adding '*e*' suffix after verb root and it denotes the completed action. e.g., *mujhe khānā khāye bahuta der ho gayī.* This kṛdanta is similar to a '*vikārī*' form of past kṛdanta. So the development of this kṛdanta is similar to past kṛdanta

4. **Immediate kṛdanta**: This type of kṛdanta is formed by adding *'te hī'* after verb root. e.g., *māṃ ke āte hī baccā prasanna ho gayā.* This kṛdanta is similar to a *'vikārī'* form of present kṛdanta and *'hī'* is added later. So the development of this kṛdanta is similar to present kṛdanta.

Chapter – III

Kṛdanta Recognition and Analysis

Chapter 3

Kṛdanta Recognition and Analysis

3.1 Introduction

Computational morphology has various approaches of analysis and most popular of them are rule based, example based and statistical approaches. The mechanism for *kṛdanta* analysis developed under this research follows the rule based approach broadly with necessary lexical interfacing.

This chapter describes the lexical resources needed for recognition and analysis of *kṛdanta* suffixes in a Sanskrit text according to *Pāṇinian* formalism. Strategy for using lexical database from within a rule based algorithm is essentially *Pāṇinian* in nature. Pāṇini's *sūtrapāṭha* calls specific items from lexical databases as and when needed. The lexical data used in the system is in different tabular and textual formats. The sample of the same is also given in the chapter. The lexical database includes-

1. a lexical *kṛdanta* database containing complicated forms with their *kṛdanta* information,
2. adapted Monier Williams Sanskrit Digital Dictionary (MWSDD) by Louis Bontes with words tagged with *kṛdanta* information,
3. corpus of current Sanskrit prose with words tagged with *kṛdanta* information.

3.2 Kṛdanta identification and analysis mechanisms

The process of *kṛdanta* analysis mechanism is divided into two sections - recognition and analysis.

3.2.1 Kṛdanta identification mechanism

1. The *kṛdanta* recognition starts by an exclusion process. The verb forms, *avyayas* and punctuations are excluded by running POS Tagger by checking verb, *avyaya* and pronoun databases and punctuation lists. The nominal bases

are obtained by the *subanta* analyzer which is a part of the POS tagger. These nominal bases are then checked in fixed lists by the POS tagger. This may result in some of the *subantas* being marked for *kṛdanta*. The remaining *subantas* are sent to the *kṛdanta* recognizer and analyzer system for recognition and analysis using following steps -

2. check the *kṛdanta* database, annotated corpus and *kṛdanta*-tagged MWSDD.
3. the *subantas* still untagged for *kṛdanta* are sent to the rule base for kṛdanta checking.
4. the rule base applies *Pāṇinian* rule base in reverse for marking *kṛdantas*.
5. it is possible that even after these systematic identification procedures, there may remain an untagged *kṛdanta subanta*. This will count as failure of the system.

3.2.2 Kṛdanta analysis mechanism

The system is divided into two parts- lexical database and rule-base. Lexical database of examples has been created for analyzing those forms which would be otherwise very complex to analyze if passed through the rule base. Lexical database has three major parts- a lexical *kṛdanta* database with complicated *kṛdanta* forms and their lexical information, Monier Williams Sanskrit Digital Dictionary and corpus of the current Sanskrit prose with *kṛdanta* words tagged with *kṛdanta* information.

The rule-base is for analyzing more regular forms. It consists of mainly three tables, namely, *upasargavikāra* table, *dhātuvikāra* table and *pratyayavikara* table. To restrict *dhātuvikāra* and *pratyayavikara* from inconsistent combinations, both are bound with a unique id. This will be described latter.

3.3 Strategy of lexical database

The first component of lexical database is lexical *kṛdanta* database. It has data from different sources like *Kṛdantarūpamālā* (a concordance of verbal derivatives) and *Siddhāntakaumudī.* It is in the format of –

Roopa,Dhātu,Gaṇa,Kṛt-suffix,Verbal-der-suffix,POS_gender

For example-

```
अतृ,अद,2,तृच्,0,noun_m
अतृ,अद,2,तृच्,0,noun_n
अत्री,अद,2,तृच्,0,noun_f
```

Table 3.1: sample of example base data

Roopa here means a *kṛdanta* derivative in *prātipadika* form or with feminine suffix. *Dhātu* is verb root in its original form with *anubandha.* *Gaṇa* is the class of verb root which the *dhātu* belongs to. The *gaṇas* are ten in number and *gaṇa* number in the table refers to a *gaṇa* as follows:

gaṇa_id	**Gaṇa_name**
1	bhvādigaṇa
2	Adādigaṇa
3	juhotyādigaṇa
4	Divādigaṇa
5	Svādigaṇa
6	Tudādigaṇa
7	rudhādigaṇa
8	Tanādigaṇa
9	kryādigaṇa
10	curādigaṇa

Table 3.2: dhātu gaṇa table

Kṛt suffix is nominal derivative suffix with its *anubandhas.* Verbal-der-suffix is a suffix which when added to a verb root, derives a new verb root with extended meaning. Verbal derivative suffixes are *ṇic, san, yaṅ, yak.*

POS is the category of derived form whether it is noun or indeclinable (*Avyaya*)
Gender is for noun derivative only that is masculine (m), feminine (f) or neuter (n).

The Second component of lexical database is Monier Williams Sanskrit Digital Dictionary (MWSDD). It has *kṛdanta* tagged with its information in the same format as lexical *kṛdanta* database. Other words are tagged as noun, pronoun, verb etc. Some examples from this dictionary are given as follows-

आख्यापक=आख्यापक,आङ्+चक्षिङ्,2,वुन्,णिच्,noun_m

आख्यापन=आख्यापन,आङ्+चक्षिङ्,2,ल्यु,णिच्,noun_n

आख्यापित=आख्यापित,आङ्+चक्षिङ्,2,क्त,णिच्,noun_m

उदाशंस्=Verb

उदास्=Verb

The Third component of lexical database is *kṛdanta* tagged Sanskrit corpus. These texts are from different sources like contemporary Sanskrit magazines, story books etc. These are running texts in which only *kṛdanta* forms are manually tagged with *kṛdanta* information. Example of some such text is given as follows-

शब्दायमानः[शब्दायमान,शब्दाय,0,शानच्,0,noun_m]

जातः[जात,जनी,4,क्त,0,noun_m] । अभ्यासबलेन दक्षिणेन हस्तेन

अधिकारी[अधिकारी,अधि+डुकृञ्,3,इनि,0,noun_m] ध्वनिग्राहिकाम्

उत्थाप्य[उत्थाप्य,उत्+ष्ठा,1,ल्यप्,0,avy] उक्तवान्[उक्तवत्,वच,2,क्तवतु,0,noun_m] – "हरिः ओम् अहम् उपमण्डलाधिकारी वदन्[वदत्,वद,1,शतृ,0,noun_m] अस्मि" इति ।

3.4 Strategy of rule base

When a nominal base is not analyzed as a *kṛdanta* by lexical database, it goes through rule base for *kṛdanta* analysis. The data base for rules consists of mainly three tables, namely, *upasargavikāra* table, *dhātuvikāra* table and *pratyayavikara* table.

3.4.1 Upasargavikāra table

This table has information of *upasargas* which can be found at the beginning of a *kṛdanta* form. *Upasarga*s are primarily 22 in number but by way of *sandhi* combinations, there can be many more possibilities as they get modified and also affect the *dhātu.* Also more than one *upasarga* can come consecutively such as *vyāpādita = vi + ā + pada + kta.* This table contains key columns *upasargavikāra,* basic *upasarga* and *upasargavikāra* expansion (uv expan). This is used to separate an *upasarga* from *dhātu* returning the original form of *dhātu* without *upasarga.* The separated *upasargas* are later prefixed to *dhātu* after analysis.

Id	Upasarga	Upasargavikara	uv expan
1	प्र	प्र	प्र+
1	प्र	प्रा	प्र+अ
1	प्र	प्राप	प्र+आप
1	प्र	प्राञ	प्र+आञ्
1	प्र	प्रे	प्र+इ
1	प्र	प्रे	प्र+ई
1	प्र	प्रो	प्र+उ
1	प्र	प्रो	प्र+ओ
1	प्र	प्रोह	प्र+ऊह
1	प्र	प्रे	प्र+ए
1	प्र	प्रै	प्र+ए
14	अधि	अध्य	अधि+अ
18	उत्	उद	उत्+अ
11	वि	व्या	वि+आ+

Table 3.3 Sample of upasargavikāra table

3.4.2 Dhatuvikāra table

This table has the informations of *dhātuvikāra* in *kṛdanta,* i.e., how a *dhātupāṭha* appears in a *kṛdanta* form. *Dhātuvikāra* is the key column of the table. Other information are *dhātu, gaṇa,* verb derivation suffix if any, *dhātuvikāra* type id. *Dhātu* is the verb root as read in *dhātupāṭha*. Derived *dhātus* are also given in the same column as basic *dhātu* and derivational suffix connected with plus sign.

Sr. No.	**Gana**	**Dhatu id**	**Dhatu**
1	भ्वादि	1	पठ
3	भ्वादि	1	पठ+णिच्
5	भ्वादि	1	पठ+यङ्
6	भ्वादि	1	पठ+सन्
7	भ्वादि	1	पठ+णिच्+यङ्
19	भ्वादि	1	पठ+णिच्+यक्
20	भ्वादि	1	पठ+यक्
21	भ्वादि	1	पठ+सन्+यक्

Table 3.4: sample of dhātu information in dhātuvikāra table

Gaṇa is the class of verb root which the *dhātu* belongs to. The *gaṇa* are ten in number as mentioned in table no.

Dhātu derivation suffixes are *ṇic, san, yaṅ, yak,* which when added to a *dhātu* derive a new *dhātu* with extended meaning.

Dhātuvikāra type id (dv type id) is not *dhātuvikāra* id but a unique number bound with a process undergone with a *dhātu* or derived *dhātu*. Different processes may result in the same modification so more than one dv type ids are bound with same *dhātuvikāra*. This dv type id is a constraint for *pratyayavikāras* which can concatenate with a particular *dhātuvikāra* to form a *kṛdanta*.

Dhatu	**Dhatuvikara**	**dv type id**
पठ	पठ	1,7,8,53,89,97,
पठ	पाठ	23,34,
पठ+णिच्	पाठय	1,4,68,
पठ+णिच्	पाठ	28,
पठ+यङ्	पापठ	1,31,71
पठ+सन्	पिपठिष	1,29,69,
पठ+णिच्+यङ्	पिपाठय	1,
पठ	पठिष्य	1,

पठ+णिच्	पाठयिष्य	1,4,
पठ+सन्	पिपठिषिष्य	1,
पठ+यङ्	पापठ्य	1,4,
पठ+यङ्	पापठिष्य	1,4,
पठ	पट्	
पठ	पड्	
पठ+णिच्+यङ्	पिपाठयिष्	1,
पठ	पाठ्	45,
पठ+सन्	पिपठिष्	
पठ+यङ्	पापठ्	
पठ+णिच्+यक्	पाठ्य	4,
पठ+यक्	पठ्य	4,
पठ+सन्+यक्	पिपठिष्य	5,
पठ	पट्टि	
पठ	पठ्	

Table 3.5: dhātuvikāra type id for dhātuvikāras

3.4.3 Pratyayavikāra table

This table has the information of *pratyayavikāra* in a *kṛdanta* form, i.e., how a suffix appears in a derived *prātipadika* form. The information of the key columns of *pratyavikāra* table are suffixes prescribed by *Paṇini*, suffixes in efficient form, dv (*dhātuvikāra*) compatibility id and dv compatibility description.

By the phrase 'suffixes prescribed by Paṇini' is meant *kṛt* suffixes with the *anubandhas* as, *ktvā, kta, ktvatu* etc.

A suffix in the efficient form is that part of a *pratyaya* which is directly added to a *dhātu* and it is many times different from *pratyayavikāra* as *tvā, ta, tavat, yu, vu* etc.

The dv compatibility id is a unique number bound to a *pratyayavikāra* according to the process a group of *dhātu* and a *pratyaya* undergo while forming a derived noun. These are bound to *dhātuvikāras* in *dhātuvikāra* table. As different processes may result in the same modification of *dhātu*, more than one dv types, i.e., *dhātus* modified as result of different processes, can concatenate with a *pratyayavikāra* to form a *kṛdanta*. Therefore, the similar *pratyayavikāra* is repeated and given different dv compatibility ids. This dv compatibility id is bound to dv type id in *dhātuvikāra* table. To this type of *dhātuvikāra* this *pratyayavikāra* can concatenate to form a *kṛdanta*. This requirement will restrict, the apparently possible but not technically correct, combinations of *dhātuvikāra* and *pratyayavikāra*.

dv compatibility description is the explanation of dv compatibility id as well as of dv type id of *dhātuvikāra* table. This is on the basis of what these ids are given and they differ with each other. Though being a very important part, this column is not directly used in programming and role of these is performed by their ids.

id	**suffixes prescribed by Panini**	**pratyaya vikara**	**dv compatibility id**	**dv compatibility description**
1	शतृ	त्	1	*dhātus* of *bhvādi, divādi, tudādi* and *curādigaṇa* with *vikaraṇa* and *ṇijanta, sannanta, kyajanta, kyaṣanta, yaṅluganta dhātus*
1	शतृ	त्	2	*dhātus* of *kryādigaṇa* with *vikaraṇa*
1	शतृ	त्	3	*dhātus* of *svādi, tanādi, adādi, juhotyādi* and *rudhādigaṇa* with *vikaraṇa*
2	शानच्	मान	4	*dhātus* of *bhvādi, divādi, tudādi* and *curādigaṇa* with *vikaraṇa* and *ṇijanta, kyaṅanta, kyaṣanta* and *yaṅanta dhātus*
2	शानच्	माण	5	*dhātus* of *bhvādi, divādi, tudādi* and *curādigaṇa* with *vikaraṇa* and *dhātu* of *ṇijanta, sannanta, kyaṅanta, kyaṣanta,*

				yaṅanta which have probability of *ṇatva*
2	शानच्	◌ान	6	*dhātus* of *kryādigaṇa* with *vikaraṇa*
2	शानच्	◌ान	7	*dhātus* of *svādi, tanādi, adādi, juhotyādi* and *rudhādigaṇa* with *vikaraṇa*
2	शानच्	◌ाण	8	*dhātus* of *svādi, tanādi, adādi, juhotyādi* and *rudhādigaṇa* with *vikaraṇa* which have probability of *ṇatva*
22	क्त्वा	त्वा	60	vowel ending *dhātus* with *guṇa-niṣedha, samprasāraṇa* (expansion) and after which *tvā* suffix does not change
22	क्त्वा	त्वा	61	*dhātus* of class *'ghu'* which have *'dath'* replacement with deletion of last consonant
22	क्त्वा	त्वा	64	*dhātus* with *samprasāraṇa* and *kutva*
22	क्त्वा	◌ित्वा	88	*i, ī, u, ū, ṛ, ṝ* ending *seṭ dhātus* with *guṇa, ay, av, ar*, replacement in a ending form
22	क्त्वा	◌ित्वा	105	Vowel ending *seṭ dhātus* with *guṇa-niṣedha* and *iyaṅ, uvaṅ* replacement in a ending form
22	क्त्वा	◌ित्वा	65	*seṭ dhātus* with *guṇa-niṣedha*, penultimate *n* deletion and *samprasāraṇa* in a ending form
22	क्त्वा	◌ित्वा	101	*seṭ* consonant *dhātus* in *akittva* case, penultimate *n* non deletion in *a* ending form

22	क्त्वा	त्वा	67	*k* ending and which become *ka* ending , *aniṭ dhātus* with *guṇa-niṣedha*, penultimate *n* deletion and *samprasāraṇa* in *k* ending form
22	क्त्वा	ट्वा	68	*c, j, s* ending specific *aniṭ dhātus* and *ch, ś, ṣ* ending *aniṭ dhātus* in *ṣ* ending form
22	क्त्वा	ट्वा	69	*ṭ, ṇ* ending *aniṭ dhātus* in consonant ending form
22	क्त्वा	त्वा	71	*t* ending and *t* ending becoming *d* ending *dhātus*, *p* ending *dhātus* with *guṇa-niṣedha*, penultimate *n* deletion and *samprasāraṇa* in consonant ending form
22	क्त्वा	ध्वा	72	*dh*, *bh*, ending *aniṭ dhātus*, *h* ending *aniṭ dhātus* with *dh, gh* replacement, with *guṇa-niṣedha*, penultimate *n* deletion, *samprasāraṇa* and *jaś* replacement
22	क्त्वा	त्वा	75	*ṇ, n, m* ending *aniṭ dhātus* with final nasal deletion or *m* ending penultimate lengthening in *n* ending form
22	क्त्वा	त्वा	76	*v* ending *dhātus* with *ūṭh* replacement and *yaṇ, ay, vṛddhi* replacement in vowel ending form
22	क्त्वा	त्वा	78	*s* ending *dhātus* with penultimate *n* deletion and *guṇa-niṣedha* in consonant ending form
22	क्त्वा	ढ्वा	79	Consonant ending *dhātus* with *ḍh* replacement with *samprasāraṇa, guṇa-*

				niṣedha, penultimate *n* deletion, *ḍh* deletion and vowel lengthening
22	क्त्वा	ित्वा	80	*ṇijanta dhātus* with *guṇa* and *ay* replacement in *a* ending form
22	क्त्वा	ित्वा	81	*San* derived *dhātus* in *a* ending form
22	क्त्वा	ित्वा	82	*yaṅ* derived *dhātus* which have vowel before *y* with *ya* ending form
22	क्त्वा	ित्वा	83	*yaṅ* derived *dhātus* which have consonant before *y* with *ya* deletion in a ending form
22	क्त्वा	ित्वा	84	*kyac, kyaṅ, kyaṣ* suffix derived *dhātus* with *ya* deletion in *a* ending form
22	क्त्वा	्यित्वा	85	*kyac, kyaṅ, kyaṣ* suffix derived *dhātus* with *ya* deletion in *a* ending form

Table 3.6: Pratyayavikāra table with dv compatibility and description

3.4.4 Algorithm for the rule base for kṛdanta analysis

The rule base processing will start after the input will have gone through lexical database, and it is not analyzed in lexical database.

1. The input string (KR) will be cut into two parts starting with one character from left (LP) and rest part (RP) of string. We start with LP=1 character, because that is *dhātu* part and *dhātu* is never completely deleted, it remains atleast 1 character.
2. LP will be searched in *dhātuvikār*a column of *dhātuvikāra* table.
3. If not found, go to step 7.
4. If found, search RP in *pratyayavikāra* column in *pratyayavikāra* table where dv type id of *dhātuvikāra* is equal to dv compatibility id of *pratyayavikāra.*
5. If not found, go to step 7.
6. If found, it/these will be stored temporarily as a pair with *dhātuvikāra* alongwith the information of both in the tables and this will be called a solution (SOL).

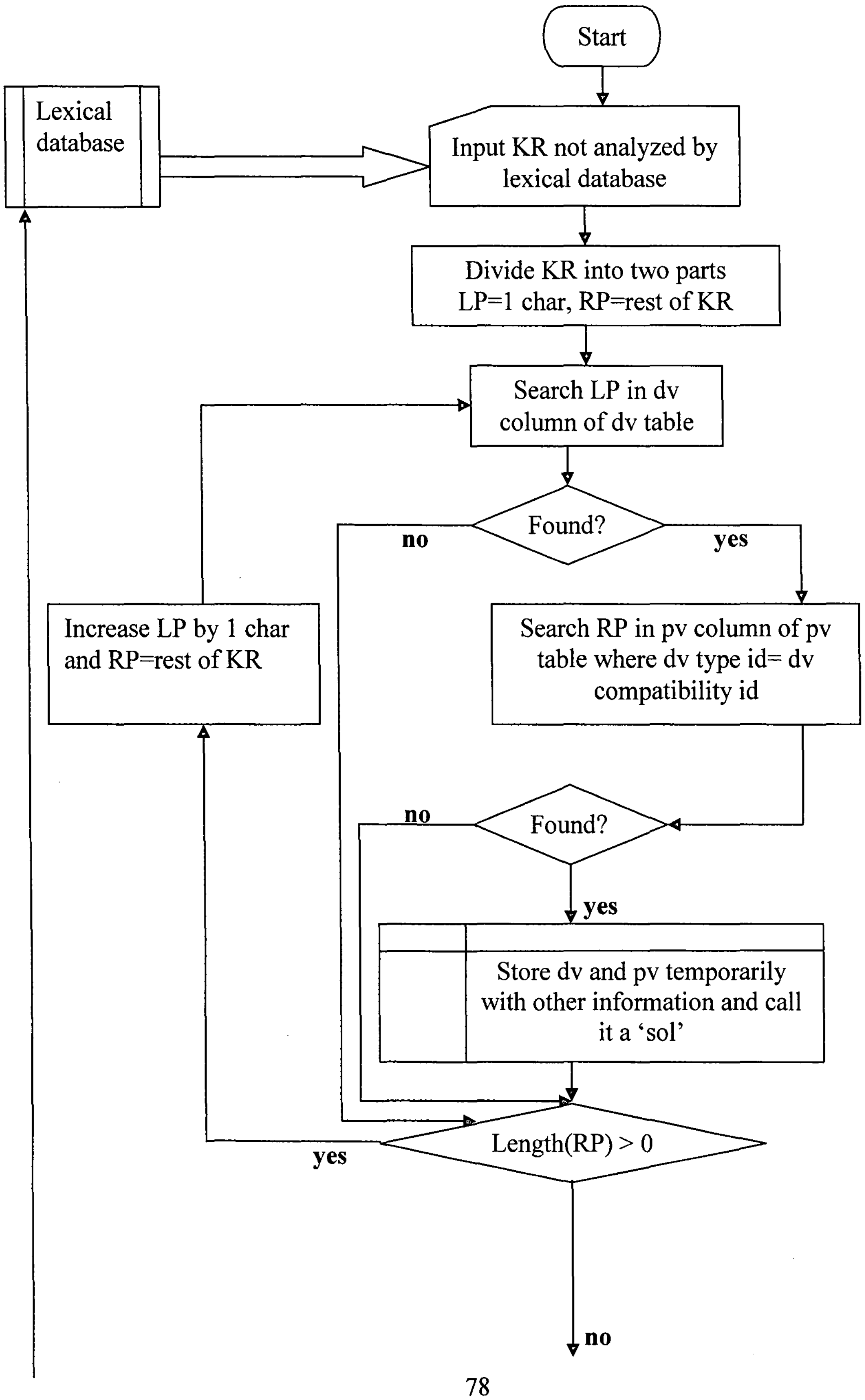
7. Till the RP is not empty string
8. First part (LP) will be increased by one character and second part (RP) will be the rest of the KR string. Now go to step 2.
9. If one or more SOL are found, replace *dhātuvikāra* with *dhātu* and *pratyayavikāra* with *pratyaya* and insert plus sign in between them. This will be called solution.
10. If no SOL is found then existence of *upasargavikār*a in the beginning of the KR string will be searched.
11. If upasarga exists, *upasargavikāra* will be replaced with *upasarga* expansion (uv expan).
12. The part of expanded string (ES) after the last plus sign will now be taken as KR and rest left part of ES will be taken as UP and this KR would be sent to be processed by example based system.
13. If *upasarga* is not found, the KR analyzable will be regarded as inseparable or basic *prātipadika* and it will itself be called solution.
14. If there is no value of string UP, define a new string UP as an empty string.
15. Now the result will be given in the form of

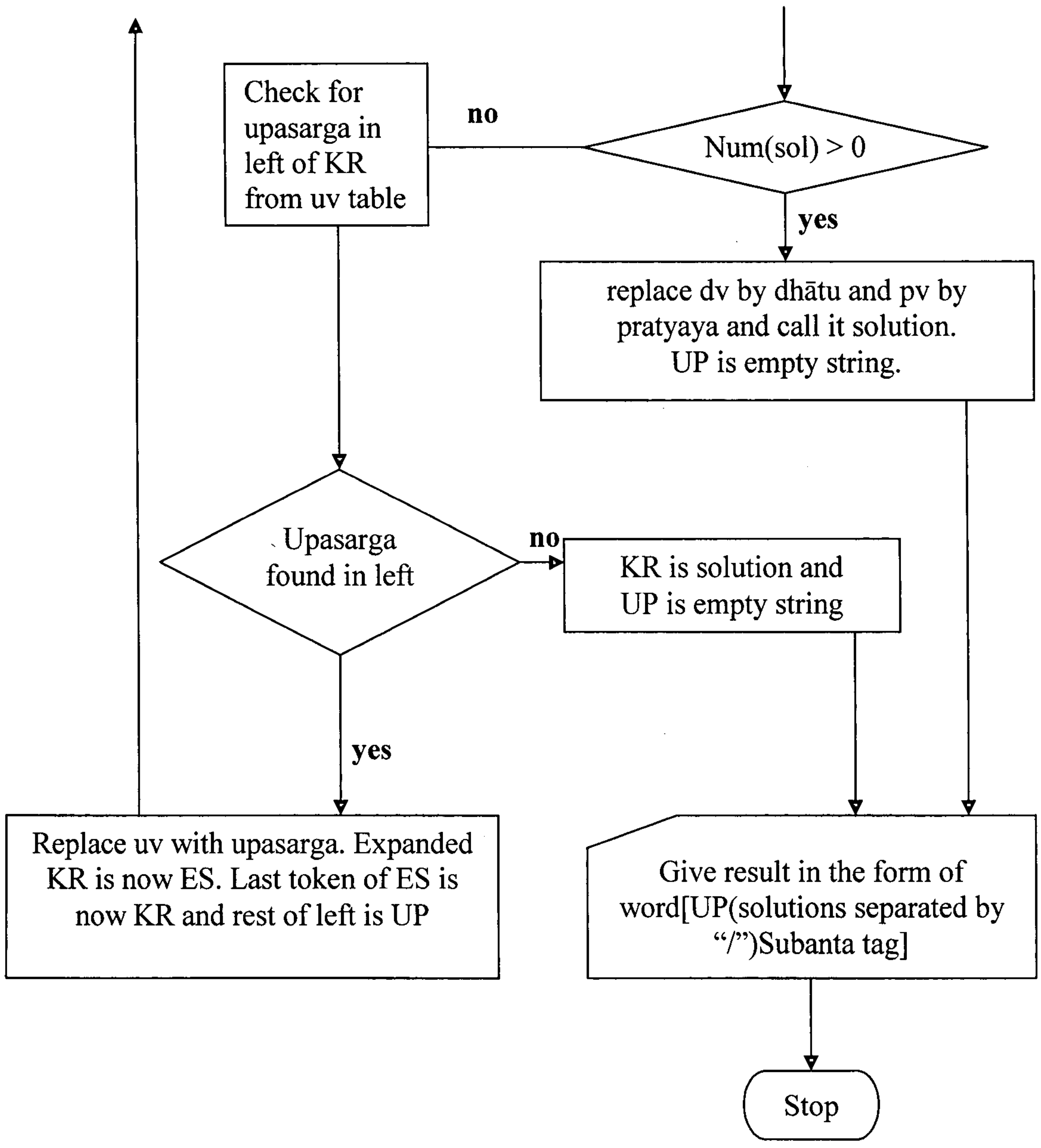
    word[UP(solutions separated by "/")Subanta tag]

    for example

    पाठकः[(पठ+ण्वुल्/पठ+णिच्+ण्वुल्)प्रथमा-एकवचन]

3.4.5 Flowchart illustration of rule-base processing


Start
Lexical database
Input KR not analyzed by lexical database
Divide KR into two parts LP=1 char, RP=rest of KR
Search LP in dv column of dv table
Found?
no
yes
Increase LP by 1 char and RP=rest of KR
Search RP in pv column of pv table where dv type id= dv compatibility id
no
Found?
yes
Store dv and pv temporarily with other information and call it a 'sol'
Length(RP) > 0
yes
no


Check for upasarga in left of KR from uv table
no
Num(sol) > 0
yes
replace dv by dhātu and pv by pratyaya and call it solution. UP is empty string.
Upasarga found in left
no
KR is solution and UP is empty string
yes
Replace uv with upasarga. Expanded KR is now ES. Last token of ES is now KR and rest of left is UP
Give result in the form of word[UP(solutions separated by "/")Subanta tag]
Stop

Flowchart 3.1: Illustration of rule based system

3.5 Illustration of the kṛdanta identification/analysis process

INPUT: पठ्यमानम् (*paṭhyamānam*)-

1. After *subanta* analysis → पठ्यमानम्[पठ्यमान+सु/अम्_प्रथमा/द्वितीया_एकवचन]
2. select *prātipadika* पठ्यमान for *kṛdanta* analysis
3. assume that it is complex form, and check in the POS database, *kṛdanta* database, k-annotated MWSDD, k-annotated corpus. If found, the process ends, else continues to the rule base

4. Rulebase check: INPUT: पठ्यमान
5. This string is cut into two parts LP= प and RP= ठ्यमान
6. प (LP) is searched in *dhātuvikāra* column of *dhātuvikāra* table
7. प is not found in *dhātuvikāra* column of *dhātuvikāra* table

८. LP is increased by one character. Now LP is पठ and RP is ्यमान

9. पठ (LP) is searched in *dhātuvikāra* column of *dhātuvikāra* table
10. पठ is found and its dv type ids are 1,7,8,53,89,97,
11. Now ्यमान is searched in the *pratyayavikāra* column in *pratyayavikāra* table where dv compatibility id is 1,7,8,53,89,97,
12. ्यमान is not found in the *pratyayavikār*a column in *pratyayavikāra* table where dv compatibility id is 1,7,8,53,89,97,
13. LP is increased by one character. Now LP is पठ् and RP is यमान
14. पठ् (LP) is searched in *dhātuvikāra* column of *dhātuvikāra* table
15. पठ् is found and its dv type ids are 200 (compaitable to lyap)
16. Now यमान is searched in the *pratyayavikāra* column in *pratyayavikāra* table where dv compatibility id is 200.
17. यमान is not found in the *pratyayavikār*a column in *pratyayavikāra* table where dv compatibility id is 200.
18. LP is increased by one character. Now LP is पठ्य and RP is मान
19. पठ्य (LP) is searched in *dhātuvikāra* column of *dhātuvikāra* table
20. पठ्य is found and its dv type id is 4.
21. Now मान is searched in the *pratyayavikāra* column in *pratyayavikāra* table where dv compatibility id is 4.
22. मान is found in the *pratyayavikāra* column in *pratyayavikāra* table where dv compatibility id is 4.
23. पठ्य and मान both are found, so this pair is stored temporarily with informations as

[पठ्य(भ्वादि–1–पठ+यक्)]+[मान(शानच्)]

24. LP is increased by one character. Now LP is पठ्यम and RP is ान
25. पठ्यम (LP) is searched in *dhātuvikāra* column of *dhātuvikāra* table.
26. पठ्यम is not found in *dhātuvikāra* column of *dhātuvikāra* table.
27. LP is increased by one character. Now LP is पठ्यमा and RP is न
28. पठ्यमा (LP) is searched in *dhātuvikāra* column of *dhātuvikāra* table.
29. पठ्यमा is not found in *dhātuvikāra* column of *dhātuvikāra* table.
30. LP is increased by one character. Now LP is पठ्यम and RP is ""
31. पठ्यमान (LP) is searched in *dhātuvikāra* column of *dhātuvikāra* table.
32. पठ्यमान is not found in *dhātuvikāra* column of *dhātuvikāra* table.
33. Result is found so there is no need of *upasarga* check.
34. In the stored pair, *dhātuvikāra* is replaced with *dhātu* and *pratyayavikāra* with *pratyaya*
35. Now we have पठ+यक्+शानच्
36. Now output result is given as

पठ्यमानम्[(पठ+यक्+शानच्)+सु/अम्_प्रथमा/द्वितीया_एकवचन]

Chapter – IV

Kṛdanta Recognizer and Analyzer for Sanskrit (KRAS)

Chapter 4

Krdanta Recognizer and Analyzer for Sanskrit (KRAS)

This chapter describes the development part of the *kṛdanta* recognizer and analyzer for Sanskrit (KRAS) which is a partial implementation of the research done for M. Phil dissertation. The *kṛdanta* recognition and analysis mechanism discussed in the previous chapter has been applied to develop a computational system which can identify and analyze *kṛdanta.* The computational model uses Java in the web format for the identification and analysis of *kṛdanta* from Sanskrit texts according to *Pāṇinian* and *Siddhānta Kaumudī (SK)* formalism. The system accepts words/sentences/ text Devanagari utf-8 input in the text area and gives analyzed output in the same format. The identification of *kṛdanta* forms depends on recognizing the *kṛt* suffix or ending in the words of the given text. There is an optional 'run-in-debug mode' which displays the internal states of KRAS.

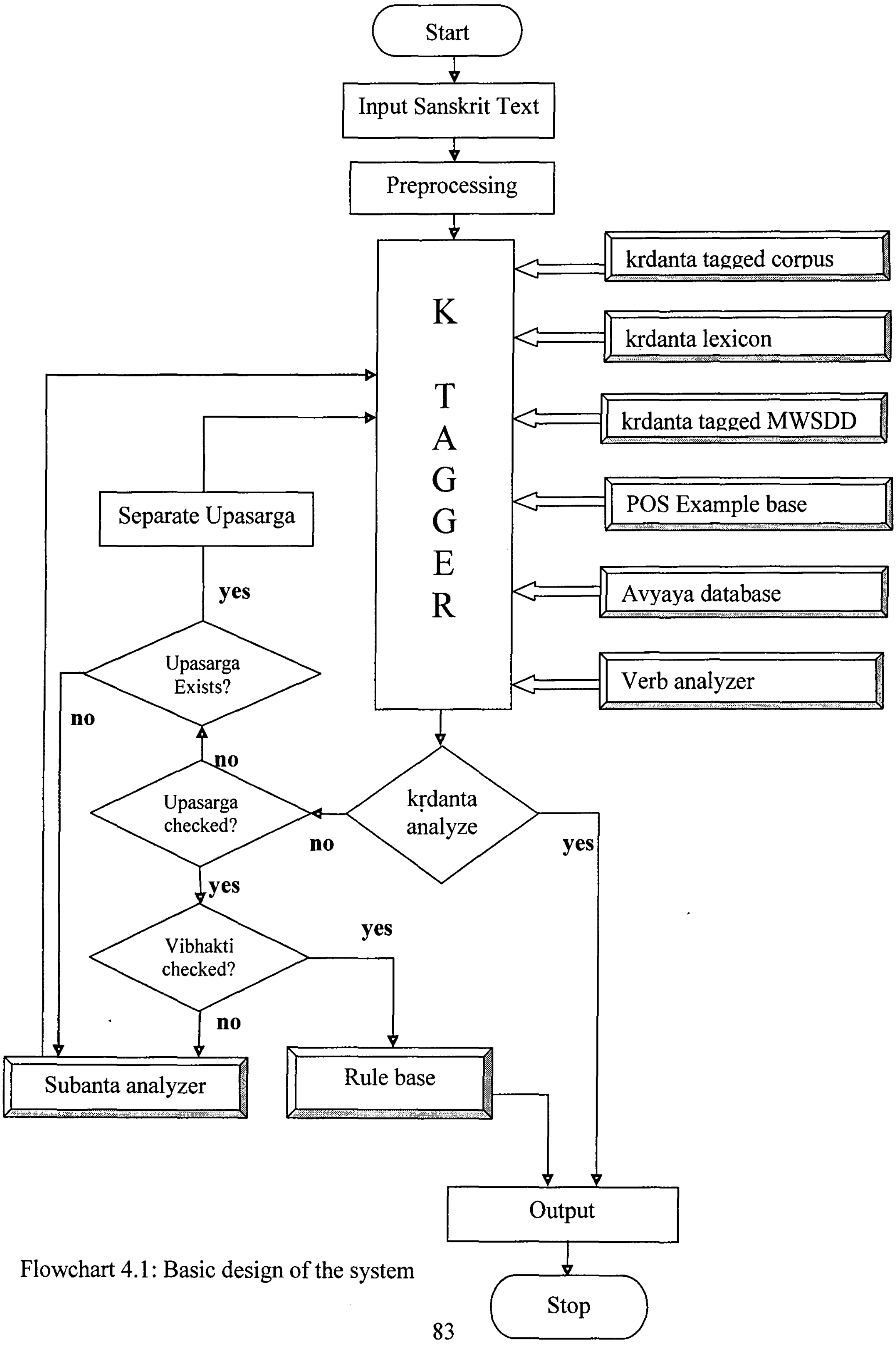
4.1 Architecture of the system

The following model describes the interaction between multi-tier architecture of the KRAS

$$\begin{tikzcd}
& \textbf{U S E R} \arrow[dl] & \\
\textbf{request} \arrow[dr] & & \textbf{response} \arrow[ul] \\
& \textbf{Apache-tomcat} \arrow[ur] \arrow[d, shift right] & \\
& \textbf{Java servlet} \arrow[u, shift right] \arrow[d, shift right] & \\
& \textbf{Data Files} \arrow[u, shift right] &
\end{tikzcd}$$

4.2 Module Description

The module description of the KRAS is given below



Flowchart 4.1: Basic design of the system

4.2.1 Pre-Processing

The system analyzes the Sanskrit Unicode text only. So at first, it is required to check if the text is in Unicode Devanagari and in Sanskrit language. In preprocessing the text is checked whether its encoding is UTF-8 and the characters are in the range of Devanagari.

4.2.2 The Kṛdanta Tagger

The K-Tagger is a program which tags the given Sanskrit text with the help of *kṛdanta* tags corpus, *kṛdanta* lexicon, *kṛdanta* tags MWSDD, *avyaya* database and verb analyzer. It marks *vibhakti* also with the help of *Subanta* analyzer.

4.2.3 Verb Analyzer

Verb Analyzer has a verb database which contains the ninety thousand verb forms of commonly used 500 *dhātus*. The verb forms which are unanalyzed by the commonly used verb database, are analyzed through rule-base.

4.2.4 Subanta Analyzer

Subanta analyzer has an example database which contains the exceptional forms with their constituent information. The remaining regular forms are analyzed by the rule base.

4.3 The front-end: online interface

The front end of the system is the Graphical User Interface (GUI), visible to the users. It is live at http://sanskrit.jnu.ac.in/kridanta/ktag.jsp. It has been created using JSP (Java Server Pages), Java and HTML components. The main JSP file *ktag.jsp* allows the user to feed the input in Devanagari utf-8 format using HTML text area component. Upon clicking the button labeled "Click to tag the above text" it calls the Java object "Kridanta" to process the input. The output returned by the Java objects is displayed to the user in Devanagari utf-8 format.

4.3.1 The web server

The *kṛdanta* analyzer runs on Apache Tomcat 4.0 platform. The details for this Java based web server follows –

4.3.1.1 Apache Tomcat 4.0

Apache Tomcat is the servlet container that is used for the Java Servlet and JavaServer Pages technologies. The Java Servlet and Java Server Pages specifications are developed by Sun under the Java Community Process. Apache Tomcat is developed in an open and participatory environment and released under the Apache Software License. Apache Tomcat is intended to be a collaboration of the best-of-breed developers from around the world[1].

4.3.1.2 Java Servlet Technology

Java Servlet technology provides web developers with a simple, consistent mechanism for extending the functionality of a web server and for accessing existing business systems. A servlet can almost be thought of as an applet that runs on the server side--without a face. Java servlets make many web applications possible[2].

4.3.1.3 Java Server Pages

Java Server Pages (JSP) technology provides a simplified, fast way to create dynamic web content. JSP technology enables rapid development of web-based applications that are server and platform-independent[3]. JSP pages are, however, compiled into servlets. Still, it is better to use JSP pages instead of always using servlets because JSP technology separates the web-presentation from the web-content and thus simplifies the process of creating pages. Basically JSP pages use XML tags and scriptlets written in the Java programming language to encapsulate the logic that generates the content for the web page. On the other hand, it passes any formatting (HTML or XML) tags directly back to the response page. In this way, JSP pages separate the page logic from its design and display. It is one of the most sophisticated tools available for high performance and secures web applications.

---

[1] Apache Tomcat website, http://www.apache.org/
[2] http://java.sun.com/products/servlet/
[3] http://java.sun.com/products/jsp/

4.3.2 Java Server Pages Class: ktag.jsp

The following code sets the encoding and character set and imports the java packages to be used in this class.

```
<%@ page
     language="java"
     pageEncoding="utf-8"
     contentType="text/html; charset=utf-8"
     import="java.util.*"
%>
```

The following code sets the encoding and content type of received input and returned output and obtains the text entered in the textarea as input text.

```
<%

     request.setCharacterEncoding("UTF-8");
     response.setContentType("text/html");
     String itext = request.getParameter("itext");
```

The following code checks if the result in debug mode is required or not.

```
     int dbg = 1;
     if  (request.getParameter("debug") == null)
          dbg = 0;

     String ch = "checked";

     if (dbg == 0)
          ch = "";

     if (itext==null)
          itext = "";
```

The following code creates a copy of the main class Kridanta to be used in this page.

```
     Kridanta k = new Kridanta(dbg);


%>
```

Following is the code of textarea in which the Unicode Devanagari text is to be entered

```
     <FORM METHOD=post ACTION=ktag.jsp name="iform" accept-
Charset="UTF-8">

        <TEXTAREA    name=itext    COLS=90    ROWS=7><%=itext
%></TEXTAREA></td>
```

The following is the code of button to start the process of tagging.

```
          <input  type=submit  value="Click  to  tag  the  above
text">
```

The following is the code of checkbox to select whether a result should be displayed in debug mode or not.

```
          Run in debug mode<input type=checkbox name="debug"
<%= ch %> value="ON">
```

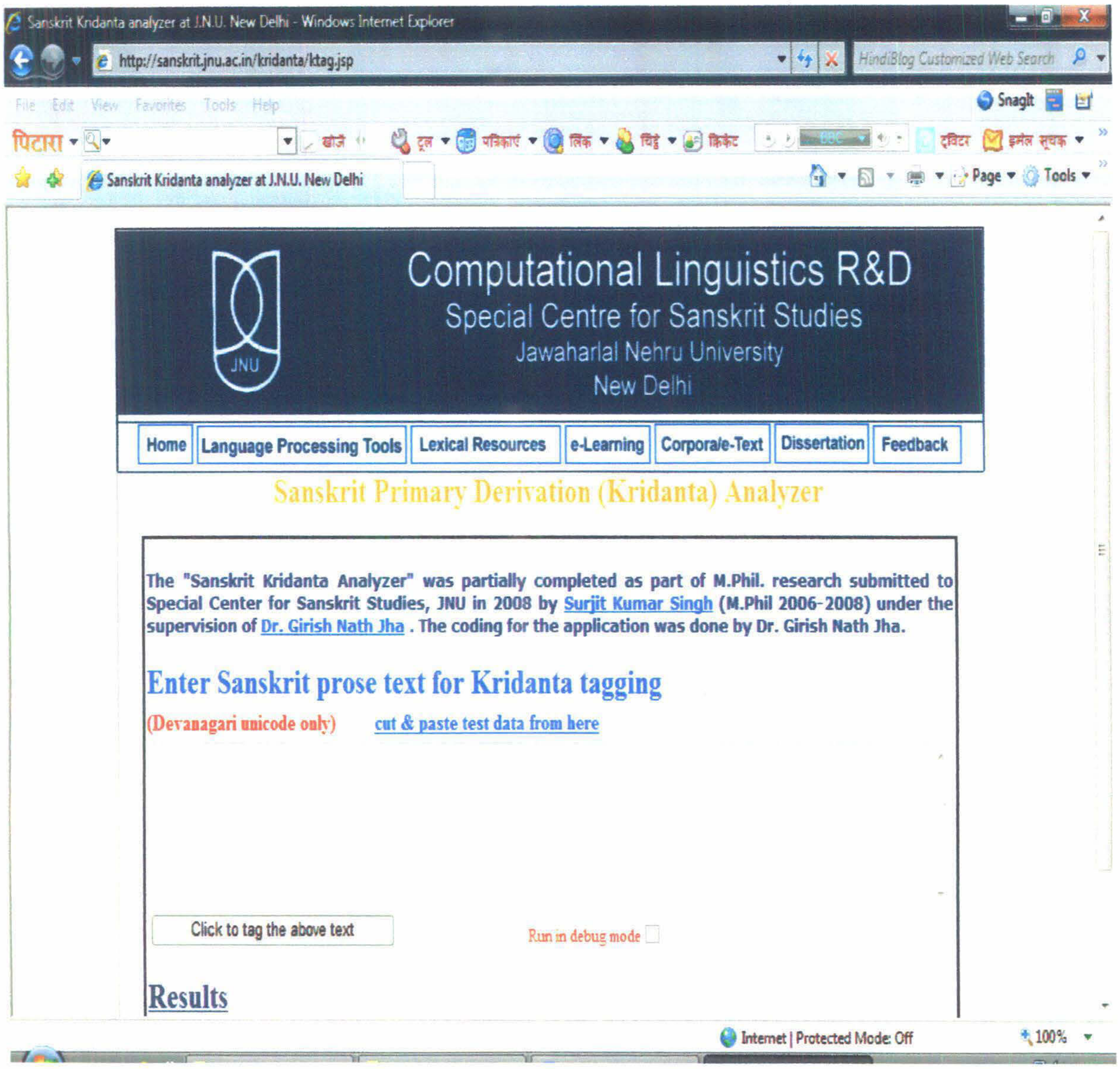
The following code displays the tagged text as result.

```
          <% if (itext.length()>0) { %>
               <%=k.tagText(itext) %>
          <% } %>
```

This code displays the process of analysis if the user wants the result in debug mode.

```
          <% if (itext.length()>0) { %>
               <%=k.printErr() %>
          <% } %>
```

4.3.3 The screen shot of the interface


Sanskrit Kridanta analyzer at J.N.U. New Delhi - Windows Internet Explorer
http://sanskrit.jnu.ac.in/kridanta/ktag.jsp
Computational Linguistics R&D
Special Centre for Sanskrit Studies
Jawaharlal Nehru University
New Delhi
Home
Language Processing Tools
Lexical Resources
e-Learning
Corpora/e-Text
Dissertation
Feedback
Sanskrit Primary Derivation (Kridanta) Analyzer

The "Sanskrit Kridanta Analyzer" was partially completed as part of M.Phil. research submitted to Special Center for Sanskrit Studies, JNU in 2008 by Surjit Kumar Singh (M.Phil 2006-2008) under the supervision of Dr. Girish Nath Jha . The coding for the application was done by Dr. Girish Nath Jha.

Enter Sanskrit prose text for Kridanta tagging

(Devanagari unicode only) cut & paste test data from here

4.4 The back-end: database/txt files

Kṛdanta analyzer is mainly developed for the purpose of server based program to be accessed widely and updated easily but it has CD version also which comes without the RDBMS component of the tiṅanta analysis system. The back end of server version contains lexical resources in the form of database tables and text files and CD version has the data in only the text files. In the CD version the tables of databases are also converted into text files. The full version of the system is available on the website only. CD version does not contain the full version of the system because of the limitations of RDBMS and supplying huge data-sets in the CD. The data structure of database files has been described in the third chapter. The data files which were developed under separate research and are being used in this program are not given here. Such files are *avyaya* file, POS example base file, verb example base file, *Subanta* example base and rule base files, proper noun file. One of the files contains Sanskrit Unicode UTF-8 text to test the system. Here sample data of the text files which are directly related with *kṛdanta* analyzer have been given.

4.4.1 Sample data of the kṛdanta corpus file

```
aaa;उपमण्डलाधिकारिणः;प्रकोष्ठः;शब्दायमानः=["शब्दायमान","शब्दाय","0","शानच्","0","no
un_m"];जातः=["जात","जनी","4","क्त","0","noun_m"];।;अभ्यासबलेन;दक्षिणेन;हस्तेन;अधि
कारी=["अधिकारिन्","अधि+डुकृञ्","7","इनि","0","noun_m"];ध्वनिग्राह्हिकाम्;उत्थाप्य=["उ
त्थाप्य","उत्+ष्ठा","1","ल्यप्","0","noun_m"];उक्तवान्=["उक्तवत्","वच","2","क्तवतु","0","no
un_m"];_;"हरिः;ओम;अहम्;उपमण्डलाधिकारी;वदन्=["वदत्","वद","1","शतृ","0","noun_m"]
;अस्मि";इति;।;
```

4.4.2 Sample data of the kṛdanta lexicon file

```
aa;आयन=इ,1,ल्युट्,णिच्,noun_n;मन्त्रयत्=मत्रि,10,शतृ,0,noun_m;मन्त्रयन्ती=मत्रि,10,शतृ,0
,noun_f;मन्त्रयत्=मत्रि,10,शतृ,0,noun_n;स्मयित्वा=ष्मिङ्,1,क्त्वा,णिच्,avy;जीवन=जीव,1,
ल्युट्,0,noun_n;स्कन्दन=स्कन्दिर्,1,ल्युट्,0,noun_n;दुधूनयिषित्री=धूञ्,10,क्त,सन्,noun_f;दुधू
नयिषितृ=धूञ्,10,क्त,सन्,noun_m;दुधूनयिषितृ=धूञ्,10,क्त,सन्,noun_n;जरित्री=जॄ,10,शतृ,0,no
un_f;जिजरिषित्री=जॄ,10,शतृ,सन्,noun_f;आञ्छक=आछि,1,ण्वुल्,0,noun_m;आञ्छक=आछि,1
,ण्वुल्,0,noun_n;आञ्छक=आछि,1,ण्वुल्,णिच्,noun_m;आञ्छक=आछि,1,ण्वुल्,णिच्,noun_n;
आञ्छनीया=आछि,1,अनीयर्,0,noun_f;आञ्छनीया=आछि,1,अनीयर्,णिच्,noun_f;आञ्छनीय=
आछि,1,अनीयर्,0,noun_m;आञ्छनीय=आछि,1,अनीयर्,0,noun_n;आञ्छनीय=आछि,1,अनीय
र्,णिच्,noun_m;आञ्छनीय=आछि,1,अनीयर्,णिच्,noun_n;आञ्छन=आछि,1,ल्युट्,0,noun_n;
```

आञ्छन=आछि,1,ल्युट्,णिच्,noun_n;आञ्छन्ती=आछि,1,शतृ,0,noun_f;आञ्छत्=आछि,1,शतृ,
0,noun_m;आञ्छत्=आछि,1,शतृ,0,noun_n;आञ्छयमाना=आछि,1,शानच्,0,noun_f;आञ्छय
मान=आछि,1,शानच्,0,noun_m;आञ्छयमान=आछि,1,शानच्,0,noun_n;आञ्छयन्ती=आछि,1,
शतृ,णिच्,noun_f;आञ्छयत्=आछि,1,शतृ,णिच्,noun_m;आञ्छयत्=आछि,1,शतृ,णिच्,noun_n
;आञ्छयिष्यमाणा=आछि,1,शानच्,णिच्,noun_f;आञ्छयिष्यमाण=आछि,1,शानच्,णिच्,noun_
m;आञ्छयिष्यमाण=आछि,1,शानच्,णिच्,noun_n;

4.4.3 Sample data of the MWSDD kṛdanta lexicon file

अंश=अंश,अंश्,1,अच्,0,noun_m;अंशकरण=अंशकरण,अंश+कृ,1,ल्युट्,0,noun_n;अंशकल्पना=
अंशकल्पना,अंश+कल्प,1,ल्यु,0,noun_f;अंशप्रकल्पना=अंशकल्पना,अंश+प्र+कल्प,1,ल्यु,0,noun
_f;अंशप्रदान=अंशप्रदान,अंश+प्र+दा,1,युच्,0,noun_n;अंशभागिन्=Noun;अंशभाज्=अंशभाज्,अं
श+भज्,1,क्विप्,णिच्,noun_m;अंशभू=अंशभू,अंश+भू,1,क्विप्,0,noun_m;अंशभूत=अंशभूत,अंश
+भू,1,क्त,0,noun_m;अंशवत्=Noun;अंशसवर्णन=अंशसवर्णन,अंश+स+वर्ण्,1,ल्युट्,0,noun_m;
अंशस्वर=अंशस्वर,अंश+स्वर्,1,अच्,0,noun_m;अंशहर=अंशहर,अंश+हृ,1,अप्,0,noun_m

4.4.4 Sample data of the upasarga lexicon file

ক=aप्राप=प्र आप;प्रात्र=प्र आत्र;प्रोह=प्र ऊह;प्रा=प्र अ;प्रे=प्र इ;प्रे=प्र ई;प्रो=प्र उ;प्रो=प्र
ओ;प्रे=प्र ए;प्रै=प्र ए;प्र=प्र ;परा=परा ;परे=परा इ;परो=परा उ;अपा=अप अ;अपे=अप
इ;अपो=अप ओ;अपो=अप ऊ;अप=अप ;सङ्=सम् ;सन्=सम् ;सञ्=सम् ;सण्=सम्
;सम्=सम् ;सं=सम् ;समि=सम् इ;समी=सम् ई;समु=सम् उ;समू=सम् ऊ;समृ=सम्
ॠ;समे=सम् ए;समे=सम् आ इ;समो=सम् ओ;सम=सम् अ;अनु=अनु ;अन्वा=अनु
आ;अन्वि=अनु इ;अन्वी=अनु ई;अनू=अनु उ;अनू=अनु ऊ;अन्वे=अनु ए;अन्वय=अनु
ऐ;अन्वो=अनु ओ;अन्वौ=अनु औ;अन्व=अनु अ;अव=अव ;अवा=अव अ;अवा=अव
आ;अवे=अव इ;अवे=अव ई;अवे=अव ए;अवो=अव उ;अवो=अव ऊ;निस्=निस् ;निष्=निस्
;निश्=निस् ;निर्=निर् ;निरि=निर् इ;निरी=निर् ई;निरु=निर् उ;निरू=निर् ऊ;निरे=निर्

ए;निरा=निर् आ;निरो=निर् ओ;निर=निर् अ;दुस्=दुस् ;दुष्=दुस् ;दुश्=दुस् ;दुरुप=दुर् उप ;दुर्=दुर् ;दुरा=दुर् आ;

4.5 Main Class: Kridanta

This is the main class of the program

```
public class Kridanta{
}
```

This class takes input Sanskrit text and gives finally output by using other subclasses.

The following are the initializations of buffered readers which get data from eleven data files.

```
BufferedReader br1 = null;
BufferedReader br2 = null;
BufferedReader br3 = null;
BufferedReader br4 = null;
BufferedReader br5 = null;
BufferedReader br6 = null;
BufferedReader br7 = null;
BufferedReader br8 = null;
BufferedReader br9 = null;
BufferedReader br10 = null;
BufferedReader br11 = null;
```

These are the initializations of different subclasses called in this main class.

```
Preprocessor pre = null;
SupAnalyzer sa = null;
TipAnalyzer ta = null;
FixedListTagger flt = null;
```

```
    KridantaRule krb = null;
```

The following code defines new values for the eleven buffered readers which read the data files of both the example base and rule base. After taking data from these, these are closed.

```
         try{
              br1 = new BufferedReader(.. .. ..);
              br2 = new BufferedReader(.. .. ..);
              br3 = new BufferedReader(.. .. ..);
              br4 = new BufferedReader(.. .. ..);
              br5 = new BufferedReader(.. .. ..);
              br6 = new BufferedReader(.. .. ..);
              br7 = new BufferedReader(.. .. ..);
              br8 = new BufferedReader(.. .. ..);
              br9 = new BufferedReader(.. .. ..);
              br10 = new BufferedReader(.. .. ..);
              br11 = new BufferedReader(.. .. ..);

              .. .. ..;

              br1.close();
              br2.close();
              br3.close();
              br4.close();
              br5.close();
              br6.close();
              br7.close();
              br8.close();
              br9.close();
              br10.close();
              br11.close();
```

```
          catch(Exception e){
          System.out.println("error  in  reading  data  files
"+e.toString());
          }
```

The following code is the main function of the class for analyzing *kṛdanta*, it applies preprocessor, *tiṅanta* analyzer, *Subanta* analyzer and *avyaya* check. If the token is not tagged it tries to analyze it through the rule base of *kṛdanta* analyzer.

```
     public String tagText(String s){
          String tkn="";
          String ts = "";
          s = s.trim();
          String tmp = "";
```

The following code applies after assuring that the text to tag is not empty.

```
          if (s.length()>0){
```

The following code calls the preProcessing() function of class Kridanta. This function checks if the text entered is in UTF-8 encoding and in Sanskrit language.

```
                    tkn= preProcess(tkn);
```

The following code calls checkFL() function of class Kridanta which tags the words which are found in fixed lists of *avyaya*, punctuation etc.

```
                    if ( .. .. ..)==-1 ){ //not yet tagged
                         tkn = checkFL(tkn);
```

```
                        }
```

The following code calls sup_analyze() function of Kridanta class which separates sup suffix from the word if the text is not tagged as *kṛdanta* and *sup* suffix is found.

```
                        if ( .. .. ..)==-1 ){ //not yet tagged
                             tkn = sup_analyze(tkn);
                        }
```

The following code calls tip_analyze() function of Kridanta class which checks and analyzes the verb forms if the word is not tagged as *kṛdanta.*

```
                        if ( .. .. .. ){ //not yet tagged
                             tkn = tip_analyze(tkn);
                        }
```

The following code calls applyRuleBase() function of Kridanta class which is applied on the input text which is not still tagged after applying the above functions.

```
                        if ( .. .. .. ){//if still not tagged
                             tkn = applyRuleBase(tkn);
                        }
```

The following code displays the tagged, untagged and ambiguous texts in different colours.

```
                        if (tkn.indexOf("[") ==-1) // untagged
                             tkn = "<font
color=red>"+tkn+"</font>";
```

```
                    else if (tkn.indexOf("]/[") >0) //
ambiguous
                         tkn = "<font
color=blue>"+tkn+"</font>";

                    ts = ts + " " + tkn;
```

The following code is the ending of the tagText() function of Kridanta class.

```
               }
               return ts;

          }
          else return "Please enter some Sanskrit text to
process";
     }
```

The following is the code of preProcess() function called above which takes the work done by the preProcess function of the class Preprocessor.

```
     private String preProcess(String tkn){

          if (tkn.length()>0){
               tkn = pre.preProcess(tkn);
          }
          return tkn;
     }
```

The following is the code of checkFL() function called above which tags the available *avyayas*, verbs, *kṛdantas* etc. from the fixed list data files through checkLists() function of class FixedListTagger.

```
private String checkFL(String tkn){

      if (tkn.length()>0){
            tkn = flt.checkLists(tkn);
      }
      return tkn;
}
```

The following is the code of sup_analyze() function called above which checks the word if it is *Subanta* and then separates its suffix with the help of analyzeSup() function of SupAnalyzer class.

```
private String sup_analyze(String tk){

      if (tk.length()>0){
            tk = sa.analyzeSup(tk);
      }
      return tk;
}
```

The following is the code of tip_analyze() function called above which checks the word if it is verb form and then separates its suffix with the help of analyzeTip() function of TipAnalyzer class.

```
private String tip_analyze(String tk){

      if (tk.length()>0){
            tk = ta.analyzeTip(tk);
      }
      return tk;
}
```

The following is the code of applyRuleBase() function called above which tries to tag the word as *kṛdanta* if the word is not tagged by above functions. It gets its work done by examine() function of KridantaRule class.

```
private String applyRuleBase(String tk){

     if (tk.length()>0){
          tk = krb.examine(tk);
     }
     return tk;

}
```

4.6 How to use the System

Program has a web version and a CD version. To use web version simply log on to the URL http://sanskrit.jnu.ac.in/kridanta/ktag.jsp and use any Devanagari UTF-8 mechanism to enter the text in the textarea or copy a text from sample file and paste in textarea.

CD version requires installing java development kit, Apache Tomcat web server 4.0, MS-SQL server 2005, MS-JDBC driver and a UTF-8 Devanagari input mechanism such as Baraha7.0[4]. First install the jdk any version then set the path of java by editing the path variable and adding the path of bin folder of java C:\jdk1.5.0_14\bin. Now install the Apache Tomcat web server. To run Tomcat Server two environment variables are to be defined-

JAVA_HOME= C:\jdk1.5.0_14

CATALINA_HOME= C:\Program Files\Apache Tomcat 4.0

Install MS-SQL server 2005 and MS-JDBC driver to connect the database with Apache Tomcat web server. Run Tinanta database script to create the database. Import data from excel file. Set permissions to database objects.

Now copy the Kridanta folder from the CD to the following folder-

[4] http://www.baraha.com/

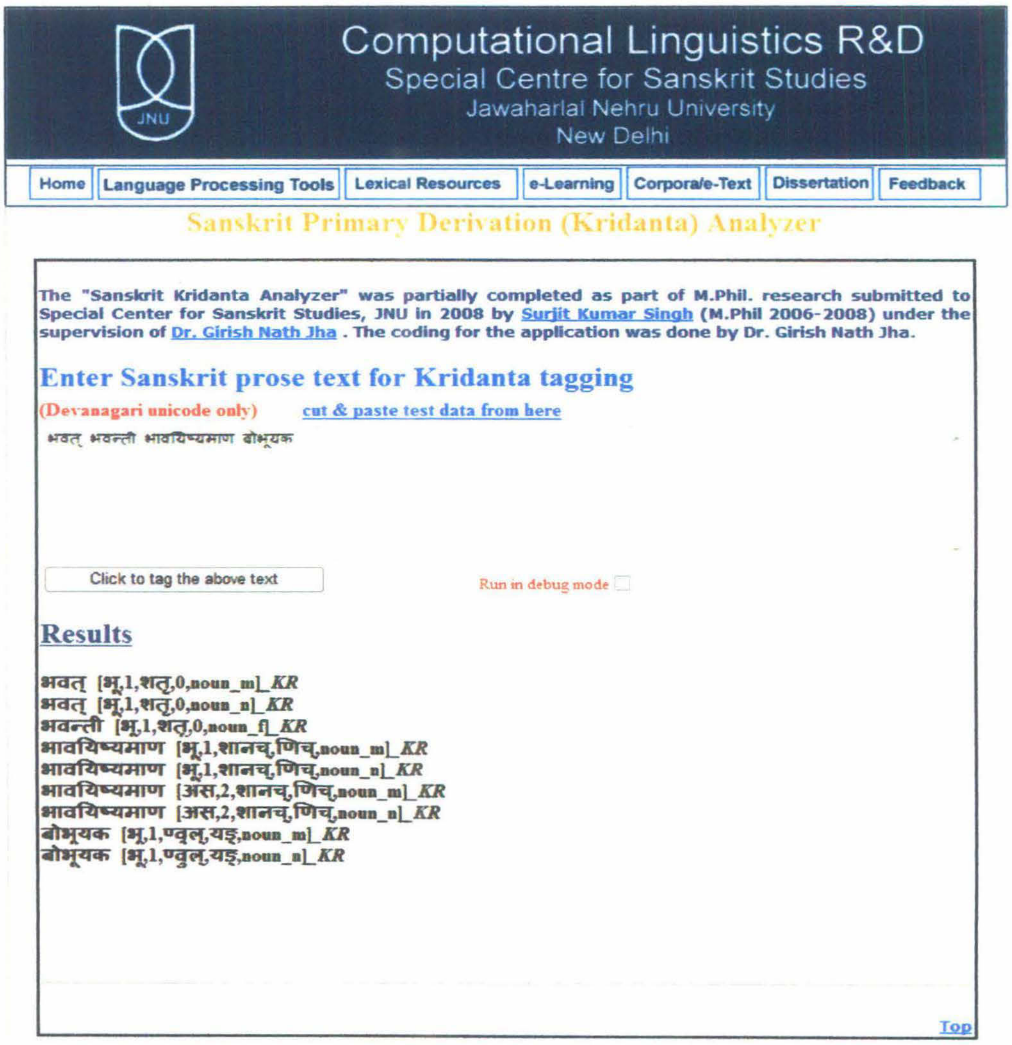
C:\Program Files\Apache Tomcat 4.0\webapps

Now start Tomcat Web Server and type the following address in the web browser-

http://localhost:8080/kridanta/ktag.jsp

Now either copy the text from sample into the textarea or open baraha direct, select language Sanskrit Unicode and type the text according to baraha transliteration scheme.

4.7 The screen shot of the interface with analysis of data



Computational Linguistics R&D
Special Centre for Sanskrit Studies
Jawaharlal Nehru University
New Delhi

Home | Language Processing Tools | Lexical Resources | e-Learning | Corpora/e-Text | Dissertation | Feedback

Sanskrit Primary Derivation (Kridanta) Analyzer

The "Sanskrit Kridanta Analyzer" was partially completed as part of M.Phil. research submitted to Special Center for Sanskrit Studies, JNU in 2008 by Surjit Kumar Singh (M.Phil 2006-2008) under the supervision of Dr. Girish Nath Jha . The coding for the application was done by Dr. Girish Nath Jha.

Enter Sanskrit prose text for Kridanta tagging

(Devanagari unicode only) cut & paste test data from here

भवत् भवन्ती भावयिष्यमाण बोभूयक

Click to tag the above text

Run in debug mode ☐

Results

```
भवत्  [भू,1,शतृ,0,noun_m]_KR
भवत्  [भू,1,शतृ,0,noun_n]_KR
भवन्ती  [भू,1,शतृ,0,noun_f]_KR
भावयिष्यमाण  [भू,1,शानच्,णिच्,noun_m]_KR
भावयिष्यमाण  [भू,1,शानच्,णिच्,noun_n]_KR
भावयिष्यमाण  [अस,2,शानच्,णिच्,noun_m]_KR
भावयिष्यमाण  [अस,2,शानच्,णिच्,noun_n]_KR
बोभूयक  [भू,1,ण्वुल्,यङ्,noun_m]_KR
बोभूयक  [भू,1,ण्वुल्,यङ्,noun_n]_KR
```

Top

Conclusion

Conclusion

Morphological analysis is known as shallow parsing but it is not so shallow, it is in fact a very serious and challenging task. Morphological analysis is challenging whether it is inflectional or derivational. Generally, the forward morphology is considered easier when compared with reverse morphology. The present R&D was on the latter and hence more challenging. The Pāṇinian system is very clear on the treatment of kṛdantas, but formalizing is required to make it adaptable for computing purposes. In Sanskrit, kṛdanta process is very productive and leads to a large number of derived nouns. The Pāṇinian analysis of kṛdanta was very helpful in this task. Hoewever, ancillary texts like Siddhāntakaumudī, Aṣṭādhyāyī Sahajabodha by Puspa Dixit and Kṛdantarupamālā by Pt. Ramasubba Shastri also proved very useful in this research.

The system developed as part of present R&D is live at http://sanskrit.jnu.ac.in and has two major components – the rule-base and the example-base. The rule-base is applied for regular forms after the example-bases have been exhausted.

The rule-base has been developed following the Pāṇinian methodology. A verb root undergoes several modifications, such as, guṇa, vṛddhi, upadhā-guṇa, upadhā-vṛddhi, samprasāraṇa, nasal deletion and num āgama when followed by a specific type of suffix such as ñit, ṇit, kit, ṅit, ghit. Moreover, a suffix undergoes modifications when preceded by different verb roots. Following this, in this research, verb modifications and the suffix modifications were collected as dhātuvikāra and pratyayavikāra respectively and to control unusual combinations the valid pairs of verb modifications and suffix modifications have been bound through a compatibility connection.

In this research, the example base is created for analyzing those forms which are difficult to analyze by the rule base. Those suffixes which are completely elided when joined with verb roots are difficult to find in a form. Those forms which are very complex to solve through rule base are easy to store. Exceptions, special cases and complex forms have been handled by creating exhaustive example bases. As it has been said earlier that this technique of example base is used by Pāṇini also as in sūtra kṣayya-jayyau śakyārthe (6.1.81).

Limitations

1. Sandhi splitting is not done. Therefore kṛdantas with sandhi will not be analyzed correctly
2. disambiguation betweenmultiple results has not been done, as it involves syntactic and kāraka analyses
3. kṛdanta recognition and analysis system internally uses POS Tagger, Subanta analyzer and tiṅanta analyzer. So the accuracy of the result of kṛdanta analyzer also depends on the output they give.
4. The strategy is unable to differentiate among the various types of kṛdanta forms which are similar. In a dhātu different kṛt suffixes result in similar forms as ṇvul and vuñ or ñuṭ and ṇyuṭ. Sometimes more than one dhātu with same suffix result in same form. It can cause unnecessary expansion of analyzed output. Present strategy cannot differentiate between them because it only analyzes morphemes.
5. Many kṛdanta forms are likely to come as part of a compound form while some are always found in compounds. Those forms will not be analyzed because system does not analyse compounds as of now.
6. kṛdantas found in the taddhitas (secondary derived nouns) also will not be analysed at this point

Possibilities of Research and development

1. kāraka analysis can be brought in for disambiguation purposes
2. The scope of this kṛdanta analyzer can be extended to the analysis of kṛdanta in compounds, taddhitas and nāmadhātus.
3. Help of sandhi analyzer can be taken for analyzing those which are not independently apparent in a sentence but joined with other words as samāsa or otherwise.
4. kṛdanta tagged corpus should be made richer for the disambiguation purposes.

5. Mapping of Sanskrit and other Indian languages kṛdanta forms can be done which will be useful in MT systems.
6. The current system accepts only Devanagari Unicode as input. It will be convenient for users if it supports Sanskrit input in roman and other schemes also.

APPENDICES

APPENDIX-I

i) <u>Sample data of the avyaya database</u>

abc=abc;अज्ञात्वा=[AVK];अदत्त्वा=[AVK];धृत्वा=[AVK];नीचैःकृत्वा=[AVK];न्यग्भूत्वा=[AVK];परापृष्ठीभूत्वा=[AVK];पीत्वा=[AVK];पृष्ट्वा=[AVK];प्रीणयित्वा=[AVK];बाधित्वा=[AVK];भुक्त्वा=[AVK];भोजयित्वा=[AVK];मुक्त्वा=[AVK];योषित्वा=[AVK];लात्वा=[AVK];विनाभूत्वा=[AVK];शसित्वा=[AVK];शसित्वा=[AVK];शान्त्वा=[AVK];स्थापयित्वा=[AVK];स्थित्वा=[AVK];स्थिरित्वा=[AVK];स्नात्वा=[AVK];हित्वा=[AVK];हुवा=[AVK];सदा=[AVKV];अन्यदा=[AVKV];इदा=[AVKV];एकदा=[AVKV];नित्यदा=[AVKV];सदा=[AVKV];सर्वदा=[AVKV];सर्वथा=[AVKV];वृथा=[AVKV];अन्यथा=[AVKV];अमुथा=[AVKV];अवृथा=[AVKV];इतरथा=[AVKV];इत्था=[AVKV];इमथा=[AVKV];उभयथा=[AVKV];ऊर्ध्वथा=[AVKV];ऋतुथा=[AVKV];एवथा=[AVKV];नामथा=[AVKV];प्रत्नथा=[AVKV];बहुथा=[AVKV];वृथा=[AVKV];अथवा=[AVC];अथोवा=[AVC];तद्वा=[AVC];दिवा=[AV];वा=[AVC];अकामतः=[AVKV];अकुतः=[AVKV];अग्रतः=[AVKV];अग्रतः=[AVKV];अज्ञानतः=[AVKV];अतः=[AVKV];अथातः=[AV];अधरतः=[AVKV];अधर्मतः=[AVKV];अनिमित्ततः=[AVKV];अनुरूपतः=[AVKV];अनुसारतः=[AVKV];अन्ततः=[AVKV];अन्यतः=[AVKV];अभितः=[AVKV];अभितः=[AVKV];अभिवाहतः=[AVKV];अभीपतः=[AVKV];अभ्यन्तरतः=[AVKV];अमुतः=[AVKV];अर्जुनतः=[AVKV];अर्थतः=[AVKV];अवरतः=[AVKV];अवरार्धतः=[AVKV];

ii) <u>Sample data of the noun list</u>

a=a;निर्जर=N_n;सेवक=N_m;सेविका=N_f;वर्ग=N_m;आज्ञा=N_f;जीवित=N_n;तृण=N_n;राज्यद्वितयसैन्यसामग्री=N_f;सामग्री=N_f;नीति=N_f;तनय=N_m;तनया=N_f;पद=N_n;राजसूनु=N_m;पुत्रत्व=N_nसाहाय्य=N_n;किंवदन्ती=N_f;कृत्य=N_n;राजतनय=N_m;राजतनया=N_f;अस्मत्स्वामिन्=N_m;स्वामिन्=N_m;स्वामिनी=N_f;पुत्र=N_m;पुत्री=N_f;प्रसाद=N_m;एतद्राज्य=N_n;राज्य=N_n;अश्मकेशसैन्य=N_n;सैन्य=N_n;भवानीसाहाय्य=N_n;देवी=N_f;देव=N_m;शक्ति=N_f;मानव=N_m;विग्रह=N_m;चित्त=N_n;मौल=N_m;प्रकृति=N_f;सुत=N_m;सुता=N_f;अभ्युदय=N_m;दान=N_n;मान=N_m;आदि=N_m;आवर्जन=N_n;वर्जन=N_n;विश्वास=N_m;विशेष=N_m;राजन्=N_m;राज्ञी=N_f;इन्द्र=N_m;अन्तरङ्ग=N_m;भृत्य=N_m;भृत्या=N_f;पुरुष=N_m;प्रीति=N_f;रहस्=N_n;मित्र=N_n;मित्र=N_m;वचस्=N_n;हस्त=N_m;पक्ष=N_m;तत्पक्ष=N_m;अन्तक=N_m;अतिथि=N_m;भवन=N_n;अन्तकातिथिभवन=N_n;अनन्ता=N_f;भय=N_n;प्रवृत्ति=N_f;वृत्ति=N_f;परिजन=N_m;जन=N_m;सुख=N_n;त्रिशूल=N_n;मैत्री=N_f;अस्मन्मुख=N_n;मुख=N_n;भवानीवर=N_m;वर=N_m;मनस्=N_n;वचन=N_n;वश=N_m;वृत्तान्त=N_m;प्रजा=N_f;प्रभु=N_m;कृपा=N_f;मूल्य=N_n;क्षमा=N_f;गमनागमनस्थल=N_n;स्थल=N_n;गमन=N_n;

APPENDIX-II

i) <u>Sample data of the POS Example base</u>

अज्ञात्वा=[AVK];अदत्त्वा=[AVK];धृत्वा=[AVK];नीचैःकृत्वा=[AVK];न्यग्भूत्वा=[AVK];परापृष्ठीभूत्वा=[AVK];पीत्वा=[AVK];पृष्ट्वा=[AVK];प्रीणयित्वा=[AVK];बाधित्वा=[AVK];भुक्त्वा=[AVK];भोजयित्वा=[AVK];मुक्त्वा=[AVK];योषित्वा=[AVK];लात्वा=[AVK];विनाभूत्वा=[AVK];शसित्वा=[AVK];शसित्वा=[AVK];शान्त्वा=[AVK];स्थापयित्वा=[AVK];स्थित्वा=[AVK];स्थिरित्वा=[AVK];स्नात्वा=[AVK];ह्रित्वा=[AVK];हुवा=[AVK];सदा=[AVKV];अन्यदा=[AVKV];इदा=[AVKV];एकदा=[AVKV];नित्यदा=[AVKV];सदा=[AVKV];सर्वदा=[AVKV];सर्वथा=[AVKV];वृथा=[AVKV];अन्यथा=[AVKV];अमुथा=[AVKV];अवृथा=[AVKV];इतरथा=[AVKV];इत्था=[AVKV];इमथा=[AVKV];उभयथा=[AVKV];ऊर्ध्वथा=[AVKV];ऋतुथा=[AVKV];एवथा=[AVKV];नामथा=[AVKV];प्रत्नथा=[AVKV];बहुथा=[AVKV];वृथा=[AVKV];अथवा=[AVC];अथोवा=[AVC];तद्वा=[AVC];दिवा=[AV];वा=[AVC];अकामतः=[AVKV];अकुतः=[AVKV];अग्रतः=[AVKV];अग्रतः=[AVKV];अज्ञानतः=[AVKV];अतः=[AVKV];अथातः=[AV];अधरतः=[AVKV];अधर्मतः=[AVKV];अनिमित्ततः=[AVKV];अनुरूपतः=[AVKV];अनुसारतः=[AVKV];अन्ततः=[AVKV];अन्यतः=[AVKV];अभितः=[AVKV];अभितः=[AVKV];अभिवाहृतः=[AVKV];अभीपतः=[AVKV];अभ्यन्तरतः=[AVKV];अमुतः=[AVKV];अर्जुनतः=[AVKV];अर्थतः=[AVKV];अवरतः=[AVKV];अवरार्धतः=[AVKV];अविधानतः=[AVKV];आगमनतः=[AVKV];

ii) <u>Sample data of the verb example base</u>

भवति=[P_laT_1.1:1];चेतति=[P_laT_1.1:27];खादति=[P_laT_1.1:28];मन्थति=[P_laT_1.1:29];नर्दति=[P_laT_1.1:30];गर्दति=[P_laT_1.1:31];तर्दति=[P_laT_1.1:32];कर्दति=[P_laT_1.1:33];अन्तति=[P_laT_1.1:34];इन्दति=[P_laT_1.1:35];निन्दति=[P_laT_1.1:36];नन्दति=[P_laT_1.1:37];चन्दति=[P_laT_1.1:38];क्रन्दति=[P_laT_1.1:39];क्लिन्दति=[P_laT_1.1:40];शुन्धति=[P_laT_1.1:41];फक्कति=[P_laT_1.1:50];तङ्कति=[P_laT_1.1:51];बुक्कति=[P_laT_1.1:52];शाखति=[P_laT_1.1:53];ओखति=[P_laT_1.1:54];उङ्खति=[P_laT_1.1:55];अङ्गति=[P_laT_1.1:56];शोचति=[P_laT_1.1:64];कुञ्चति=[P_laT_1.1:65];लुञ्चति=[P_laT_1.1:66];अञ्चति=[P_laT_1.1:67];गुञ्जति=[P_laT_1.1:68];अर्चति=[P_laT_1.1:69];वाञ्छति=[P_laT_1.1:70];मूर्च्छति=[P_laT_1.1:71];उञ्छति=[P_laT_1.1:72];कूजति=[P_laT_1.1:73];अर्जति=[P_laT_1.1:74];गर्जति=[P_laT_1.1:75];तर्जति=[P_laT_1.1:76];अजति=[P_laT_1.1:77];खञ्जति=[P_laT_1.1:78];एजति=[P_laT_1.1:79];क्षयति=[P_laT_1.1:80];व्रजति=[P_laT_1.1:81];अटति=[P_laT_1.1:90];रटति=[P_laT_1.1:91];नटति=[P_laT_1.1:92];लोटति=[P_laT_1.1:93];चेटति=[P_laT_1.1:94];अयति=[P_laT_1.1:95];मण्डति=[P_laT_1.1:96];स्फोटति=[P_laT_1.1:97];मुण्डति=[P_laT_1.1:98];पठति=[P_laT_1.1:99];कुण्ठति=[P_laT_1.1:100];क्रीडति=[P_laT_1.1:101];क्रामति=[P_laT_1.1:108];ईर्ष्यति=[P_laT_1.1:118];

APPENDIX-III

i) <u>Sample data of the Subanta example base</u>

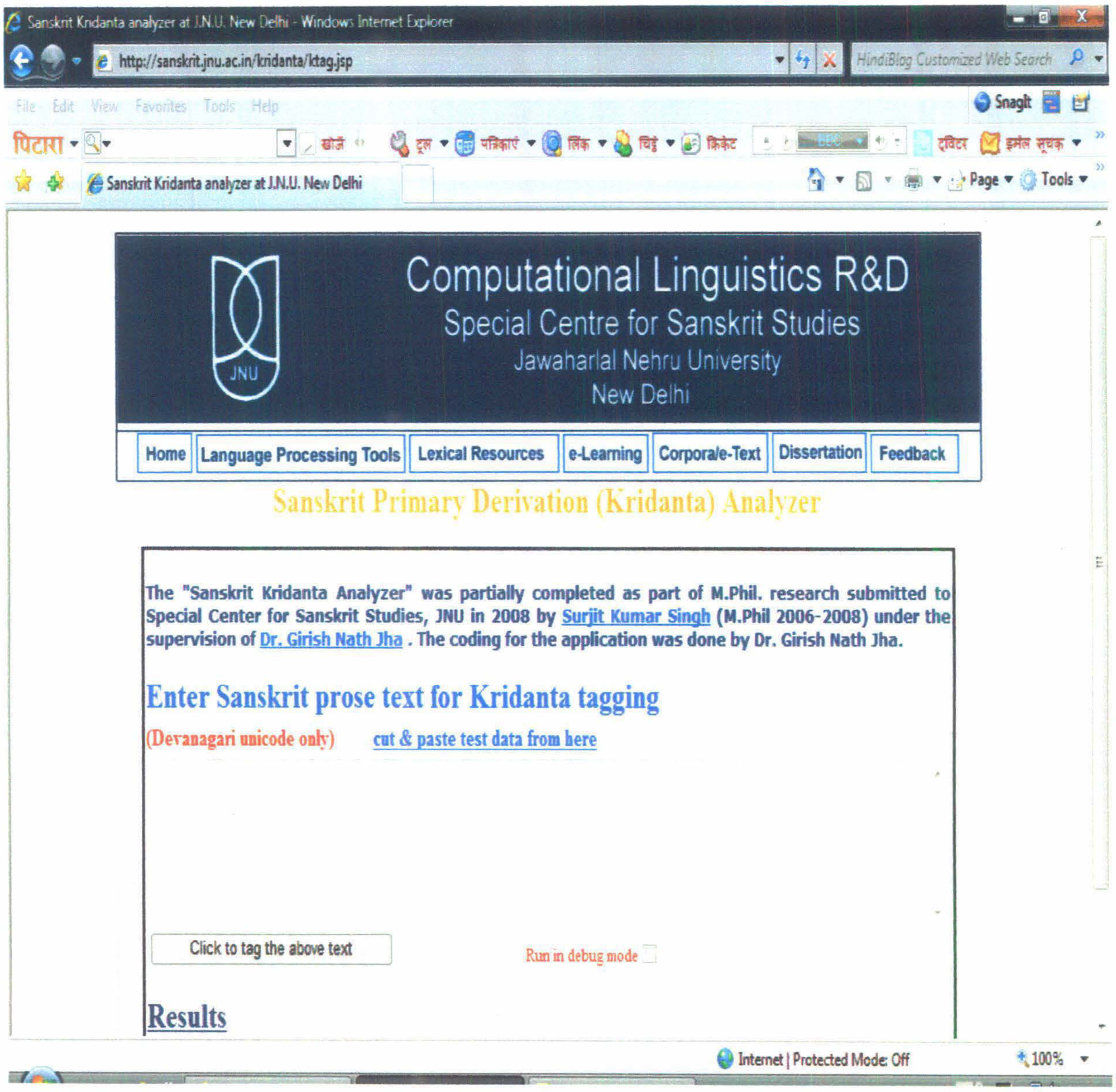
निर्जरेण=निर्जर+टा+3.1;ऋतुपर्णः=ऋतुपर्ण+सु+1.1;ना=नृ+सु+1.1;नरः=नर+सु+1.1/नृ+जस्+1.3;नरौ=नर+औ/औट्+1.2/2.2/नृ+औ/औट+1.2/2.2;नृन्=नृ+जस्3;सः=तद्+सु+1.1;स=तद्+सु+1.1;एतं=एतत+सुअम्,+1.1/2.1;नरम्=नर+अम्,+1.1/2.1नृ+अम्,+1.1/2.1;नरं=नर+अम्,+1.1/2.1,नृ+अम्,+1.1/2.1;प्रशामः=प्रशाम्+जस्/शस्/ङसि/ङस्+1.3/2.3/5.1/6.1;प्रशामम्=प्रशाम्+अम्+2.1;प्रशामं=प्रशाम्+अम्+2.1;प्रशामौ=प्रशाम्+औ/औट्+1.2/2.2;प्रशामा=प्रशाम्+टा+3.1;प्रशामे=प्रशाम्+ङे+4.1;प्रशामाम्=प्रशाम्+आम्+6.3;प्रदामः=प्रदाम्+जस्/शस्/ङसि/ङस्+1.3/2.3/5.1/6.1;प्रदामम्=प्रदाम्+अम्+2.1;प्रदामं=प्रदाम्+अम्+2.1;प्रदामौ=प्रदाम्+औ/औट्+1.2/2.2;प्रतामः=प्रताम्+जस्/शस्/ङसि/ङस्+1.3/2.3/5.1/6.1;प्रतामम्=प्रताम्+अम्+2.1;प्रतामं=प्रताम्+अम्+2.1;प्रतामौ=प्रताम्+औ/औट्+1.2/2.2;प्रकामः=प्रकाम्+जस्/शस्/ङसि/ङस्+1.3/2.3/5.1/6.1;प्रकामम्=प्रकाम्+अम्+2.1;प्रकामं=प्रकाम्+अम्+2.1;प्रकामौ=प्रकाम्+औ/औट्+1.2/2.2;सर्वः=सर्व+सु+1.1;सर्वौ=सर्व+औ1+2.2;सर्वे=सर्वा+जस+1.2/2.2(स्त्री.),सर्व+जस+1.3(पु.),सर्व+जस+1.2/2.2(नपु.);सर्वम्=सर्व+सु+1.1/2.1(नपु.),अम+2.1(पु.);सर्वं=सर्व+सु+1.1/2.1(नपु.),अम+2.1(पु.);सर्वान्=सर्व+शस+2.3;सर्वेण=सर्व+टा+3.1;न्रे=नृ+टा+3.1;सर्वाभ्याम्=सर्व+भ्याम+3.2/4.2/5.2(पु.),सर्वा+भ्याम+3.2/4.2/5.2(स्त्री.);सर्वाभ्यां=सर्व,सर्वा+भ्याम+3.2/4.2/5.2;सर्वैः=सर्व+भिस+3.3;सर्वस्मै=सर्व+ङे+4.1;न्रे=नृ+ङे+4.1;सर्वेभ्यः=सर्व+भ्यस+4.3/5.3;सर्वस्मात्=सर्व+ङसि+5.1;नुः=नृ+ङसि/ङस्+5.1/6.1;सर्वस्माद्=सर्व+ङसि+5.1;सर्वस्य=सर्व+ङस+6.1;सर्वयोः=सर्व,सर्वा+ओस+6.2/7.2;सर्वेषाम्=सर्व+आम+6.1;सर्वेषां=सर्व+आम+6.1;सर्वस्मिन्=सर्व+ङि+7.1;

ii) <u>Sample data of the Subanta rule base</u>

a=a;म्=+अम्+1.1/2.1;◌ः=+सु+1.1;◌ा=◌ा+सु+1.1;◌ाभ्याम्=+भ्याम्+3.2/4.2/5.2;◌ाभ्यां=+भ्याम्+3.2/4.2/5.2;भ्याम्=+भ्याम्+3.2/4.2/5.2;भ्यां=+भ्याम्+3.2/4.2/5.2;◌ेभ्यः=+भ्यस्+4.3/5.3;भ्यः=+भ्यस्+4.3/5.3;◌ानाम्=+आम्+6.3;◌ानां=+आम्+6.3;◌ाणाम्=+आम्+6.3;◌ाणां=+आम्+6.3;नाम्=+आम्+6.3;नाम्=+आम्+6.3;णाम्=+आम्+6.3;णां=+आम्+6.3;स्य=+स्यङस्+6.1;◌े=+ङि+7.1;◌ी=◌ी+सु+1.1;◌ी+सु+1.2.2वचन;णा=+टा+3.1;◌ौ=+औ/औट्+1.2/2.2;◌ा◌ः=+जस्+1.3;◌ाणि=+जस्+1.3/2.3;◌ानि=+जस्+1.3/2.3;◌ं=+अम्+2.1;◌ान्=+शस्+2.3;◌ेण=+टा+3.1;◌ेन=+टा+3.1;◌ै◌ः=+भिस्+3.3;◌ाय=+ङे+4.1;◌ेभ्यः=+भ्यस्+4.3/5.3;◌ात्=+ङसि+5.1;◌ाद्=+ङसि+5.1;स्य=+ङस्+6.1;योः=+ओस्+6.2/7.2;◌ानां=+आम्+6.3;◌ानाम्=+आम्+6.3;◌ाणां=+आम्+6.3;◌ाणाम्=+आम्+6.3;◌े=+ङि+7.1;◌ेषु=+सुप्+7.3;◌ेसु=+सुप्+7.3;◌ा=◌ा+सु+1.1;◌े=◌ा+औ/औट्+1.2/2.2;◌ा◌ा◌ः=◌ा+जस्+1.3;◌ा◌ं=◌ा+अम्+2.1;◌ाम्=◌ा+अम्+2.1;या=◌ा+टा+3.1;यै=+ङे+4.1;◌ायाः=◌ा+ङसि/ङस्+5.1/6.1;योः=+ओस्+6.2/7.2;◌ानां=+आम्+6.3;◌ानाम्=+आम्+6.3;◌ाणां=+आम्+6.3;◌ाणाम्=+आम्+6.3;◌ायां=◌ा+ङि+7.1;◌ायाम्=◌ा+ङि+7.1;◌ेषु=+सुप्+7.3;◌ेसु=+सुप्+7.3;◌ि◌ः=◌ि+सु+1.1;

APPENDIX-IV

Screen Shot of the Input screen


Sanskrit Kridanta analyzer at J.N.U. New Delhi - Windows Internet Explorer

http://sanskrit.jnu.ac.in/kridanta/ktag.jsp

Computational Linguistics R&D

Special Centre for Sanskrit Studies

Jawaharlal Nehru University

New Delhi

Home | Language Processing Tools | Lexical Resources | e-Learning | Corpora/e-Text | Dissertation | Feedback

Sanskrit Primary Derivation (Kridanta) Analyzer

The "Sanskrit Kridanta Analyzer" was partially completed as part of M.Phil. research submitted to Special Center for Sanskrit Studies, JNU in 2008 by Surjit Kumar Singh (M.Phil 2006-2008) under the supervision of Dr. Girish Nath Jha . The coding for the application was done by Dr. Girish Nath Jha.

Enter Sanskrit prose text for Kridanta tagging

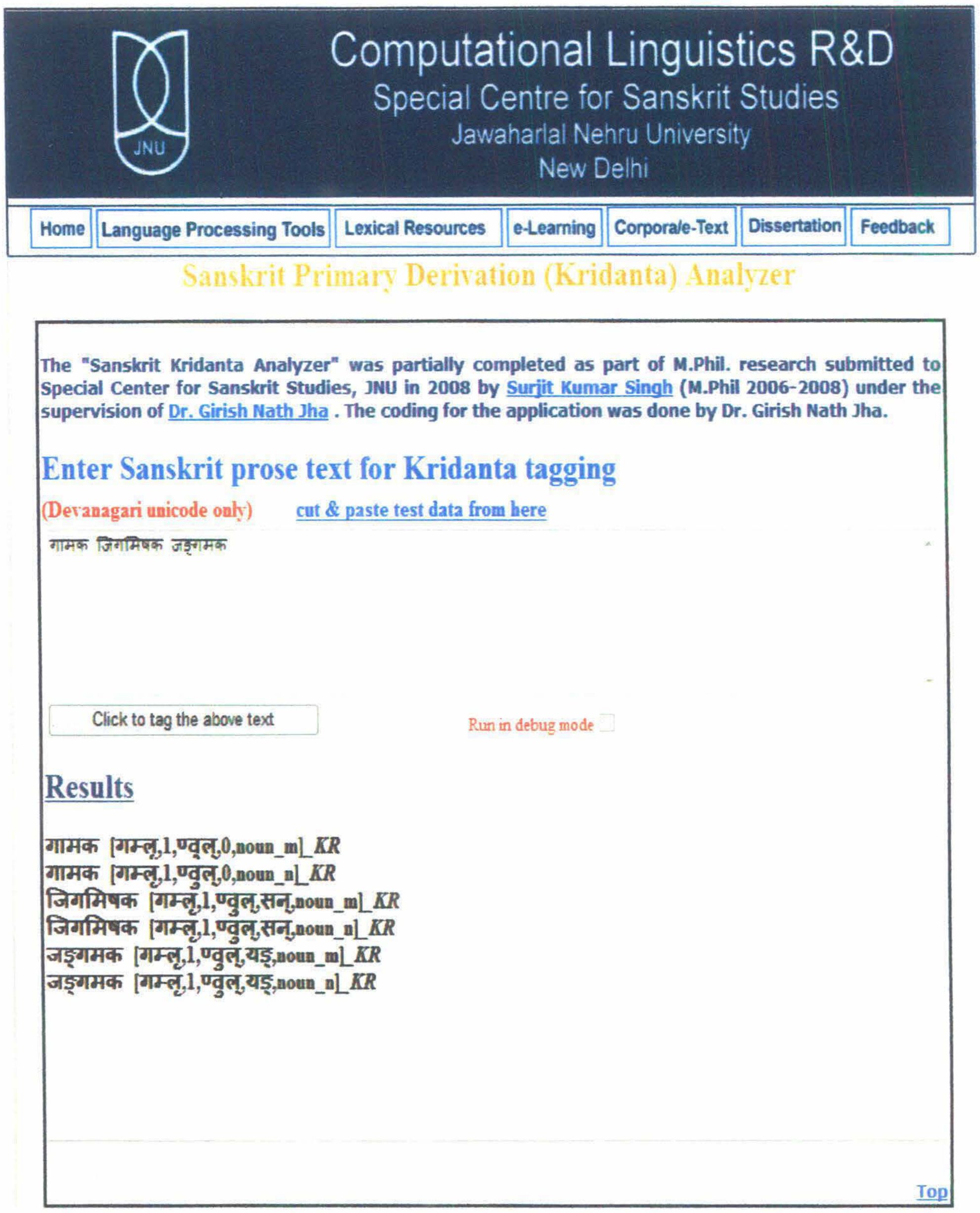
(Devanagari unicode only)   cut & paste test data from here

Click to tag the above text   Run in debug mode

Results

APPENDIX-V

Screen shot of the krdanta analysis


Computational Linguistics R&D
Special Centre for Sanskrit Studies
Jawaharlal Nehru University
New Delhi
Home
Language Processing Tools
Lexical Resources
e-Learning
Corpora/e-Text
Dissertation
Feedback

Sanskrit Primary Derivation (Kridanta) Analyzer

The "Sanskrit Kridanta Analyzer" was partially completed as part of M.Phil. research submitted to Special Center for Sanskrit Studies, JNU in 2008 by Surjit Kumar Singh (M.Phil 2006-2008) under the supervision of Dr. Girish Nath Jha . The coding for the application was done by Dr. Girish Nath Jha.

Enter Sanskrit prose text for Kridanta tagging

(Devanagari unicode only) cut & paste test data from here

गामक जिगमिषक जङ्गमक

Results

```
गामक [गम्लृ,1,ण्वुल्,0,noun_m]_KR
गामक [गम्लृ,1,ण्वुल्,0,noun_n]_KR
जिगमिषक [गम्लृ,1,ण्वुल्,सन्,noun_m]_KR
जिगमिषक [गम्लृ,1,ण्वुल्,सन्,noun_n]_KR
जङ्गमक [गम्लृ,1,ण्वुल्,यङ्,noun_m]_KR
जङ्गमक [गम्लृ,1,ण्वुल्,यङ्,noun_n]_KR
```

APPENDIX-VI

<u>Debugging process of the input word</u> गमयिष्यमाण

```
गमयिष्यमाण |गम्लृ,1,शानच्,णिच्,noun_m|_KR
गमयिष्यमाण |गम्लृ,1,शानच्,णिच्,noun_n|_KR



-----------------START OF Kridanta.preProcess()------------

input=गमयिष्यमाण
output=गमयिष्यमाण

-----------------END OF Kridanta.preProcess()------------



-----------------START OF Kridanta.checkFL()------------

input=गमयिष्यमाण
output=गमयिष्यमाण |गम्लृ,1,शानच्,णिच्,noun_m|_KR
गमयिष्यमाण |गम्लृ,1,शानच्,णिच्,noun_n|_KR


-----------------END OF Kridanta.checlFL()------------



---------------------------START OF preProcess()------------------------


---------------------------START OF checkPuct()------------------------

token=गमयिष्यमाण tkn length=10

start of 3+ char punc check tkn = गमयिष्यमाण c =ग
idx=-1
i=0
checking puncs inside words ts= ग
```

```
c =म
idx=-1
i=1
checking puncs inside words ts= गम
c =य
idx=-1
i=2
checking puncs inside words ts= गमय
c =ि
idx=-1
i=3
checking puncs inside words ts= गमयि
c =ष
idx=-1
i=4
checking puncs inside words ts= गमयिष
c =्
idx=-1
i=5
checking puncs inside words ts= गमयिष्
c =य
idx=-1
i=6
checking puncs inside words ts= गमयिष्य
c =म
idx=-1
i=7
checking puncs inside words ts= गमयिष्यम
c =ा
idx=-1
i=8
checking puncs inside words ts= गमयिष्यमा
c =ण
idx=-1
i=9
checking puncs inside words ts= गमयिष्यमाण
ts= गमयिष्यमाण
```

```
---------------------------END OF checkPunct()-----------------------

---------------------------END OF preProcess()-----------------------


---------------------------START OF FLT:checkLists()-----------------------


after tagtoken1 tkn=गमयिष्यमाण
FLT:tagToken():checking kridanta corpus tkn='गमयिष्यमाण'(10)
---------------------------START OF FLT:tagToken()-----------------------


FLT:tagToken():checking kridanta lexicon tkn='गमयिष्यमाण'(10)
---------------------------START OF FLT:tagToken()-----------------------


match found in lexicon... adding upasarga='' upa_viccheda=''
inside FLT:tagToken() tagged data=गमयिष्यमाण [गम्लृ,1,शानच्,णिच्,noun_m]_KR

match found in lexicon... adding upasarga='' upa_viccheda=''
inside FLT:tagToken() tagged data=गमयिष्यमाण [गम्लृ,1,शानच्,णिच्,noun_m]_KR
गमयिष्यमाण [गम्लृ,1,शानच्,णिच्,noun_n]_KR

upasarga=
upa_viccheda=

---------------------------END OF FLT:checkLists()-----------------------
```

Bibliography

Bibiliography

Books

- Sharma, Rama Nath, 2003, '*The Astādhyāyi of Pāṇini*', Munshiram Manoharlal Publishers Pvt. Ltd., Delhi.
- Pt.Brahmadatta jigyasu (ed.), 1998, '*Pāṇini-Astādhyāyi*', Ramlal kapoor trust, Sonipat.
- Joshi, Sivaram Dattatray & Roodbergen J. A. F., 1998, '*The Astādhyāyi of Pāṇini*', Sahitya Akademi, New Delhi.
- Mishra, Narayan (ed.), 1996, *'Kāśikā of Pt.Vāmana and Jayāditya'*, Chaukhamba Sanskrit Sansthan, Varanasi.
- Guru Prasad Shastri (ed). 1999, *'Vyakarana-Mahābhāṣya with Pradipoddhat ṭikā'* Pratibha Prakashan, Delhi.
- Vasu, Srisa Chandra (ed. & Translated into eng.), 1982, *'The Siddhānta Kaumudī of Bhaṭṭoji Dīkṣita, Vol. II*, MLBD. Delhi.
- Panashikar, Vasudev Lakshman Shastri (ed), 1994, '*Siddhāntakaumudī',* Chaukhamba Sanskrit Pratisthan, Delhi.
- Ballantyne, James R., 1967, '*Laghukaumudī of Varadarāja',* Chaukhamba Sanskrit Pratisthan, MLBD. Delhi.
- Pandey, G.D., (ed.), 2003, *'Vaiyākaraṇa Siddhāntakaumudī'*, Chaukhamba Surbharati Prakashana, Varanasi.
- Shastri, Bhimasena, 2001, *'Laghu-Siddhānta-Kaumudī (kṛdanta-kāraka-prakaraṇam)',*Bhaimi Prakashan, Delhi.
- Choudhary, Ramvilas, 2002, *'Vaiyākaraṇa Siddhāntakaumudī (Pūrva-kṛdanta-prakarṇam & Saṁjñāvivecanam',*MLBD.,Delhi.
- Dixit, Pushpa , 2004, *'Astādhyāyi Sahajabodha Vol. III Kṛdanta Prakaraṇam',* Pāṇinīya Sodha Sansthan, Bilaspaur.
- Mishra, Gopabandhu, 2007, *'Kṛt-Pratyayaviślesana',* Choukhamba Vidyabhavana, Varanasi.

- Mishra, Azad, 1989, *'Sanskrit ke pratyao kā bhāṣāśāstrīya adhyayana',* Madhukar prakashan, Lucknow.
- Harakare, Gunderava, 1983, *'Pratyayakośah',* Usmania Visvavidyalaya, Haidarabad.
- Vina, Sharma, 1996, *'Krt-pratyayavimarśa',* Viidyapuri, Patna.
- Shastri, Ramasubrahmanya, 1989, *'Krdantarūpamālā',* Sanskrit Education Society, Madras.
- Shastri Charu Deva, 1991, *'Pāṇini : Re-interpreted'* Motilal Banarasidass, Delhi.
- Cardona, George, *'Pāṇini: his work and its traditions',* Motilalal banarasidass, New Delhi.
- Cardona, George, 1999, *'Recent Research in Pāṇinīan Studies',* Motilal Banarasidass, New Dalhi.
- Kapoor Kapil, 2005, *'Dimensions of Pāṇini Grammar: the Indian Grammatical System'*, D.K. Printworld (P) Ltd., New Delhi.
- Iyer, K.A.Subramania, 1969, *'Bhartrhari : A study of the vākyapadīya in the light of ancient commentaries'* Deccan College, Poona.
- Stall, J. F., 1985, *'A Reader on Sanskrit Grammarians'*, Motilal Banarsidas, Delhi.
- Singh, Jag Deva, 1991, *'Pāṇini: His Description of Sanskrit (An Analytical Study of the Aṣtādhyāyī)'* , Munshiram Manoharlal, Delhi.
- Fromkin & Rodman, 2003, *'An Introduction to Language'*, Thomson Wadsworth.
- Kale, M.R., 1972, *'A Higher Sanskrit grammar'* , Motilal Banarasidas, Delhi.
- Katre, S.M., 1968, *'Dictionary of Pāṇini'*, Deccan College, Poona.
- Abhyankar, K.V., 1961, *'A Dictionary of Sanskrit Grammar'*, Gaekwad's Oriental Series, Baroda.
- Muller, F. Max, 1983, '*A Sanskrit grammar*' Asian Educational Services, Delhi.
- Whitney, William Dwight, 1983, '*Sanskrit Grammar*', MLBD, Delhi.
- Williams, Monier, '*A Practical Grammar of the Sanskrit Language*', Clarendon Press, Oxford
- Bhandarkar, R.G., 1924, '*First book of Sanskrit*', Radhabai Atmaram Sagoon, Bombay.

- Bhandarkar, R.G., 1924, '*Second book of Sanskrit*' Radhabai Atmaram Sagoon, Bombay.
- Wilson, H. H., 1841, '*An Introduction to the Grammar of the Sanskrit Language*', J. Madden & Co., London.
- Kielhorn, F., 1970, '*Grammar for Sanskrit Language*' Chowkhamba Sanskrit Series office, Varanasi.
- Macdonell, A. A., 1997, '*A Sanskrit Grammar for Students*', D. K. Printworld (P) Ltd., New Delhi.
- Nautiyal, Chakradhar Hans, 1995, *'Brhada-anuvād-candrikā',* Motilal Banarasidass, Delhi.
- Bhola Nath, Tiwari, *'Hindi-Bhāṣā',* Kitab Mahal, Delhi.
- Bharati, Akshar, Vineet Chaitanya and Rajeev Sangal, 1995, *'Natural Language Processing: A Paninian Perspective',* Prentice-Hall of India, New Delhi.
- Jurafsky, Daniel & Martin, 2005, *'Speech and languages processing',* Pearson Education Pvt.Ltd., Singapore.
- Troast, Harald. 2003, '*Morphology', in 'The Oxford Handbook of Computational Linguistics'*, Edited by Ruslan Mitkov, Oxford University press, New York, pp. 25-47.
- Bakharia, Aneesha. 2001, *'Java Server Pages',* Prentice Hall of India Private Limited, New Delhi.
- Russell, Joseph P. 2002, '*Java Programming*', Prentice Hall of India Private Limited, New Delhi.
- Date, C.J., 1987, *'Introduction to Database Systems',* Addison-Wesley, Reading.
- Reyle, U. and C. Rohrer (eds.), 1988, *'Natural Language Parsing and Linguistic Theories',* D. Reidel, Dordrecht.

Articles and Papers

- Bharati, Akshara, Amba Kulkarni and V. Sheeba, *'Building a wide Coverage Sanskrit Morphological Analyzer: A Practical approach*, MSPIL-06.
- Saha, Saranya, 'Parsing and generation of verbal nouns and other verb-derivatives in OIA', SIMPLE-04.
- Cardona, George, 2004, *'Some Questions on Pāṇini's Derivational system'* In SPLASH proc. of iSTRANS, pp. 3.
- G.V. Singh, Girish Nath Jha, '*Indian theory of knowledge: an AI perspective*' proc. of seminar, ASR, Melkote, Mysore, 1994
- Jha Girish N, 1994, '*Indian theory of knowledge: an AI perspective*' (proc. of national seminar on "Interface Mechanisms in Shastras and Computer Science", Academy of Sanskrit Research, Melkote, Mysore, April, 1994)
- Jha Girish N, 1995, '*Proposing a computational system for Nominal Inflectional Morphology in Sanskrit*' (Proc. of national seminar on "Reorganization of Sanskrit Shastras with a view to prepare their computational database", January, 1995)
- Jha Girish Nath, Mishra, S K, Chandrashekar R, Subash, August, 2005, '*Developing a Sanskrit Analysis System for Machine Translation*' presented, at the National Seminar on Translation Today: state and issues, Deptt. of Linguistics, University of Kerala, Trivandrum.
- Jha Girish Nath, October 2003, '*A Prolog Analyzer/Generator for Sanskrit Subanta Padas*', Language in India, Volume 3: 11.
- Jha Girish Nath, November, 2005 '*Language Technology in India: A survey*' Issue of C.S.I. magazine
- Jha Girish Nath, February 2004, '*The System of Panini*', Language in India, volume 4:2

- Jha Girish Nath, March 2004, '*Generating nominal inflectional morphology in Sanskrit*' SIMPLE 04, IIT-Kharagpur Lecture Compendium, Shyama Printing Works, Kharagpur, WB,
- Jha, Girish Nath, December, 2003, '*Current trends in Indian languages technology*', Langauge In India, Volume December.
- Jha, Girish Nath. 2007, '*Introduction to Computational Morphology*', Lecture delivered on 5 January 2007 at CDAC, Noida.
- Jha, Girish Nath. '*Regional & linguistic perspective on internationalization: the case of Hindi/Sanskrit*', 2007.
- Singh, Surjit Kumar & Girish Nath Jha. 2006, '*Strategies for Identifying and Processing Derived Nouns in Sanskrit*', In the Souvenir Abstracts of 28th AICL, BHU, Varanasi, p. 131.
- Bharati Akshar, Amba P. Kulkarni Vineet Chaitanya, 1996, '*Challenges in developing word analyzers for Indian languages*', Presented at Workshop on Morphology, CIEFL, Hyderabad.
- Bharati, A., Sangal R., 1990, '*A karaka based approach to parsing of Indian languages*', proc of the 13 th COLING vol 3, pp 30-35, Finland.
- Bharati, Akshar and Rajeev Sangal, '*Parsing free word order languages using the Paninian framework*', In ACL93: Proc.of Annual Meeting of Association for Computational Linguistics, Association for Computational Linguistics, New York, 1993.

- Bharati, Akshar, Vineet Chaitanya and Rajeev Sangal, '*A computational framework for Indian languages*', Technical Report TRCS-90-100, Dept. of CSE, IIT Kanpur, July 1990b. (Course Notes for Intensive Course on NLP for Linguists, Vol.1)
- Huet, Gerard, '*Towards Computational Processing of Sanskrit*'
- Bhate, Saroja and Subhash Kak, 1993, '*Pāṇini's Grammar and Computer Science*' Annals of the Bhandarkar Oriental Research Institute, vol. 72, pp. 79-94.
- Kapoor, Kapil, 1996. '*Panini's derivation system as a processing model*' (to appear in the proc. of "A Symposium on Machine Aids for Translation and Communication, 11-12 April, School of Computer & Systems Sciences, J.N.U. New Delhi, 1996)
- Kiparsky, P. and Stall, J. F., 1969, '*Syntactic and Semantic Relation in Panini*' (Foundations of Language, Vol.5, 83-117).
- R.M.K. Sinha, 1989, '*A Sanskrit based Word-expert model for machine translation among Indian languages*', Proc. of workshop on Computer Processing of Asian Languages, Asian Institute of Technology, Bangkok, Thailand, Sept.26-28, 1989, pp. 82-91.12.
- Ramakrishnamacharyulu, K.V., '*Paninian Linguistics and Computational Linguistics*', Samvit, Series no. 27. Pp. 52-62, Academy of Sanskrit Research, Melkote, Karnataka (India), 1993.

Thesis and Dissertations

- Jha Girish N, 1993, '*Morphology of Sanskrit Case Affixes: A Computational analysis*' Dissertation of M.Phil submitted to Jawaharlal Nehru University, New Delhi-110067.
- Chandrashekhara, R., 2006, '*POS Tagging for Sanskrit*', submitted for Ph.D degree at SCSS, JNU.
- Chandra, Subash, 2006, '*Machine Recognition and Morphological Analysis of Subanta-padas*', submitted for M.Phil degree at SCSS, JNU.
- Mishra, Sudhir Kumar, 2007, '*Sanskrit Karaka Analyzer for Machine Translation*', submitted for Ph.D. degree at SCSS, JNU.
- Bhadra, Manji, 2007, '*Computational Analysis of Gender in Sanskrit Noun Phrases for Machine Translation*', submitted for M.Phil degree at SCSS, JNU.
- Kumar, Sachin, 2007, '*Sandhi Splitter and Analyzer for Sanskrit' (with special reference to aC sandhi)*, submitted for M.Phil degree at SCSS, JNU.
- Agrawal, Muktanand, 2007, '*Computational Identification and Analysis of Sanskrit Verb-forms of bhvādigana*', submitted for M.Phil degree at SCSS, JNU.

Web References

- Huet, Gerard, *http://sanskrit.inria.fr/* (accessed: 10 April, 2008).
- Peter M. Scharf and Malcolm D. Hyman, *http://sanskritlibrary.org/morph/* (accessed: 18 April, 2007).
- Peter M. Scharf and Malcolm D. Hyman, *http://sanskritlibrary.org/* (accessed: 18 April, 2008).
- *http://en.wikipedia.org/wiki/Clay_Sanskrit_Library* (accessed: 2 July 2008).
- John Clay, *http://www.claysanskritlibrary.org/* (accessed: 2 July 2008).

- Academy of Sanskrit Research, Melkote, *http://www.sanskritacademy.org/About.htm* (accessed: 22 April 2008).
- RSV Tirupati, *http://rsvidyapeetha.ac.in* and *http://www.sansknet.org* (accessed: 22 April 2008).
- RCILTS, JNU – Achievements, *http://tdil.mit.gov.in/SanskritJapaneseChinese-JNUJuly03.pdf* (accessed: 25 April 2008).
- Computational Linguistic R&D, J.N.U., *http://sanskrit.jnu.ac.in/index.jsp* (accessed: 2 May 2008).
- RCILTS, School of Computer & System Science, *http://rcilts.jnu.ac.in* JNU, New Delhi. (Accessed: 20 April 2008).
- RCILTS, Utkal University, *http://www.ilts-utkal.org/nlppage.htm* (accessed: 15 April 2008).
- Desika, *http://tdil.mit.gov.in/download/Desika.htm* (accessed: 10 May, 2008).
- Anusaaraka, *http://www.iiit.net/ltrc/Anusaaraka/anu_home.html* (accessed: 10April, 2008).
- Wikipedia, *http://en.wikipedia.org/wiki/Natural_language_processing* (accessed: 15 April 2008).
- *http://www.acroterion.ca/Morphological_Analysis.html* (accessed: 15 April 2008).
- Oflazer,Kemal,*http://folli.loria.fr/cds/2006/courses/Oflazer.ComputationalMorphology.pdf* (accessed: 15 April 2008).
- *http://tdil.mit.gov.in/languagetechnologyresourcesapril03.pdf* (accessed: 20 April 2008).
- *http://www.sas.upenn.edu/~vasur/project.html* (accessed: 18 April 2008).

- *http://www.sil.org/pckimmo/* (accessed: 18 April 2008).
- *http://www.comp.lancs.ac.uk/ucrel/claws* (accessed: 20 April 2008).
- C-DAC, *http://www.cdac.in/html/ihg/activity.asp* (accessed: 20 April 2008).
- Language Processing Tools: TDIL website, *http://tdil.mit.gov.in/nlptools/ach-nlptools.htm* (accessed: 20 April 2008).
- Shakti, LTRC, IIIT, Hyderabad, *http://www.iiit.net/ltrc/index.html* (accessed: 25 April 2008).
- IIT, Bombay, *http://www.cse.iitb.ac.in* (accessed: 25 April 2008).
- *http://www.cfilt.iitb.ac.in/wordnet/webhwn/wn.php* (accessed: 25 April 2008).
- AU-KBC Research Centre - *http://www.au-kbc.org/frameresearch.html* (accessed: 25 April 2008).
- *http://www.languageinindia.com/feb2004/panini.html* (accessed: 2 May 2008).
- Shabdabodha, TDIL, Gov. of India, *http://tdil.mit.gov.in/download/Shabdabodha.html* (accessed: 25 April 2008).
- Pen to Paper, The House Magazine of C-DAC, *http://www.cdacindia.com/html/adp/mactrans.asp* (accessed: June 15 May, 2008).
- Pen to Paper, The House Magazine of C-DAC, *http://www.cdacindia.com/html/connect/3q2000/art10a.htm#* (accessed: 15 May, 2008).
- Baraha, *http://www.baraha.com/BarahaIME.htm* (accessed: 6 July 2008).
- The Web server, Apache Tomcat, *http://www.apache.org/* (accessed: 5 July 2008).
- Java Server Pages, *http://java.sun.com/products/jsp/* (accessed: 5 July, 2008).
- Java, Servlet, *http://java.sun.com/products/servlet/* (accessed: 5 July, 2008).