

**Computational Analysis of Gender in Sanskrit
Noun Phrases for Machine Translation**

Dissertation submitted to Jawaharlal Nehru University

In partial fulfillment of the requirements

For the award of the

Degree of

MASTER OF PHILOSOPHY

MANJI BHADRA



Special Centre for Sanskrit Studies

Jawaharlal Nehru University

New Delhi-110067

INDIA

2007



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

July 30, 2007

DECLARATION

I declare that the dissertation entitled “**Computational Analysis of Gender in Sanskrit Noun Phrases for Machine Translation**” 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.

Manji Bhadra
(Manji Bhadra)




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

July 30, 2007

C E R T I F I C A T E


This dissertation entitled “**Computational Analysis of Gender in Sanskrit Noun Phrases for Machine Translation**” submitted by **Manji Bhadra** 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.

Dr. C. Upender Rao
(Chairperson)

 **Dr. C. Upender Rao**
Chairperson
Special Centre for Sanskrit Studies
Jawaharlal Nehru University
New Delhi - 110067

Dr. Girish Nath Jha

(Supervisor)

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

Acknowledgement

Writing an acknowledgement is difficult as the text can never do full justice to all the names and faces that crowd to the alleys of memory. At the very beginning, I extend my thanks to my guide, Dr.Girish Nath Jha. Without his help, constant motivation and valuable suggestions, this work would not have been complete. Despite his busy schedule he helped me with programming. His invaluable intellectual inputs, scholarly guidance, insightful suggestions, and bountiful patience helped this work to be completed successfully.

I also express my heartfelt thanks and gratitude for the other faculty members of The Special Center of Sanskrit Studies Jawaharlal Nehru University, New Delhi, for the encouragement they provided me. My sincere thanks also goes out to the staff of the central library of JNU, especially to Mallik Sir, who helped me to find necessary materials for work.

I am really grateful to my roommate Geeta who has adjusted with my haphazard and disorganized lifestyle throughout M. Phil. My special thanks to my classmates, Muktanand and Sachin who always gave me valuable suggestions and encouragement for the entire M. Phil course. I express my sincere thanks to my seniors and juniors for helping me systematize and structure the ideas for my work and to build the lexical resource.

I shall always remember my friends, Irani, Debosree, Nayanee who have made me my stay away from home an enjoyable one. I would never forget my friends and relatives who gave me support.

I must remember Somdatta and Srimonda who took the pain of reading and editing the whole dissertation.

Without the constant help of my parents, I could not have reached this stage in my life. Thank You Ma and Baba, thank you once again.

Having been privileged in receiving such support, I am solely responsible for all the errors and omissions in this work.

Transliteration key used in the dissertation

अ	=	a
आ	=	ā
इ	=	i
ई	=	ī
उ	=	u
ऊ	=	ū
ऋ	=	r̥
ॠ	=	r̄
ऌ	=	l̥
ॡ	=	l̄
ए	=	e
ऐ	=	ai
ओ	=	o
औ	=	au
अं	=	am̐
अः	=	ḥ
क	=	k
ख	=	kh
ग	=	g
घ	=	gh
ङ	=	ṅ
च	=	c
छ	=	ch
ज	=	j
झ	=	jh
ञ	=	ñ
ट	=	ṭ
ठ	=	ṭh
ड	=	ḍ
ढ	=	ḍh
ण	=	ṇ
त	=	t
थ	=	th
द	=	d
ध	=	dh
न	=	n
प	=	p
फ	=	ph
ब	=	b
भ	=	bh
म	=	m
य	=	y
र	=	r
ल	=	l
व	=	v

प्र
व
स
ह
क्ष
त्र
प्र
५

=
=
=
=
=
=
=

६
१
९
ह
क
त्र
५

Contents

Acknowledgements	i
Transliteration scheme used for Devanāgarī	ii-iii
Contents	iv-v
Introduction	1-5
Chapter1: Computational Morphology and Grammatical Gender in Indian Languages	
1.1 Computational Morphology	6-9
1.2 Existing research in this area	10-11
1.3 R&D for other Indian languages	11-18
1.4. Machine translation systems	18-22
1.5 Text-processors for Indian language	22-25
1.6 Jawaharlal Neheru University	25-26
1.7 Morphological analyzer for Non -Indian languages	27-34
1.8 Development by private enterprises	34- 41
Chapter2: Gender in Sanskrit Grammar	
2.0 Gender	42-43
2.1 Grammatical gender in Sanskrit	43-47
2.1.1 Gender in Aṣṭādhyāyī	47-60
2.1.1.1 Gender rules in Liṅgānuśāsana	60-67
2.2 Gender in Hindi	67-72
Chapter3: Gender recognition and analysis in Sanskrit: Simple Noun Phrases	
3.0 Noun Phrase	73-74
3.1 Simple NPs in Sanskrit	74-76
3.2 Gender Agreement in Sanskrit	76-78
3.3 Gender recognition and analysis in simple NPs	79-84

3.4 Structure of lexicon

84-86

**Chapter4: Gender Recognizer and Analyzer System for Sanskrit
(GRASS)**

4.0 GRASS

87

4.1 The GRASS Model

88

4.1.1 How does GRASS work

89-90

4.1.1.1 Sample code

90-91

4.1.2 Lexical resources

91-92

4.2 The environment on which system is going to run

92-93

4.3 Result Analysis

93-94

Conclusion

95-98

Appendices

Sample of Lexicon

99-100

Sample of Verb Lists

100-101

Debug mode

101-102

Bibliography

103-109

Introduction

Natural language processing (NLP) is one of the ways to establish interaction between humans and computers. Machine translation is one of the applications of NLP. In a linguistically diverse country like India, linguistic variations may cause hindrance to communication. Cost effective automatic translations from Sanskrit to Indian languages are highly desirable. Attempts are made towards this direction by Dr. Amba Kulkarni, Sanskrit Research Academy, Melkote through their Semusi project and also the Desikā system of CDAC has touched upon the issue. These researches are done at the morphological level. At the stage of machine translation it is necessary to have the idea of collocation gender at the level of noun phrase. Many Indian languages have gender agreement between substantive and adjective and also have gender agreement between the verb and the agent in a sentence. Without correct gender information of the source language and also of the target language the whole translation will be wrong.

The scope of this dissertation is to comprehensively analyze simple Sanskrit Noun phrases (NP) with a hierarchy of increasing complexity, and develop an online system which will recognize and analyse the gender of the individual NP and higher NPs by – morphological analysis and lexical interfacing and cross linkages with and among linguistic resources. The immediate application of such a system will be to test grammaticality (in terms of gender agreement) of Sanskrit sentences. The system can also be used as a sub-component in a larger NL system for Sanskrit. Finally, this system will be critical to any Sanskrit – Indian language Machine Translation application or research.

The objectives of this dissertation are,

- to evolve a mechanism to figure out gender from *prātipadika* from *subanta* padas,
- to develop an extensive example base which can be used to identify gender of Sanskrit words,
- to develop rules to identify gender and as well as to separate multiple noun phrases in a sentence,

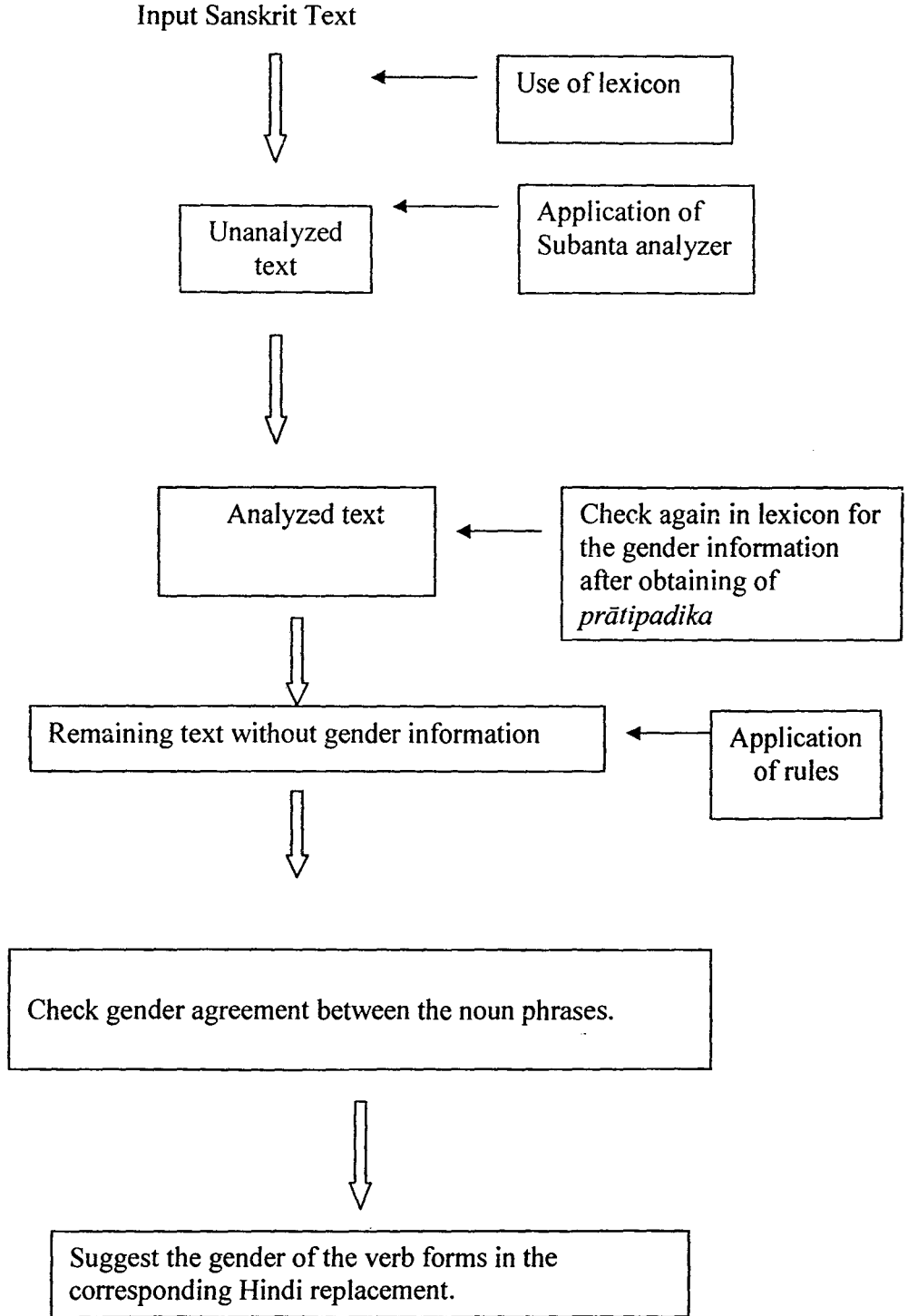
- to develop rules to check the gender agreement in a noun phrase, depending on Pāinian rules,
- to develop a mechanism that could suggest the collocation gender of a Sanskrit noun phrase in respect to a target language, for example, Hindi.
- to adapt Monier Williams Sanskrit Digital Dictionary (MWSDD) of Louis Bontes for analytical purpose,
- to adapt available e-corpora and customize them for gender analysis purpose,
- to build a servlet based online Java engine which will consult the rule base, example base and the linguistic resources to analyze gender in a Sanskrit text and to be on identical platform with the other application modules of Sanskrit analysis

For this Research and Development (R&D), the methodology of computational Sanskrit and software engineering has been used. This R&D is based on a hybrid approach of rule base and example base. The study consists of 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 the Pāinian Aṣṭādhyāyī, Lingānuśāsana, Siddhāntakaumudī (SK) of Bhaṭṭojidīkṣita, Kāśikāvṛtti of Vāmana and Jayāditya, Mahābhāṣya (MB) of Patanjali, Vākyapadīya of Bṛṅhari, Amarakoṣa of Amarasimha, adapted MWSDD by Louis Bontes and adapted and customized e-corpora. Secondary materials include several books of grammar, published articles and information on the internet.

As part of the research, various linguistic resources were developed and adapted according to the need of the system. To build a corpus of Sanskrit words, a lexicon of place name, nouns was developed and MWSDD was adapted to *Devanāgarī* UTF-8. A verb and indeclinable database is also adapted to exclude the *tinanta* (verb form) and indeclinable (*avyaya*) of processing. An example base of prātipadikas which appear in Lingānuśana are developed with adequate gender information for both Sanskrit and Hindi. Another example base of words of Amarkoṣa are also developed with gender information.

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

The basic design of the system is given below



The dissertation is divided into four chapters.

- The First Chapter discusses about computational morphology and the place of gender in morphology. It also discusses about the necessity of gender analysis in Sanskrit language and about the importance of collational gender of noun phrase of the source language and also of the target language in Machine translation. It also consists of a brief survey of gender analyzer as well as morphological analyzer of especially Sanskrit and other Indian Languages. In this chapter there are also surveys of the major projects related to morphological analysis for other Indo European languages.
- The Second chapter discusses the gender in Sanskrit. It consists of the theories of what is to be considered as grammatical gender in Sanskrit, based on Patanjali's Mahābhāṣya and Baṭṭhari's Vākyapadīya. In this chapter there are discussions about the rules for determining the gender of Sanskrit words based on Aṣṭādhyāyī and Liṅgānuś āsana. In Sanskrit, words are formed after adding different affixes. Feminine affixes are used after nominal stem to denote feminine. A list of feminine suffixes and the examples of those words are given in this chapter. At the end of this chapter, there is a brief discussion that how gender determines the changes in agreement in Hindi sentences, and how the gender is important while translating from source language (here it is Sanskrit) to target language (here it is Hindi).
- The Third chapter contains a discussion about Sanskrit noun phrases .The characteristics of simple Sanskrit NPs and the basic structure of them are described here. This chapter consists of how the gender agreements are taking place among the various components of NPs. It also discusses the basic algorithm of the gender analyzer system. The rules which are adapted to figure out gender from Sanskrit text are also described here in details. This chapter also has a list of corpora which are used to test the system.

- The Fourth chapter discusses the implementation aspects of the system, such as the front end, Java objects, databases, linguistic resources (corpus and rule bases and example bases), how they work and what the basic requirements of the system are.

The limitations of the system and its implications for future research have been summarized in the concluding part of the dissertation. A portable CD has also been enclosed with the dissertation which comprises the sample data of each linguistic resources and a screenshot of the interface with the URL of the system. The system is already accessible at <http://sanskrit.jnu.ac.in>

Chapter1: Computational Morphology and Grammatical Gender in Indian Languages

1.1 Computational Morphology

Computational Morphology (CM) is concerned with lexical analysis and generation using computational techniques and algorithms. Indian languages are rich in morphology and relatively word-order free. Consequently, words have a larger role to play in Indian languages. In the Indian context, the building of morphological analyzers is critical. Natural languages contain an open list of words and which changes dynamically. However, this infinity of words may have been produced from a finite collection of smaller units comprising of roots, stems and affixes. The task of CM therefore is to account for the underlying mechanism responsible for such processes in such a manner, that cross-linguistic generalizations can be done and language neutral models be evolved.

Grammatical gender is a morphological category associated with the expression of gender through inflection or agreement. In Sanskrit, except verb and *avyaya*, every word has gender and there is an intra-phrasal gender agreement between compatible non verb categories and also between non verb and *kṛdanta* verbs.

In most modern Indian languages, grammatical gender plays an important role in sentence formation, for example in Hindi, Marathi, etc. So while translating the Sanskrit text in these languages it is necessary to keep track of gender of the words and their appropriate gender agreement. In absence of a correct gender analysis of Sanskrit NPs, the target language translations will be wrong. For example –

1) Sanskrit:

{ ([sundaraḥ]ADJ_MAS [bālakah]N_MAS ([sundarī]ADJ_FEM [bālikā]N_FEM) [c]CONJ) NP ([āgachataḥ]V)VP }

(NP --NUM--VP)AGR

2) Hindi:

{ ([sundara]ADJ_MAS [laḍakā]N_MAS [aur]CONJ ([sunardar]ADJ_FEM [laḍakī]N_FEM)) NP ([āte ha:ī]V)VP }S
(NP ----NUM_GEN----VP)AGR

Comparing (1) and (2), we can see that Hindi requires gender information in addition to the number information. Therefore from a sentence like (1) where gender of each noun phrases in the conjunct noun phrase has to be figured, and then a collocational gender has to be assigned to the NP keeping in mind the rules of the target language (Hindi in this case).

Therefore, the intra-phrasal gender agreement (within a phrase, for example between adjective and noun) and long-distance or inter-phrasal gender agreement (across phrases, for example between a noun phrase and a verb phrase) in Indian languages has been very important for natural language analysis engines for obvious reasons. As shown in (1) and (2), an incorrect recognition of gender marking in the head noun can have disastrous results. It can lead to a wrong understanding of the collocation gender of the noun phrase and then to an ungrammatical sentence due to invalid subject-verb agreement. While latter is not true of Sanskrit and many eastern Indo Aryan languages like Maithili, Bangla, Assamese and Oriya; most other Indian languages are very strict about gender agreement at the level of verbs. For Sanskrit, the gender agreement at the noun phrase level is critical, and the language enforces this agreement through what we call 'intra-phrasal constraints'.

One of the important goals CM R & D in the Indian context should be, to understand these constraints and correctly formalize them for machines so that any larger natural language system like a translation system can give correct translation in the target language. Therefore, such computer systems are in great demand not only for Sanskrit but for other Indian languages as well. Unfortunately none such system exists for Sanskrit till date.

1.1.1 Goals and methods in CM

The most basic task in computational morphology is to take a string of characters or phonemes as input and deliver an analysis as output or generate new words. The problem of recognizing that 'foxes' breaks down in two morphemes, 'fox' and '-es' is an example of morphological parsing. In the domain of NLP, this parsing can be morphological, syntactic, semantic, and pragmatic, in the form of a string, or a tree, or a network. In the domain of information retrieval, stemmers are used to reduce as many related words and word forms as possible to a common canonical word. In Chinese, Japanese or Korean, words in a sentence are not separated by blanks or punctuation marks. Morphological analysis is used to perform the task of automatic word separation. In Japanese language, many input systems use kana- kanji character conversion. The whole text is typed in kana character and the relevant portions are subsequently converted to kanji. This mapping from kana to kanji is quite ambiguous. Combinations of statistical and morphological methods are applied to solve such tasks. Another application of computational morphology is lemmatization. It is the process of finding the corresponding dictionary form for a given input word. Broadly the applications can be divided into two sections, analysis and word builders. Analysis can be divided into taggers, chunkers, word breaker or lemmatizer. Paradigm generator comes under word builders.

1.1.2 Methods followed so far

While performing different kinds of tasks, different methods are used in computational linguistics. These are rule based string processing techniques, as well as finite state techniques.

1.1.2.1 Cut and paste method

Cut and paste is a very popular method in computational linguistics. Here canonical form is obtained by removing and concatenating morphemes at the end of a string. The best known ancestor of these systems dates back to the 1960s¹.

Another system known as Morphogen (Petheroudakis, 1991) is a commercial toolkit for creating sophisticated cut and paste analyzers². 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 morpheme³

1.1.2.2 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 in early 80s. Rules of morphological alternations could be implemented using FSTs as a finite state network (Johnson 1972, Kaplan and Kay 1994)⁴. First practical application of this model appeared in the 90s (Koskenniemi 83, Karttunen 1993, Antworth 1990, Karttunen and Beesley 1992, Ritchie, Russell et al, 1992, sproat 1992)⁵. 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 Allen, J., Hunnicutt, M. S., and Klatt, D. (1987). *From text to speech---the MITalk system*. MIT Press, Cambridge, Massachusetts.

² Petheroudakis, J. (1991). *MORPHOGEN automatic generator of morphological information for base form reduction*. Technical report, Executive Communication Systems ECS, Provo, Utah.

³ Schuller, G., Zierl, M., and Hausser, R. (1993). *MAGIC. A tutorial in computational morphology*. Technical report, Friedrich-Alexander Universitat, Erlangen, Germany.

⁴ 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.

⁵ 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.

1.2 Existing research in this area

In Indo European, Dravidian and Semitic language families, gender agreement plays an important role in sentence construction. So linguists have tried to build different kinds of analyzer systems for those languages. The computational analysis of gender is related to those languages through morphological analysis and grammar checking in those languages.

1.2.1 Existing computational analyzer of grammatical gender for Sanskrit

Though no applied work in Sanskrit grammatical gender has been done so far, some related work has been done. The Indian Heritage Group of the Centre for Development of Advanced Computing (C-DAC) has done some work in the related areas. The system called *DESIKĀ*⁶ claims to process all the words of Sanskrit; includes generation and analysis (parsing); has an exhaustive database based on *amarakoṣa*, a rule-base using the grammar rules of *Pāṇini's Aṣṭādhyāyī* and heuristics based on *nyāya & mimāmsa śāstras* for semantic and contextual processing. *Desikā* which is available at the TDIL site does only subanta generation.

The Academy of Sanskrit Research,⁷ ASR Melkote claims to have achieved the identification of gender of words. The name of this project is Semusi (Sanskrit noun generation and analysis). This Subanta module generation has a Lexicon consisting of words in *Amarakoṣa* (9000 *subanta* base) which can generate more than 2,16,000 cases of inflected forms. It generates all the forms in 7+1 *vibhakti* (cases) and 3 *vacana* (Numbers) (24 forms) or any one of the opted forms of any chosen *subanta* base available in the database. It also generates multiple forms, if any. It can analyze the forms of any given case inflected form of the Subanta base available in database. It claims to identify the *anta* (Ending), *liṅga* (Gender) and *prātipadika* (Base). It can display multiple identifications.

⁶ Deśika (Natural Language Understanding System), <http://tdil.mit.gov.in/download/Desika.htm>

⁷ <http://www.sanskritacademy.org/About.htm>

Sanskrit morph analyzer, developed by Amba Kulkarni,⁸ analyzes Sanskrit string into gender number etc. The system can not analyze *kṛdanta* forms and *karmaṇi* forms of verb. The system can not check the gender agreement between words in a sentence.

Dr. Gerard Huet, Director, INRIA has developed various computational tools for Sanskrit, which are available online.⁹ Among them lemmatizer and Sanskrit readers are the analyzers. The Lemmatizer tries to tag a given simple inflected noun or a verb (without *upasarga*-s), the Sanskrit Reader Companion analyses a given phrase or a simple sentence, segments it into individual words, tags each word and parses the input. Modular transducers are applied to constrain the lexical analyzer to recognize the stream of forms. Over generation of segmentation and ambiguity of tags are checked by semantic role analysis similar to Pāṇini's *kāraṅka* theory and also by governance patterns of verbs.¹⁰

1.3 R&D for other Indian languages

Other than Sanskrit, the researches are going on to develop morphological analyzer for other Indian Languages in different institutes in India. In the Indian institute of Science and Technology, Mumbai, the morphological analyzer and stemmer have been developed. These will assist in Hindi part of speech tagging task. Here the system analyses the input (a sentence) in the category of vowel ending, valid suffix of a word, gender, number, and person and case information. The verb analysis covers tense, aspect, modal, gender and number. Here stemmer is functioning as initial tagger providing grammatical category of input words. It helps in finding the category of unknown words (not present in dictionary) by looking at the attached suffix. Compound nouns and proper nouns have not been handled yet.

⁸ http://lrc.iiit.net/~anusaaraka/SAN_MO/test_san_mo.html

⁹ <http://sanskrit.inria.fr/>

¹⁰ Ibid.

1.3.1 Hindi

A Hindi morphological tagger was developed by Vasu Renganathan.¹¹ After entering a Hindi sentence it tags the words of the sentence in person, number and also in gender.

1.3.1.2 Hindi Word Net

A Hindi word net along with Marathi word net developed by Dr. Puspaka Bhattacharya and his team is available at <http://www.cfilt.iitb.ac.in/wordnet/webhwn/>. The word net has both Hindi and English interface. According to Dr. Bhattacharya, the Hindi word net is a system for bringing together different lexical and semantic relations between the Hindi words. It organizes the lexical information in terms of word meanings and can be termed as a lexicon based on psycholinguistic principles. The design of the Hindi word net is inspired by the famous English WordNet.

1.3.2 Bangla

Morphological analyzer of inflectional compound words in Bangla¹² has been developed. It is implemented in PC-KIMMO version 2, which is based on two-level morphology. It is said that the system gives 100% correct results. The compound words are used from Bangla grammar book and from literature. Morphological rules for simple words found in literature have been used in this system. It also handles the ambiguities resulting from inflection deletion or the lack thereof.

1.3.2.1 Spell checker

A spell checker has been developed by ISI Kolkata.¹³ It is a tool to detect errors in **Bangla** words and correcting them by providing a set of correct alternatives which includes the intended word. An erroneous word can belong to one of two distinct categories, namely, non-word error and real-word error. In this spell-checker, only non-

¹¹ <http://ccat.sas.upenn.edu/plc/tamilweb/hindi.html>

¹² <http://www.springerlink.com/content/m2gr7lt8v0cjre54/fulltext.pdf>

¹³ B. B. Chauhduri and T. Pal, Detection of word error position and correction using reverse word dictionary', Intl. Conf. On Computational Linguistics, Speech and Document Processing ICCLSDP'98, February 18-20, 1998, pp, C41-C46, B. B. Chauhduri, A Novel Spell-checker for Bangla Text Based on Reversed-Word Dictionary, Vivek, Vol. 14(4), pp. 3-12, October 2002.

word errors are considered. Word errors can be classified into four major types namely, substitution, deletion, insertion, and transposition error. In Bangla, wrong uses of characters, which are phonetically similar to the correct ones is observed. A great deal of confusion occurs in the use of long and short vowels, aspirated and unaspirated consonants, dental and cerebral nasal consonants due to phonetic similarity. Another type of error is typographical error which is caused by the accidental slip of fingers on the keys which are neighbours of the intended key. In this spell-checker, the main technique of error detection is based on matching the candidate string in the normal as well as in the dictionary. To make the system more powerful, this approach is combined with a phonetic similarity key based approach where phonetically similar characters are mapped into a single symbol and a nearly-phonetic dictionary of words is formed. Using this dictionary, phonetic errors can be easily detected and corrected. Here a candidate string first passes through the phonetic dictionary. If the word is not found in the dictionary and also failed to give suggestion then it tries to divide the word in root part and suffix part by separately verifying both. If an error is found, the spell-checker attempts to provide suggestion. If it fails, it checks whether the string is a conjunct word generated by appending two noun word and suffix. Option for adding new words permanently or temporarily is provided in the spell checker.

1.3.2.2 Morphological Analyzer and Synthesizer¹⁴

Media Lab Asia Research Laboratory at IIT Kharagpur¹⁵ has designed a morphological analyzer and synthesizer for Bengali. The morphological analyzer can identify the tense, aspect, modality and person of an inflected verb. The morphological analyzer has been modelled as finite state transducers which are given a root and TAM and GNP information as input. The transducer's job is to generate the correct inflected form of the word by adding the appropriate suffix for Bengali. Verbs were classified according to the vowels present in them. The modus operandi of the synthesizer is to determine the category of the root, given an input. It then accordingly synthesizes the ultimate form of

¹⁴ <http://www.mla.iitkgp.ernet.in/technology.html>

¹⁵ <http://www.mla.iitkgp.ernet.in/Resource/index.html>

the word depending on the input TAM GNP parameters. The numbers of exceptions are limited, and have been trapped easily.

1.3.3 Manipuri

A morphological analyzer for Manipuri language¹⁶ has been developed at IIT Guwahati. The morphological analysis determines the syntactic properties of Manipuri words and it comprises of the following three major functions, Morphographemics, Morphotactics and Feature Combination. The analyzer is modeled to treat orthographic variations, sequential and non-sequential morphotactic constrains and combination of morphosyntactic features. The morphological processing is based on the grammatical rules and the dictionaries: root and affix dictionary. A model tagger is used to tag the analyzed word. The tagger tags the lexical category of the root and the grammatical category of the affixes. Here the focus is mainly on the derivational and inflectional morphology of nouns and verbs. It takes an unknown word as input and produces the morphological structure of the word. The Morphological Analyzer is composed mainly of three modules: Segmentation, Morphosyntactic Analyzer and Tagging. The morphological analyzer takes the input and refers to segmentation module, which divides the input into root and affixes (prefix or suffix). After the segmentation is done, the root and affixes are supplied to the next module for checking the morphosyntactic features or rules. The tagging module performs the identification of morpheme meaning or category of each morpheme. The output text comes only after the tagging is completed.

Along with Manipuri morphological analyzer, the Assamese morphological analyzer has also been developed by IIT Guwahati. Both analyzers are used in the spell checker system of respective languages.

¹⁶ <http://www.springerlink.com/content/1fnfnn8qwd6xq4kr/fulltext.pdf>

1.3.4 Oriya

The PG Department of Comp. Sc. and Application, Utkal University, Bhubaneswar has developed Oriya Morphological Analyzer (OMA), a computational model for the analysis and generation of Oriya language.¹⁷ The major contents of OMA are:

- i) Pronoun Morphology (PM),
- ii) Inflectional Morphology (IM) and
- iii) Derivational Morphology (DM).

Developers have developed and implemented the Decision Tree (DT) and its respective algorithm of each type of morphology, through which OMA runs. While performing morphological analysis, OMA not only deals with the study of words but also its morphemes. The OMA system has been designed on the basis of Object-Oriented Approach (OOA). By use of this design methodology, different functions can be added or deleted to the existing system conveniently. Pronoun Morphology and Inflection Morphology have been implemented in the OMA in such a manner that it successfully runs with the OriNet system, Oriya Spell Checker (OSC) and Oriya Grammar Checker (OGC). The OSC handles any type of word (derived, inflectional or root) using the OMA. It also provides sufficient interface for applications involved in Oriya Machine Translation (OMT), Word-Net for Oriya (OriNet), Oriya Spell Checker (OSC) and Oriya Grammar Checker (OGC). All these developments have been worked out on the basis of the syntactic approach of Sanskrit language

1.3.5 Punjabi

The Advanced Centre for Technical Development to Punjabi Language Literature and Culture has developed the Punjabi morphological analyzer and developer. The software on click of a mouse, displays the list of all the possible word forms of any Punjabi root word, along with their respective grammatical information. The grammatical information will be different for different word classes, like for nouns it will provide its gender, number, and case, for verbs it will give tense, phase, aspect, etc. The software can also

¹⁷ <http://www.ilts-utkal.org/oriyamorphological.htm>

identify the grammatical attributes of any Punjabi word. The software can also be used to search for any Punjabi word in it to know its root and other grammatical information. The database used in the software consists of more than 1.72 lakh Punjabi words, grouped into 22 word classes such as noun, personal pronoun, reflexive pronoun, verb, inflected and uninflected adverb, inflected and uninflected adjective, conjunction, interjection etc.

1.3.6 Gujarati

The Gujarati spell checker along with Gujarati morphological analyzer is in the last stage of development by the resource centre for Indian language technology solutions-Gujarati, Maharaja Savajirao University of Baroda. Morphological Analyzer covers the analysis of nouns and verbs of the language.

1.3.7 Tamil

The Morphological analyzer for Tamil¹⁸ has been developed by the AU-KBC Research centre of Anna University. The aim of the analyzer is to strip a word of its inflections. The system is based on a finite automata state table. The analyzer first checks whether the given word is a root or not. If the given word is a root, then the analyzer will print the grammatical features of that word. Otherwise it will check whether there are morphemes in the given word. Every time the word is checked to find whether any root is present in it or not after cutting the morpheme. If there is no match found in the state table then the analyzers will check whether there is any *sandhi* present in the given word.

At the www.languageinindia.com site, there is a theoretical discussion about computational morphology of Tamil verbal complex by S. Rajendran, S.Viswanathan, and Ramesh Kumar. The system takes help from above mentioned Tamil morphological analyzer and is based on finite state automaton. It analyzes verbal complex into minimal meaningful units (which includes suffixes and bound auxiliary verbs) and assigns semantic features to each unit, and generates verbal complex out of the same meaningful

¹⁸ http://www.au-kbc.org/research_areas/nlp/projects/morph/document.html

units. The preparation of a morphological analyzer, in which the processing of verbal complex forms a part, has many natural language applications such as parsing, text generation, machine translation, preparing dictionary tools and lemmatization. It also helps in speech applications such as text-to-speech synthesizing and speech recognition and in word processing applications like spell checking and text input and in retrieval of documents. Finding the categorical details of the verbal forms helps in assigning parts-of-speech tags to the verbal complex.

1.3.8 Telegu

CALTS, University of Hyderabad has developed a morphological analyzer cum spell checker for Telegu¹⁹ with 97% recognition rate. It is tested on five million word corpora and it can be used with windows OS or in Linux. English-Telugu Machine Translation System is being built at CALTS in collaboration with, IIIT, Hyderabad; Telugu University, Hyderabad; Osmania University, Hyderabad. CALTS are also developing morphological generation for various Indian languages particularly for Telegu, Kannada, Tamil, Malayalam and Oriya.

1.3.9 Malayalam

RCILTS-Malayalam, C-DAC, Thiruvananthapuram has designed a software subsystem named Nerpadam²⁰ that can be integrated with Microsoft word as a macro or the Malayalam editor stylepad developed by them, to check the spelling of words in a Malayalam text file. While running as a macro in word, it functions as an offline spell checker in the sense that one can use this software with a previously typed text file only. Both off line and online checking are possible when it is integrated with the text editor. It generates suggestions for wrongly spelt words. The system adapts a rule cum dictionary-based approach for spell checking. It incorporates a fully developed Morphological Analyzer for Malayalam. This module splits the input word into root word, suffixes, post positions etc. and checks the validity of each using the rule database. Finally it will check

¹⁹ <http://72.14.235.104/search?q=cache:qkl6DegG3AJ:www.aubc.org/dfki/igws/calts.ppt+morphological+analyzer&hl=en&ct=clnk&cd=178&gl=in>

²⁰ tdil.mit.gov.in/Malayalam-CDACThiruvananthapuramJuly03.pdf

the dictionary to find whether the root word is present in the dictionary. If anything goes wrong in this checking it is detected as an error and the error word is reprocessed to get 3 to 4 valid words, which are displayed as suggestion. The user can add new words into a personalized data base file, which can be added to the dictionary if required.

1.4. Machine translation systems

Different machine translation systems from one Indian language to another Indian language have been developed since two decades. These systems are namely Anusaaraka, Anglabharati, Anubharati, Shakti etc. A MT system basically has three major components, viz, morphological analyzer of source language, mapping unit and the target language generator. The morphological analyzer splits a word into its constituent morphemes. These morphemes collectively describe the word grammatically. The root word is obtained from this process, and this root word is tagged with grammatical information such as tense marker if it is a verb. If it is known then gender, number, vibhakti etc information is obtained. The mapping unit maps the root word and its inflection to equivalent target language terms. Generator is the reverse process of analyzer. Given a root word and its inflections this generates the equivalent word of the target language. While generating, this takes into account all the information like the gender, tense etc. and the equivalent word is generated accordingly. Thus in a machine translation system, the analysis of grammatical gender is necessary if the target and the source languages both have gender agreement. As a consequence, the existing machine translation systems are mentioned.

1.4.1 Anusaaraka

Anusaaraka²¹ is a computer software which renders text from one Indian language into another. It produces output which is comprehensible to the reader, although at times it might not be grammatical. Anusaaraka analyzes the source language text and presents exactly the same information in a language close to the target language. It does not try to guess using world knowledge etc. It tries to preserve information from the input to the output text. Anusaaraka systems are available in Kannada – Hindi, Marathi- Hindi, Punjabi- Hindi, Tamil- Hindi, and also English – Hindi. It is necessary to have Linux operating system to use these systems.

1.4.2 Anglabharati

IIT Kanpur has developed a machine-aided translation methodology for translation from English to Indian languages, named Anglabharati.²² It is a pattern directed rule based system with context free grammar like structure for English (source language). It generates a 'pseudo-target' (Pseudo-Interlingua) applicable to a group of Indian languages (target languages) such as Indo-Aryan family (Hindi, Bangla, Asamiya, Punjabi, Marathi, Oriya, Gujrati etc.), Dravidian family (Tamil, Telugu, Kannada & Malayalam) and others. A set of rules obtained through corpus analysis is used to identify plausible constituents with respect to which movement rules for the 'pseudo-target' is constructed. Within each group the languages exhibit a high degree of structural homogeneity. A language specific text-generator converts the 'pseudo-target' code into target language text. Paninian framework based on Sanskrit grammar using kāraka relationship provides a uniform way of designing the Indian language text generators. It uses an example-base to identify noun and verb phrases and resolve their semantics. An attempt is made to resolve most of the ambiguities using ontology, syntactic & semantic tags and some pragmatic rules. The unresolved ambiguities are left for human post-editing. Some of the major design considerations in design of Anglabharti have been aimed at providing a practical aid for translation wherein an attempt is made to get 90% of the task done by the machine and 10% left to the human post-editing; a system which could grow incrementally to handle

²¹ <http://www.iit.net/ltrc/index.html>

²² <http://anglahindi.iitk.ac.in/>

more complex situations; an uniform mechanism by which translation from English to majority of Indian languages with attachment of appropriate text generator modules; and human engineered man-machine interface to facilitate both its usage and augmentation. The translation system has also been interfaced with text-to-speech module and OCR input. (Add source ref here)

1.4.3 Anubharati

AnuBharti²³ system is designed to translate Hindi to English and other languages. It has also been developed by IIT Kanpur. Here the pre-stored examples form the basis for translation. The translation is obtained by matching the input sentence with the minimum 'distance' example sentence. In our approach, we do not store the examples in the raw form. The examples are abstracted to contain the category/class information to a great extent. This makes the example-base smaller in size and further partitioning reduces the search space. The creation and growth of the example-base is also done in an interactive way. The strategy has now been generalized in AnuBharti-II to cater to Hindi as source language for translation to any other language, though the generalization of the example-base is dependent upon the target language. The core of AnuBharti-II architecture is a generalized hierarchical example-base. In absence of availability of large parallel corpora, the example-base is augmented interactively during the development phase. Development of such an example-base for Hindi to other Indian languages is lot easier as compared to a dissimilar language. Hindi like all other Indian languages is a relatively free word-group order language. The example-base size grows enormously if all variations are incorporated into it. The input Hindi sentence is converted into a standardized form to take care of word-order variations. This requires a shallow grammatical analysis of Hindi. This makes the paradigm used in AnuBharti-II a hybrid paradigm. The standardized Hindi sentence is matched with a top level standardized example-base. In case no match is found then a shallow chunker is used to fragment the input sentence into units that are then matched with a hierarchical example-base. The translated chunks are positioned by matching with sentence level example base. Analysis of Hindi sentence is rule and heuristic based and is primarily used for deriving standardized form and for shallow

²³ <http://www.cse.iitk.ac.in/users/langtech/anubharti.htm>

chunking. The boundary friction problem of chunk translation composition is handled by the sentence level example-base with chunks. Here the chunk properties are used for distance computation and the chunk translation among the alternatives yielding minimum distance is picked up. Both of these system architectures, AnglaBharti and AnuBharti, have undergone a considerable change from their initial conceptualization. In 2004, phase-II of system development has been launched which addresses many of the shortcomings of the earlier architectures. These are named AnglaBharti-II and AnuBharti-II. While AnglaBharti-II is primarily a rule-based system and AnuBharti-II uses EBMT as the basic paradigm for translation, both of these systems are hybridized with varying degree of hybridization of different paradigms.²⁴

1.4.4 Shakti

Shakti system is being developed from English to Indian languages by IIIT Hyderabad.²⁵ It combines rule-based approach with statistical approach and its modular design makes it possible to produce machine translation systems for new target languages rapidly. It can be tried for three target languages- Hindi, Telugu and Marathi.

1.4.5 MaTra

MaTra²⁶ is a Human-Assisted translation project for English to Indian languages, currently Hindi, essentially based on a transfer approach using a frame-like structured representation. The focus is on the innovative use of man-machine synergy—the user can visually inspect the analysis of the system, and provide disambiguation information using an intuitive GUI, allowing the system to produce a single correct translation. The system uses rule-bases and heuristics to resolve ambiguities to the extent possible – for example, a rule-base is used to map English prepositions into Hindi postpositions. The system can work in a fully automatic mode and produce rough translations for end users. Currently, it works for simple sentences, and work is on to extend the coverage to complex sentences. The MaTra lexicon and approach is general-purpose, but the system has been applied

²⁴ <http://www.cse.iitk.ac.in/users/rmk/proj/proj.html#mt>

²⁵ <http://shakti.iiit.ac.in>

²⁶ <http://www.ncst.ernet.in/matra/http://www.ncst.ernet.in/matra/about.shtml>

TH-17801

mainly in the domains of news, annual reports and technical phrases, and has been funded by TDIL²⁷.

1.4.6 Mantra

The Mantra²⁸ project is based on the TAG formalism from University of Pennsylvania. A sub-language English-Hindi MT system has been developed for the domain of gazette notifications pertaining to government appointments. In addition to translating the content, the system can also preserve the formatting of input Word documents across the translation. The Mantra approach is general, but the lexicon/grammar has been limited to the sub-language of the domain. Recently, work has been initiated on other language pairs such as Hindi-English and Hindi-Bengali, as well as on extending to the domain of parliament proceeding summaries.

1.4.7 English-Hindi MAT for news sentences²⁹

The Jadavpur University at Kolkata has recently worked on a rule-based English-Hindi MAT for news sentences using the transfer approach.

1.4.8 Anuvadak English-Hindi software

Super Infosoft Pvt Ltd is one of the very few private sector efforts in MT in India. They have been working on a software called Anuvadak³⁰, which is a general-purpose English-Hindi translation tool that supports post-editing.

1.4.9 English-Hindi Statistical MT

The IBM India Research Lab³¹ at New Delhi has recently initiated work on statistical MT between English and Indian languages, building on IBM's existing work on statistical MT.

²⁷ <http://www.tdil.mit.gov.in/TDILmeet2001May01.pdf>

²⁸ <http://www.cdacindia.com/html/about/success/mantra.asp>

²⁹ <http://www.jadavpur.edu/>

³⁰ <http://www.mysmartschool.com/pls/portal/portal.MSSStatic.ProductAnuvaadak>

³¹ <http://www.elda.org/en/proj/scalla/SCALLA2001/SCALLA2001Rao.pdf>

1.5 Text-processors for Indian languages

Text processor is one of the most frequently used applications. This is a program that allows writing pretty formatted documents, from simple letter to large manuscripts, even containing tables and illustrations. Text processing programs often come as part of a software bundle, called an office suite. It usually contains at least a word processor, a spreadsheet, a data base and a presentation program. Probably the most widespread office suite is MS Office from Microsoft, containing Word, Excel, Access and PowerPoint.

1.5.1 Akshara

Akshara an advanced Multi-Lingual Text Processor³² encodes texts in a standard character encoding scheme such as ISCII or UNICODE. It uses character-encoded documents for all its operations instead of font-encoded pages. Mapping to fonts is done only for the purposes of display and printing - all other operations are performed on character encodings. Attributes are included in an open XML style markup language called Extensible Document Definition Language (XDL) developed by us. This makes it easy to convert to and from various other encoding schemes thereby ensuring highest levels of portability and platform independence. A unique feature of Akshara is that it understands the script grammar and warns if one tries to build ungrammatical syllables. It has been successfully used to clean up all the corpora at CIIL, Mysore. It is platform independent and can be used on MS Windows, Linux and many other platforms. Dictionaries, morphological analyzers, spell checkers, OCR systems, TTS systems, text processing tools including searching, sorting etc. are part of it. Several text processing tools, Telugu spell checker and Telugu TTS have already been integrated.

1.5.2 Bilingual Punjabi/English Word Processor

A Bilingual Punjabi/English Word processor named Likhari³³ has been developed by RCILTS-Punjabi at Thapar Institute of Engineering & Technology, Patiala keeping in view the difficulties being faced by the users while working on the currently available word processors. Likhari supports word processing under the windows environment and allows typing and processing in Punjabi Language through the common typewriter

³² <http://ildc.gov.in/telugu/htm/Akshara.htm>

³³ <http://tdil.mit.gov.in/Punjabi-TIETPatialaJuly03.pdf>

keyboard layout. It has MS-Word compatible features and commands. It provides a number of features that make the use of Punjabi Language on a computer easy and provides a number of tools to increase the efficiency of the user. These tools include Bilingual Spell Checker with suggestion list, onscreen keyboard layouts with composition reference for Punjabi Language typing, bilingual search and replace, sorting as per the language, alphabetical order, technical glossaries and onscreen Bilingual dictionaries.

1.5.3 Nashir

RCILTS-Urdu, Sindhi & Kashmiri, Centre for Development of Advanced Computing, Pune has developed a word-processor named Nashir.³⁴ It is claimed to be capable of creating documents in Perso-Arabic languages, and at the same time to layout complete newspapers and magazines in Urdu, Sindhi, Kashmiri, Arabic and Persian. Each document of the Nashir consists of a number of pages. On each page of the document, one can place items like text blocks, graphics, etc. It is kind of a Word processor & also best suited for publishing segment. Spellchecker support is added in the full version. Apart from the base dictionary for spellchecker, various domain specific dictionaries addition is planned. It supports Nastaliq True Type fonts (presently 2 fonts) as well as Naskh fonts (presently 12 fonts). Fonts for Sindhi, and Kashmiri are also added. It supports C-DAC & Phonetic Keyboards and also user-defined keyboards. It provides various drawing objects and also supports the OLE Automation. Nashir supports Urdu, Sindhi & Kashmiri along with English. A transliteration engine (uTRANS) has also been implemented in it. One can insert an “.aci” (ISCII file) file into Nashir and see the transliterated version in Urdu script (Naskh or Nastaliq). Rule based transliteration has been developed for Hindi & Punjabi. The user can save the document as HTML page, and thus Naskh as well Nastaliq scripts can be viewed on the Internet. A Table control is provided to feed tabular information in the document. Phonetic keyboard supported. Both Horizontal and Vertical kerning is provided to manually adjust a text. It has Horizontal and Vertical rulers in the GUI, dynamic font settings for the Urdu and English fonts. The

³⁴ tdil.mit.gov.in/UrduSindhiKashmiri-CDACPuneJuly03.pdf

full version of Nashir is now equipped with the spell checker facility. The Spellchecker program works as a word-level error detection and correction (single or multiple correction) tool.

1.6 Jawaharlal Neheru University

The RCILTS – Sanskrit, Japanese, Chinese unit of JNU, under the leadership of Prof. G.V.Singh claims to have developed web based Sanskrit Language 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.³⁵

Girish Nath Jha³⁶ 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.

1.6.1 Special Center for Sanskrit Studies, JNU

Several language processing tools for Sanskrit and Andamanese have been developed under the supervision of Dr. Girish Nath Jha at the Special Center for Sanskrit Studies, JNU. These tools can be used online at <http://sanskrit.jnu.ac.in>. The details of some of them are as follows -

Sudhir Kumar Mishra³⁷, is working on a Kāraka Analyzer for laukika Sanskrit prose text based on Pāṇini's kāraka formulations which can be an important component in any Sanskrit-Indian language translation system. As part of this research work, he has also worked on identification of verb inflections in Sanskrit morphology³⁸.

³⁵ RCILTS, JNU – Achievements: <http://tdil.mit.gov.in/SanskritJapaneseChinese-JNUJuly03.pdf> (accessed 15.10.2006)

³⁶ Jha, Girish Nath, 1993, 'Morphology of Sanskrit Case Affixes: A Computational Analysis', M.Phil., submitted to JNU, New Delhi.

³⁷ Mishra, Sudhir Kumar & Girish Nath Jha, 2004, 'Sanskrit Karaka Analyser for Machine Translation', in the proceedings of iSTRANS-2004, New Delhi, pp.224-225.

³⁸ Mishra, Sudhir Kumar & Girish Nath Jha, 2005, 'Identifying Verb Inflections in Sanskrit Morphology', in the proceedings of SIMPLE-05, IIT-Kharagpur, pp.79-81

As part of Subash Chandra's³⁹ M.Phil. dissertation, a Sanskrit *subanta* Recognizer and Analyser System has been developed which is an online system running at <http://sanskrit.jnu.ac.in> on Apache Tomcat platform using Java Servlet. This system has been developed according to Pāṇinian formulation which accepts only non-joint (*sandhi-rahita*) Sanskrit text in Unicode Devanāgarī and fully depends on both the rule base, example base and a database of other linguistic resources. The system claims to give an average accuracy of 91.65% accuracy, tested on some selected simple Sanskrit prose texts.

Research work is also being done on learning Sanskrit language using e-learning approach⁴⁰, *tiṅanta* analyzer using reverse Pāṇinian and database approach⁴¹, and *sandhi* analyzer applying Pāṇinian and some heuristic rules.⁴²

R.Chandrashekhar's Ph.D. thesis on POS tagging of Sanskrit led to the development of a POS tagger for Sanskrit. The tagger is live at <http://sabnskrit.jnu.ac.in>. Narayan Chaudhary's M.Phil. thesis on Great Andamanese verb analyzer led to the development of a tagger for Great Andamanese Verbs. This web application is also live at the above site.

³⁹ Chandra, Subash, 2006, 'Machine Recognition and Morphological Analysis of Subanta-padas', submitted for M.Phil degree at SCSS, JNU

⁴⁰ Bhowmik, Preeti & Jha, Girish Nath, 2006, 'Sanskrit Language Pedagogy: an e-learning approach', In the Souvenir Abstracts of 28th AICL, BHU, Varanasi, p.150.

⁴¹ Agrawal, Muktanand, 2006, 'Computational Identification and Analysis of Sanskrit Verb-forms', In the Souvenir Abstracts of 28th AICL, BHU, Varanasi, pp.126-127.

⁴² Kumar, Sachin & Jha, Girish Nath, 2006, 'Issues in sandhi processing of Sanskrit', In the Souvenir Abstracts of 28th AICL, BHU, Varanasi, p.129.

1.7 Morphological analyzer for Non -Indian languages

1.7.1 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 systems⁴³. 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, and
- 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.

Figure 1 shows the main components of the PC-KIMMO system⁴⁴.

⁴³ <http://www.sil.org/pckimmo/> accessed on 25th December 2005

⁴⁴ <http://www.sil.org/pckimmo/>

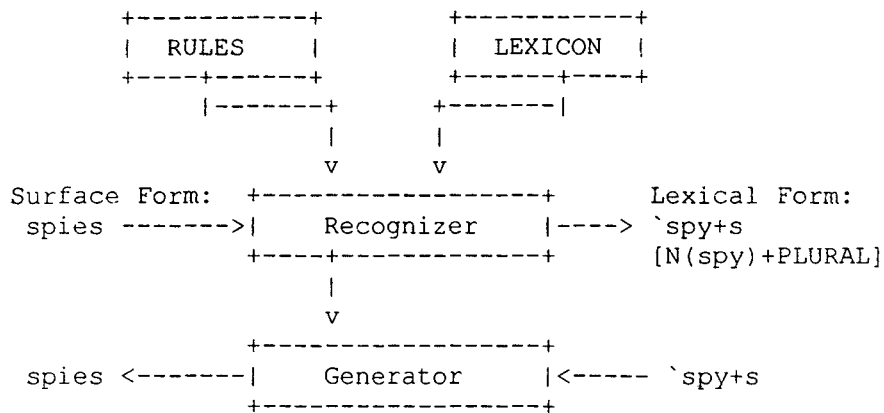


Figure 1.1 Main components of PC-KIMMO

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 data⁴⁵.

⁴⁵ <http://www.sil.org/pckimmo/>

1.7.2 CLAWS

CLAWS (Constituent Likelihood Automatic Word-tagging System)⁴⁶ 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 Manual⁴⁷.

1.7.3 Spanish

The Panglyzer, Spanish language analysis system, follows a multi-pass approach consisting of preprocessing, part-of-speech tagging, phrase recognition, proper name classification, phrase analysis, clause recognition, clause analysis and reading ranking. Here while analysis of phrase the system tags gender of noun.

Grampal: This is a morphological processor for Spanish implemented in prolog. It is a model for the full treatment in Spanish inflection for verbs, nouns and adjectives. While analyzing nouns and adjectives the gender of those are also marked.

1.7.4 French

Though it is a corpus analysis for predictability of gender attribution in French, still there is analysis of gender of French noun phrases. The INFL analyzer analyses French words in context free morphological analysis. It is a licensed product of the MultiLingual Theory and Technology team at the Rank Xerox Research Center, in Grenoble, France. made available to ARTFL through a technology Exchange agreement.⁴⁸ The principal developers of INFL are Lauri Karttunen and Annie Zaenen.

⁴⁶ <http://www.comp.lancs.ac.uk/ucrel/claws> access on 15th January 2006

⁴⁷ www.comp.lancs.ac.uk/ucrel/claws

⁴⁸ Lauri Karttunen and Todd Yampol, *INFL Morphological Analyzer*, (XEROX Palo Alto Research Center, Palo Alto, CA, 1990).

1.7.5 Swedish

Granaska⁴⁹ is a grammar checker of Swedish language. It basically deals with spelling and other grammatical mistakes. Among them agreement between noun phrases is there. While detecting the wrong agreements the system checks the gender agreement among noun phrases and suggests the correct ones.

1.7.7 Greek

The Greek morphological lexicon⁵⁰ was used to develop a lemmatiser and a morphological analyzer that were included in a controlled language checker for Greek. The goal of this work is to produce a wide – coverage morphological lexicon of Greek that can be easily exploited in several natural language processing applications. While covering the morph syntactical details of lexicon this lexicon gives information about gender of nouns of Greek languages. A computational model for the morphological analysis of the Greek noun category presents the first stage of research for the development of an automatic lemmatisation system for Modern Greek. The noun category of the language has been examined so far. The objective of the study was to identify in the form of a suffix database, all the suffixes which provide single grammatical information about the lexical item they mark. The novelty of the approach lies first in what has been treated as suffix and secondly, in the attempt to grammatically disambiguate words in a text on the basis of morphology alone, reducing dictionary matching to its minimum. The term graphological groupings (G.G.) proved the most appropriate for the suffixes of our database, as they range from one to seven characters and they cover in many cases both the inflectional and derivational suffixes of the language. Six hundred and fifty one graphological groupings (G.G.) out of a total of eight hundred and fifty have been identified as specifying, rendering dictionary matching redundant. The remaining one hundred and ninety nine provide two to three grammatical alternatives in the form of Gender/Number/Case (G.N.C.) combinations, requiring either dictionary matching or interactivity facilities for the disambiguation process. This database, supplemented by a number of simple transformation rules, gave birth to a

⁴⁹ <http://www.csc.kth.se/tcs/projects/granska/rapporter/granskareport.pdf>

⁵⁰ <http://www.springerlink.com/content/hcdjrlvj5nlybf5c/fulltext.pdf>

program which achieves automatic lemmatisation for the majority of Greek nouns. The lemmatiser on the whole can be used:

- a. for teaching Modern Greek grammar to the first classes of Primary School.
- b. as a filter:
 - i) during Modern Greek computerized dictionary consultation for foreign learners of the language
 - ii) on the morphological analysis level for Machine Translation systems
 - iii) in automatic spelling-checking systems.

1.7.8 Norwegian language

The automatic proof reading system of Norwegian language⁵¹ named Scarrie, tries to check the spelling and the grammar Scandinavian languages and specially the grammar variations in Norwegian Bokmal. While performing this it checks gender agreement among noun phrases.

1.7.9 Polish

In the system of Polish text summarization⁵², to solve the anaphora resolution and to generate a proper form of antecedent one database is used. In the database there is gender information of the noun along with other morphological information. The system also checks the gender agreement between anaphora and its antecedent. If the agreement is wrong then it supplies the right one.

1.7.9.1 Tagger

Polish nouns agree in gender with adjectives and verbs, so this system of polish tagger⁵³ tags the gender information with other morphological information.

1.7.10 Bulgarian

Morphological processor is one of the parts of the language processing tools for Bulgarian⁵⁴. The input in this system is in ASCII or in ANSI text and the output is in SGML text. The system check the gender agreement while checking the syntactic

⁵¹ <http://ling.uib.no/~desmedt/papers/nodalida99pub.html>

⁵² <http://www.enformatika.org/ijci/v1/v1-4-56.pdf>

⁵³ <http://acl.ldc.upenn.edu/W/W07/W07-1701.pdf>

⁵⁴ <http://tanev.dir.bg/LingTunis.htm>

agreement between words, while resolving morphological disambiguation and while NP extraction.

1.7.11 Czech

The Xerox research centre in Europe has developed morphological analyzer, tokenizer and part of speech disambiguation for Czech language.⁵⁵

1.7.12 German

Morphy,⁵⁶ an integrated tool for German morphology, part-of speech tagging and context-sensitive lemmatization has been developed by Wolfgang Lezius, Reinhard Rappa and Manfred Wettler. Its large lexicon of more than 320,000 word forms plus its ability to process German compound nouns, guarantee a wide morphological coverage. Syntactic ambiguities can be resolved with standard statistical part-of-speech tagger. By using the output of the tagger, the lemmatizer can determine the correct root even for ambiguous word forms. The morphological analyzer is based on the standard Duden grammar and provides wide coverage due to a lexicon of 324,000 word forms and the ability to process compound nouns at runtime. It gives for each word form of a text, all possible lemmata and morphological descriptions. The ambiguities of the morphological descriptions are resolved by the tagger, which provides about 85% accuracy for the large and 96% accuracy for the small tag set. The lemmatizer uses the output of the tagger to disambiguate word forms with more than one possible lemma. It achieves an overall accuracy of about 99.3%.

1.7.13 Arabic

In the machine translation from English to Arabic, it is necessary to mark gender of Arabic nouns phrases, because gender agreement is important there. So the system analyses gender and looks in to the gender agreement between noun phrases. Kenneth R. Beesley at the Xerox Research centre Europe, chemin de Maupertuis, 38240 MEYLAN, France has been developed an Arabic Morphological Analysis and Generation. Arabic

⁵⁵ <http://www.xrce.xerox.com/competencies/content-analysis/demos/czech>

⁵⁶ <http://www.ims.uni-stuttgart.de/projekte/corplex/paper/lezius/coling1998.pdf>

morphological analyzer and generator, which was built using Xerox Finite-State Technology. The system accepts Modern Standard Arabic words and returns morphological analyses and English glosses. Arabic words are displayed in Arabic script using Java applets. This lexicography and Arabic-language consultation for the research system was provided by Tim Buckwalter. The Arabic "Yarb" font used in the interfaces was created by Yannis Haralambous. System design, interfaces, Arabic-script rendering, and computational linguistics were done by Ken Beesley. Arabic language consulting and lexicography for the commercialization in 2002 was provided by Martine Pétrod⁵⁷

1.7.14 African languages

In the department of African languages of South Africa University, a project namely Computational Morphological Analysis Project⁵⁸, has been taken to develop finite state morphological analyzers for Bantu languages, using the natural language independent Xerox Finite-State Tools. This integrated set of tools is used to model and implement the complexities of word-formation rules as well as morphophonological alternations by means of finite-state networks, which in turn are combined together algorithmically into larger networks that perform morphological analysis. Lexical challenges are addressed by means of the development of machine-readable lexicons in XML format, containing knowledge about individual words in the languages. In this project Zulu morphological analyzer (ZulMorph) has been developed. Preliminary testing of the current prototype was done on a test corpus consisting of 30 000 types. The application of the morphological analyser to the test corpus results in the recognition of approximately 77% of the types in the corpus. The electronic monolingual Zulu word list used to build a rudimentary XML lexicon contains a total of over 28 000 lemmas (e.g. 13 000 noun stems, 7 400 verb roots, 950 ideophones, 60 adjective stems (and their variations), 12 conjunctions and 10 interrogatives). The same kind of project is going on regarding the language Xhosa. The development of the machine-readable Xhosa lexicon continues with capturing of data in a generic template in XML format. Northern Sotho morphological analyzer (NsoMorph) has been developed as a part of this project. This system can

⁵⁷ <http://www.xrce.xerox.com/competencies/content-analysis/arabic/>

⁵⁸ http://www.unisa.ac.za/contents/faculties/humanities/afrl/docs/NRF_Project_Webpage.pdf

analyze verbs by providing for past tense forms of the verb forms. The basis for an analyzer of verbs, nouns and derivative nouns has been established. Editing of the scanned Comprehensive Northern Sotho dictionary, by means of Perl scripting, has nearly been completed. A basic XML schema, based on the structure of the Comprehensive Northern Sotho dictionary, has been created. The development of a tokeniser has made progress to the point that testing has begun. The prototype for a morphological analyzer of Tswana nouns based on a selected lexicon has been finalized. Derivatives have been added to the prototype. The researches are going on to develop a morphological analyzer for Swati language.

1.7.15 Morphological analyzer for Chinese, Japanese, Korean

Asian language analyzers are used in some of the world's most transaction-heavy environments, like Google's search engine and Amazon's e-commerce site. Rosette Base Linguistics for Chinese, Japanese and Korean are extremely accurate and reliable solutions to help complex applications process unstructured Asian language text by conquering some of these languages' many challenges, such as the use of numerous scripts and absence of spaces between words. Using advanced morphological analysis, Asian Base Linguistics perform functions critical for analyzing Asian text such as segmentation, lemmatization, noun decomposing, part-of-speech tagging, sentence boundary detection, and base noun phrase analysis⁵⁹.

1.8 Development by private enterprises

In the age of globalization there is needed to reach upto many people, through the path of technology. Language is a major barrier in this process. To overcome this problem many NGOs and private enterprises too have found interest in NLP in the whole world. Companies like Microsoft, IBM, Google etc are constantly developing new methods for the implementation of globalization and as well as localization of their products.

⁵⁹ <http://www.basistech.com/base-linguistics/asian/>

Culturally and linguistically India is a land of great diversity. As a consequence, many developments are going on in India in this field.

1.8.1 Microsoft India

Indian Language support in Microsoft products⁶⁰ started around 1999 with Windows 2000 Professional being the first official product to provide Indic support. It supported 2 scripts, Devanagari and Tamil, and 5 languages, Hindi, Konkani, Marathi, Sanskrit and Tamil. Windows XP saw more Indian scripts and languages being supported in MS products: Gujarati, Kannada, Punjabi and Telugu. Currently with Service Pack 2, Windows XP supports 11 Indian Languages, such as Bengali, Gujarati, Hindi, Kannada, Konkani, Malayalam, Marathi, Punjabi, Sanskrit, Tamil, and Telegu. Windows XP is the operating system introduced in 2001 from Microsoft's Windows family of operating systems. With built in support for over 60 scripts, hundreds of languages, and 126 locales, every language version (and each flavor) of Windows 2000 provides unprecedented support for international and multilingual computing. Windows XP expands this global support and supports multilingual computing into the client space and home computing. The users of Windows 2000 / XP can do the following in their systems:

- Display Indic text;
- Input Indic text / script,
- Format and print Indic text;
- Create file names with Indic text;
- Correctly display dates, times, currencies and numbering schemes in Indic text and local cultural format;
- Correctly sort and search on Indic text;
- Set default browser language to an Indic language

⁶⁰ http://tdil.mit.gov.in/Oct_2004/GDemand%20world%20-10.pdf

1.8.1.1 MS Office

MS Office can help users to perform following tasks, using Indic language,

- Automatic language detection
- Date, Time & Number Style
- Based on Regional Settings
- Automatic Font Linking
- Editing Indic Content
- Find & Replace for Indic Content
- Spell Checking
- Checking correct sequence of characters
- Sorting – based on Indic Text
- Auto Filter
- Custom List
- Create Graphs with Indic Text
- Currency Functions
- Sending & Receiving Indic Content
- Date Formats
- Type appointments in Indic Language

Office Hindi is a product from Microsoft. It includes a Hindi language interface and supports nine Indian languages, empowering Indian users to leverage the global, standards based Office applications suite in the language of their choice. The Professional version offers a switchable interface, providing users with the flexibility of easily deploying the Hindi or an English interface as required. The product also offers multiple keyboard options, such as transliteration, inscript, remington and godrej, along with other variations.VS.

1.8.1.2 Technology behind Indic Support in MS products

To accommodate multilingual computing and to properly display text in a multilingual context, Microsoft introduces new technologies, such as open type fonts, font fallback, font linking, and uniscribe.

Open Type fonts is developed jointly by Microsoft and Adobe, is in fact an extension to the TrueType font specification. The Open Type fonts in Windows 2000 include expanded repertoires of glyphs to accommodate Pan-European, Arabic, Hebrew, and Thai scripts.

Font fallback mechanism, made available through Uniscribe, provides a fallback font (or a default font) when dealing with complex scripts. If the selected font face does not include any glyphs for the complex script that is about to be displayed, Uniscribe selects a default hard coded font for the given script. For example, if one has Hindi text and the font is Courier, then Uniscribe will use the Mangal font. This technique is internal to Uniscribe and developers can not add additional fonts to the list of fallback fonts.

Font linking technique is mainly used to accommodate East Asian languages and uses a registry setting of fonts linked in a list to a face name. If the required glyph is not in the nominal face name, GDI searches each font in the list of those linked to the face name until it finds one with the required glyph. This new font is then linked to the nominal font (base font). Users and developers should not modify the list of linked fonts in the registry, since it might cause serious display problems.

Uniscribe is the layout and shaping engine for these scripts, is built into Windows 2000/XP and provides consistent support across its clients (Windows2000, Microsoft Office 2000 and Office XP/2003, Microsoft Internet Explorer 5.0 and beyond).

1.8.1.3 Indian Language Application Development using .Net

.NET has support for Indian languages using Unicode. One can use VB.NET, VC++.NET, C#.NET, etc. to develop any application for the .NET framework. The application developed can be deployed on any system which has the dot net framework installed. Since the environment is Unicode one can have Indian language text in Menus, textboxes, error messages, etc. One can have web services in Indian language. There could be a web service running from a server that provides news headlines in Indian language.

1.8.1.4 SharePoint Products and Indic Support

SharePoint Products and Technologies facilitate collaboration within an organization and with partners and customers. Using the combined collaboration features of Windows SharePoint Services and SharePoint Portal Server 2003, users can easily create, manage, and build their own collaborative Web sites and make them available throughout the organization. SharePoint Services is data management and analysis software that will deliver increased scalability, availability, and security to enterprise data and analytical applications while making them easier to create, deploy, and manage.

SharePoint Services can also be used as a development platform for creating collaboration and information sharing applications. SharePoint Portal Server 2003 is a secure, scalable, enterprise portal server built upon Windows SharePoint Services that can be used to aggregate SharePoint sites, information, and applications into a single portal. All the above can be achieved in the 11 languages supported by Windows XP SP2.

1.8.1.5 Indic Support on SQL Server

SQL Server is a relational database management and analysis system for e-commerce, line-of-business, and data warehousing solutions. In SQL Server 2000, the data in Indic languages can be stored using the data types like nchar, nvarchar and ntext. These data types are defined to be equivalent to UTF-16 Unicode. Unicode data is stored using two bytes per character, which is one byte per character in the case of character data. The sorting of Unicode data is based on the collation name selected for the database. In the case of Hindi language the collation name can be selected as "Hindi_BIN" Data Types

used Indic Languages in various products, such as SQL Server 2000 – NVarchar, NChar, NText, MS Access – Text, .Net – String, C, C++ - wchar, LPWSTR.

1.8.2 IBM Software support for Indian languages⁶¹

- **Lotus Notes and Workplace:** This promotes effective collaboration and communication between users. It helps rise in productivity and user-satisfaction.
- **Universal Database DB2 Data and Content Management:** This facilitates storage, archival and management of data and content created by users. It helps to complete and perpetual availability of data and content.
- **Websphere Application Server:** It is claimed that it is the most popular middleware in the world. It hosts applications on the internet that are 'behind' browser and at the heart of the internet. It is a globalized application that caters to users anywhere in the world.
- **Rational Application Development Tools:** Under these tools, high quality rapid application development tools using open source integrated development environments like Eclipse. It is the foundation of simplifying application development for simple and complex business problems.
- **Tivoli Infrastructure Management Solutions:** Managing Information Technology infrastructure that could be spread over wide areas of cultural diversity. The simplified infrastructure management tailored to the conventions of the local users.

Other IBM offerings to supporting Indian languages (either now or soon) are: IBM Printers, IBM AIX Operating System for pSeries and IBM OS400 Operating System for iSeries. Researchers have extended IBM's ViaVoice speech recognition technology to develop a system for Hindi and "Indian English." This system understands and transcribes human speech with minimal use of keyboards thereby helping people unfamiliar with computers or the English language. Since there are no standard keyboards available in Indian languages, speech recognition eliminates the need to learn non-standard keyboard mapping. The system has been tested and trained for variations over a large number of speakers from different regions of the country.

⁶¹ http://tdil.mit.gov.in/Oct_2004/GDemand%20world%20-10.pdf

1.8.3 HP Labs

HP Labs India is trying to break the English language barrier by exploring the use of handwriting as an input. Millions of forms, like the ones for railway reservations, are filled out every day and in different Indian languages. Therefore, the lab is developing a technology that can recognize handwriting as well as capture image data for further processing. A prototype of such a script-independent device called Script Mail has been developed for sending and receiving hand written e-mails.

Voice or speech is one of the most prevalent forms of communication. HP Labs India has come up with a telephone-based railway inquiry system providing information on ticket availability and the status of wait-listed tickets for trains running between four Indian metros. In the next stage, online booking may be developed. This system is also accent-independent and works for Hindi and "Indian English." HP Labs took a general-purpose engine for speech recognition and synthesis, generated by a team of researchers worldwide and freely available to the software professional community, and combined it with their own recognition models specific to Hindi and "Indian English." This system too can accept and process a variety of Indian accents and speaking styles.

1.8.4 Google

Google has different interfaces in different languages, with many Indian languages and even in Sanskrit.⁶² Google tool can be used to translate texts from English to Spanish, Portuguese, German, French, Italian and vice versa.

Except these companies in India the Tata Institute of Fundamental Research (TIFR), Tata Consultancy Service (TCS) and other companies have started funding the Indian languages and KBCS R&D. The Computer Systems and Communication Group of TIFR, Bombay aims at developing a Voice Oriented Interactive Computing Environment (VOICE) using KBCS technologies. Already, some progress has been made in preparing speech database, phoneme-to-speech synthesis system, Multi-speaker speech recognition systems, and Computer Tutor in speech mode.

⁶² <http://www.google.com/intl/sa/>

From the above discussion it can be said that, Sanskrit can take a place in natural language processing. Sanskrit grammar can be used as a model for Indian language in natural language processing. In the next chapter the gender in Sanskrit grammar is going to be discussed.

Chapter2: Gender in Sanskrit Grammar

2.0 Gender

Gender in common usages refers to the differences between man and woman. According to Encyclopaedia Britannica, it is an individual's self-conception as being male or female, as distinguished from the actual biological sex. Though in the world, gender is used interchangeably across the academic field, especially in social sciences, in cultural studies or in gender studies. But gender in language, especially in grammar, has a different connotation. Grammatical gender is a morphological category associated with the expression of gender through inflection or agreement. In a language, every noun can be categorised under one class of gender, pronouns are also marked by gender. Natural languages exhibit gender agreement between nominal categories and/or between subject nouns and verbs. But this feature varies from language to language.

Most of the Indo-Aryan languages have gender agreement at the level of nouns and also at the level of verbs. Among them Sanskrit has three kinds of gender. In the Middle Indo-Aryan stage of language development, Pali and Prakrit show a two gender system. New Indo-Aryan languages can be classified into three groups based on their grammatical marking of gender.¹ Languages in which the gender is not grammatical, but it is only a semantic category, come under the first group, as for example languages of eastern Indo-Aryan subgroup such as Assamese, Bengali, and Oriya belong to this group. Languages which have a two-way grammatical gender, that is, masculine and feminine, constitute another group. Hindi and Punjabi may be cited as examples of this group. Languages which have three-way grammatical genders, like Marathi, Gujarati, and Bhili are in third group. Gender goes along with number and it is marked in various word classes like nouns, pronouns, adjectives, verbs, and, interestingly enough, with some of the post-positions that function as genitive, locative, and ablative markers. Generally the person-number suffixes also go together. These word categories show the

¹ <http://www.languageinindia.com/jan2002/sirmauri.html>

agreement of gender-number with other word categories in syntactic constructions, like phrases and sentences.

It is really necessary to ascertain the grammatical gender of the source language while doing translation, more so if the target language has gender agreement. Otherwise, the translation would be wrong. In the context of present R&D, Sanskrit is taken as source language and Hindi is assumed as a target language. Since both languages have gender agreement at the level of nominal forms, and Hindi has gender agreement at the level of the verb as well, it is critical that the gender analysis of Sanskrit, the source language be done.

2.1 Grammatical gender in Sanskrit

The word gender is used in Sanskrit language as *liṅga*. On the one hand, it means the characteristics of men and on the other hand it signifies grammatical gender. The terms referring to gender in Sanskrit - *pum̐sa*, *strī* and *napum̐saka* - which are found in later works first occur in Śatapatha Brāhmaṇa². The idea of gender is also very clear from a story about a battle of gods and demons.³ The Gopatha Brāhmaṇa⁴ mentions three grammatical genders while giving the definition of *avyaya*. The Nirukta (3.21) also clearly refers to the three genders. Using gender in language creates problem, because in life only animate objects have gender in terms of sex, but in language, both animate and inanimate objects have gender and the assignment of gender to the words usually does not follow the worldly rule. For example, in Sanskrit *jayā* and *dātrī* are used for wife in two different genders - feminine and masculine respectively.⁵ The gender assignment varies from language to language, for example, the word *agni* is used in Sanskrit in masculine gender, but the word *āga* bearing the same meaning in Hindi is used as feminine. It is really difficult to form some specific rules for gender assignment even in one language. Under the sutra *strīyām* (p.4.1.3) Patañjali has discussed gender in Mahābhāṣya. At first he refers to the worldly

² ŚB 1.3.1.9

³ Ibid 1.5.4.6-11

⁴ 1.1.26

⁵ Theory of Gender and its Generation in Pāṇini, Kapil Kapoor, Santosh Kumar Shukla pg-114 Sahitya Academy New Delhi 2002

view.⁶ Depending upon this rule, he argued that the word *bhrakuṃsa* etc will have to be considered as feminine because the meaning of the word is a male actor playing the role of female characters in drama. Consequently, after the word *bhrakuṃsa*, the feminine suffix *ṭāp* is added according to the rule *ajādyataṣṭāp* (P1.4.3).⁷ Though the worldly gender or *laukika liṅga* has no place in grammar, it does seem to have some bearing on the grammatical gender.⁸ Here through a *kārikā*⁹, the word *strī* and *puruṣa* are analyzed. Accordingly, *saṃsthāna* is disappearance (*tirobhāva*) and *prasavau* (*ā*) is manifestation. Now Patañjali did not say these specific words, it is according to Kaiyata's *pradīpa*.¹⁰ Now depending on *sāṃkhya* philosophy, these concepts were explained. These are qualities of *sattva*, *rajas* and *tamas*. Since these qualities are beyond the sense organs, the qualities like *śabda*, *sparśa*, *gandha* are meant here. All the substances do possess these qualities and thus the gender can be said to be universally present. But all the qualities are not present everywhere all the time. So where the *tirobhāva* is found it is regarded as having feminine gender. When *āvirbhāva* is noticed the masculine gender is considered and when both are noticed then it is considered as having neuter gender. Later the mention of *saṃgrakāra* by Helārāja supports this view.¹¹ Subramania Iyer explains it as 'all things are based on the following notions. All things are combinations of the five qualities *śabda*, *rasa*, *rūpa*, *sparśa*, and *gandha* which again are made up of *sattva*, *rajas*, *tamas*. Everything has therefore the properties of these qualities and their properties are: *prakāśa* (light), *prasaṃsā* (accretion), *āvirbhāva* (manifestation) for *sattva pravṛttiḥ* (alert principle), *kriyā*(action) for *rajas* and *varaṇam*, *tirobhāva* (decline), *sthitiḥ*(state) for *tamas*. These three properties are the three genders. The entire time qualities *rūpa* etc are constantly changing. But these changes are not perceptible to all. We see only their final result. Patañjali himself has declared that nothing remains as it is constantly changing like boiling water. Everything is constantly appearing and disappearing (*āvirbhāva* and *tirobhāva*) and these are the two main characteristics of the two

⁶ *stanakeśavatī strī syāt lomaś puruṣaḥ smṛta/ ubhyontaram yacca tadabhābe napuṃsakam* Mahābhāṣya(MBH)

⁷ *Semantic Perspective on Strīpratyas*, Prahlada Char pg-125 Sahitya Academy New Delhi 2002

⁸ pg-301 *Sankrit Vyakarana Darshan*

⁹ *saṃstānaprasavau liṅgamāstheyau svakṛtāntataḥ/sṃstāne syāyateirṭ strī sūte sap prasave pumān .*

¹⁰ *saṃstānamititirobhāvaḥ pravṛttirāvirbhāvaḥ.*

¹¹ *Theory of Gender and its Generation in Pānini*, Kapil Kapoor, Santosh Kumar Shukla pg-115 Sahitya Academy New Delhi 2002

main genders.’’¹² Patañjali and later Bhartṛhari accounted for the variability of gender of same object as expressed by different words in terms of the philosophy of flux. Many properties are constantly changing.¹³ At a given point of time according to which property is seen to be manifest or in ascendance, the particular gender is assigned and this gender may be actually seen in the object or it is superimposed as though it were seen and is nothing more than part of the correctness of the world.¹⁴ This is like the singularity. But the words are capable of expressing plurality. It is the same with gender.¹⁵ Thus according to differences in limiting conditions, a particular property or the *guṇas* becomes the cause of correctness and the object of *vivakṣā*.¹⁶ Gender in this way is only a factor in the formation of words.¹⁷ Gender is an aspect of meaning of the word. In *Vākyapadīya*¹⁸, Bhartṛhari has talked about seven kinds of thoughts concerning gender. According to him, gender is the relation of a thing which is signified by sex-signs such as breast and hair and the sex-signs themselves are characterized by that relation. The third view is that the gender is universally manifested by sex-signs. The next view is that gender is the three conditions of the three *guṇas*, such as *sattva*, *rajas* and *tamas* and it is the three *guṇas* themselves in this condition. The sixth view is that gender is an attribute created in objects by word. The last view is that gender is an attribute of words by themselves.¹⁹ While sex is part of the meaning of word, gender in Sanskrit is a property of the word-form that is why when a different word is used for the same object it may have a different gender. As Kātyāyana observes *ekārthe śabdānyatvāvad dṛṣṭam Liṅgānyatvam*. If the gender of the word and the gender of the object in the world happen to be the same it is seen as merely a matter of coincidence.²⁰ According to Peter M Scarf gender is considered to be a property of the object a word denotes.²¹ If a generic

¹² *Vākyapadīya* (vp) iii .ii.iii

¹³ *Vākyapadīya* 3-13-16

¹⁴ *Vākyapadīya* 3-13-28

¹⁵ *Vākyapadīya* 3-13-29

¹⁶ *Vākyapadīya* 3-13-24

¹⁷ *Vākyapadīya* 3-13-30

¹⁸ *Vākyapadīya* 3.13-43 sl

¹⁹ K.A . Subramania Iyer chapter III part II MLBD Delhi

²⁰ Theory of Gender and its Generation in Pānini, Kapil Kapoor, Santosh Kumar Shukla pg-119 Saranya Academy New Delhi

²¹ Scarf .M.Peter The Denotation of Generic Terms in Ancient Indian Philosophy: Grammar, Nyāya, and Mimāṃsa, *Transactions of the American Philosophical Society*, New Ser., Vol. 86, No. 3. (1996), pp.i-x+1-336./ <http://links.jstor.org/sici?sici=00659746%281996%292%3A86%3A3%3Ci%3ATDOGTI%3E2.0.CO%3B2-T>

term denotes a class property, the fact that the class property is permanently associated with a certain gender accounts for the consistent use of a generic term in a single gender. The class property always has its innate gender and the gender suffixes and inflectional terminations appropriate to that gender arise naturally after the nominal base of the generic term. As Peter scarf has observed, Patañjali's presentation of the grammarians' conception of gender under P.1.2.64²² is intended at solving the gender problem. This would indicate a limitation on the application of a speaker's intention. It highlights a property already present in the object, but it cannot superimpose a property which is absent. One does not make a horse a cow just by a speaker's intention that it is so. Similarly, one cannot make an object have a gender that is not present in it, by a speaker's intention. In other words, gender is still considered to be an objective. Different genders succeed on the view that a generic term denotes a generic property because a speaker may intend a generic property as having one gender in one instance and as having another gender in another instance. These different intentions are the conditions for the use of a generic term in different genders. In *vārttika* 54, *Gunavacanavad vā*, Kātyāyana says that gender and number terminations occur for generic terms as they do for quality words. According to Patañjali, different genders and numbers occur for words denoting qualities according to the substrata in which the quality resides." For example, in *śuklam vastram. śuklā śāṭī, śuklaḥ kambalaḥ*, quality word 'śukla' occurs in the neuter to refer to the quality white residing in a neuter substance, in the feminine to refer to the quality white residing in a feminine substance, and in the masculine singular to refer to the quality white residing in a masculine substance. P.1.2.51 *lupi yuktavad vyaktivacane*, provides that when a suffix is deleted (replaced by [up]) the object denoted by the derivate is considered to have the gender and number of the object denoted by the pre-suffixal base. The result is that gender and number suffixes occur for the derivate as they do for the base. *viśeṣaṇām cājāteḥ*, provides that qualifiers of the object denoted by the derivate are also considered to have the gender and number of the object denoted by the pre-suffixal base (*viśeṣaṇām ca*). It is the gender and number of an object other than that denoted by the nominal base. The gender and number terminations arise after a nominal base denotes a quality according to the gender and number of the individual substances, in which the quality

²² sarūpāṇāmekaśeṣa-ekavibhaktau

resides. Patañjali concludes that, the reason P1.2.52 is stated is, to inform us of the general principle of gender and number agreement for words denoting qualities: A word denoting a quality takes the gender and number of the substratum in which the quality resides²³

2.1.1 Gender in Aṣṭādhyāyī

Pāṇini has discussed the gender in the section of compound, for feminine suffix and in Līṅgānuśāsna (one of the ancillary texts among three). It appears from Pāṇini's work that his main concern seems to determine the gender of words, because without knowing the gender of words it is not possible to use the language. Rules 2.4.1-31 delineate the gender of compounds. The process of deriving feminine forms by affixation is laid down in 4.1.3 to 4.1.77. In the derivational process, the formation of the feminine takes place immediately before the process of nominal inflection begins. The feminine affixes are *tāp*, *cāp*, *dāp*, *nīp*, *nīṣ*, *nīn*, *uṅg*, *ti*. According to Kapoor and Shukla (2006) Pāṇini is seeking generalization and systematization in an area of grammar which is recognized to be arbitrary and not amenable to rules.

Through intensive phonological, morphological and semantic analysis, Pāṇini makes the feminine derivation almost completely rule bound by specifying the phonological, morphological and semantic conditions for the attachment of the affixes. Rules which state feminine suffixes are as follows:-

- 4.1.4 *ajādyataṣṭāp*
- 4.1.5 *ṛnacho nīp*
- 4.1.6 *ugitaś ca*
- 4.1.20 *vayasi prathame*
- 4.1.21 *dvigoḥ*
- 4.1.40 *anyato nīṣ*
- 4.1.41 *ṣidgaurādibhyaś ca*
- 4.1.44 *voto guṇavacanāt*

²³ "na tarhiddnim ayam yogo vaktavyah. vaktavyas ca. kimprayojanam? Idam tatra tatrocyate, guṇavacandnm sahddnc n dsrayatolitigavacandni hhavantiiti. tad anena kriyate." MB-K, vol. 1, p. 228, lines 19-21.

- 4.1.63 *jāterastrīṣayādayopadhāt*
- 4.1.65 *ito manuṣyajate*
- 4.1.66 *ūnutaḥ*

In the above mentioned part of Aṣṭādhyāyī, there are rules on the types of feminine suffixes, which are added after different nominal bases under different condition. Some rules prevent feminine suffixes to add after nominal bases under certain conditions. A brief survey of feminine suffixes is presented here, to give an overview of different feminine suffixes.

Name of the feminine affix	Clause	Example	special comment, if any
<i>tāp</i> ²⁴	occurs after nominal stem which is abstracted from the group headed by <i>ajā</i> , or after a stem which ends in <i>aṭ</i>	<i>ajā, eḍakā, caṭakā, aśvā</i> etc	
<i>tāp</i> ²⁵	occurs after nominal stem which ends in <i>pād</i> , when a hymn is signified	<i>dvipāda, tripāda, cauṣpāda</i> etc	
<i>nīp</i> ²⁶	occurs after nominal stems which end in <i>ṛ</i> and <i>n</i>	<i>kaṭṛ, haṭṛ, daṇḍinī, chaṭṛiṇi</i>	
<i>nīp</i> ²⁷	occurs after a form which ends in an <i>it</i> denoted by the abbreviatory term <i>uk</i>	<i>bhavaṭī, atibhavaṭī, pacantī, yajantī</i>	

²⁴ 4.1.4 *ajādyataṣṭāp*

²⁵ 4.1.9 *ṭābṛci*

²⁶ 4.1. *ṛnebhyo nīp*

²⁷ 4.1.5 *ugitaśca*

<i>nīp</i> ²⁸	occurs after a nominal stem which terminates in <i>van</i> , here <i>r</i> comes in the place of final sound of that which ends in <i>van</i>	<i>dhivarī, pivarī, sarvarī, paralokadṛśvarī</i>	<i>van</i> covers affixes <i>kvanip, vīnip, ṛvip</i>
<i>nīp</i> ²⁹	occurs after a nominal stem which ends in <i>pād</i>	<i>dvipādī, catuṣpādī</i>	optionally as <i>dviṣpāt, catuṣpāt</i>
<i>nīp</i> ³⁰	occurs after nominal stems which ends in <i>a</i> , and have their final, non secondary form either marked with <i>ṭ</i> , or ending in affixes <i>ḍha, aṅ, aṅ, dvaysac, mātrac, tayap, ṭhak, ṭhañ, kañ, kavaraṇ</i>	<i>kurucarī, madracarī, sauparṇeyī, kumbhakarī, śubhaṅgakarāṇī, tāḍṛśī</i> etc	
<i>nīp</i> ³¹	occurs after a non secondary nominal stem which ends in <i>yañ</i>	<i>gārgī, vatsī</i>	
<i>nīp</i> ³²	occurs after a non secondary nominal stem which ends in <i>a</i> and denotes the first stage of life	<i>kumārī, kiśorī, varkarī</i>	
<i>nīp</i> ³³	occurs after a non secondary nominal stem which ends in <i>a</i> and is termed <i>dvigu</i>	<i>pañcapūlī, daśapūlī</i>	

²⁸ 4.1.7 *vano ra ca*

²⁹ 4.1.8 *pādo 'nyatarasyām*

³⁰ 4.1.15 *ṭiddhāṅaṅdvayasajdaghnaṅmātracayapṭhakṭhaṅkvarapaḥ*

³¹ 4.1.16 *yañāśca*

³² 4.1.20 *vayasi prathame*

<i>nīp</i> ³⁴	occurs after a <i>bahuvrīhi</i> compound which ends in <i>ūdhas</i> , and begins either with a number or an indeclinable	<i>dvyūdhnī, nirūdhnī</i>	
<i>nīp</i> ³⁵	occurs after a <i>bahuvrīhi</i> compound which begins with a word denoting number and ends in <i>dāmana</i> or <i>hāyana</i>	<i>dvidāmanī, trihāyanī</i>	
<i>nīp</i> ³⁶	occurs after a <i>bahuvrīhi</i> compound which ends in <i>an</i> and anticipates the deletion of its penultimate sound	<i>bahurājini</i>	optionally
<i>nīp</i> ³⁷	occurs after a <i>bahuvrīhi</i> nominal stem which ends in <i>an</i> and anticipates the deletion of its penultimate sound, provided denotes a name or else is restricted to the vedic language	<i>surājiñī, dvidāmnī</i>	obligatory
<i>nīp</i> ³⁸	occurs after nominal stems <i>kevala, māmaka, bhāgadheya, pāpa, apara,</i>	<i>kevalī, sumagalī, pāpī, bheṣajī</i>	

³³ 4.1.21 *dvigoḥ*

³⁴ 4.1.26 *saṁkhyāvyayāderñīp*

³⁵ 4.1.27 *dāmahāyanāntācca*

³⁶ 4.1.28 *ana upadhālopinī'nyatarasyām*

³⁷ 4.1.29 *nīyam saṁjñāchandasoḥ*

³⁸ 4.1.30 *kevalmāmakabhāgadheya pāpapurarasasamānāryakṛtasumaṅgalabheṣajācca*

	<p><i>samāna,</i> <i>āryakṛta,</i> <i>sumaṅgala,</i> <i>bheṣaja,</i> provided derivatives denotes a <i>samjñā</i> or they are restricted to vedic language</p>		
<i>ñīp</i> ³⁹	<p>occurs after <i>rātri</i> when the derivative denotes a name, or constitutes an usages of vedic, provided the context involves an ending other than <i>jas</i></p>	<i>rātrībhiḥ</i>	
<i>ñīp</i> ⁴⁰	<p>occurs after nominal stem <i>pati,</i> when <i>n</i> to come in place of its final sound, when sacrificial connection is signified</p>	<i>yajamānasya patnī</i>	
<i>ñīp</i> ⁴¹	<p>occurs after a non <i>upasarjana</i> nominal stem which contains <i>pati</i> as its final constituent, ~used in combination after an initial constituent with the provision of <i>n</i> coming to replace the final sound of the base.</p>	<i>vṛdhapatnī, sthulapatnī</i>	optionally

³⁹ 4.1.31 *rātreścājasau*

⁴⁰ 4.1.33 *patyurnoyajñasamyoge*

⁴¹ 4.1.34 *vibhāṣā sapūrvasya*

<i>nīp</i> ⁴²	occurs after <i>pati</i> , provided <i>pati</i> is included in a nominal stem listed in the group headed by <i>sapatnī</i> , with an additional provision of <i>n</i> replacing its final sound	<i>sapatnī, ekapatnī</i>	obligatory
<i>nīp</i> ⁴³	occurs after non- <i>upasarjana</i> nominal stem <i>pūtakratu</i> , and its final <i>u</i> is replaced with <i>ai</i>	<i>pūtakratāyī</i>	
<i>nīp</i> ⁴⁴	occurs after non- <i>upasarjana</i> nominal stems <i>vṛṣakapi</i> , <i>agni</i> , <i>kusita</i> , <i>kusīda</i> , and the sound <i>ai</i> is marked as <i>udātta</i> come in place of their final vowel.	<i>vṛṣakapī, āgnāyī, kusitāyī, kusīdāyī</i>	
<i>nīp</i> ⁴⁵	occurs after nominal stem <i>manu</i> and <i>au</i> or <i>ai</i> marked <i>udātta</i> , come in the place of final vowel.	<i>manāyī, mānavī</i>	optionally
<i>nīp</i> ⁴⁶	occurs after a nominal stem which terminates in <i>a</i> , marked <i>anudātta</i> at the end, signifies <i>varṇa</i> , the penultimate <i>t</i> is replaced with <i>n</i>	<i>enī, śyenī, hariṇī</i>	optionally

⁴² 4.1.35 *nityam sapatnyādiṣu*

⁴³ 4.1.36 *pūtakratorai ca*

⁴⁴ 4.1.37 *vṛṣākapyagnikusitakusīdānām udāttaḥ*

⁴⁵ 4.1.38 *manorau vā*

⁴⁶ 4.1.39 *varṇādnudātāttopdhātto naḥ*

<i>nīp</i> ⁴⁷	occurs after a nominal stem which begins with a constituent denoting <i>diś</i>	<i>prāñnāsikī</i>	
<i>nīṣ</i> ⁴⁸	occurs after a <i>bahuvrīhi</i> compound which ends in <i>ūdhas</i>	<i>ghaṭodhnī, kuṇḍodhinī</i>	
<i>nīṣ</i> ⁴⁹	occurs after a non- <i>upasarjana</i> nominal stem which denotes <i>varṇa</i> and ends in <i>a</i> marked <i>anudātta</i> , even when it does not have <i>t</i> in its <i>upadhā</i>	<i>sāraṅgi, kalmāṣī</i>	
<i>nīṣ</i> ⁵⁰	occurs after non- <i>upasarjana</i> nominal stems which are either marked with <i>ṣ</i> as an <i>it</i> or are enumerated in the list headed by <i>gaura</i>	<i>nartakī, rajakī, gaurī, matsī, gārgāyanī</i>	
<i>nīṣ</i> ⁵¹	occurs after non- <i>upasarjana</i> nominal stems <i>jānapada</i> , <i>kuṇḍa goṇa</i> , <i>sthala</i> , <i>bhāja</i> , <i>nāga</i> , <i>kāla, nīla</i> , <i>kuśa</i> , <i>kāmuka</i> , <i>kabara</i> , when derivatives denote <i>vṛtti</i> , <i>amatra</i> ,	<i>jānapadī, kuṇḍī, kālī, kabarī</i>	

⁴⁷ 4.1.60 *dikpurvāpadān nīp*

⁴⁸ 4.1.25 *bahuvrītherūdhaso nīṣ*

⁴⁹ 4.1.40 *anyato nīṣ*

⁵⁰ 4.1.41 *ṣidgaurādibhyaśca*

⁵¹ 4.1.42 *jānapadakuṇḍagaṇṣthalabhājanāgakālānīlakuśakāmukakabarādvṛtyamatrāvaonākṛtrimāśrāñsthaulya-varṇāñcchādanāyovikāramaitihuncchākeśaveśeṣu*

	āvapana, akṛtrimā, śrāṇā,sthūlya, varṇa, anācchādāna, ayovikāra, maithuncchā, kabara respectively		
<i>nīṣ</i> ⁵²	occurs after a nominal stem which is constituted by śoṇa, and is not an <i>upasarjana</i>	<i>śoṇī</i>	according to eastern
<i>nīṣ</i> ⁵³	occurs after a non- <i>upasarjana</i> nominal stem which end in <i>u</i> and has sifnification of quality	<i>paṭvī, mṛdvī</i>	optionally
<i>nīṣ</i> ⁵⁴	occurs after nominal stems which are listed in the group headed by <i>bahu</i>	<i>bahvī</i>	optionally
<i>nīṣ</i> ⁵⁵	occurs after nominal stems which are listed in the group headed by <i>bahu</i> and are not termed as <i>upasarjana</i> in Vedic	<i>bahvī</i>	obligatory
<i>nīṣ</i> ⁵⁶	occurs after non <i>upasarjana</i> nominal stem which ends in <i>bhu</i> in Vedic	<i>parvī, vibhvī, sambhvī</i>	

⁵² 4.1.43 *śoṇāpraācām*

⁵³ 4.1.44 *voto guṇavacanāt*

⁵⁴ 4.1.45 *bahvādibhyaśca*

⁵⁵ 4.1.46 *nityam chandaśi*

⁵⁶ 4.1.47 *bhuvaśca*

<i>nīṣ</i> ⁵⁷	occurs after nominal stem which ends in <i>a</i> and signifies a male relation to whom a female is to be expressed	<i>gaṇakī, praṣṭhī</i>	
<i>nīṣ</i> ⁵⁸	in association with a corresponding male, after non- <i>upasarjana</i> nominal stems <i>indra, varuṇa, bhava, śarva, mṛḍa, rudra, hima, ariya, yava, yavana, mātula, ācārya</i>	<i>indrāṇī, bhavānī, śarvānī, rudrāṇī, mṛdāṇī, himānī, ariyānānī, yavanī, yavanānī, mātulānī, ācāryānī</i>	additionally they also received augment ānuk
<i>nīṣ</i> ⁵⁹	occurs after a nominal stem which ends in <i>krīta</i> and is used in combination after an initial compound constituent with the signification of <i>karāṇa</i>	<i>vasanakrīṭī</i>	
<i>nīṣ</i> ⁶⁰	occurs after a non- <i>upasarjana</i> nominal stem which ends in <i>ka</i> and is used in the combination after an initial constituent denotes <i>karāṇa</i> providing the derivative denotes <i>alpa</i>	<i>abhraviliptī, sūpaviliptī</i>	

⁵⁷ 4.1.48 *pūnyogādākhyāyām*

⁵⁸ 4.1.49 *indravaruṇabhavaśarvarudramṛḍahimāranyayavayavanamātulācāryāṇāmānuk*

⁵⁹ 4.1.50 *krīṭākarāṇapurvāt*

⁶⁰ 4.1.51 *ktādalpākhyāyām*

<i>nīs</i> ⁶¹	occurs after a <i>bahuvrīhi</i> nominal stem which has a constituent ending in <i>ka</i> as its final and is also marked <i>udātta</i> at the end	<i>keśalunī, galakotkr̥tī</i>	
<i>nīs</i> ⁶²	occurs after a nominal stem termed <i>bahuvrīhi</i> which ends in <i>ka</i> , has its final vowel marked <i>udātta</i> and does not contain an initial constituent denoting <i>svāṅga</i>	<i>sārṅgajagadhī, palāṅdubhakṣitī</i>	optionally
<i>nīs</i> ⁶³	occurs after nominal stem which contains an <i>upasarjana</i> used as its final constituent with the denotatum of <i>svāṅga</i> and which does not contain a conjunct in its <i>upadhā</i>	<i>candramukhī</i>	optionally
<i>nīs</i> ⁶⁴	occurs after a nominal stem which ends in an <i>upasarjana</i> with the signification of <i>svāṅga</i> namely <i>nāsikā, udara, oṣṭha, jaṅghā, danta, karṇa śrīṅga</i>	<i>tuṅganāsikī, tilodarī, bimboṣṭhī, cārudantī</i>	optionally

⁶¹ 4.1.52 *bahuvrīheścāntodattāt*

⁶² 4.1.53 *asvāṅgapūrvapadādvā*

⁶³ 4.1.54 *svāṅgāccopasarjanādasamyogopadhāt*

⁶⁴ 4.1.55 *nāsikoddaraouṣṭhajaṅghādantakarṇaśrīṅgācca*

<i>nīṣ</i> ⁶⁵	occurs via <i>nipātana</i> , to derive <i>dīrghajihvī</i> in Vedic, when the derivate denotes a female name	<i>dīrghajihvī</i>	
<i>nīṣ</i> ⁶⁶	occurs in Vedic after a nominal stem which ends in a non-secondary constituent, namely <i>vāha</i>	<i>dityauhī, praṣṭhauhī</i>	
<i>nīṣ</i> ⁶⁷	occurs via <i>nipātana</i> to derive <i>sakhī</i> and <i>aśiśvī</i> in language	<i>sakhī, aśiśvī</i>	
<i>nīṣ</i> ⁶⁸	occurs after a nominal stem which ends in <i>a</i> with the significance of <i>jāti</i> , is not obligatorily limited to the scope of feminine and does not have <i>y</i> in its penultimate position	<i>kukkuṭī, mayūrī, brāhmaṇī, nāḍāyanī</i>	
<i>nīṣ</i> ⁶⁹	occurs after a nominal stem which denotes <i>jāti</i> , and has <i>pāka, karṇa, parṇa, puṣpa, phala, mūla</i> , and <i>bāla</i> in the end	<i>odanapāki, śaṅkukarṇī, dāsīphalī, darbhamūlī</i>	

⁶⁵ 4.1.59 *dīrghajihvī ca cchandasi*

⁶⁶ 4.1.61 *vāhaḥ*

⁶⁷ 4.1.62 *sakhyāśiśvīti bhāṣāyām*

⁶⁸ 4.1.63 *jāterstrīviṣyādyopadhāt*

⁶⁹ 4.1.64 *pākakarṇaparṇapuṣpaphalamūlabālottrapadācca*

<i>nīṣ</i> ⁷⁰	occurs after a non-secondary nominal stem which ends in <i>i</i> and denotes a class of human	<i>avantī, dākṣī</i>	
<i>ḍāp</i> ⁷¹	occurs after a nominal stem which ends in <i>man</i> and after a <i>bahuvrīhi</i> compound which ends in <i>an</i>	<i>pāmā, sīmā</i>	optionally
<i>cāp</i> ⁷²	occurs after a nominal stem which ends in affix <i>yañ</i>	<i>kausalyā, kariṣagandhyā</i>	
<i>cāp</i> ⁷³	occurs after non-upasarjana nominal stem <i>āvātya</i>	<i>āvātyā</i>	
<i>nīn</i> ⁷⁴	occurs after a non-secondary nominal stem which denotes <i>jāti</i> , and is either listed in the group headed by <i>śārṅgarava</i> or terminates in <i>a</i> of affix <i>añ</i>	<i>śārṅgaravī, auravī</i>	
<i>ūn</i> ⁷⁵	occurs after a nominal stem which end in <i>uṭ</i> , does not have a <i>y</i> in its penultimate position and signifies a class of human	<i>brahmabandhūh, vīrabandhūh</i>	

⁷⁰ 4.1.65 *ito manūṣyajāteḥ*

⁷¹ 4.1.13 *dābubhāyāmyatarasyām*

⁷² 4.1.74 *yañāścāp*

⁷³ 4.1.75 *āvātyāśca*

⁷⁴ 4.1.73 *śārṅgaravādyaño nīn*

⁷⁵ 4.1.66 *ūnutaḥ*

<i>ūñ</i> ⁷⁶	occurs after a nominal stem which ends in <i>bāhu</i> , provided the derivate denotes a name	<i>bhadrabāhūḥ, jālabāhūḥ</i>	
<i>ūñ</i> ⁷⁷	occurs after <i>paṅgu</i>	<i>paṅgūḥ</i>	
<i>ūñ</i> ⁷⁸	occurs after a nominal stem which contains <i>ūru</i> , as its final constituent, provided the derivative signifies comparison	<i>kadalīstambhorūḥ</i>	
<i>ūñ</i> ⁷⁹	occurs after a nominal stem which contains <i>ūru</i> , as its final constituent, and is used in combination after an initial constituent, namely <i>saṃhitā, śapha, lakṣaṇa, vāma</i>	<i>saṃhitorūḥ, lakṣṇorūḥ, vāmorūḥ</i>	
<i>ūñ</i> ⁸⁰	occurs after in Vedic nominal stems <i>kadru</i> and <i>kamaṇḍalu</i>	<i>kadrūḥ</i>	
<i>ūñ</i> ⁸¹	occurs after nominal stems <i>kadru</i> and <i>kamaṇḍalu</i> , when derivatives denote a name	<i>kadrūḥ, kamaṇḍalu</i>	
<i>tī</i> ⁸²	occurs after nominal stem which signifies <i>yuvan</i>	<i>yuvatīḥ</i>	taddhita

⁷⁶ 4.1.67 bāhvantātsaṃjñāyām

⁷⁷ 4.1.68 paṅgośca

⁷⁸ 4.1.69 urūttaraṇapadādaupamyē

⁷⁹ 4.1.70 saṃhitaśaphalakṣaṇavāmādeśca

⁸⁰ 4.1.71 kadrūkamaṇḍalvośchandasi

⁸¹ 4.1.72 saṃjñāyām

⁸² 4.1.77 yūnastīḥ

According to George Cardona (1997)⁸³, Pāṇini simply lets particular affixes occur with certain nominals if they are used in feminine. The affixation rules which provide for this are not put in terms of agreement. Indeed, there are no explicit concord rules for number and gender agreement in the Aṣṭādhyāyī, and Paṇini's derivational system does not require such rules.

2.1.1.1 Gender rules in Liṅgānuśāsana

Generally, in Liṅgānuśāsana, there are eight methods to determine the gender of word, such depending on *samāsa* (compound), on special meaning, on specific word, on *abhidhāna* (meaning), *upadhā*, the last sound, the affixes, and on worldly gender. In the *tatpuruṣa* and *karmadhāraya* compounds, gender depends on the later *pada* of *samāsa*.⁸⁴ For example *ātmano jñānam*, *ātma-jñānam*. Here in the compound *ātmajñānam*, the gender of the later word (*jñānam*) is neuter, so the gender of the whole word becomes neuter. In *dvigu samāsa*, some words have gender of feminine and some words have gender of masculine.⁸⁵ The word having *dvigu samāsa* gets neuter gender if the word means collocation,⁸⁶ for example *pañcagavam*. If the *dvigu samāsa* word ends in 'a' then the gender of the word would be feminine,⁸⁷ and the word gets the feminine suffix *nīp*⁸⁸, as for example *trilokī*. But if the last word in *samāsa* is *pātra*, *bhuvana*, *yuga* etc then instead of getting feminine gender it gets neuter gender,⁸⁹ as for example *pañcapātram*. In the case of *itaretara dvanda samāsa*, (*samāhara*, *ekvat* are different) the word gets the gender according to the gender of the last word,⁹⁰ for example *kukkuṭamayūrau*. In the case of *samāhara dvandva*, the gender of the word would be neuter,⁹¹ for example *saṃjñāparibhāṣām*. When the words transform into *dvandva samāsa*, with the meaning of *samāhara* only then the word is used in singular number and the gender of the word would be neuter⁹², as for example *pāṇipādam*. In the case of *bahuvrīhi samāsa* the word

⁸³ George Cardona, 1988 Pāṇini, His Work and its Traditions, vol ... i (Delhi: MLBD, 1988)

⁸⁴ P 2.4.26 *paravallīṅgaṃ dvandatatpuruṣayoḥ*

⁸⁵ Liṅgānuśāsana (Ing)133- *dviguh striyām ca*

⁸⁶ P 2.4. 17 *sa napuṃsakam*

⁸⁷ vārttika of P 2.4.17

⁸⁸ P 4.1.21 *dvigoḥ*

⁸⁹ vārttika of P 2.4.17

⁹⁰ P 2.4.26 *paravallīṅgaṃ dvandatatpuruṣayoḥ*

⁹¹ P 2.4.17 *sa napuṃsakam*

⁹² Ing 124 *dvandaiekatvam*

means other things except the meaning of words which constitute the *samāsa* word. The gender of the *bahuvrīhi samāsa* word would be according to the meaning of the resulting word. It can be masculine, feminine or neuter, depending on the meaning of the word.⁹³ The word *caturmukhaḥ* means a god having four faces. Here the word means another thing other than the meaning of four or face. The word god is in masculine gender; as a consequence the word gets the masculine gender. The example of feminine gender is *vidhavā*, and the example of neuter gender is *nirarthakam*. There are some words which are used in specific *samāsa* and in specific position. Then there are differences in assigning gender to those words as opposed to general rule. In *tatpuruṣa samāsa*, if the word *aṅga* is used as last word and the word *apa* is used as the first word then instead of getting the gender of the last word it gets masculine gender.⁹⁴ In any *samāsa*, if the last word is *aha* (*n*) then the *samāsa* gets the masculine gender.⁹⁵ In the *karmadhāraya samāsa* if the first word is *punya* (*śubha*, *maṅgala*) and the last word is *aha*(*n*) then instead of getting the gender of the last word, the gender of the word would be neuter.⁹⁶ If in any *samāsa* which is having any word as the first word, *ahna* is used as the last word then the gender of the word would be masculine⁹⁷. In *dvandva samāsa* if the first word is *vaḍavā* and *aśva* is used as the last word then instead of getting the gender of last word, the feminine gender is assigned to the word.⁹⁸ In the same *samāsa* if the same last and first words change their position respectively then the gender of the word would be masculine.⁹⁹ In *tatpuruṣa samāsa* with any word as first word and if the last word is *upakrama* or *upajñā* then the gender of the word would be neuter.¹⁰⁰ In the same kind of *samāsa* if *ūrṇā*, *śaśa* are used as the last word and the first word respectively the word gets neuter gender.¹⁰¹ If the word *chāyā* is used with the meaning of large in *tatpuruṣa samāsa* as a last word then the gender of that word would be neuter.¹⁰² The same word is used in *tatpuruṣa samāsa* or in any other *samāsa* as the last word

⁹³ lng 185 *guṇavacanam ca*

⁹⁴ lng 107 *nāḍyapajanopapadāni braṇāgapadāni*

⁹⁵ P 2.4.39 *bahulam chandasi*

⁹⁶ lng 131 *apatha punyāhe napuṃsake*

⁹⁷ p 2.4.39 *bahulam chandasi*

⁹⁸ p 2.4.27 *pūrvavadaśca vaḍavau*

⁹⁹ ibid

¹⁰⁰ p 2.4.21 *upajñopakramam tadādyācikyāsāyām*

¹⁰¹ lng 22 *grhaśaśāmyām klībe*

¹⁰² lng 127 *analpe chāyā*

other than the meaning of large, the word can be used either in neuter gender or in feminine gender.¹⁰³ In *tatpuruṣa samāsa* if the word *niśā* is used as the last word then according to the general rule the gender of the word would be feminine, but along with it the word can be used in neuter gender also.¹⁰⁴ In *nañ-ītatpuruṣa samāsa* if the first word is *a(nañ)* and *patha* is used as the last word then the word gets neuter gender.¹⁰⁵ In *tatpuruṣa samāsa* the words *pada* and *jano* are used as last word and the first word respectively then the gender of the word would be masculine.¹⁰⁶ According to the same rule the whole word in the same *samāsa* gets masculine gender if *braṇa* and *nāḍī* are used as the last *pada* and the first *pada* respectively in that specific *samāsa*. In *tatpuruṣa samāsa* or in any other *samāsa* having any word as the first *pada*, if the word *rajju* is used as the last *pada* then the whole word can be used in both feminine and neuter.¹⁰⁷ In *tatpuruṣa samāsa* with any word as the first *pada*, if the word *śālā* is used as the last *pada* in that *samāsa*, the word can be used in both feminine and neuter gender.¹⁰⁸ In the same *samāsa* if the word *sabhā* is used as the last *pada* and in the first *pada* if the word *rājā* and its synonymous words and the words *rākṣasa* etc other than human beings are used then that *tatpuruṣa samāsa* gets the neuter gender.¹⁰⁹ In *tatpuruṣa samāsa* if *sthūṇā* and *grha* are used as last and first *pada* respectively then the gender of the whole word would be neuter.¹¹⁰ In *tatpuruṣa samāsa* or in any other *samāsa*, with any word as the first *pada* if the word *surā* or *senā* is used as the last word, then the whole word can be used in both feminine and neuter gender.¹¹¹

In *dvigu samāsa* if the word *rātri* is used as the last *pada* and the synonyms of the *saṃkhyā* are used as the first word then the gender of the whole word would be neuter.¹¹² Any *samāsa* with

¹⁰³ Ing 129 *surāsenācchāyāśālāniśāḥ striyām ca*

¹⁰⁴ ibid

¹⁰⁵ Ing 13 *lāpathapūnyāhe napuṃsake*

¹⁰⁶ Ing 107 *nāḍyapajanopapadāni braṇāgapadāni*

¹⁰⁷ Ing 53 *samāse rajjuḥ puṃsi ca*

¹⁰⁸ Ing 129 *surāsenācchāyāśālāniśāḥ striyām ca*

¹⁰⁹ Ing 128 *rājāmanuṣyapūrvā sabhā*

¹¹⁰ Ing 22 *grhaśaśāmyām klībe*

¹¹¹ Ing 129 *surāsenācchāyāśālāniśāḥ striyām ca*

¹¹² Ing 132 *saṃkhyāpūrā rātriḥ*

any word as its first *pada* and the word *rātra* is used as the last *pada* then the whole word is used in masculine gender.¹¹³

In Sanskrit language there are some words used in different meanings. The gender of those words gets changed according to their meaning. In Pāṇini's Līṅgānuśāsana, there are lists of some of these words. The word *akṣa* in the sense of eye as an organ is used in neuter gender.¹¹⁴ The same word in another sense is used in masculine.¹¹⁵ The word *kaṃsa* in the sense of king is used in masculine,¹¹⁶ but in the sense other than living being, the same word can be used either in masculine or in neuter. The word *kāṣṭhā* in the sense of direction is used in feminine¹¹⁷ but in the sense of wood it is used in neuter.¹¹⁸ The word *duṇḍubhi* in the meaning of a pair of three spots on a die is used in feminine.¹¹⁹ The same word in the meaning of large kettle-drum and as a demon the word is used in masculine.¹²⁰ The word *nābhi* in the meaning of naval is used in feminine;¹²¹ the same word in the meaning of musk is used also in feminine. The same word in the meaning of *kṣatriya* is used in masculine.¹²² Other than those mentioned above, in all other meanings the same word is used in masculine. The word *vasu* in the sense of wealth is used in neuter¹²³. The same word in the meaning of 'ray of light', 'light' and 'a medicinal root' is used in feminine. Except these meanings, in all other meanings the same word is used in masculine.¹²⁴ In the meaning of 'ray of light', the word can be used in both feminine and masculine. The word *śukra* in the meaning of planet and preceptor of demons is used in masculine,¹²⁵ in the meaning of semen virile it is used in neuter,¹²⁶ in the sense of white the same word can be used in masculine or in feminine or in neuter. Other than these seven words there are many words in Sanskrit language, whose gender gets changed according to the meaning.

¹¹³ p 2.4.39 *bahulam chandasi*

¹¹⁴ lng 169 *akṣamindriye*

¹¹⁵ lng 93 *ṣopadha*

¹¹⁶ lng 96 *sopadha*

¹¹⁷ lng 72 *kāṣṭhā digaryā strīyām*

¹¹⁸ lng 71 *kāṣṭhaprṣṭhasikthokathāni napuṃsake*

¹¹⁹ lng 14 *duṇḍubhirakṣeṣu*

¹²⁰ lng 16 *ubhāvapyanytra puṃsi*

¹²¹ lng 15 *nābhirakṣtriye*

¹²² lng 16 *ubhāvapyanytra puṃsi*

¹²³ lng 55 *vasu cārthavāci*

¹²⁴ lng 51 *dhenurajjukuhūsarayutanureṇupriyanīgavaḥ strīyām*

¹²⁵ lng 89 *ropadhah*

¹²⁶ lng 91 *śukramadevatāyām*

In Liṅgānuśāsana there are mentions of individual words along with their gender. For example, the word *aṃśa* is mentioned in masculine gender. (if the word in any gender is used as a name, then it follows *laukika liṅga*.) There are some words whose name or *abhidhāna* are used in same gender. As for example *bhūmi*, *vidyut*, *sarit*, *latā vanitā*¹²⁷ these words or their paryāya are used in feminine gender. Like these there are mentions of masculine and neuter gender also.¹²⁸ The gender can be recognised from *upadhā*¹²⁹ also. *Upadhā* is a technical term of Sanskrit grammar; it means the penultimate sound of a word. For example in the word *bālaka* 'k' is the *upadhā*. The word *bālaka* is *kopadha*, because it's *upadhā* is 'k' and this *kopadhā* word is used in masculine gender if the word is *a* ending.¹³⁰ Under the same condition, all the words whose *upadhā* are *ṭ*¹³¹ or *ṇ*¹³² or *th*¹³³ or *n*¹³⁴ or *p*¹³⁵ or *bh*¹³⁶ or *m*¹³⁷ or *y*¹³⁸ or *r*¹³⁹ or *ṣ*¹⁴⁰ or *s*¹⁴¹ are used in masculine gender. The word whose *upadhā* is *l*¹⁴² is used in neuter gender.

Sometimes the genders of words depend on the last sound of the word. This can be divided into one syllable and more than one syllable. Both of these categories are further more divided into vowel ending and consonant ending. Under one vowel ending sound category words included are - *mātr*, *duhitr*, *svasr*, *potr*, *nanādr*. Except these five words, other words whose end syllable is *r* are used in masculine or neuter gender. These five words are used in feminine gender.¹⁴³ Generally words ending in *u* are used in masculine gender,¹⁴⁴ as for example *sādhu*. But if words ending in *u* are used as adjectives, then the gender depends on nouns and the word can be used in

¹²⁷ Ing 18 *bhumividyutsarillatāvanitābhidhānāni*

¹²⁸ Ing 43, 49, 100, 103, 137, 157

¹²⁹ P.1.1.65 *alo 'ntyātpurva upadhā*

¹³⁰ Ing 61 *kopadhāḥ*

¹³¹ Ing 64 *ṭopadhāḥ*

¹³² Ing 67 *ṇopadhāḥ*

¹³³ Ing 70 *thopadhāḥ*

¹³⁴ Ing 74 *nopadhāḥ*

¹³⁵ Ing 77 *popadhāḥ*

¹³⁶ Ing 80 *bhopadhāḥ*

¹³⁷ Ing 83 *mopadhāḥ*

¹³⁸ Ing 86 *yopadhāḥ*

¹³⁹ Ing 89 *ropadhāḥ*

¹⁴⁰ Ing 93 *ṣopadhāḥ*

¹⁴¹ Ing 96 *sopadhāḥ*

¹⁴² Ing 141 *lopadhāḥ*

¹⁴³ Ing 3 *ṛkārāntāmātrduhitṛsvasṛpotṛnanāndārāḥ*

¹⁴⁴ Ing 51 *ukārāntāḥ*

three genders.¹⁴⁵ As for example, the word *madhu* with the meaning of honey is used in neuter gender. The same word with the meaning of spring and a name of a demon is used in masculine gender. Generally the words ending in *na* syllable is used in masculine,¹⁴⁶ as for example *guṇin*, *rājan*, *vṛtrahan* etc. Words which end in *tu*¹⁴⁷ are used in masculine gender, as for example *dhātu*. Words which end in *tra* are used in neuter gender¹⁴⁸, as for example *patra* and the words which end in *ru* are used in masculine gender¹⁴⁹, as for example *taru*. Words which end in *is* and *us* are used neuter gender, as for example *sarpis*, *trapus* respectively.¹⁵⁰ In some specific condition depending on *taddhita* suffix and *kr̥ta* suffix the gender of word can be recognise. *Taddhita* suffixes are other than *sup* and they are added to words. Feminine suffixes are part of this. *Kṛta* suffixes are other than *tin̄* and they are added to verb roots. *Uṇādi* and *kr̥tya* suffixes come under this suffixes. For example-

Affix	Definition	Added after root or base	In the meaning	Any special condition	Gender
<i>ac</i>	<i>kr̥t</i>	root	in the sense of action		masculine ¹⁵¹
<i>aṅ</i>	<i>taddhita</i>	base	in the sense of action & karma		neuter ¹⁵²
<i>āni</i>	<i>kr̥t-uṇādi</i>	root			feminine ¹⁵³
<i>as</i>	<i>kr̥t-uṇādi</i>	root		if in the whole word there are only two vowels	neuter ¹⁵⁴

¹⁴⁵ lng 173 *guṇavacanamukārantam napuṃsakam ca*

¹⁴⁶ lng 48 *nāntaḥ*

¹⁴⁷ lng 57 *rutvantaḥ*

¹⁴⁸ lng 153 *trāntaḥ*

¹⁴⁹ lng 57 *rutvantaḥ*

¹⁵⁰ lng 134 *isusantaḥ*

¹⁵¹ lng 37 *ghājantāśca*

¹⁵² lng 123 *yadyadhgyogañ-aṅ-vuñchāśca bhāvakarmani*

¹⁵³ lng 4 *anyūpratyaṅānto dhātuḥ*

<i>kr̥tya</i>	<i>kr̥t</i>	root	in the sense of action & karma		neuter ¹⁵⁵
<i>cha</i>	<i>taddhita</i>	base	in the sense of action & karma		neuter ¹⁵⁶
<i>tal</i>	<i>taddhita</i>	base	in the meaning of action and svārtha etc		feminine
<i>mi</i>	<i>kr̥t-uṅādi</i>	root			feminine ¹⁵⁷
<i>lyut</i>	<i>kr̥t</i>	root	in the sense of action		neuter ¹⁵⁸
<i>lyut</i>	<i>kr̥t</i>	root	in sense of instrumental & location		feminine, masculine ,neuter ¹⁵⁹

There are other methods to mark gender in Liṅgānuśāsana, such as the gender of adjectives, which are according to the gender of nouns. It can be masculine or feminine or neuter.¹⁶⁰ The gender of modifiers of adjective in a sentence is used in neuter, for example *rāmaḥ atyantam paṭuḥ asti* in this sentence the word *atyantam* is the modifier of the adjectives of *paṭuḥ*. Most of the numerals have forms in three genders. Forms of pronouns are also different in three genders, except the words *asmad*, *yuṣmad*. These words have one form only, and it is said that the gender is not expressed in them. Some genus of fruit is used in neuter gender, for example *āmra*, but

¹⁵⁴ Ing 151 *asanto dvayackaḥ*

¹⁵⁵ Ing 187 *karaṇādhikaraṇyoryaṭ*

¹⁵⁶ Ing 123 *yadyadhgyagañ-aṅ-vuñchāśca bhāvakarmani*

¹⁵⁷ Ing 6 *minyantaḥ*

¹⁵⁸ Ing 119 *bhāve lyudanta*

¹⁵⁹ Ing 187 *karaṇādhikaraṇyoryaṭ*

¹⁶⁰ Ing 185 *guṇavacanam ca*

kadali, *vadari* etc fruits are used in masculine gender.¹⁶¹ Some genus of tree are used in feminine, but with an exception, same as previous.¹⁶² The gender of *anukāraka śabda* is used in neuter, as for example '*rāma!*' *iti uktam tena* . The genders of adverbs are neuter, as for example *mandam vahati*.

2.2 Gender in Hindi

Hindi has two genders - masculine and feminine. Hindi verbs are inflected with respect to gender of the subject (masculine, feminine). In a Hindi sentence there are gender agreements between subject and verb and also among nouns, especially between adjectives and nouns. When pronoun is used in the place of noun, then also there is gender agreement between pronoun and verb. As for example Ram *calegā*¹⁶³, (Hindi (McGregor 1972),¹⁶⁴ here the subject *rām* is in masculine, so there are agreement between subject and verb. From the verb *calegā* one can understand that it is used in masculine gender. Object agreement in Hindi involves agreement in number and gender and not in person, while subject agreement can involve agreement in person, number, and gender. Here the structure of gender agreement would be masc & masc = masc, masc & fem = masc, fem & fem = fem. Tense heads a functional projection and hosts person-number features, while aspect heads another functional projection with number-gender features. Aspect forms express a pragmatic relation, the speaker's decision to view an event as bounded or in progress. The morphological expression of aspect in Hindi is by suffixes. As for example *Rāhul kitāb paRḥ-tā thā*. The past tense auxiliary, being derived historically from a participial form, agrees only in number and gender and not in person. The present tense auxiliary agrees in person and number but not in gender. The future agrees in person, number, and gender. The future tense in Hindi is marked by a single synthetic form which realizes person, gender and number, but the present tense is marked by a participle that realizes number and gender and an auxiliary that realizes person and number, while the simple past tense is marked by a participle that marks gender and

¹⁶¹ Ing 161 *phalajāti*

¹⁶² Ing 162 *vrkṣajātiḥ strīyāmeva*

¹⁶³ <http://72.14.235.104/search?q=cache:8LB3pp3f5QJ:www.essex.ac.uk/linguistics/LFG/wwwlfg.stanford.edu/lfg2004/school/material/agreement/finalday2.pdf+gender+agreement+in+Hindi+grammar&hl=en&ct=clnk&cd=3&gl=in>

¹⁶⁴ McGregor, R. S. 1972. *Outline of Hindi Grammar*. Oxford/Delhi: Oxford University Press.

number with no auxiliary and no expression of person. One can see this through Sanskrit sentences and subsequent Hindi translation of those sentences. In the case of larger noun phrases, in a sentence if one single element is mentioned in masculine and others are in feminine, then while agreement with verb, it would take the gender of masculine.

As for example-

SNS1- *vimale! etām ujjvalām mālām tvam etasyai guṇavatyaī bālikāyaī katham na yacchasi?*

HS1- *vimalā yaha ujjvala mālā tu isa guṇavatī laḍakī ko kyūṅī nahī detī hai?*

In Sanskrit sentence ‘*etām ujjvalām*’ are adjectives of *mālā*, so its take the same gender of *mālā*. Here it is feminine. But in Hindi, the adjectives of *mālā* are not in feminine, but adjective of *laḍakī* in both language is taking the gender (here feminine) of noun. In Hindi there is agreement of between verb and objects in terms of gender.

SNS2 - *etāni svādūni miṣṭānnāni cañcalā uṣā sāntāyaī svapnāyaī dāsyati*

HS2 – *ye svādiṣṭa miṣṭhāiyāṃ cañcala uṣā sānta svapnā ko degī*

Here in Sanskrit sentence *miṣṭāni* is used in neuter so its adjective is used in neuter. But in Hindi it is used in feminine gender. Here also adjectives in Hindi sentence of *uṣā* and *svapnā* are not taking nouns gender, because these words are used in Hindi in masculine gender only. Like the previous sentence Hindi sentence has gender agreement between verb and noun.

SNS3 - *ātmani paramātmāni ca ayameva bheda yad ātmā alpajña asti, paramātmā ca sarvajña.*

HS3- *ātmā aur paramātmā meṃ bheda yahī hai ki ātmā alpajña hai aur paramātmā sarvajña hai*

Here the major difference between in these two sentences that in Sanskrit the word *ātmā* is used in masculine but in Hindi the same word is used in feminine

SNS4- *ayodhyārājasya cvatvāraḥ sūnavah babhūbuḥ*

HS4- *ayodhyā ke rājāke cāra ladake the*

SNS5- *he ṛtvijau! yuvām srajau dhāraytam maṅtrān ca uccārayatam*

HS5- *he ṛtvijo! āpa donoṃ mālā dhāraṇa karo aur maṅtra bolo*

SNS6- *asyaṃ sarisi tasya parivārasya duhitarah, tanayāḥ, vadhvāḥ ca snāsyanti*

HS6- *isa nadī meṃ usa parivār kī laḍakiyā, laḍake, bahueṃ nahāyeṃge*

Here in Sanskrit sentence there is no agreement between object and verb, but in Hindi sentence one object is in masculine, so the verb of whole sentence takes masculine gender.

SNS7- *eteṣāṃ vastūnāṃ yojanāya jatunaḥ śakalāni āhara*

HS7- *ina vasuoṃ ko joḍane ke liye lākha ke tukḍe la*

In these two sentences, in the first sentence, word *vastu, jatu, śakala* are used in neuter gender. In Hindi these words are used in feminine and masculine gender respectively.

SNS8 -*mahatāṃ karmaṇāṃ, katṛṇāṃ nāmāni jagati prathitāni bhavanti*

HS8-*baḍe baḍe kāma karne vāloṃ ke nāma jagat meṃ prasidha ho jāte haiṃ*

In this example in Sanskrit the word *jagata* is used in feminine, whereas in Hindi it is used in masculine.

SNS9 - *yusmādrśah janāḥ eva kanakabhūṣaṇasaṅgrānocitān maṅṅin trapuṣi pratyabadhnan*

HS9 -*tum jaise logoṃ ko ne hī son eke gahanoṃ jaḍane yogya maṅiyom ko katīra meṃ phit kiyā hai*

In these two sentences in two different languages, there are differences regarding gender assignment. As for example in a Sanskrit sentence *maṇi* and *trapus* are used in masculine and neuter respectively. But in Hindi, the parallel words of these two are used in feminine and masculine gender respectively.

SNS-10 *gāvaskareṇ sadṛkṣā daṇḍapaṭṭakrīḍakāḥ bhuvī viralāḥ santi*

HS10- *gāvaskar jaise ballebāja saṃsārme bahut kama haiṃ*

Here the word *bhuva* is used in feminine, where as in Hindi *saṃsār* is used in masculine.

SNS11- *prakṣālitāni tu etāni vastrāṇi anyādr̥śāṇi eva samavarttiṣata*

HS11- *dhone para to ye kapḍe dusre se hogaye*

In this example in the Sanskrit sentence word *vastra* is used in neuter gender, while the same word in the Hindi sentence is used in masculine gender.

SNS12- *rājapathe gacchataḥ pathikān hastau prasārya yācataḥ bhakṣayatām manuṣyānām karebhyaḥ patanti patrāṇi ledhum śvabhiḥ saha dhāvataḥ ca bālakān vālikāḥ dr̥ṣṭvā api śāsakānām hṛdayeṣu duḥkham na utpadyate*

HS12- *saḍaka par jāte huye rāhagīrom se hāth phailā kara māṃgte huye aur khāte huye manuṣyom ke hātho se girte huye pattom ke chātne ke liye kuttom ke sātha daurtem huye laḍakom ko dekhkar bhi gavarnment ke hṛdayame mem duḥkha nahi hotā*

In sentence no 12 in Sanskrit, words *rājapatha*, *patra*, *hṛdaya* are used in masculine and neuter gender respectively, but those words in Hindi sentence are used in feminine and masculine respectively.

SNS13 - *sitām hṛtavati ravaṇe rāmalakṣmaṇau kutram āpatuḥ*

HS13-*jaba rāvaṇa sitā ko hara le gayā tab rāma lakṣmaṇa kutiyāpe pahuṃce*

In this example in Sanskrit, the word *kutīra* is used in neuter, where as in Hindi the same word is used in feminine. In both sentences one specific time is mentioned. In Hindi it is expressed with verb, whereas in Sanskrit according to the rule *yasya ca bhābena bhāvalakṣaṇam*(2.3.37) *saptamī* is used. In Hindi while denoting the time there is gender agreement between agent and verb. In Sanskrit while using *saptamī* masculine gender is used.

SNS14- *harivarṣe bhramantaḥ bhavantaḥ kiyataḥ deśān adhyuṣivantaḥ?*

HS14 -*yoropa meṃ ghumte huye āpa kitane deśom meṃ rahe?*

SNS15 - *harivarṣe bhramadbhiḥ kiyantaḥ deśāḥ adhyuṣitā?*

HS15- *yoropa meṃ ghumte huye āpa kitane deśom meṃ rahe?*

In these two sentences there is no difference between Hindi and Sanskrit regarding the uses of gender.

SNS16- *dayānandaḥ sarvāṇi śāstrāṇi paṭhitavān*

HS16- *dyānanda ne sāre śāstra paṭhe*

SNS17- *dayāndena sarvāṇi śāstrāṇi paṭhitāni*

HS17- *dyānanda ne sāre śāstra paṭhe*

In these examples in Hindi sentence the word *śāstra* is used in masculine, but in Sanskrit sentence it is used as neuter gender.

SNS18- *vṛkṣāt patanti pakvāni āmrāṇi dṛṣtvā dhāvan surendraḥ kriḍantīm dipikām uktavān – ‘dhāva āmrāṇi khāda’*

HS18 - *pedase girte huye pake huye āma dekhkar dauḍate huye surendrane kheltī huyī dīpikāse kahā - ‘dauḍa dauḍa āma khā.*

SNS19- *vṛkṣāt patanti pakvāni āmrāṇi dṛṣtvā dhāvatā surendreṇa kriḍantī dipikā uktā –*
'dhāva āmrāṇi khāda'

HS19- *peḍase girte huye pake huye āma dekhkar dauḍate huye surendrane kheltī huyī*
dīpikāse kahā - 'dauḍa dauḍa āma khā.

In above mentioned two sentences in Sanskrit *āma* is used in neuter, but in Hindi it is used in masculine.

SNS20- *adya capalā dīptavatī*

HS20- *āja bijalī camakī*

In sentence 28 there are no difference among Hindi and Sanskrit regarding gender.

SNS21 *puṣṭikāmaḥ manuṣyaḥ puṣṇe pāyasam aśnīyāt?*

HS21 *puṣṭike icchuka ādamīko puṣya nakṣtra ke samaya khīra cāhiye?*

In these examples in Sanskrit *pāyasam* is used in neuter but in Hindi *khīra* is used in feminine.

From the above mentioned examples it can be understood that grammatical gender plays an important role while translating from one language to another language. In Sanskrit there are different kinds of noun phrases, such as adjectives, nouns; pronouns, nouns; etc. Next chapter is going to discuss about Sanskrit noun phrases.

Chapter3: Gender recognition and analysis in Sanskrit: Simple Noun Phrases

3.0 Noun Phrase

Bhatṛhari, the famous grammarian observes “*na so'sti pratyayo loke yaḥ śabdānugamādrte / anuviddham jñānam sarvam śabden bhāsate*”.¹ However *śabda* is not the technical term for word in Sanskrit. A syntactic unit is called *pada* in Sanskrit. In different languages, words are arranged in a sentence according to different syntactic rules. A family of expressions that can substitute one another without any grammatical loss is called a syntactic category.² Noun phrase (NP) is one such syntactic category of the language. NPs may function as the subject or as an object in a sentence. They often contain some form of a noun or proper noun, but may also consist of a pronoun only, or even contain a clause. Though proper nouns like John etc, and pronouns, such as he or him are single words, technically they are NPs, because they function like NPs in being able to fill a subject or object or any other NP slot. As for example-

(1) John found the puppy.

In (1) *John* is the subject and *puppy* is the object.

(2) He found the puppy.

Here pronoun *he* is used in place of John. *He* is the subject NP here, the object NP remains the same in both sentences.

(3) The puppy loved him.

(4) The puppy loved John.

In both sentences, the same NPs are used but in different positions. But still they behave as NPs without any grammatical loss.

¹ vākyapadīya 115, part I, edited by K.A Subrmanya Iyer, (Pune 1995)

² pg-125 An Introduction to Language, seventh edition, Victoria Fromkin.

(5) Romeo, who was a Montague, loved Juliet, who was a Capulet.

NP subject of this sentence is *Romeo, who was a Montague* and NP object is *Juliet, who was a Capulet*.

3.1 Simple NPs in Sanskrit

Sanskrit NPs are constituted by different grammatical categories. The structure of a simple NP can be given as following-

(6) {(Adj) (...) N} (...) (conj)

In the above skeleton, Adj stands for adjectives, numerals, pronouns and *kṛdantas*.

Simple NPs can be recursively infinite using conjunctions. Even without recursive conjunctions, there may be infinite number of adjectives/modifiers before the noun. This structure can be very complex, like Bāṇbhata's compositions, such as *Kādambarī* and *Harṣacarita*. For this gender analyzer the focus would be on simple Sanskrit NPs. The example of these simple NPs are given below.

(7) *cañcalaḥ bālakaḥ cañcalā bālikā ca*

In (7) *cañcalaḥ* and *cañcalā* are the adjectives of *bālaka* and *bālikā* respectively and *ca* is the conjunct. Words *cañcala* and *cañcalā* are under the category of noun. Here the nouns are used as adjectives and *cañcalaḥ bālakaḥ* and *cañcalā bālikā* are separate NPs.

Numerals can also serve purpose of adjective inside a noun phrase. Such as

(8) *ekaḥ bālakaḥ ekā bālikā ca*

In (8) numerals *eka* and *ekā* modifies nouns *bālaka* and *bālika*. Here *eka* and *ekā* are considered as adjectives.

These numerals can also be used along with other adjectives. Such as (9) *ekaḥ cañcalaḥ bālakaḥ ekā cañcalā bālikā ca*

Here *ekaḥ cañcalaḥ* and *ekā cañcalā* are together adjectives of *bālaka* and *bālikā* respectively. Forms of *kṛdanta* verb are used as adjectives to the nouns. As for example

(10) *hasan bālakaḥ hasantī bālikā ca*

Here *hasan* and *hasantī* are *kṛdanta* forms of root *has*. These two are used here as adjectives towards *bālaka* and *bālikā*. Now with these all sentences pronouns can also be added as a part of NPs. As for example(11) *saḥ hasan ekah cañcalah bālakah sā hasantī ekā cañcalā bālikā ca-* here from *saḥ* to *bālakah* is one noun phrase and from *sā* to *bālikā* is another noun phrase. The nouns are *bālaka* and *bālikā* here and all other words are adjectives to them.

As J.S.Spejir observes that in Sanskrit the difference between adjective and substantive is not strongly marked. Both classes of nouns have the same declension, and a great numbers of them have sometimes adjectival meanings, sometimes they are used as substantives. Adjective of a substantive follows the gender and number of the substantives. As for example *śuklah varṇah, śuklā sudhā, śuklam vasanam*, here adjective *śukla* takes the gender of substantives which are masculine, feminine and neuter gender respectively. In a noun phrase, adjectives can be of different types. A noun can be used as an adjective, like the above mentioned example. Numerals can be used as adjectives alone, and along with other adjectives, as for example *ekā bālikā gacchati*, or *ekā gaurī bālikā gacchati*. In the first sentence the numeral *ekā* is used to modify the noun *bālikā*. In the second sentence numeral *ekā* is used with an adjective *gaurī*. In the first sentence *ekā bālikā* is the noun phrase and in the second sentence *ekā gaurī bālikā* is the noun phrase.

Pronouns are also used within a noun phrase. For example, *sā saralā bālikā gacchati*. Here pronoun *sā* is used in a noun phrase and it functions as an adjective here. Sometimes pronouns are used in a clause, as for example *yasyam pāṭhasālāyām mohansya putraḥ paṭhati tasyāmeva jagadīśaya tanayaḥ api paṭhati*. In this sentence pronoun *yasyam* and *tasyām* are showing the relation between two sentences. Here *yasyam pāṭhasālāyām mohansya putraḥ* is one noun phrase and *tasyāmeva jagadīśaya tanayaḥ* is another noun phrase. Sometimes participles are used as adjectives, as for example *hasantī bālikā gacchati*. Here the word *hasantī* is formed by adding affix after root, and grammatically it is termed as a participle. In this sentence it functions as an adjective and *hasantī bālikā* is a noun phrase. These noun phrases are arranged

according to some rules. Those rules with respect to gender agreement are discussed below.

3.2 Gender Agreement in Sanskrit

In Sanskrit there is no gender agreement between subject and verb (except when the verb is a *kr̥danta*). The adjectives agree with the substantives in gender and number

3.2.1 Agreement of the verb with the subject

In Sanskrit, the predicate may not always be a finite verb; a participle, or an adjective or a noun may take its place. When a participle is used as the predicate it must agree with the subject and in number and gender. As for example (12) *saaduktavān*, (13) *sā taduktavat* (14) *teṣāṃ vandhanāni chitrāni* (15) *kāryam kṛtam* (16) *latā chinnā* etc. In the first two sentences, participles are used as verbs. Here verb agrees with the subject in number and gender. In the last example, there is no finite verb, but the subject and predicate part have agreement in terms of gender and number.

When an adjective or a noun is used as the predicate, a form of the roots *as* or *bhū* may be used or may be omitted. The adjective used predicatively agrees with the subject in number and gender, words like *āspada*, *patra*, *bhājana*, *sthāna*, *pada* etc, retain their number. As for example, *subhṛtya durlabhaḥ*, *sampadaḥ padamāpadā*, etc.

3.2.3 Agreement of the adjective with the substantive

An adjective participial or qualitative, must agree with the substantive, as for example, (17) *rūpavān puruṣaḥ*, (18) *rūpavatī strī*, (19) *tāni pustakāni* etc. But numeral adjectives of fixed gender and number remain unchanged, as for example (20) *śatam brāhmaṇāḥ*, (20) *śatam striyaḥ*, (21) *viṃśatiḥ bālakāni* etc.

When an adjective qualifies two or more substantives it agrees with them in their combined number. When the two substantives differ in gender, the adjective will be

masculine, as for example (22) *rājā rājñī ca stutyacaritau staḥ*, here due to the masculine *rājā* the adjective *stutyacarita* is used in masculine. If substantives are of three different genders such as masculine, feminine and neuter, then the neuter predominates over two. As for example (23) *dhṛtiḥ kṣamādamo'steyam śaucamindriyanigrahaḥ dhīrvidyābudhirakrodho etāni dharmalakṣaṇāni manunā proktāni*. Here *kṣamā*, *vidyā* etc are used in feminine gender, *dama*, *budhi* are used in masculine, and *asteya* is used in neuter gender. Here *dharmalakṣaṇa* is noun subject and its adjectives are in three genders, as a consequence of the above mentioned rule the neuter prevails over the other two genders, and so the word *etāni* is used in neuter gender. If there are two genders, such as masculine and neuter then also the gender defaults to neuter. As for example

(24) *dharmaḥ kāmāśca, darpaśca harṣaḥ krodhaḥ sukham vayaḥ/ arthāhetāni sarvāṇi pravartante na saṃsayaḥ*

Here *dharma*, *kāma* etc are used in masculine and *sukham* is used in neuter, and as a consequence, pronoun *sarvāṇi* defaults to neuter .

3.2.4 Agreement of relative with its antecedent

The relative agrees with its antecedent in gender, number and person, the cases of the relative and its antecedent being determined by their relation to their respective clauses, as for example (25) *buddhīryasya balam tasya*, here the rel *yasya* and co-rel *tasya* are referring to a masculine person. When the rel has for its predicate a noun differing in gender from the antecedent the relative generally takes the gender of the antecedent noun, the demonstrative pronoun following that of the noun it qualifies, as for example (26) *śaityaṃ hi yatsā prakṛtirjalasya*, here *śaitya* is in neuter gender which is different than the gender of *jalam* which is in masculine. As a consequence, the pronoun *yat* is used in neuter.

The relative pronoun *yat* is used to introduce a clause in Sanskrit sentence. The gender of the demonstrative pronoun is being the same as that of the antecedent noun, as for example, (27) *satyoyam janapṛavādaḥ yatsampatsampadamnubadhnātī*, here *yat* is

taking the gender of the word *satya*, i.e. neuter. Sometimes in Sanskrit the antecedent noun or pronoun is omitted and has to be inferred from the gender and number of the relative, as for example (28) *dhanen kiṃ yo na dadāti yācake*. The pronouns of the first (asmat) and second person (yuṣmat) have no gender. They are used in same forms in all three genders. The other pronouns follow the gender of the nouns they refer to.

3.2.5 Participles

All declinable participles function like adjectives. They agree with the nouns, in gender, number and case. The participles often discharge the functions of verbs. They are largely employed to take the place of the past and future tenses and more especially of passive verbs.

3.2.5.1 Present Participle

When a contemporary action is indicated the present participle is used, as for example (29) *aranya caran saḥ byāghram apaśyat*. Here *caran* is used to denote the contemporary action of the watching tiger. Word *caran* has gender agreement with the subject *saḥ*. Here *caran saḥ* is one noun phrase. The present participle is also used to denote a manner in which an action is done,³ as for example (30) *gacchan bhakṣyati*.

3.2.5.2 Past Participle

In Sanskrit past passive participle is very frequently used in the place of verb. It is used as an adjective and agrees with the object in gender, number and case. In the sentence, the agent is put in instrumental, while the past active participle is treated like verb in past tense. As for example (31) *tena kāryam kṛtam*, here *kṛtam* is used as verb and it agrees in number, gender and case with the object *kāryam*. In the case of past passive participle of intransitive roots the agent is put in the nominative case, as for example *tadā prarudito bālakaḥ*. The past passive participle is used as a neuter substantive, as *gatam*, *suptam*, *dattam* etc.

³ page-511, Kale.M.R, A Higher Sanskrit Grammar, MLBD(1995)

3.3 Gender recognition and analysis in simple NPs

As mentioned earlier also, in a Sanskrit sentence, except the verb, the other words have gender information with them. As a consequence the major goal of this thesis is to analyse the gender of Sanskrit noun phrases. First the structure of the online gender analyzer is described here.

Input text/sentence/word (*sundarī bālikā gatavatī*)

↓

Gender Recognition qualifying constituents (non-*tinanta* , non *avyaya*)

↓

Gender Analysis

↓

Agreement checking

↓

Determine collocational gender

The algorithm can be given as under –

-get each sentence as a token

-preprocess

-find verbs and *avyayas* (exclude them)

-if multiple NPs with conj or commas

-->get simple NP chunks separated by conj or comma

-run subanta analyzer (obtain *pratipadikas*)

-for each NP

--> get gender info (Sanskrit and Hindi) of each word from lexical resources (example base, amarakosha, MWDD, corpus etc)

--> if not found in lexicon

--> apply gender rule base (vibhakti, patterns, upadha etc)

--> obtain gender information

-check agreement (any non N left or right of N is assumed to be a modifier)

-evaluate collational gender

The present subanta analyzer⁴ of Sanskrit analyses the Sanskrit words into *prātipadika* with *vibhakti* markers. Now some *vibhakti* markers help to identify the gender of word.

Pada with vibhakti	Prātipadika	Vibhakti-marker	Number	Gender
<i>narān</i>	<i>nara</i>	<i>ān</i>	plural	masculine
<i>phalāni</i>	<i>phala</i>	<i>āni</i>	Plural	neuter
<i>munīn</i>	<i>muni</i>	<i>īn</i>	Plural	masculine
<i>sādhūn</i>	<i>sādhu</i>	<i>ūn</i>	Plural	masculine
<i>dātrn</i>	<i>dāta</i>	<i>ṛn</i>	Plural	masculine
<i>vāriṇī</i>	<i>vāri</i>	<i>ṇī</i>	Dual	neuter
<i>vāriṇi</i>	<i>vāri</i>	<i>īṇi</i>	Plural	neuter
<i>madhunī</i>	<i>madhu</i>	<i>nī</i>	Dual	neuter
<i>madhūni</i>	<i>madhu</i>	<i>ūni</i>	Plural	neuter

From this table a rule can be made that if after any word these *vibhakti* markers appear then the gender of those would be according to the table. Here some problems can be solved. As for example the word *mitra* with the meaning of friend is used in neuter, but to

⁴ Chandra, Subash, 2006 , 'Machine Recognition and Morphological Analysis of Subanta-padas', submitted for M.Phil degree at SCSS, JNU.

signify the meaning sun it is used in masculine. So if the word appears in the input in this particular *vibhakti*, then the gender recognition of the word would be easy. The problem with this method is that the particular word has to arrive in the input with this particular *vibhakti*. As a consequence after this step there would be huge numbers of words whose gender would be unrecognized by the system.

To solve the problem, one rule can be implemented from *Lingānuśāsana*. Gender can be recognised from the last but one syllable of the word. The technical name of this category is *upadhā*⁵(penultimate).

Upadha	Clause	Gender
<i>k, ṭ, ṇ, ṭh, n, p, bh, ma, y, r, ṣ, s</i>	if the word ends in <i>a</i>	masculine
<i>l</i>	if the word ends in <i>u</i>	neuter

This should cover words like *stabaka, ghaṭa, pāṣāṇa, śoṭha, phena, bhānu, dīpa, stambha, soma, samaya, kṣura, vṛṣa, vāyasa, phala etc.* But there are exceptions of this rule, such as *cibuka, śāluka, prātipadika, amśuka, utsaka, kirīṭa, muḥuṭa, lalāṭa, śṛṅgāṭa, ṛṇa, lavaṇa, parṇa, uṣṇa, kāṣṭha, pṛṣṭha, rikṭha, ukṭha, jaghana, ajina, tuhina, kānana, vipina, vana, vṛjina, vetana, śāsana, sopāna, mithuna, śmaśāna, ratna, cihna, pāpa, śilpa, puṣpa, śaṣpa, antarīpa, kuṅkuma, rukma, sidhma, yudhma, adhyātma, hṛdaya, indriya, uttariya, dvāra, agra, tamkra, vaktra, vapra, chidra, nīra, kṛchra, randhra, śvabhra, astra, timira, vichitra, kēyūra, udara, śarīra, kandara, pañjara, jaṭara, ajira, vaira, catvara, puṣakara, gavhara, kuhara, kulīra, kāśmīra, ambara, śisīra, tantra, yantra, kṣatra, kṣetra, mitra, kalatra, citra, sūtra, netra, gotra, aṅulitra, śastra, sāstra, vakra, patra, pātra, śukra, ṛjīṣa, ambarīṣa, pīyūṣa, purīṣa, kilmiṣa, bias, busa, sāhasa. All these words are used in the neuter gender. Some words whose penultimate sound is *l*, are used in the masculine gender, as for example *tūla, upala, tāla, kusūla, tarala, kambala, devala, vṛṣala etc.* But among them if any word is used as a proper name, then it would follow the gender of the person, especially if the name is a mythical and famous one. As for example the words *ambarīṣa, śukra* are used as the names of a king and the teacher of demons respectively.*

⁵ P.1.1.65 alo'ntyāt pūrva upadhā

In these sense these words would be used in masculine gender only. But still there would be confusion regarding many words.

At this level, to solve the problem, another rule from Liṅgānuśāsana would be implemented here. The rule depends on the last varṇa of the word as shown in following table -

Ending syllable	Gender	Example	Exception
<i>r</i>	Masculine	<i>pitṛ, bhāṭṛ</i>	<i>māṭṛ, duhitṛ, svasṛ, potṛ, nanāḍṛ</i> (feminine)
<i>u</i>	Masculine	<i>sādhu</i>	<i>dhemu, rajju, kuhu, sarayu, tanu, kareṇu, priyaṅgu</i> (feminine) , <i>śmṣu, jānu, vasu, aśru, jatu, trapu, talu, dāru, madhu, svādu, vastu, mastu</i> (neuter), adjectives depend on the gender of nouns.
<i>n</i>	Masculine	<i>guṇin, rājan, vṛtrahan, maghavan etc</i>	
<i>tu</i>	Masculine	<i>dhātu</i>	
<i>tra</i>	Neuter	<i>Patra</i>	
<i>ru</i>	Masculine	<i>Taru</i>	
<i>is</i>	Neuter	<i>Sarpis</i>	<i>chadis</i> (feminine)
<i>us</i>	Neuter	<i>Trapus</i>	<i>chātra, maṅtra, putra, vṛtra</i> [name of a demon] (masculine)
<i>as</i>	Neuter	<i>Srotas, yādas, manas</i>	
<i>man</i>	Neuter	<i>Carman,</i>	<i>sīman</i> (feminine)

		<i>varman</i>	
--	--	---------------	--

After the application of the Pāṇinian rules, there are some Sanskrit noun phrases whose gender recognition is very difficult. For these kinds of words the gender can be recognized from the last syllable of the word apart from the Pāṇinian rules.

The ending syllable of word	Gender	Exception
<i>a</i>	masculine or neuter	
<i>ā</i>	Feminine	<i>viśvapā, dārā</i>
<i>i</i>	masculine, feminine, neuter	
<i>ī</i>	Feminine	<i>sudhī</i>
<i>u</i>	masculine, feminine, neuter	
<i>ū</i>	Feminine	
<i>ṛ</i>	Masculine	<i>māṛ, duhitṛ, svasṛ, potṛ, nanāḍṛ</i> and <i>dhāṛ</i> is neuter
<i>o</i>	feminine, masculine (forms are same)	
<i>ou</i>	feminine, masculine (forms are same)	
<i>c</i>	Masculine	<i>ṛc, tvac, vāc, ruc, śuc, sruc</i>
<i>j</i>	Masculine	
<i>ṭ</i>	Masculine	
<i>k</i>	Masculine	
<i>ṭh</i>	masculine	
<i>t</i>	masculine	
<i>d</i>	masculine, feminine	
<i>dh</i>	feminine	<i>vudh</i>
<i>n</i>	masculine, neuter	
<i>p</i>	feminine	

<i>bh</i>	feminine	
<i>r</i>	feminine	
<i>b</i>	feminine	
<i>ś</i>	feminine	
<i>ṣ</i>	feminine, masculine, neuter	
<i>s</i>	masculine, feminine, neuter	
<i>h</i>	masculine, feminine	

3.4 Structure of lexicon

To build a computational analysis system, it is necessary to have enough lexical resources along with the rules. The lexical resources used for this purpose include tagged resources as shown under –

- Amarakoṣa
- Example base consisting of
 - strī pratyaya examples from grammar books (Sidhāntakaumudī and others)
 - selected words from a corpus of 140 texts of ordinary Sanskrit.
- Description of the corpus

File name	Source	Theme	Number of sentences	Comments
Corpus-1	http://sanskrit.jnu.ac.in/currentSanskritPrase/	<i>yadā aham uddyānam gacchāmi tadā jānāmi</i>	10	Simple sentences

Corpus -2	<a href="http://sanskrit.jnu.ac.in/currentSanskritPr
ose/">http://sanskrit.jnu.ac.in/currentSanskritPr ose/	<i>gurubhakta</i> <i>ḥ āruṇi</i>	32	
Corpus -3	<a href="http://sanskrit.jnu.ac.in/currentSanskritPr
ose/">http://sanskrit.jnu.ac.in/currentSanskritPr ose/	<i>apriṣya na</i> <i>kartvyam</i>	20	
Corpus -4	<a href="http://sanskrit.jnu.ac.in/currentSanskritPr
ose/">http://sanskrit.jnu.ac.in/currentSanskritPr ose/	<i>bālakaḥ</i> <i>dhruvaḥ</i>	25	
Corpus -5	<a href="http://sanskrit.jnu.ac.in/currentSanskritPr
ose/">http://sanskrit.jnu.ac.in/currentSanskritPr ose/	<i>yasya</i> <i>budhirtasya</i> <i>balam</i>	22	
Corpus -6	<a href="http://sanskrit.jnu.ac.in/currentSanskritPr
ose/">http://sanskrit.jnu.ac.in/currentSanskritPr ose/	<i>cvtāri</i> <i>preraṇā</i> <i>srotāṃsi</i>	55	
Corpus -7	sandēśa May, 2005	<i>gatirodhak</i> <i>khemrāj</i> <i>nepālakath</i> <i>ā</i>	130	
Corpus -8	<a href="http://sanskrit.jnu.ac.in/currentSanskritPr
ose/">http://sanskrit.jnu.ac.in/currentSanskritPr ose/	<i>trayaḥ</i> <i>praśnāḥ</i>	45	
Corpus -9	sandēśa December 2005	<i>nandīśvara</i>	64	
Corpus -10	sandēśa September 2005	<i>nilamādhva</i>	52	

- Adapted MWDD, Apte
- Rule base

Here the lexicon would be a long list of the *prātipadikas* or nominal head-words of Sanskrit. These words will be annotated with their gender information. Along with that these words would have the gender of their Hindi counterparts. There are various dictionaries and koṣas for Sanskrit. The proposed lexicon will depend on the Sanskrit – English Dictionary of Monier Williams and the dictionary of Apte along with various

traditional koṣas, such as Amarkoṣa, Medini koṣa etc. In Liṅgānuśāsana there are lists of base words with gender information. The whole list is stored in lexicon, as for example-

Words	Gender
<i>akṣata</i>	Masculine
<i>agni</i>	Masculine
<i>āji</i>	Feminine
<i>indriya</i>	Neuter
<i>upala</i>	Masculine
<i>kaṭa</i>	masculine, neuter
<i>cāmara</i>	Neuter
<i>pada</i>	Neuter
<i>mukuta</i>	Neuter

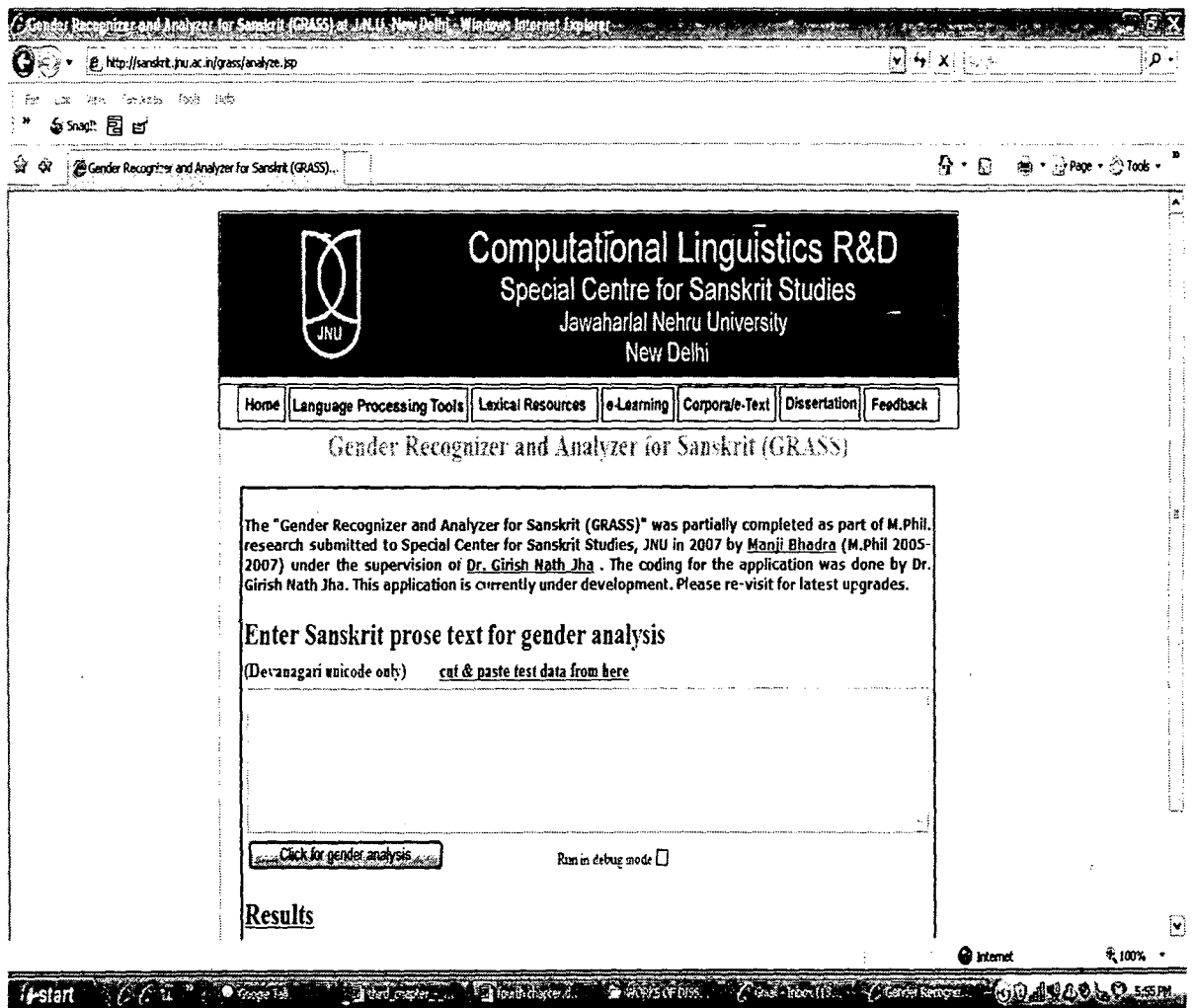
The lists of numerals and pronouns with the information of gender are stored as a part of lexicon. In the Amarkoṣa also there are words with gender information. This would also be there in the lexicon. Along with these lexicons there would be lists words with of feminine suffix with the respective feminine suffix as a part of gender analysis. If in input these kinds of words come in the result it would show the feminine suffix as a part of result. In the lexicon examples are taken from of Sidhāntakaumudī.

The next chapter discusses the implementation of above mentioned rules along with the front end, Java objects.

Chapter4: Gender Recognizer and Analyzer System for Sanskrit (GRASS)

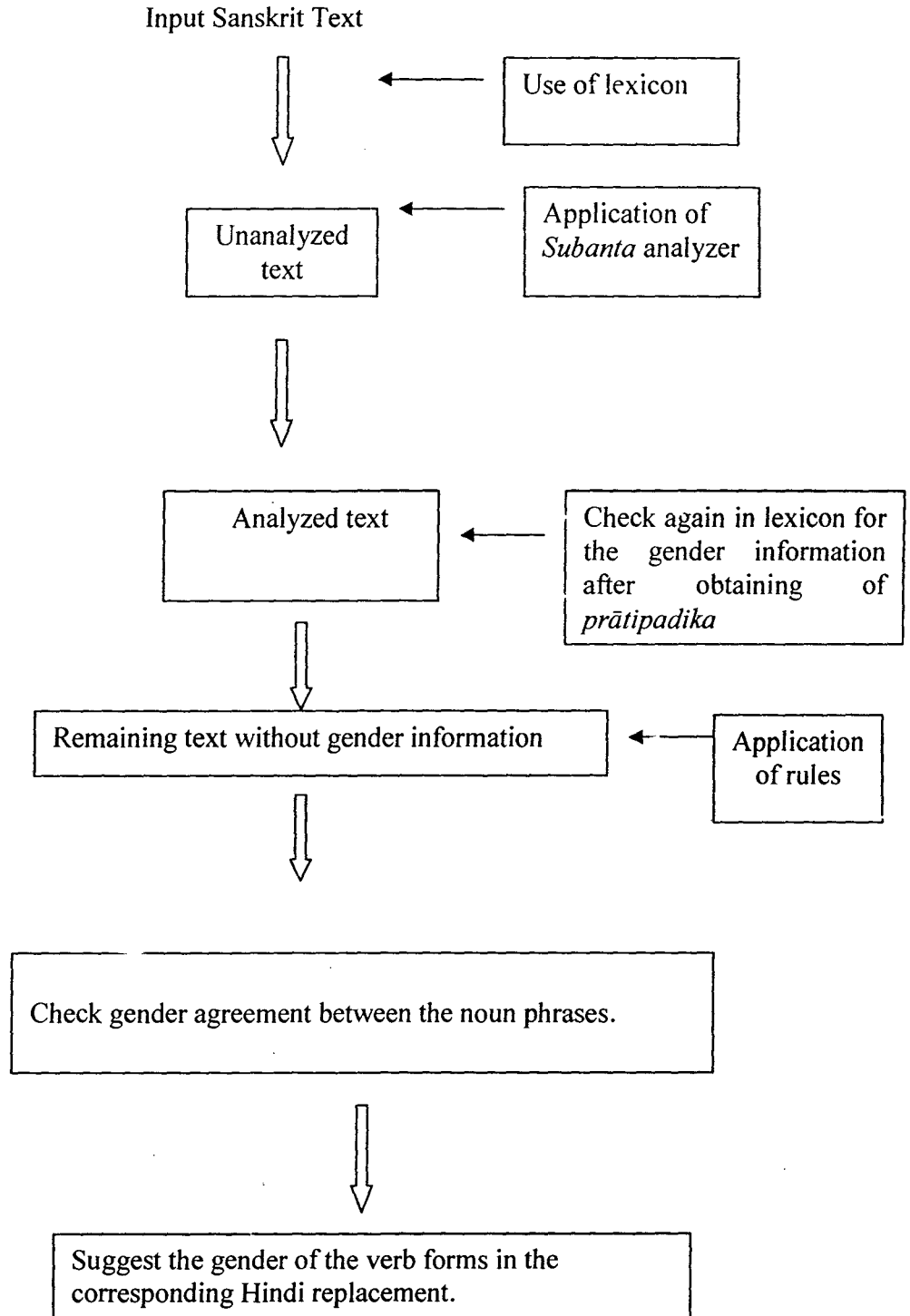
4.0 GRASS

This chapter describes the Gender Recognizer and Analyzer System for Sanskrit (GRASS) which is a partial implementation of the work done for this dissertation. This system is a web application developed in Java Servlet. It takes devanāgarī input in Unicode (UTF-8) and displays the result in devanāgarī. GRASS has a debug mode also which displays the internal states of the application. The screen shot of the interface is given below-



4.1 The GRASS Model

The following model describes the structure of GRASS



4.1.1 How does GRASS work

After getting the Sanskrit text as input, the system analyses it according to the requirements. These requirements are instructed to the computer by the GRASS with the help of the main Public class which is called GRASS. Under this main class there are other classes which perform various actions. After taking the input, it takes each sentence as a token for further analysis.

After this step the system normalizes the input with the help of the pre processor, and then checks if there are constituents to process. First it excludes the verb, and indeclinables and marks them as [V] and [AV] respectively. This component checks punctuation in input text and tags it with a specific tag. If any punctuation is found between two characters without space “ ” then it deletes them. Also, if any punctuation comes to the left or right of a word without space, then this program adds a space between the word and the punctuation. It also checks the punctuation, and excludes them from the input.

After exclusion of verbs and indeclinables, the system will search for NPs. First it will search for conjuncts or commas to get separate NPs, if there are multiple NPs. Among them it will chunk simple NPs, which are going to be attached to GRASS subsequently. Then from the main class, it calls for the fixed lists taggar class for gender marking of each word of Sanskrit except verbs and indeclinables. Through this function GRASS checks in lexicon for the gender information of the NPs. But the major part of the input is not covered by this step.

As a consequence, GRASS is taking help of existing *subanta* analyzer¹ to obtain *prātipadika*. After obtaining the *prātipadika* it needs to look again in the example base for gender information, both in Sanskrit and in Hindi.

¹ Chandra, Subash, 2006 , ‘Machine Recognition and Morphological Analysis of Subanta-padas’, submitted for M.Phil degree at SCSS, JNU.

Still there are many words the gender of which are not recognized by the system. To solve this problem the rule base is applied. With this private class application, the gender recognition process is completed. After getting the gender of each word, the system looks for the gender agreement within a single noun phrase which is going to developed subsequently.

4.1.1.1 Sample code

```

public class Grass{
public String markGender(String s){
StringTokenizer st = new StringTokenizer(s, "\u0964"); //tokenization based on sentence
boundary (stop or danda)
private String preProcess(String tkn){
if (tkn !=null && tkn.length()>0){
tkn = pre.preProcess(tkn);
}
return tkn;
}
private String checkFL(String tkn){
if (tkn.length()>0)
tkn = flt.checkLists(tkn);
return tkn;
}
private String sup_analyze(String tk){
if (tk.length()>0)
tk = sa.analyzeSup(tk);
return tk;
}
private String applyRuleBase(String s){
String ts = "";
String tkn = "";
int cgCounter=0;

if ( tkn.indexOf(".")!=-1 && tkn.indexOf("AV")!=-1 && tkn.indexOf("V")!=-1 ) //if
not verb/Avyaya and not tagged for gender yet
tkn = flt.checkRuleBase(grass_rb, tkn);

if (tkn.indexOf("hm")>0)
cgCounter++;

ts = ts + " "+tkn;
}

if (cgCounter >0)

```

```
ts = ts +" > Masculine";
else
ts = ts +" > Feminine";
```

4.1.2 Lexical resources

- Example base- In the example base data are stored in two formats, in *pada* form and in also *prātipadika* form. All data are stored in text file format (.txt). Pronouns are stored in grass_eb.txt. The format is given below-

सर्वः=sm:hm;सर्वो=sm:hm;सर्वे=sm_f_n:hm;सर्वम्=sm_n:hm;सर्वान्=sm:hm;सर्वेण=sm_n:h
m;सर्वाभ्याम्=sm_f_n:hm;सर्वैः=sm_n:hm;सर्वस्मै=sm_n:hm;सर्वेभ्यः=sm_n:hm;सर्वास्मात्=
sm_n:hm;सर्वस्य=sm_n:hm;सर्वयोः=sm_f_n:hm;सर्वेषाम्=sm_n:hm;सर्वस्मिन्=sm_n:hm;स
र्वेषु=sm_n:hm;सर्वाणि=sn:hm;सर्वाः=sf:hm;सर्वा=sf:hf;सर्वाम्=sf:hf;सर्वया=sf:hf;सर्वाभिः=sf:
hm;सर्वस्यै=sf:hm;सर्वाभ्यः=sf:hm;सर्वास्याः=sf:hm;सर्वासाम्=sf:hm;सर्वास्याम्=sf:hm;सर्वा
सु=sf:hm;

Here *s* stands for Sanskrit and *h* stands for Hindi language, *m*, *f*, and *n* stand for masculine, feminine and neuter respectively. In the structure *sm:hm* stands that the word is used in Sanskrit in the masculine gender and its Hindi counterpart is also used in masculine gender. The gender information is here after the mentioning both the languages, by *s* or *h*.

- Along with this example there are also words taken from the Sidhāntakaumudī and those words are tagged with gender information and also with feminine suffix. As for example- नदी=sf:hf[डीप्];देवी[डीप्];सौपर्णयी=sf:hf[डीप्];ऐन्द्री=sf:hf[डीप्]. Here along with gender information, feminine suffixes are also mentioned within third bracket.

- There is another example base file where data are stored in prātipadika format with the same structure of earlier txt file. As for example-

आदि = sm:hm; आभा = sf:hf; आभरण = sm:hm; आभीर = sm:hm; आदर्श = sm:hm; आदिशङ्कर = sm:hm; आधुनिक = sm:hm; आदिनाथ = sm:hm; आदित्य = sm:hm.

- In the Amarakoṣa also there are words with gender information. It is also a part of lexicon resource. The data of dictionary of Monier Williams and Apte are also used in example base.

4.2 The environment on which system is going to run-

On the localhost (CD version), the website has the URL <http://localhost:8080/grass/analyze.jsp> . On the actual server, the URL is <http://www.sanskrit.jnu.ac.in/gender> . The site accepts devanāgarī data in UTF-8 format, and for that Unicode IME like Baraha² has to be installed. Otherwise, the user can enter some of the test files provided. The JSP interface sends data to the GRASS object, which after tokenizing the input sends each word for preprocessing and the result of preprocessor to GRASS. The Grass keeps on building the display depending on the output from the pre processor-recognizer and analyzer objects.

4.2.1 The web server

The GRASS runs on Apache Tomcat 4.0 platform. The details for this Java based web server are as follows -

4.2.2 Apache Tomcat 4.0

Apache Tomcat is the servlet container that is used for the Java Servlet and Java Server 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

² <http://www.baraha.com/BarahaIME.htm>

Tomcat is intended to be a collaboration of the best-of-breed developers from around the world³.

4.2.3 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 possible⁴.

4.2.4 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-independent⁵. It is one of the most sophisticated tools available for high performance and secures web applications.

4.3 Result Analysis

For the input-

sundarah bālakah sundarī hālikā ca gacchatah

GRASS is coming up with the result-

सुन्दर[1.1][sm:hm]बालक[1.1][sm:hm]सुन्दरीsf:hf[डीप्]बालिकाsf:hf[टाप्]च[AV]गच्छतः[V]>masculine.

The first square bracket which is occurring after the words is bearing number and *vibhakti*. The second square bracket has the information about the gender of the word in Sanskrit and the gender of its counterpart in Hindi. Here *sm:hm* describes that the word is used in masculine gender in both languages, Sanskrit and Hindi, and *sf:hf* indicates that

³ <http://www.apache.org/>

⁴ <http://java.sun.com/products/servlet/>

⁵ <http://java.sun.com/products/jsp/>

the word is used in the feminine gender in both Sanskrit and Hindi. The words, *sundarī* and *bālikā* have been formed in Sanskrit after adding feminine suffix डीप् and टाप् respectively after the nominal stem *sundara* and *bālaka*. In the square bracket which is appearing after the gender information there is mention of these feminine suffixes. At the end with a greater than sign the collocation gender is suggested by the system. A screenshot of output is given below.

The screenshot shows a web browser window displaying the GRASS application. The browser's address bar shows the URL <http://localhost/grass/analyze.jsp>. The page header includes the JNU logo and the text "Special Centre for Sanskrit Studies, Jawaharlal Nehru University, New Delhi". A navigation menu contains links for Home, Language Processing Tools, Lexical Resources, e-Learning, Corporate/Text, Dissertation, and Feedback. The main content area is titled "Gender Recognizer and Analyzer for Sanskrit (GRASS)".

The application text reads: "The 'Gender Recognizer and Analyzer for Sanskrit (GRASS)' was partially completed as part of M.Phil. research submitted to Special Center for Sanskrit Studies, JNU in 2007 by Mauli Bhadra (M.Phil 2005-2007) under the supervision of Dr. Girish Nath Jha. The coding for the application was done by Dr. Girish Nath Jha. This application is currently under development. Please re-visit for latest upgrades."

The interface prompts the user to "Enter Sanskrit prose text for gender analysis" (Devanagari unicode only) and provides a "cut & paste test data from here" link. The input field contains the text: "सः सुन्दरः एक बालकः सा सुन्दरी एका बालिका च गच्छतः ।". Below the input field is a "Click for gender analysis" button and a "Run in debug mode" checkbox.

The "Results" section displays the following output: "सुन्दर[1.1][sm:hm] बालक[1.1][sm:hm] सुन्दरीsf:hf[डीप्] बालिकाsf:hf[टाप्] च[AV] गच्छतः[V] > Masculine".

Conclusion

This dissertation is an R&D effort at the M.Phil level for developing a gender analyzer for Sanskrit. Rules of gender marker for Sanskrit words are collected from Aṣṭādhyāyī and from Liṅgānuśāsana (one of the ancillary text of Aṣṭādhyāyī). The explanation of these rules are taken from Sidhāntakaumudī of Baṭṭojī Dīkṣita and Kāśikāvṛtti of Jayāditya Vāmana. The lists of separate words with the gender information are collected from Liṅgānuśāsana and from Amarakoṣa. Various linguistic resources for gender recognition and analysis were also developed and adapted. Besides this, the evaluation of tools and techniques-JSP for front end, Java for servlet objects and Apache Tomcat for web server was studied and an online interface was developed.

Limitation

At present, this system covers only simple NPs of prose Sanskrit. Sanskrit is a relatively free word order language. As a consequence of this, it is really very difficult to determine the exact position of noun phrase even in simple Sanskrit prose for the computer. In a Sanskrit sentence there is no fixed place for the verb. This gender analyzer system lacks any kind of parsed tree for the Sanskrit sentence. As a result of this, the system can not properly understand how to check gender agreement between NPs.

While recognizing the gender of Sanskrit words there are some ambiguities regarding kṛdanta verb forms. In some case it is marked as verb. As a result of this, those words are not considered for analysis. The proposed gender analyzer system is not able to solve the semantic problems at present. In Sanskrit, same words are used in different genders depending on the meaning. As for example, the word *mitra* is used in neuter gender while it denotes the meaning friend. The same word is used with same spelling in masculine gender while it denotes the meaning sun. The forms of both these words are the same from third case ending. The singular form of second case ending of the word *mitra* in the sense of sun and singular forms of the word *mitra* in the first and second case ending in the sense of friend are the same. It is difficult to decide that whether the word *mitra* is used in neuter or in masculine. Sometimes genders are decided from the adjectives. But if

the system can not decide about the gender definitely, checking gender agreement and suggestion of collocation gender will be at stake. In the case of these kinds of words the present system will show both the results as output. In case of suggesting collocation gender of Hindi sentence, it will come up with the suggestion that it is masculine. If there is confusion between neuter gender and feminine gender collocation gender will be displayed as feminine. When there is a problem between feminine gender and masculine gender the system will come up with the suggestion of collation gender as masculine with respect to Hindi language.

The gender of the Sanskrit words can be marked from affixes also. This method is not applied here. At presently there is no affix analyzer of Sanskrit words. To apply this rule it is necessary to have an affix analyzer first. This limitation causes many ambiguities.

The limitation of lexical resources may affect the accuracy of the result. It is not always possible to get the segmented forms validated through the corpus. But the system is online, so the quality of the lexicon will always improve.

The present gender analyzer system will not segment compounds if they are found in the linguistic resources. This means that the larger strings may stay as un-simplified.

The limitations of the analytic process of the *subanta* analyzer may affect the accuracy of *sandhi* analysis because, if *subanta* analyzer gives the wrong analysis of a word, the *sandhi* analyzer may not correctly validate its segmentation. One problem with this system may be the slow speed due to heavy processing, lexicon and corpora check. Since this system will be online and potentially multiple users could be accessing this system, speed will certainly be an issue.

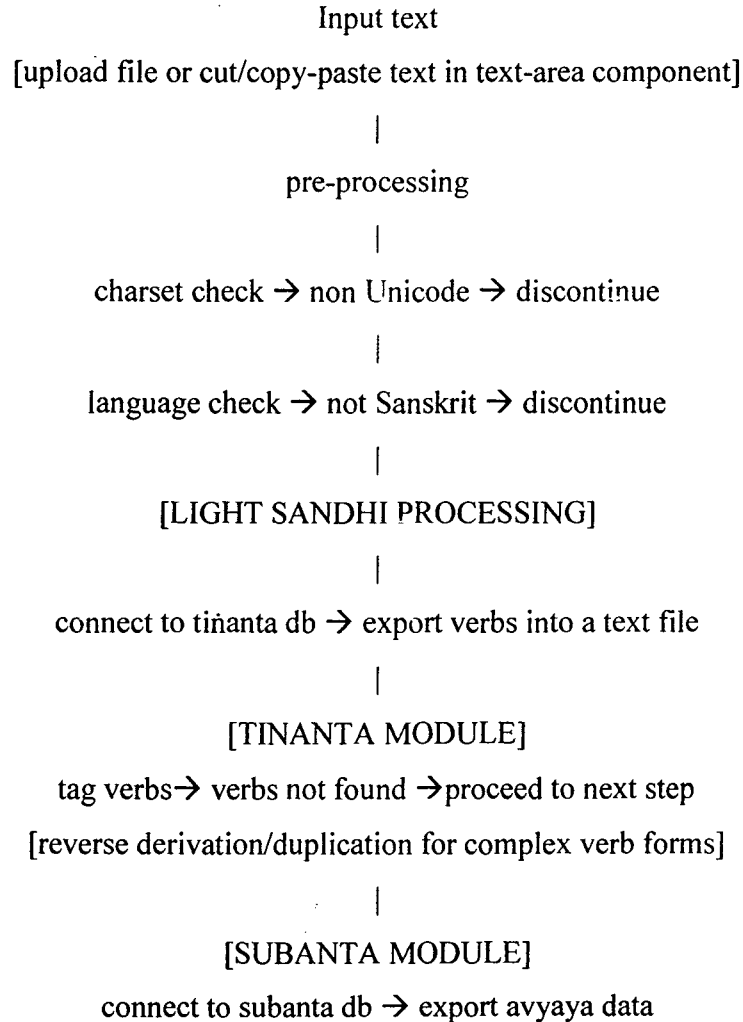
Further Research and development

The present gender analyzing system is only for simple Sanskrit. The same methodology can be applied to build a system which can comprehensively analyze all kinds of NPs, such as embedded NPs. This kind of NPs may have multiple noun phrases which are related with each other by conjunct, or in a single sentence there can be multiple NPs which are co-related by means of a clause. Further research can be put forward in the

direction of the other aspects of computational linguistics, such as anaphora resolution, discourse analysis etc. On the basis of this, structure gender recognizer for other Indian languages can be made.

To solve the gender recognition problem more accurately, all the rules of Liṅgānīśasana are to be applied. To achieve this purpose, research can be done on affix analyzer for Sanskrit words.

One of the major goals of this is to design a machine translation system from Sanskrit to Indian languages (right now Hindi). To process Sanskrit for this purpose, an algorithm of Sanskrit analysis system was presented by Dr. Jha and all the students (2006) as follows¹



¹ **Towards a Computational analysis system for Sanskrit** in the proceeding of first National symposium on Modelling and Shallow parsing of Indian Languages at Indian Institute of Technology Bombay pp 25-34 on 2nd to 4th April 2006

|
check for avyayas → tag them
|
connect to subanta db → export vibhakti data
|
check vibhaktis → tag nps for vibhaktis
|
[SAMASA MODULE]
[TADDHITA MODULE]
|
[STRI PRATYAYA MODULE]
|
[KRDANTA MODULE]
|
[AVYAYA MODULE]
|
[KARAKA MODULE]
|
yogyata check
|
karaka semantics check
|
karaka-vibhakti analysis
|
output text display/download/email

Appendix

• Appendix-1

Sample of Lexicon

अजा=sf:hf[टापु];एडका=Sf:Hf[टापु];अथा=Sf:Hf[टापु];चटका=Sf:Hf[टापु];मूषिका=sf:hf[टापु];
वाला=sf:hf[टापु];वत्सा=sf:hf[टापु];होडा=sf:hf[टापु];मन्दा=sf:hf[टापु];विलता[टापु];मेधा[टापु];
गङ्गा=sf:hf[टापु];गंगा=sf:hf[टापु];[सर्वा=sf:hf[टापु];खट्वा=sf:hf[टापु];धनिका=sf:hf[टापु];स्व
भावजा=sf:hf[टापु];कृत्रिमा=sf:hf[टापु];गता=sf[टापु];कर्त्री[डीपु];हर्त्री[डीपु];धार्त्री[डीपु];दण्डिनी=s
f:hf[डीपु];योगिनी[डीपु];रोगिणी=sf:hf[डीपु];राज्ञी=sf:hf[डीपु];भवन्ती=sf:hf[डीपु];पचन्ती=sf:hf
[डीपु];दीव्यन्ती=sf:hf[डीपु];नमन्ती=sf:hf[डीपु];पठन्ती=sf:hf[डीपु];पतन्ती=sf:hf[डीपु];चोरय
न्ती=sf:hf[डीपु];मुष्णती=sf:hf[डीपु];ददती=sf:hf[डीपु];कुर्वती[डीपु];जानती[डीपु];अदती[डीपु];श्रृ
ण्वती[डीपु];तुदन्ती=sf:hf[डीपु];तुदती=sf:hf[डीपु];लिखन्ती=sf:hf;लिखती=sf:hf[डीपु];पृच्छन्ती
[डीपु];पृच्छती[डीपु];यान्ती=sf:hf;याती=sf:hf[डीपु];पान्ती=sf:hf[डीपु];पाती=sf:hf[डीपु];भविष्य
ति=sf:hf[डीपु];भविष्यती=sf:hf[डीपु];कुरुचरी=sf:hf[डीपु];नदी=sf:hf[डीपु];देवी[डीपु];सौपर्णयी=
sf:hf[डीपु];ऐन्द्री=sf:hf[डीपु];औत्सी=sf:hf[डीपु];ऊरुद्वयसी=sf:hf[डीपु];ऊरुद्वघ्नी=sf:hf[डीपु];ऊ
रुमात्री=sf:hf[डीपु];पञ्चतयी=sf:hf[डीपु];आक्षिकी=sf:hf[डीपु];प्रास्थिकी=sf:hf[डीपु];लावणिकी=
sf:hf[डीपु];यादृशी=sf:hf[डीपु];इत्वरी=sf:hf[डीपु];स्त्रैणी=sf:hf[डीपु];पौत्स्नी=sf:hf[डीपु];शाक्ती
की=sf:hf[डीपु];महाभागा=sf:hf[टापु];कमललोचना=sf:hf[टापु];विश्वरूपा=sf:hf[टापु];विशालाक्षी
=sf:hf[डीपु];गौरी=sf:hf[डीपु];आदि=sm:hm;आभा=sf:hf;आभरण=sm:hm;आभीर=sm:hm;आद
र्श=sm:hm;आदिशङ्कर=sm:hm;आधुनिक=sm:hm;आदिनाथ=sm:hm;आदित्य=sm:hm;आफरी
न=sf:hf;आग्नेय=NA;आह्लाद=sm:hm;आह्लादित=sm:hm;आह्लानित=sm:hm;आकार=sm:hm;आ
काश=sm:hm;आकाङ्क्षा=sf:hf;आलम=sm:hm;आलाप=sm:hm;आलोक=sm:hm;आमोद=sm:h
m;अन्दलीब=sm:hm;आशीष=sm:hm;आतिश=sm:hm;आभीर=sm:hm;अब्बास=sm:hm;अभय=
sm:hm;अभयनन्द=sm:hm;अभयप्रद=sm:hm;अभीक=sm:hm;अभिभव=sm:hm;अभिचन्द्र=NA;
अभिहित=sm:hm;अभिजात=sm:hm;अभिजय=sm:hm;अभिजित=sm:hm;अभिजात=sm:hm;अ
भिज्वल=sm:hm;अभिलाष=sm:hm;अभिमन्द=sm:hm;अभिमानी=sm:hm;अभिमन्यु=sm:hm;
अभिमन्युसुत=sm:hm;अभिमोद=sm:hm;अभिनभस=sm:hm;अभिनन्द=sm:hm;अभिनन्दन=s
m:hm;अभिनाथ=sm:hm;अभिनव=sm:hm;अभिरुप=sm:hm;अभिषेक=sm:hm;अभिशोक=sm:h
m;अभिसुमत=sm:hm;अभिस्यन्त=sm:hm;अभिवीर=sm:hm;अभ्र=sm:hm;अभ्रकसिन=sm:hm;
अभ्यग्नि=sm:hm;अभ्युदय=sm:hm;अभ्युदित=sm:hm;अब्जयोनि=sf:hf;अब्जित=sm:hm;अच

ल=sm:hm;अचल=sm:hm;अचलपति=sm:hm;अचलेन्द्र=sm:hm;अचलेश्वर=sm:hm;अचन्द=sm:hm;आचार्य=sm:hm;आचार्यनन्दन=sm:hm;आचार्यसुत=sm:hm;आचार्यतनय=sm:hm;अचिन्द्र=sm:hm;अचिन्त्य=sm:hm;अच्युत=sm:hm;अच्युतराय=sm:hm;अदलरसु=sm:hm;आदर्श=sm:hm;आदेश=sm:hm;अधिक=sm:hm;अधिकार=sm:hm;अधिप=sm:hm;अधीत=sm:hm;अधीत=sm:hm;आदिल=sm:hm;आदिनाथ=sm:hm;आदिकवि=sm:hm;आदित=sm:hm;आदितेय=sm:hm;

• Appendix 2

Sample of Verb Lists

बुक्कति=[V];वसते=[V];आसन्=[V];उपकरोति=[V];अनिच्छातः=[V];इच्छन्ति=[V];जागर्ति=[V];विशदीकरोतु=[V];सन्तु=[V];परित्यजति=[V];उपकरोति=[V];रक्षन्तु=[V];अरोचत=[V];प्रचलति=[V];पश्यतु=[V];याति=[V];आसीत=[V];प्रासारयत्=[V];उपाविशत्=[V];आदिशत्=[V];उत्पतत्=[V];उदपतन्=[V];अधावत्=[V];आगच्छन्=[V];व्यवर्तत्=[V];कर्तिष्यति=[V];अकृन्तत्=[V];प्रत्यागच्छत्=[V];आलोकयन्=[V];लिम्पन्ति=[V];भूषयन्ति=[V];प्रेषयन्ति=[V];प्रज्वलयन्ति=[V];अपनयति=[V];धारयन्ति=[V];भक्षयन्ति=[V];समायोजयन्ति=[V];प्रभवति=[V];कथयन्ति=[V];मन्यन्ते=[V];पूजयन्ति=[V];उपयुज्यते=[V];उत्पादयति=[V];सञ्चालयति=[V];प्राविशत्=[V];अपतत्=[V];अव्रजत्=[V];अभणत्=[V];व्रस्यत=[V];अस्मि=[V];परिपालय=[V];अकथयन्=[V];आज्ञापयति=[V];आकर्णयत्=[V];अकुर्वन्=[V];अलभत्=[V];असि=[V];संजायते=[V];अजायत=[V];प्रारभत=[V];प्राशंसत्=[V];अलङ्करोति=[V];प्रयतेमहि=[V];भ्रमति=[V];उद्भवन्ति=[V];हरति=[V];धारयन्ति=[V];भ्रमन्ति=[V];गुञ्जन्ति=[V];प्रसीदन्ति=[V];समायोज्यते=[V];धारयन्ति=[V];खादन्ति=[V];प्रसीदन्ति=[V];अनुभवन्ति=[V];रञ्जयति=[V];रोचते=[V];नास्ति=[V];वर्जयेत=[V];जयते=[V];प्रविचलन्ति=[V];प्रविशति=[V];नमर करोमि=[V];इच्छसि=[V];ददामि=[V];याचे=[V];प्रदास्ये=[V];ददामि=[V];पिबामि=[V];इच्छामि=[V];आरोहामि=[V];नेच्छामि=[V];श्रूयताम्=[V];आस्ताम्=[V];आगच्छताम्=[V];कुरु=[V];चालय=[V];स्वीकरोतु=[V];अक्रन्दत्=[V];अवदत्=[V];जीवतु=[V];अर्हति=[V];ग्रहीष्यति=[V];अक्षाम्यत्=[V];असहत=[V];जीवति=[V];अतिष्ठन्=[V];अहरत्=[V];मरिष्यन्ति=[V];आगमिष्यति=[V];अनुभवन्तु=[V];तिष्ठन्तु=[V];गच्छत्सु=[V];चिन्तयामास=[V];विनङ्क्षयति=[V];विषीदत=[V];हन्ति=[V];संवादयामि=[V];आनुष्ठिते=[V];उवाच=[V];समायातः=[V];उच्यताम्=[V];श्रुणु=[V];करिष्यामि=[V];गच्छ=[V];क्षम्यताम्=[V];आसते=[V];विद्यते=[V];मनोरञ्जयामः=[V];प्रभवामः=[V];अलभत=[V];गृह्णाति=[V];प्रदर्शयन्ते=[V];प्रतीयन्ते=[V];ददाति=[V];शक्नुमः=[V];वर्धयन्ति=[V];अपश्यन्=[V];आस्ते=[V];अचिन्तयत्=[V];म्रियते=[V];नीयते=[V];क्रियते=[V];प्राप्स्यामि=[V];अन्वभवत्=[V];अवसत्=[V];प्राप्नोत्=[V];न्यवसत्=[V];अलभत=[V];कुर्याम=[V];

V];कारयति=[V];वितरति=[V];अभिधीयते=[V];शक्यन्ते.अन्तर्भवन्ति=[V];अभिलषति=[V];प्रवर्तयति=[V];अतिक्रामति=[V];तरति=[V];कथ्यते=[V];निर्मोयन्ते=[V];वर्धते=[V];आपद्यते=[V];उड्डाययति=[V];शक्येत्=[V];अर्हत्=[V];शक्नोति=[V];शक्नुयात्=[V];श्रुतवति=[V];प्राप्नोति=[V];आगच्छति=[V];चोरयामि=[V];अदशन्=[V];प्रभवति=[V];कारयन्ति=[V];उवाच=[V];भाति=[V];आनयति=[V];प्रयतते=[V];समाह्वयति=[V];प्राप्यते=[V];स्थापयति=[V];अनुतिष्ठितः=[V];समाचरन्ति=[V];आसनस्थं=[V];पश्य=[V];अन्वतिष्ठन्=[V];यान्ति=[V];पलायन्तः=[V];प्रावर्तत=[V];अत्यजत्=[V];चिन्तयति=[V];रक्षिष्यति=[V];प्रार्थयत=[V];प्रस्थीयताम्=[V];रक्षिष्यति=[V];अवर्धत=[V];तिष्ठति=[V];प्रायतत=[V];आगच्छ=[V];प्रतिन्यवर्तत=[V];रक्षिष्यति=[V];उद्गच्छन्ति=[V];अवर्धत=[V];आरोहत्=[V];अश्रावयत्=[V];गच्छत्=[V];आगच्छत्=[V]

- Appendix-3

Debug mode

Gender Recognizer and Analyzer for Sanskrit (GRASS)

The "Gender Recognizer and Analyzer for Sanskrit (GRASS)" was partially completed as part of M.Phil. research submitted to Special Center for Sanskrit Studies, JNU in 2007 by Manji Bhadra (M.Phil 2005-2007) under the supervision of Dr. Girish Nath Jha . The coding for the application was done by Dr. Girish Nath Jha. This application is currently under development. Please re-visit for latest upgrades.

Enter Sanskrit prose text for tagging

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

सरलः बालकः

Click for gender analysis

Run in debug mode

Results

सरल[_1.1][sm:hm] बालक[_1.1][sm:hm] > Masculine

-----START OF Grass.markGender()-----

token = सरल: बालक:

-----START OF POST.preProcess()-----

input=सरल: बालक:

output= सरल: बालक:

-----END OF POST.preProcess()-----

-----START OF POST.checkFL()-----

input= सरल: बालक:

output= सरल: बालक:

-----END OF POST.checkFL()-----

-----START OF Grass.sup_analyze()-----

input= सरल: बालक:

output= सरल[_1.1] बालक[_1.1]

-----END OF Grass.sup_analyze()-----

-----START OF Grass.applyRuleBase()-----

input= सरल[_1.1] बालक[_1.1]

output= सरल[_1.1][sm:hm] बालक[_1.1][sm:hm] > Masculine

-----END OF Grass.applyRuleBase()-----

Bibliography

Primary source

- Bhattacharya Srimadgurunatha Vidyanidhi (Ed), 1988. Amarakośa, Sanskrit Pustak Bhandar, Kolkata
- Chaturvedi Giridharsharma (ed.), 2004, Vaiyakaraṇasiddhāntakaumudī with Bālamānorama and Tattvabodhini tikā, Motilal Banarasidass, Delhi.
- Iyer K.A Subramania (ed).1973, 'Vākyapadīya of Bhaṭṭhari with the Prakīrṇa Prakāśa of Helārāja' part 3 Deccan college Poona
- Iyer K.A Subramania (ed).1995, 'Vākyapadīya of Bhaṭṭhari with the commentary of Vṛtti and Padhati of Vṛṣvadeva' Deccan College, Pune
- Jigyasu, Pt. Brahmadata (ed.), 1998, 'Panini-Astadhyayi', Ramlal Kapoor trust, Sonapat.
- Mishra Narayan (ed), 1996. kāṣikā of Pt.Vamana and Jayaditya, Chaukhamba Sanskrit sansthan, Varanasi.
- Shastri, Guru Prasad (ed.), 1939. Patañjali-Mahābhāṣya with Pradīpodyota tikā, Varanasi.
- Shastri Vasudev Lakshman Panashikar (ed.), 1994. Siddhāntakaumudī, Chaukhamba Sanskrit Pratisthan, New Delhi

Secondary source

- Acharya Vamadeva, 1990. 'Līṅga-Parijñanam', Shabdātattva Prakasan Varanasi
- Allen, J., Hunnicutt, M. S., and Klatt, D. (1987). From text to speech---the MITalk system. MIT Press, Cambridge, Massachusetts.
- Bharati, Akshar, Vineet Chaitanya and Rajeev Sangal, 1995, 'Natural LanguageProcessing: A Paninian Perspective,' Prentice-Hall of India, New Delhi.
- Bhate Saroja,1989 'Pāṇini's Taddhita Rules' University of pune
- Burrow, T, 2001, The Sanskrit Language, Motilal Banarasidass, Delhi.

- Cardona, George, 1999, Recent Research in Pāṇinīan Studies, Motilal Banarasidass, New Delhi
- Cardona, George, 1997, Pāṇini: his work and its traditions, Motilal Banarasidass, New Delhi
- Cardona George, Jain Dhanesh (ed). 2003, The Indo Aryan Languages, Routledge Language Family Series, vol.2, Routledge, London and New-York
- Chatterjee Kshitish Chandra, 2003. Technical Terms and Technique of Sanskrit grammar, Sanskrit Pustak Bhandar, Kolkata.
- Fromkin Victoria, Rodman Robert, Hymns Nina ‘An Introduction to Language, 7th edition Thomson Wordsworth
- Iyer, K.A.Subramania, 1969, ‘Bhartṛhari : A study of the vākyapadīya in the light of ancient commentaries’ Deccan College, Poona.
- Iyer, K.A.Subramania, 1971, Vākyapadīya of Bhatṛhari, English translation, part 3 Deccan College Poona
- Jigyasu, Pt. Brahmadatt, 2003, ‘Aṣṭādhyāyī Bhāṣya Vṛtti’ Ramlal Kapoor Trust, Hariyana.
- Jurafsky, Danial., James H. Martin, 2005, ‘Speech and Language Processing: An Introduction of Natural Language Processing, Computational Linguistics and Speech Recognition’, Person Education, Delhi.
- Kale, M.R., 1972, ‘A Higher Sanskrit grammar’, Motilal Banarasidass, Delhi
- Kimmo Koskeniemi. 1983. Two-level morphology: A general computational model for word form recognition and production. Publication No: 11, Department of General Linguistics, University of Helsinki
- Kumar Kavita 2001, ‘Hindi For Non –Hindi Speaking People’, Rupa& Co, Delhi
- Joworski Jamie 1999, ‘Java 2 Platform unleashed’, BPB Publication, New Delhi
- Lahiri Probodhchandra, Shastri Hrshikesh, 2000 ‘Pāṇinīyam’ The Dhaka Students Library Kolkata

- Lauri Karttunen and Todd Yampol, *INFL Morphological Analyzer*, (XEROX Palo Alto Research Center, Palo Alto, CA, 1990).
- Mitkov Rushlan(Ed) 2003, ‘The Oxford Hand Book of Computational Linguistics, Oxford University Press
- McGregor, R. S. 1972. Outline of Hindi Grammar. Oxford/Delhi: Oxford University Press.
- Naur Peter Randell Brian, Buxton J N (Ed.) 1976, Software engineering: concepts and techniques, Mason/Charter Publisher Inc, New York
- Petheroudakis, J. (1991). *MORPHOGEN automatic generator of morphological information for base form reduction*. Technical report, Executive Communication Systems ECS, Provo, Utah.
- Scarf. M Peter 1996, The Denotation of Generic Terms in Ancient Indian Philosophy: Grammar, Nyāya, and Mīmāṃsa, *Transactions of the American Philosophical Society*, New Ser., Vol. 86, No. 3
- Sharma Rama Nath, 2003. ‘The Aṣṭādhyāyī of Pāṇini’, Munshiram Manoharlal Publishers Pvt. Ltd., Delhi.
- Shastri Acharya Ram, 2005. ‘Sanskrit Sikshan Sarani’, Acharya Ram Shastri Ganpith, Delhi
- Shastri Bhimsen 2004. Laghusidhāntakaumudī, Bhaimi Prakashan, New Delhi
- Speijer.J.S, 1998. ‘Sanskrit Syntax’, MLBD, New Delhi
- Sukla Ramgobind, Vyakarana-Darshan-Pratima , Sampurnanada Sanskrit Vishvavidyalaya, Varanasi.
- Vidyasagara Isvarchandra 2003 ‘Samagra Vyakaran Kaumudi’ Deva Sahitya Kutir. Kolkata

Web Reference

- http://www.au-kbc.org/research_areas/nlp/projects/morph/document.html
accessed on 28th July,2007
- <http://ccat.sas.upenn.edu/plc/tamilweb/hindi.html>
accessed on 28th July,2007
- <http://tdil.mit.gov.in/download/Desika.htm>

- accessed on 28th July, 2007*
- <http://www.sanskritacademy.org/About.htm>
accessed on 28th July, 2007
 - http://ltrc.iiit.net/~anusaaraka/SAN_MO/test_san_mo.html *accessed on 28th July, 2007*
 - <http://sanskrit.inria.fr/> *accessed on 28th July, 2007*
 - <http://www.springerlink.com/content/m2gr7lt8v0cjre54/fulltext.pdf> *accessed on 28th July, 2007*
 - <http://72.14.235.104/search?q=cache:qIkl6DegG3AJ:www.aubkc.org/dfki/igws/calts.ppt+morphological+analyzer&hl=en&ct=clnk&cd=178&gl=in> *accessed on 28th July, 2007*
 - tdil.mit.gov.in/Malayalam-CDACThiruvananthapuramJuly03.pdf *accessed on 28th July, 2007*
 - <http://www.iiit.net/ltrc/index.html> *accessed on 28th July, 2007*
 - <http://anglahindi.iitk.ac.in/> *accessed on 28th July, 2007*
 - <http://www.cse.iitk.ac.in/users/rmk/proj/proj.html#mt> *accessed on 28th July, 2007*
 - <http://shakti.iiit.ac.in> *accessed on 28th July, 2007*
 - <http://www.ncst.ernet.in/matra/http://www.ncst.ernet.in/matra/about.shtml>
accessed on 28th July, 2007
 - <http://www.tdil.mit.gov.in/TDILmeet2001May01.pdf> *accessed on 28th July, 2007*
 - <http://www.cdacindia.com/html/about/success/mantra.asp> *accessed on 28th July, 2007*
 - <http://www.jadavpur.edu/> *accessed on 28th July, 2007*
 - <http://www.mysmartschool.com/pls/portal/portal.MSSStatic.ProductAnuvaadak>
accessed on 28th July, 2007
 - <http://www.elda.org/en/proj/scalla/SCALLA2001/SCALLA2001Rao.pdf>
accessed on 28th July, 2007
 - <http://ildc.gov.in/telugu/htm/Akshara.htm> *accessed on 28th July, 2007*
 - <http://tdil.mit.gov.in/Punjabi-TIETPatialaJuly03.pdf> *accessed on 28th July, 2007*
 - tdil.mit.gov.in/UrduSindhiKashmiri-CDACPuneJuly03.pdf *accessed on 28th July, 2007*

- RCILTS, JNU – Achievements: <http://tdil.mit.gov.in/SanskritJapaneseChinese-JNUJuly03.pdf> *accessed 15.10.2006*
- <http://www.sil.org/pckimmo/> *accessed on 25th December 2007*
- <http://www.comp.lancs.ac.uk/ucrel/claws> *accessed on 15th January 2006*
- www.comp.lancs.ac.uk/ucrel/claws *accessed on 28th July,2007*
- <http://www.csc.kth.se/tcs/projects/granska/rapporter/granskareport.pdf> *accessed on 28th July,2007*
- <http://www.springerlink.com/content/hcdjrlvj5nlybf5c/fulltext.pdf> *accessed on 28th July,2007*
- <http://ling.uib.no/~desmedt/papers/nodalida99pub.html> *accessed on 28th July,2007*
- <http://www.enformatika.org/ijci/v1/v1-4-56.pdf> *accessed on 28th July,2007*
- <http://acl.ldc.upenn.edu/W/W07/W07-1701.pdf> *accessed on 28th July,2007*
- <http://tanev.dir.bg/LingTunis.htm> *accessed on 28th July,2007*
- <http://www.xrce.xerox.com/competencies/content-analysis/demos/czech> *accessed on 28th July,2007*
- <http://www.ims.uni-stuttgart.de/projekte/corplex/paper/iezius/coling1998.pdf> *accessed on 28th July,2007*
- <http://www.xrce.xerox.com/competencies/content-analysis/arabic/> *accessed on 28th July,2007*
- http://www.unisa.ac.za/contents/faculties/humanities/afri/docs/NRF_Project_Webpage.pdf *accessed on 28th July,2007*
- <http://www.basistech.com/base-linguistics/asian/> *accessed on 28th July,2007*
- http://tdil.mit.gov.in/Oct_2004/GDemand%20world%20-10.pdf *accessed on 28th July,2007*
- http://tdil.mit.gov.in/Oct_2004/GDemand%20world%20-10.pdf *accessed on 28th July,2007*
- <http://www.google.com/intl/sa/> *accessed on 28th July,2007*
- <http://www.languageinindia.com/jan2002/sirmauri.html> *accessed on 28th July,2007*

- <http://72.14.235.104/search?q=cache:8LB3pp3f5QJ:www.essex.ac.uk/linguistics/LFG/wwwlfg.stanford.edu/lfg2004/school/material/agreement/finalday2.pdf+gender+agreement+in+Hindi+grammar&hl=en&ct=clnk&cd=3&gl=in> accessed on 28th July,2007
- <http://www.baraha.com/BarahaIME.htm> accessed on 28th July,2007
- <http://www.apache.org/> accessed on 28th July,2007
- <http://java.sun.com/products/servlet/> accessed on 28th July,2007
- <http://java.sun.com/products/jsp/> accessed on 28th July,2007

Article and Papers

- Agrawal, Muktanand, 2006, 'Computational Identification and Analysis of Sanskrit Verb-forms', In the Souvenir Abstracts of 28th AICL, BHU, Varanasi, pp.126-127.
- Bhowmik, Preeti & Jha, Girish Nath, 2006, 'Sanskrit Language Pedagogy: an e-learning approach', In the Souvenir Abstracts of 28th AICL, BHU, Varanasi, pp.150.
- Char Prahalada 2002, Semantic Perspective of Strīpratyas, in Aspects of Pāṇinian Semantics, Sahitya Academy, New Delhi pp-124-131
- Chakravarthy Meera 2002, 'Gender description in Vyākaraṇa in Aspects of Pāṇinian Semantics, Sahitya Academy, New Delhi pp-132-138
- Jha Girish Nath,2006 Towards a Computational analysis system for Sanskrit" in the proceeding of first National symposium on Modelling and Shallow parsing of Indian Languages at Indian Institute of Technology Bombay pp 25-34
- 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.
- Kapoor Kapil and Shukla Santosh ,2002 ' Theory of Gender and its Generation in Pāṇini' in Aspects of Pāṇinian Semantics, Sahitya Academy, New Delhi pp114-123
- Kumar, Sachin & Jha, Girish Nath, 2006, 'Issues in sandhi processing of Sanskrit', In the Souvenir Abstracts of 28th AICL, BHU, Varanasi, p.129.

- Mishra, Sudhir Kumar & Girish Nath Jha, 2004, 'Sanskrit Karaka Analyser for Machine Translation', in the proceedings of iSTRANS-2004, New Delhi, pp.224-225.
- Mishra, Sudhir Kumar & Girish Nath Jha, 2005, 'Identifying Verb Inflections in Sanskrit Morphology', in the proceedings of SIMPLE-05, IIT-Kharagpur, pp.79-81

Thesis and Dissertation

- Chandra, Subash, 2006, 'Machine Recognition and Morphological Analysis of Subanta-padas', submitted for M.Phil degree at SCSS, JNU
- Chandrashekar, R., 2007, 'Part-of-Speech Tagging for Sanskrit', Ph.D. thesis submitted to SCSS, JNU.
- Jha, Girish Nath, 1993, 'Morphology of Sanskrit Case Affixes: A Computational Analysis', M.Phil. submitted to JNU, New Delhi.

Dictionary

- Apte, V.S., 1997, Sanskrit – Hindi Dictionary, Motilal Banarsidass, Delhi
- M. Monier Williams, 2004, Sanskrit- English Dictionary, Munshiram Manoharlal Publishers Pvt. Ltd, Delhi.