Avyaya Analyzer: Analysis of Indeclinables using Finite State Transducers

N. Murali
Department of Computer Applications
S.V. Vedic University
Tirupati

Prof. R.J. Ramasree
Department of Computer Science
R.S. Vidyapeetha,
Tirupati

Prof. K.V.R.K. Acharyulu
Department of Vyakarana
R.S. Vidypeetha,
Tirupati

ABSTRACT
In Sanskrit, *avyaya* i.e., the indeclinable, plays an important role in the construction of a sentence and can be used as preposition, interjection, particle, conjunction or an adverb. To understand a given sentence, a thorough cognizance of the *avyaya* is necessary. This paper briefly describes the *avyayas*, types of *avyayas and* role of *avyayas*. An *avyaya* analysis tool (*avyaya* Morphological Analyzer) has been described here in using FST. The system described in this paper reads the input and produces the derivational morphological information. This paper portrays a simple and straight forward technique to analyze the *avyayas*.

General Terms
Natural Language Processing (NLP) – Morphological Analyzer

Keywords
Avyayas – Kr□danta – Nipāta – Conditional *avyayas –* Relational avyayas – *kr□ t– taddhita*

1. INTRODUCTION
Natural Language Processing (NLP) is a branch of Artificial Intelligence and deals with the Analysis of Human Languages. Each language consists of a set of words called *lexicon* and a set of rules called *grammar*. These enable the use of words to form cognitional valid sentences. NLP deals with both the Spoken as well as Written Languages. The goal of a researcher in NLP is to develop a Parser to analyze any human language. A typical language parser can be divided into 3 components [8] viz., Morphological Analyzer, Local Word Grouper and Core Parser. A *Sentence* is the input for the Parser. A sentence is a group of interrelated words. Every word in the sentence has its own meaning and is encoded grammatically. Further it is related to other words in the sentence. *Morphological Analyzer* (MA) will take each word in the sentence and retrieve the information encoded in the word. With the help of the information produced by the *MA*, *Local Word Grouper* (LWG) will try to identify the phrases in the sentence by grouping the words which have mutual relationship in the sentence. *Parser* will generate the parsed structure of the sentence.

1.1 Avyayas
Sanskrit is the oldest, morphologically rich and most complex language in the world. Pān□ inī grammar is the most widely accepted grammar of Sanskrit. In Sanskrit, the words can be broadly classified into 2 categories viz. Declinable and Indeclinable [1]. Figure 1 illustrates this.

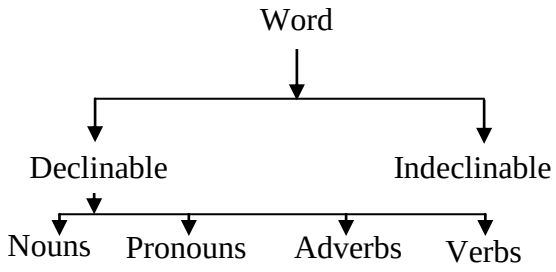

Figure 1: Classification of Words

Declinable or inflectional words are those words whose base form can be changed or inflected. For example, the base form *prātipadikam* or lemma *Rāma* can be inflected in 7 vibhaktis i.e., cases and numbers. These declinable/inflectional words can again be categorized as Nouns, Pronous, Adverbs and Verbs. In Sanskrit, indeclinable words are called *avyayas.* A Sanskrit grammarian has given a beautiful and simple definition for *avyayas*

सदृशं त्रिषु लिङ्गेषु सर्वासु च विभक्तिषु ।
वचनेषु च सर्वेषु यन्न व्येति तदव्ययम् ।।

Sadr□śam□ tris□u lin□gēs□u sarvāsu ca vibhaktis□u /
Vacanēs□u ca sarvēs□u yanna vyēti tadavyayam //

The meaning of the sloka states that the words that cannot be changed or inflected or which remain immutable in all genders, numbers and cases; are called avyayas [1][2][6][7][10]. But, there is an exception to some nouns (Nominative or Locative), which have only one declension e.g. *ādau, sam□vat, svar, svāhā, svadhā* etc. Figure 2 in the next page illustrates various types of *avyayas*. Some grammarians have classified *avyayas* into *Substantive* and *Non-substantive* [6] [7]. The examples for *Substantive avyayas* are a*sti, ādau, sam□ vat, svar* etc. *Non-substantive avyayas* can be further divided into two types i.e., *Conditional* and *Relational. cēt, yat-tat, yathā-tathā, yadī-tarhi, yatra-tatra, yāvat-tāvat* etc. are the examples of *Conditional avyayas*, while *atha, api, iti, iva, ēva, ēvam, kila, khalu, ca, tu, manyē, vat, vā* etc. for *relational avyayas*

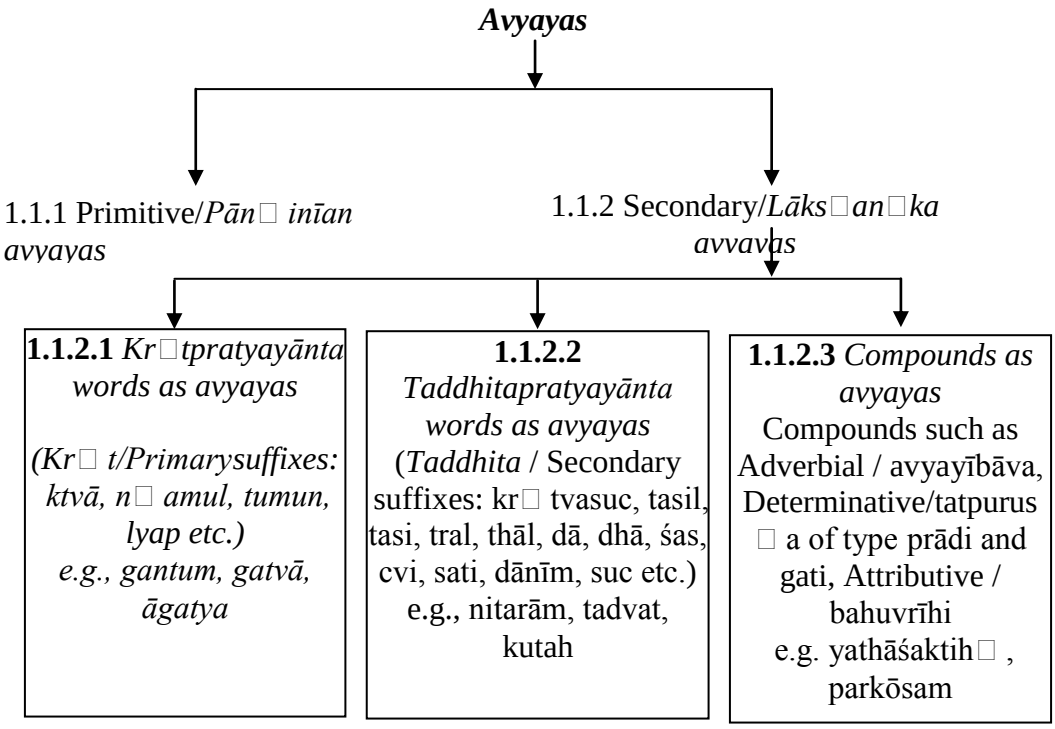

Figure 2: Classification of *avyayas*

Avyayas can be classified into two types i.e., Primitive/ *Pān□inīan avyayas* and *Secondary* / L*āks□an□ka avyayas*.

1.1.1 Primitive/Pān□inīan avyayas

These types of avyayas are directly named by the Sanskrit grammarian *Pān□inī* as *nipātas*. These *avyayas* can be used as prepositions, interjections, particles and conjunctions and can be called as *nipātas* i.e., exceptional or irregular forms [6]. All *nipātas* are indeclinable [7]. All *upasargas*/prefixes can also be treated as *nipātas* and are indeclinable.

1.1.2 Secondary / Lāks□an□ka-s avyayas

These are used as adverbs or adjectives. *Kr□ dantāvyayas*, *taddhitāntāvyayas* and adverbial, determinative (of type *prādi, gati*) and attributive types of compounds fall into this category.

1.1.2.1 Kr□ tpratyayāntawords as avyayas

Kr□tpratyaya / Primary suffixes are used to form nouns (Nominalization) and *avyayas* from roots [1][2][6][10][11]. These *avyayas* denote some action, which have a relation to the main verb in the sentence.

1.1.2.2. Taddhitapratyayānta words as avyayas

Ttaddhitapratyaya / Secondary suffixes are those suffixes that are used to form nouns or *avyayas* from other nouns [1][2][6][10].

1.1.2.3. Compounds as avyayas

Some compounds such as Adverbial / *avyayībhāva*, Determinative / *tattpurus□a* (of type *prādi* and *gati)*, Attributive / *bahuvrīhi* also act as *avyayas*.

2. FINITE STATE TRANSDUCER (FST)

FST is an extension of Finite State Automata and can generate output symbols [9]. *Morphological Analyzers/Generators* can be developed using FST. The transducer is defined as [9]

$$T = (Q, \Sigma, q_0, F, \delta)$$

Q is a finite set of N states q0, q1, q2..... qN. Σ is a finite set of alphabet of complex symbols. Each complex symbol is composed of an input – output pair i : o; one symbol *i* from an input alphabet *I*, and one symbol *o* from output alphabet *O*, thus $\Sigma \subseteq I \times O$, *I* and *O* may each also include the epsilon symbol "ε" i.e. empty symbol or transition label. State transition labels are of 4 kinds [11] (σ : γ), indicates that σ read from the surface form and γ is written; (σ : ε) indicates that σ is read from the surface form and ε i.e., nothing is written; (ε : γ) indicates that nothing is read but γ is written; (ε : ε) indicates that a state transition occurs without reading or writing. q_0 is the initial state. F is a set of final states, $F \subseteq Q$. δ (q, i : o) is the transition function between states. Given a state $q \in Q$ and complex symbol $i : o \in \Sigma$, δ(q, i : o) returns a new state $q' \in Q$. δ is thus a relation from $Q \times \Sigma$ to Q.

3. *AVYAYA* ANALYZER

Avyaya Analyzer (AA) was designed based on FST. FST can be built to accept letters, strings or patterns. A string or pattern is a group of characters. Because of the complex and agglutinative nature of most of the Indian languages, transducers based on strings and patterns are preferred. In this tool, unlike letter transducers, transitions on strings take place. Strings generally are valid *upapadas*, prefixes, roots, suffixes and *feature sequences* in the language.

Avyaya Analyzer deals fully with the primitive type *avyayas* i.e., 1, and *Kr□dantas* / Primary suffixes i.e., 1.1.2.1 and partly with *Taddhitas* / Secondary suffixes i.e., 1.1.2.2. Dealing with *avyayas* of *type 1* is very simple because it has already been defined as *avyayas* by ***Pān□ inī***in *nipātas*. A list of *avyayas* was collected from the text "***avyaya kōśa***", which contains 900 *avyayas* written by Srivatsankacharya. Here a simple lookup is enough to identify the *avyayas*. Some of the examples of type 1 *avyayas* are *asti, ādau, atha,*

api, iti, iva, ēva, ēvam, katham, kila, khalu, ca, tu, manyē, vat, vā etc., *avyayas* of type 1.1.2.2 had been encoded with information and play a vital role in understanding the sentence. Consider the following examples

Gantum means "to go"
Āgantum means "to come"
Gatvā means "having gone"
Āgatya means "having come"

All the above examples were derived from the single root *gam*, prefix *ā* and *kr□ t*suffixes tumun, ktv*ā* and *lyap*. Each word indicates a different meaning because of the suffixes. All these *avyayas* indicate incomplete action of their constituent verbal roots and have a relation to the main verb in the sentence and act as an adverb. Let us see the *avyayas* in action in a sentence

Sentence: *aham□* ***gantum*** *icCāmi.*
POS: Sub avyaya verb
Meaning: I wish to go.

Sentence: *aham□* ***āgantum*** *icCāmi.*
POS: sub avyaya verb
Meaning: I wish to come.

Sentence: *bhavān tatra* ***gatvā*** *vadatu.*
POS: sub avyaya avyaya verb
Meaning: having gone there, you tell.

Sentence: *atra* ***āgatya*** *upaviśatu.*
POS: avyaya avyaya verb
Meaning: having come, sit here.

In all the examples given above, *avyayas* act as adverbs. Both the *avyaya* and main verb are mutually dependent on each other and it is impossible to understand the meaning of the sentence without gaining a proper understanding of the *avyaya*.

3.1 Previous work

Dr. Girish Nath Jha, Special centre of Sanskrit Computational Linguistics, Jawaharlal Nehru University, New Delhi[1] has dealt with the avyayas to some extent. The sample output obtained from the website is presented below in roman transliteration form

Abhijñātum [Abhi jñā, 9, tumun, 0, avy]_KR

Abhijñātum [Abhi jñā, 9, tumun, 0, avy]_KR

Abhiūhitum [Abhi ūūha, 1, tumun, 0, avy]_KR

Abhiūhitum [Abhi ūūha, 1, tumun, 0, avy]_KR

An□ gīkr□ tya [_not found]

Akr□ vā[_1.1]

Ā krāntum [_1.1/2.1]
Apa sārya [_not found]

pra tyāgatya [_not found]

Laks□ īkr□ tya [_not found]

When the test data of 122 *avyayas* was given to this online tool, 77 *avyayas* were found unidentified and 13 *avyayas* were not recognized correctly. The precision is 74.50% and recall is 53.52%. Dr. Gerard Huet, Inria, France[2] is also working on *kr□ dantas*. Dr. P. Amba Kulkarni, Department of Sanskrit Studies, University of Hyderabad[3] and IIITH[4], Hyderabad are also working on Morphological Analyzer for Sanskrit.

3.2 Design

The structure of *avyaya* may be

[[*Upapada*][*Upasarga*]root + suffix]

Upapada and u*pasarga*/prefix are optional. The avyaya Analyzer will give the derivational morphological information of an *avyaya* i.e., the *upapada*, *upasarga*, root and suffix information.

3.2.1 Representation of linguistic information

This system relates the root and linguistic features to the surface form through a set of transformations. 2000 verbal roots, 120 *kr□t* suffixes (out of which four suffixes are related to *avyayas*), 26 *upasargas*/prefixes and nearly 500 *upapadas* were collected. With the combination of all these *upapadas*, prefixes, roots and suffixes, we can generate a large number of *avyayas* which are not productive [9]. The *upapada*, *upasarga*, root and suffixes acquire different forms due to euphonic transformations between them. Hence, each and every possible phonetic change of roots, suffixes, *upasargas*, *and upapadas* were written manually[5] along with the necessary linguistic features. Morphological dictionaries for *upapadas*, *upasargas*, roots and suffixes were created. An example of the possible phonetic changes in *upapada*, *upasarga***,** root and suffix are given below. The second field indicates the lemma and the first field indicates all possible phonetic changes of the second field in context of euphonic change. For computational purpose, the data is presented in WX transliteration scheme. There are several converters available to convert the text from Roman to Unicode and vice-versa.

Upapada
sva/sv,sva
IRax/IRan/IRac/IRaj/IRal/IRaw,IRaw
iram/iraM,ira
jala/janaM/janam/jala/jana/jal/jana, jana

*U*pasrga:
nir/niH/nil,nir
vi/vy/v,vi
anu/Anu/anv/Anv/onu/onv,anu
apa/ap/opa/op/Apa/Ap,apa

Root:

raMram/riraM/ram/raM/rAm/ran/rem/raw/ra,ram_1_A_853
iR/IR/eR/ER,iR_4_pa_1127
iR/IR/eR/ER,iR_6_pa_1351

The second field gives the information about the root, conjugation, type, and root number given in *siddhāntakaumudī*

Kr□t suffix:
iwum/Iwum/Dum/tum/wum/Xum,wumun
iwvA/IwvA/DvA/tvA/wvA/XvA/vA,kwvA
wya/ya,lyap
am,Namul

[1] http://sanskrit.jnu.ac.in/kridanta/ktag.jsp

[2] http://sanskrit.inria.fr/DICO/reader.html

[3] http://sanskrit.uohyd.ernet.in/

[4] http://iiith.ac.in

[5] Upapada, upasarga, roots, and kr□ tsuffixes from the text siddhāntakaumudī and dhāturatnākara were collected and their euphonic Changes were also created by Prof. K.V. Ramakrishnamacharyulu, Dept. of Vyakarana, R.S. Vidyapeetha, Tirupati

3.2.2. Identification of avyayas

It can be checked whether the word is *avyaya* or not by noticing whether the word is *nipāta* or by comprehending its suffix. If the word is *nipāta*, then treat it as *avyaya* and stop. If the word ends with the suffix related to *avyaya*, then it return suffix info and strips the suffix and checks for *upapada*. As *upapada* and *upasarga* are optional, they may or may not be present in the given word. Hence, if *upapada* is present, then return the *upapada* info and strip it, or else return nothing and remove nothing. Then the presence of *upasarga* may be checked for. If *upasarga* is found, then return *upasarga* info and strip it or else return nothing and remove nothing. Then check whether the remaining part of the word is the root or not. If it is, then return the root info or return nothing and stop. The sample output of the avyaya Analyzer is given below. Figure 3 in the next page illustrates the process of analyzing the *avyaya* using Finite State Transducers.

The sample output is presented here in WX transliteration scheme:

<Word: aBijFAwum><vargaH: avyaya><lifgam: xxx><ViBakwi: xxx><vacanam: xxx><level: 1><upapaxam: xxx><upasargaH: aBi><prAwipaxikam: aBijFAwum/jFAwum><XAwuH: jFA><prawyaya: wumun#><gaNaH: 9><paxI: pa>

<Word: afgIkqwya><vargaH: avyaya><lifgam: xxx><ViBakwi: xxx><vacanam: xxx><level: 1><upapaxam: afgI><upasargaH: xxx><prAwipaxikam: afgIkqwya/kqwya><XAwuH: kq#kqw><prawyaya: lyap#><gaNaH: 8#5#6#7><paxI: u#pa>

<Word: akqwvA><vargaH: avyaya><lifgam: xxx><ViBakwi: xxx><vacanam: xxx><level: 1><upapaxam: xxx><upasargaH: naF_naFwawpuruRaH><prAwipaxikam: akqwvA/kqwvA><XAwuH: kq#kqw><prawyaya: kwvA#><gaNaH: 8#5#6#7><paxI: u#pa>

<Word: AkrAnwum><vargaH: avyaya><lifgam: xxx> <ViBakwi: xxx><vacanam: xxx><level: 1><upapaxam: xxx> <upasargaH: Af><prAwipaxikam: AkrAnwum/krAnwum> <XAwuH: kram><prawyaya: wumun#><gaNaH: 1><paxI: pa>

<Word: apasArya><vargaH: avyaya><lifgam: xxx><ViBakwi: xxx><vacanam: xxx><level: 1><upapaxam: xxx><upasargaH: apa><prAwipaxikam: apasArya/sArya><XAwuH: sA#san#sAr#sq><prawyaya: lyap#><gaNaH: 1#4#8#10#3><paxI: pa#u>

<Word: prawyAgawya><vargaH: avyaya><lifgam: xxx><ViBakwi: xxx><vacanam: xxx><level: 1><upapaxam: xxx><upasargaH: prawi,Af><prAwipaxikam: prawyAgawya/gawya><XAwuH: gam><prawyaya: lyap#><gaNaH: 1><paxI: pa>

<Word: lakRyIkqwya><vargaH: avyaya><lifgam: xxx> <ViBakwi: xxx><vacanam: xxx><level: 1><upapaxam: lakRya><upasargaH: xxx><prAwipaxikam: lakRyIkqwya/kqwya><XAwuH: kq#kqw><prawyaya: lyap#><gaNaH: 8#5#6#7><paxI: u#pa>

4. Evaluation

A test data consisting of 28,000 words was prepared by collecting various children stories from *candamama* monthly magazine published during a year. Out of which 122 *avyayas* formed by the primary suffixes (1.2.2.1) have been collected and tested. The tool was evaluated directly based on the Morphological analysis of a given *avyaya*. The system was evaluated by calculating precision and recall for *avyayas*.

C = Number of Correct analysis,
M = Number of Missed analysis
W = Number of Wrong analysis
Precision= C/(C+W)
Recall= C/(C+M)

The *avyayas* were analyzed with a precision of 96.72% and with a recall of 98.36%

5. LIMITATIONS

This tool can identify up to one *upapada*, two *upasargas* in a sequence from the word. As there is no limit on number of *upapadas* and *upasargas* that should be present in the word, the present system will fail to recognize the word when it contains more than one *upapada*, and two *upasargas*.

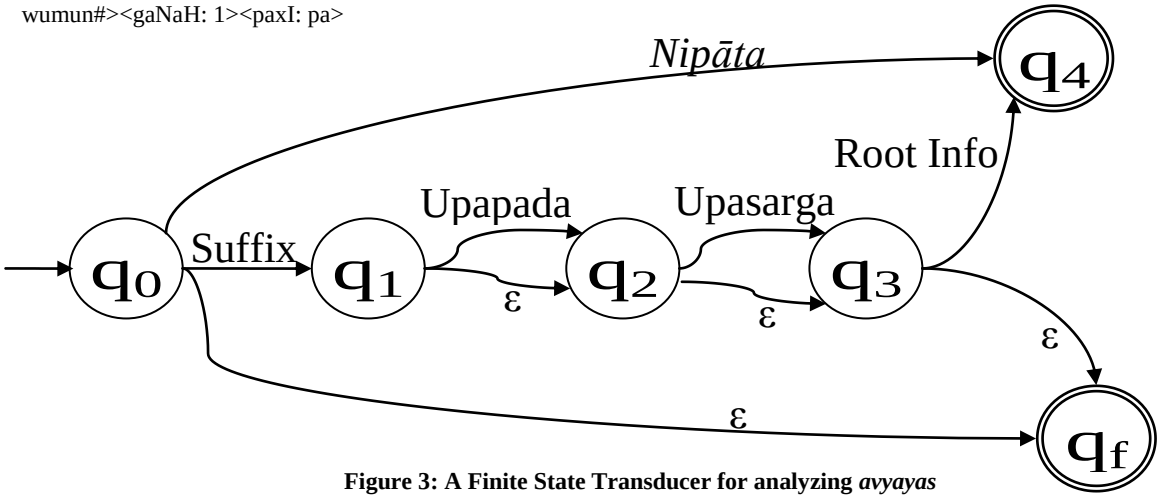

Figure 3: A Finite State Transducer for analyzing *avyayas*

This tool cannot handle when two or more primitive *avyayas* (1.1.1) are presented in *sandhi* form (euphonic transformation). Only a few secondary suffixes (1.1.2.2) like *vaw*, *atasuc*, etc have been covered here. As it is told earlier, compounds which act as *avyayas* (1.1.2.3.) have not been taken up in the present study. *Kr□t* suffixes related to *Vedas* have also not been covered in the present study.

6. CONCLUSION

This paper presents a practical and more sophisticated approach to analyze *avyayas*. As the morphological data and the program modules are independent of each other, it is easy to add new data or change the data without changing the modules. This tool was developed using PERL and it can be easily integrated with any Morphological Analyzers for Sanskrit. The morphological dictionary for roots, ***kr□t*** suffixes (except those which are related to *vaidikaprakaran□a*) and prefixes are complete, but *upapada* dictionary contains limited terms. The right to left approach was also developed to deal with those *upapadas* which are not available in morphological dictionary which is still under testing.

7. ACKNOWLEDGEMENT

Authors thank Dr. Viroopaksha V. Jaddipal, Associate Professor, R.S. Vidyapeetha, Dr. Manoj Kumar Mishra, Assistant Professor, S.V. Vedic University, Mr. Sriramachandra Sarma, Assistant Professor, S.V. Vedic University, Sri Vinayak Bhat, Assistant Professor, S.V. Vedic University, Dr. K. Ushasharma, Asst. Professor, S.V. Vedic University for giving valuable suggestions and Mr. Veluvarti Srinvasa Narayana, Acharya Student, R.S. Vidyapeetha for supplying linguistic information.

REFERENCE

[1] Chakradhar Nautiyaalhansa Shastri, 1966 "*br□ had- anuvāda-candrikā* ", Motilal Banarsidas, New Delhi

[2] Dr. R.V.R. Krishna Sastri, 1997 "*sam□ skr□ ta vyākaran□ am*", Krishnanada Mutt, Hyderabad

[3] Sri Srvatsankacharya, 1971 "avyaya kōśa (A Dictionary of Indeclinables", The Sanskrit Education Society, Chennai

[4] Bhat□ t□ ōji Diksita "siddhāntakaumudī", 1983 Motilal Banarsidas, New Delhi

[5] Muni lāvan□ya vijayasūri, "dhāturatnākara", I-VII Volumes Motilal Banarsidas, New Delhi

[6] M. Monier–Williams, "Sanskrit–English Dictionary", online dictionary www.sanskritlexicon.uni-koeln.de/ monier /indexcaller.php

[7] Declension - Indeclinables available at : www.sanskritsanscrito.com.ar/en/sanskrit_sanskrit6/indeclinables.shtml

[8] Akshar Bharati, Vineet Chaitanya, Rajeev Sanghal, "Natural Language Processing – A Paninian Perspective", Prentice Hall of India, New Delhi.

[9] Daniel Jurasky & James H. Martin, Third Reprint, 2004 "Speech and Language Processing", Pearson Education

[10] Girish Nath Jha, Muktanand Agrawal, Subash, Sudhir K. Mishra, Diwakar Mani, Diwakar Mishra, Manji Bhadra, Surjit K. Singh "Inflectional Morphology Analyzer for Sanskrit", First International Sanskrit Computational Linguistics Symposium, Paris, October 29-31, 2007

[11] Bhuvaneshwari C Melinamath, Shubhangi D.C., A G Mallikarjunmath, "Robust Morphological analyzer to Capture Kannada noun Morphology", 2011 International Conference on Future Information Technology, IPCSIT vol.13 (2011) IACSIT Press, Singapore

[12] Alberto Sanchis, David Pic´o, Joan Miquel del Val, Ferran Fabregat,Jes ´us Tom´as, Mois´es Pastor, Francisco Casacuberta, Enrique Vidal "A Morphological Analyser for Machine Translation Bsed on Finite-stateTransducers"

[13] Akshar Bharati, Amba Kulkarni, V Sheeba, "Building a Wide Coverage Sanskrit Morphological Analyzer: A Practical Approach", IIT Bombay