International Journal of Applied Research 2016; 2(6): 1086-1093

ISSN Print: 2394-7500
ISSN Online: 2394-5869
Impact Factor: 5.2
IJAR 2016; 2(6): 1086-1093
www.allresearchjournal.com
Received: 08-04-2016
Accepted: 11-05-2016

Dr. N Murali
S.V. Vedic University, Tirupati, Andhra Pradesh, India

Correspondence Author;
Dr. N Murali
S.V. Vedic University, Tirupati, Andhra Pradesh, India

Identification of multi-word groups in elementary Sanskrit texts

Dr. N Murali

Abstract
Sentences consist of a group of words. Each word has a relation with its adjacent words. Each word has its own meaning and plays a significant role in understanding the sentence correctly. Sometimes a group of words contribute a different meaning than the meaning of the constituent words. These word groups are called as phrases. The concept of phrase is not suitable to Indian languages. In Sanskrit, the order of occurrence of words has very less significance. This paper describes the role of Panini's karaka theory and discusses the issues of Sanskrit Parser developed at the University of Hyderabad as a part of Sanskrit-Hindi Accessor cum Machine Translator.

Keywords: NLP, phrases, local word grouping, chunking, karaka theory

1. Introduction
Sentence (*vakyam*) means a group of words which expresses a complete idea or thought. The basic element of a sentence is word (*padam*). Each word has its own meaning and expresses a single idea. Sometimes, the meaning of individual words has no significance in the process of analysing a sentence in order to get the overall meaning of that sentence. For example, the phrase "in front of" in English cannot be individually analysed. The concept of noun phrase and verb phrase cannot be directly applied to Indian languages and hence the term phrase in the context of Indian languages refers to the word group only (Bharati et. al, 96) [7]. Word groups for Sanskrit can be classified as Noun groups and Verb groups. The Chunker developed as a part of Sanskrit-Hindi Accessor cum Machine Translator (SHMT) has been studied to know the process of local word grouping. Certain unresolved issues in grouping the words were identified. To improve the performance of SHMT, Avyaya Analyzer (Murali N *et al*., 2012), [15] Kridanta Analyzer (Murali N and Ramasree RJ., 2011) [14] have been developed and integrated within the pipe line of SHMT. Still some unresolved issues are identified. The main reason is that the, Sanskrit is word free order language. The word may occur anywhere in the sentence. But the *vibhakti* and *vacana* helps in maintaining the coherence between the words in the sentence. Hence, grouping of words based on the sequence features is not enough for identification of word groups. (Murali and Ramasree RJ, 2013) [16] One can extract the word groups only after analyzing the entire sentence.

2. Extracting Word Groups
Grammar explains how to understand the meaning of a given sentence. Also it explains how to produce an intended sentence. There are many grammar formalisms which explain the syntax (sentence structure) and semantics (meaning) of natural languages. P*anini's* grammar is the most widely accepted grammar for Sanskrit. According to Bharati *et al*. (1996) [7], *Panini* Grammar can be described by the following points.
1. Each and every action has many nominal participants.
2. Each verb refers to an action which has
3. *Phala* (result) & *vyapara* (complex action).
4. Each action can have sub activities. Each activity has its own participants.

5. The speaker has *vivaksha* (intention/opinion) about the listener and this is conveyed in his utterance.
6. Action springs from *karta* (Agent) and is experienced by the *karma* nominal, which is known as *sakarmaka* (transitive).
7. If the action is experienced by the *karta* itself, it is known as intransitive (*akarmaka*).
8. *Karaka* relations are the relations between nominals and verbals.
9. *Karaka* relations are syntactico-semantic relations.

There are six different types of *Karaka* relations which explain the relationship of action and its participants. Different types of *Karaka* relations are described below.

2.1 *Karaka* Relations

The analysis of a sentence will be complete when all words were recognized correctly, chunked correctly and their semantic relations are identified correctly. In Sanskrit the relations between words and verbs can be understood by analyzing *Karaka* relations (Bharati A and Sangal R, 1990) [2], (Bhadra *et al*. 2009) [11]. These relations are called as transformational in modern linguistics. The term *Karaka* means any factor which contributes to the accomplishment of an action (Mishra SK, 2007) [9].

These *Karakas* specify relations who mediate between *vibhakti* of nominal and verb forms and semantic relations (Bharati A and Sangal R, 1990) [2]. Sudhir Kumar Mishra has mentioned the *Panini*'s *Karaka* theory in detail in his thesis (Mishra SK, 2007) [9]. *Karaka* relations are linked directly to the case endings or post nominal suffixes. The six *Karakas* are कर्ता-*Karta*, **कर्म**-*Karma*, **करण**-*Karana*, **संप्रदान**-*Sampradana*, **अपादान**-*Apadana* and **अधिकरणम्**-*AdhiKaranam*. The **षष्ठीविभक्ति** (*genitive case)* is not considered as a *Karaka* as it is not directly related to the verb.

To know role of all *Karaka* relations, consider the following sentence given by the UoH in their website [1]:

Da Sarathasya putrah ramah nagare koSat hastena brahmanaya dhanam dadati.

The above sentence contains all *Karaka* relations. The output of the Sanskrit-Hindi Accessor cum Machine Translator (SHMT) for the above sentence is given in the following table format which is convenient to understand the sentence. The last column "Related to" indicates the word position (word number) in the sentence.

Table 1: Completely parsed sentence

WNo	Word	Gender	Case/*prayogah*	No.	*Karaka* relation	Related to
1	दशरथस्य	M	6	1	षष्ठीसम्बन्धः	2
2	पुत्रः	M	1	1	विशेषणम्	3
3	रामः	M	1	1	कर्ता	9
4	नगरे	N	7	1	अधिकरणम्	9
5	कोशात्	M	5	1	अपादानम्	9
6	हस्तेन	M/N	3	1	करणम्	9
7	ब्राह्मणाय	M	4	1	संप्रदानम्	9
8	धनं	M/N	2	1	कर्म	9
9	ददाति	Verb	कर्तरि	1	अभिहितकर्ता	3

[1] http://sanskrit.uohyd.ernet.in/scl/SHMT/shmt.html

From the above table it can be understood that

i. All words in the sentence will agree with each other in terms of their number with the verb.
ii. The adjacent words which are in same case and number have a relation called *viseshana* - *viseshya* (पुत्रः रामः) or the entire group will have relation with some *avyayas* like "*ca*" or "*iti*" etc.
iii. The Karaka relation of each word with respect to the verb can be determined by the verb's *akanksha* or demands.

After understanding the basic concepts like word groups and *Karaka* relations, the Sanskrit-Hindi Accessor Cum Machine Translator (SHMT) developed at UoH has been tested.

3. Sanskrit Hindi Accessor cum Machine Translator

In 2006, five consortia have been formed by the Ministry of Information Technology, Government of India. Sanskrit to Hindi Machine Translation system is one among the five. Under this project, Sanskrit-Hindi Accessor Cum Machine Translator (SHMT) has been developed at University of Hyderabad, Hyderabad. The SHMT has been developed based on *Anusaraka* guidelines. SHMT translates the given text in Sanskrit language into Hindi language. Various input methods have been provided in its web interface. SHMT consists of different modules like MA, POS Tagger, Chunker, Parser and Translation Modules. The concept of Anusaraka is "preservation of information". SHMT provides the output in different layers. Each layer provides information obtained at different levels like MA, POS Tagging, Chunking, *Karaka* relations, Hindi translation etc. A test data of 2000 sentences ranging from simple to complex sentences were collected from *Candamama* fortnightly children's magazine. The main objective of the testing is to know how SHMT is treating word groups and *Karaka* relations. While testing the SHMT, the following performance issues were noticed.

- Unable to recognize *Kridanta* as a verb
- Unable to understand the sentence where verb is absent.
- Unable to recognize *avyayas*
- Unable to recognize noun in certain cases
- Wrong chunking
- Unable to chunk the word groups
- Failure in analyzing the complex sentences
- Erroneous semantic roles
- Unable to recognize the semantic roles of *Avyayas*

An attempt is made to improve the performance of the parser by solving the issues mentioned above. The next section describes the procedure adopted for making necessary improvements to the SHMT.

4. Improvements made to the SHMT

The Avyaya Analyzer (Murali *et al*., 2012) [15] and Kridanata Analyzer (Murali *et al*. 2011) [14] have been successfully integrated within the pipeline of SHMT. The improvements have been carried out only to recognize the words which were not recognized by the MA developed at University of Hyderabad. Hence, without disturbing the functionality of the Anusaraka system, Avyaya Analyzer and Kridanta Analyzers were included safely within the pipeline of Anusaraka system. The SHMT has been tested with the newly improved MA.

Consider the following sentence which is in WX notation and the output is in Unicode.
Example: *vAlmIkiH Saswra-ukwena krameNa svAgawIkqwya apqc Caw*.

The 4th word in the sentence *svAgawIkqwya* (read as *svagatikritya*) is a *kridanta* which is not recognized by the SHMT. This word has a prominent role in the sentence. The word *svAgawIkqwya* is derived from the combination of *upapada*, root and a suffix. The first one is *upapada* (*svagata*), the second is the root (*kru*) and the third is suffix (*lyap*). The *krit* suffix *lyap* acts as past participle in the sentence. In the above sentence, *valmikih* is performing two actions. First, he welcomed (*naradah*) and then asked (*naradah*). The relation of *svagatikritya* to *aprucchat* is called as "*purvakalah*". After integrating the modules that have been developed to improve the performance of the MA in the pipeline of SHMT and only after analyzing *svagatikritya* the parser is able to recognize the relation between all words correctly. The output is given below.

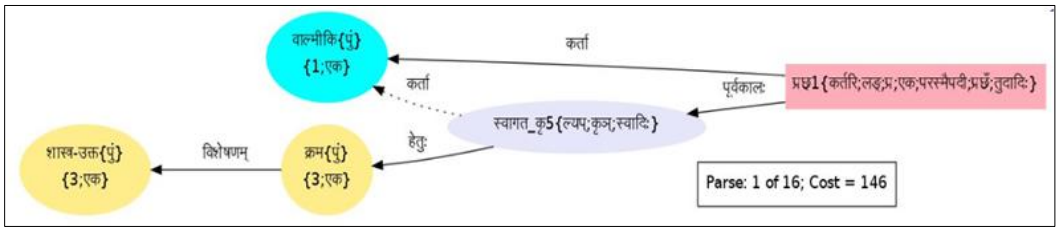

Fig 1: Output of the SHMT after integrating the module

Even after recognizing the nouns, *avyayas*, *kridantas* and certain verbs with prefixes it is found that there is no great improvement in the performance of the SHMT in giving semantic relations. Hence an attempt is made to improve the performance of SHMT by applying certain rules at different levels. The following section discusses procedure adopted to improve the SHMT.

4.1 Parser Fine Tuner

After thorough analysis of the test data and getting feed back from the initial testing, several issues with regard to the performance of SHMT has been identified and mentioned in section 4.5.1. Hence to overcome these problems a separate method has been adopted to improve the performance of the system without disturbing its originality. The output of the SHMT is the input for these modules. Figure 4.9 shows various stages in which improvements have been made to SHMT.

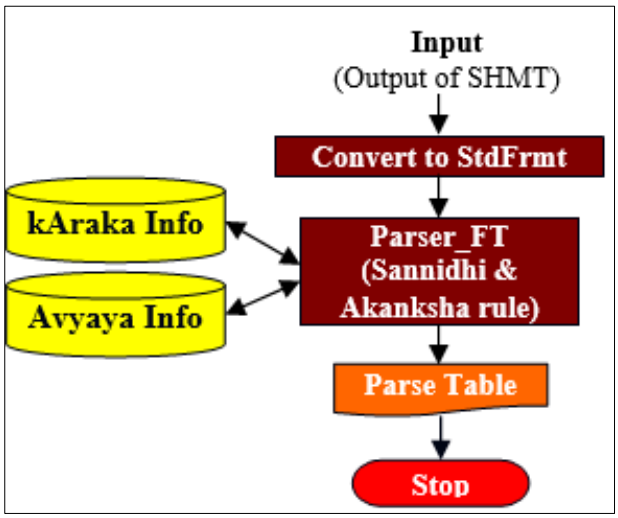

Fig 2: Stages in improvements made to SHMT

The first stage in this process is to bring the output of the parser into a standard format. SHMT provides the output at different stages like MA, Chunker, WSD, Parser and translation levels. The output of the modules developed in the present study must be tuned in accordance to the SHMT requirement. The outputs of the Avyaya Analyzer and Kridanta Analyzer have been converted into the format compatible to the SHMT. The output of WSD module has been taken to improve the performance of the SHMT. The format is very simple, easy to understand and it shows all the features like type, case, *prayoga* of verbs, number, relation name, the number of the word to which it is related, root, *krit* suffix, and *ga¸a*. If the parser returns null value for any of these features, it will be represented by 0 (zero).

4.1.1 Karaka Info: To decide the *Karaka* relations of each word in the sentence to the verb, it is essential to know those *Karakas* which a root can demand. Verb groups are called *demand groups* as they make demands about their *Karaka* and noun groups are called *source groups* as they satisfy the demands of the verb group. *Karaka* relations are mentioned in section

4.4.2. It is mentioned that *Karaka* relations can be decided based on the vibhakti. The problems with ***Vibhakti – Karaka*** mapping are mentioned below.

1. Different *Vibhaktis* may represent the same *Karaka* information.
2. Same *Vibhaktis* can be used to represent different *Karaka* information.

The important insight regarding the *Karaka-vibhakti* mapping is that it depends on the verb and TAM. Hence it has been decided to study all roots in the entire test data thorough and to find out its demands by analyzing each and every sentence wherever that root occurs. Totally 241 entries of roots with and without *upapadas* and *upasargas* and their *Karaka* expectations (*akanksha*) have been identified and recorded separately. By doing so, it will become easy to decide the *Karaka* relation of each word to the verb according to the *akanksha* (expectancy) of that particular verb. If a word in a particular *vibhakti* is found in the sentence where the verb does not expect any word in that *vibhakti* as its *karaka*, it means that there is no relation between that particular word and the verb; and the word is related to any

other word in the sentence. Table 2 presents the structure of the *karaka* info table. Consider the verbal root "*jval*" which belongs to *bvadi* and is an *akarmaka* root i.e., it will not expect any direct object. If the same root is prefixed with "*pra*", then it will function as *sakarmaka* root i.e., it will expect an object. This type of intricacies have been identified carefully and stored in the Karaka Info table.

Table 2: Karaka Info table in WX notation

Prefix	Root	Part	DNo	Type	*Karta* (1)	*Karma* (2)	*Karana* (3)	*Sampradana* (4)	*Apadana* (5)	*Adhikaranam* (7)
	as	02	1065	a	1	0	3	0	0	7
	jval	01	804	a	1	0	0	0	0	7
pra	jval	01	804	a	1	2	0	0	0	7
	RTA	01	928	a	1	0	3	0	5	7
vi#ava	RTA	01	928	a	1	2	3	4	0	7

4.1.2 Avyaya Info: One of the problems with SHMT is it has failed in identifying the relation of certain *avyayas* with the verb at several instances. To overcome this all *avyayas* in the entire test data has been collected and analyzed their nature like its role in the sentence and its relation with the verb or the word. The first field "Avyaya" indicates the *avyaya*. The second field "RelationName" indicates the relation between the *avyaya* and *padam* or verb. The third field "LinkedTo" indicates the category of the word to which the avyaya is to be linked. Table 3 describes the Avyaya Info table:

Table 3: Avyaya Info table in WX notation

Avyaya	Relation_Name	Linked_To
awaH	hewuH	paxam
awra	xeSAXikaranam	XAwuH
Jatiwi	kriyAviSeRanam	XAwuH
waxA	kAlAXikaranam	XAwuH
Prawi	anuyogI	paxam
prAwaH	kAlAXikaranam	paxam
bahuXA	kriyAviSeRanam	XAwuH

4.2 Parser Fine Tuner Module

The main task of this module is to analyze each unrecognized word or word groups in the sentence and to identify its relation with the verb. In this task, two rules have taken into account i.e., *sannidhi* and *akanksha*. According to *sannidhi* rule, when a word in the sentence is not given a relation with the verb and if there are more than one verb in the sentence then that word will be related to a verb which is nearest to it. The *akanksha* rule says that each root expects some *Karaka* relations. If an unrecognized word is in a particular *vibhakti* which is not expected by the root, then that word is not having any relation with the verb and it may be related to some other word in the sentence. Consider the following sentence as an example.

Sentence1: **देशस्य कस्मिन् कोणे किं प्रवर्तते इति सः गुप्तचराणाम् द्वारा ज्ञातुम् प्रभवति स्म।**

DeSasya kasmin koNe kim pravartate iti sah guptacaranam dvara jnatum prabhavati sma.

The karaka relations are not given by SHMT. In this sentence there is a conjunction *iti* which connects two clauses. There are two verbs at 5th and 11th positions and one *avyaya* of type *kridanta* at 10th place. The conjunction *iti* becomes object to this avyaya. The *avyayas* formed by *krit* suffix *tumun* is used as an infinitive. This suffix states a purpose or *prayojanam* (Sanskrit Manual). Every word in the sentence has been related to the verb which is nearer to them. Nearest verb has been identified by calculating the distance between the word and the verbs. In relating the nearest verb, priority was given to forward direction while searching a verb. If a verb is not found while searching in forward direction, then the search will be done in backward direction and the verb will be linked. Table 4 shows the output of the Parser Fine Tuner by applying the *sannidhi* and *akanksha* rules.

Table 4: Output of the Parser Fine Tuner for Sentence1

WNo	Word	Type	Case	No.	Relation	Related	XAwu	Kqw	GanaH
1	देशस्य	ना	6	1	षष्ठीसम्बन्धः	2			
2	कस्मिन्	सर्व	7	1	विशेषणम्	3			
3	कोणे	ना	7	1	अधिकरणम्	5			
4	किं	सर्व	1	1	कर्ता	5			
5	प्रवर्तते	क्रि.	0	0	सम्बन्धः	6	वृतु		भ्वादिः
6	इति	अव्य	0	0	कर्म	10			
7	सः	सर्व	1	1	कर्ता	11			
8	गुप्तचराणाम्	ना	6	3	षष्ठीसम्बन्धः	9			
9	द्वारा	ना	3	1	हेतुः	10			
10	ज्ञातुम्	अव्य	0	0	प्रयोजनम्	11	ज्ञा	तुमुन्	क्र्यादिः
11	प्रभवति	किं	0	0	क्रि.	7	भू		भ्वादिः
12	स्म	अव्य	0	0	सम्बन्धः	11			

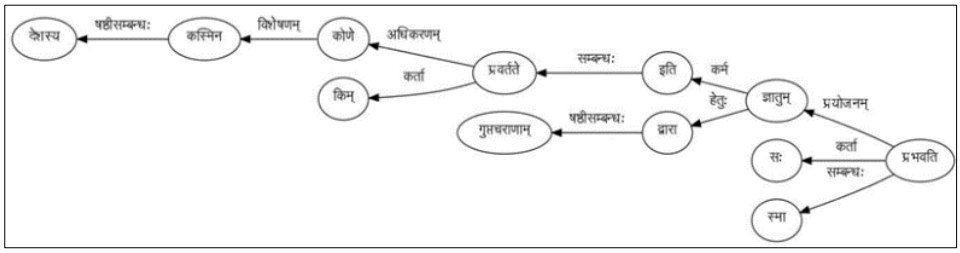

Fig 3: Graph showing the output of Parser Fine Tuner for Sentence1

From the output of the Parser Fine Tuner module we can understand the application of *akanksha* and *sannidhi* rules. The improvements made to the SHMT have been illustrated through the issues discussed in section 4.

4.2.1 *Kridanta* as main verb

E.g. तदा नारदः श्रीरास्य कथाम् विस्तरेण श्रावितवान् ।

Tada naradah Sriramasya katham vistareNa Sravitavan.

Then Narada narrated the legend of Rama in detail.

In the above sentence, MA of UoH has recognized the 6th word as a noun. This is a *Kridanta* and contains *Krit* suffix *ktavatu*. As it is mentioned earlier the *Krit* suffixes like *ktavatu* and *kta* can be treated as verbs in the absence of main verb. The Parser Fine Tuner module can recognize these types of words as verb. The relation of the 5th word has been showed as *Karanam*, which is not correct.

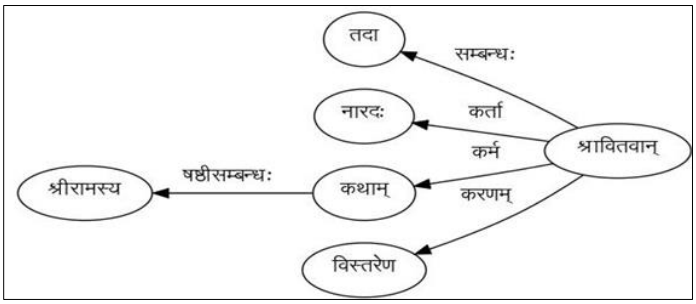

Fig 4: Parser Fine Tuner's output for a sentence with *kridanta* functioning as a verb

4.2.2 Absence of verb

वैवस्वतः सूर्यस्य पुत्रः ।

Vaivasvatah suryasya putrah.

Vaivaswata the son of the Sun.

In Sanskrit a sentence without a verb can be easily created. If a sentence does not contain a verb, then generally the Sanskrit scholars will assume any of root अस् forms like अस्ति, आसीत्Â etc. to the sentence. The verbal forms of the root "*as*" functions as an implicit "be" verb in English (Bharati *et al*., 1995). By this assumption, this problem has been solved.

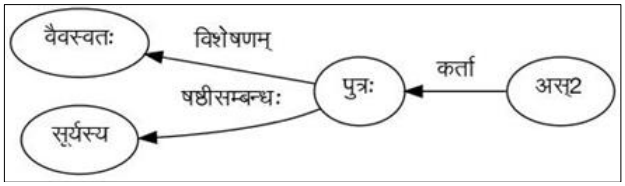

Fig 5: The graph obtained after assuming "*as*" as main verb

4.2.3 Identification of *avyaya:*

सूर्यवंशीयाः राजानः अयोध्याम् राजधानीम् परिकल्प्य कोसलदेशम् पालितवन्तः ।

SuyavamSiyah rajanah ayodhyam rajadhanim parikalpya kosaladeSam palitavantah.

Having made Ayodhya as capital, the kings of the dynasty of Sun ruled over Kosal.

The fifth word in the sentence is a *kridatna avyaya*. It has been recognized as noun and *samb°dhana vibhakti*. This word was not related to the verb by SHMT. The fourth word is karma to the fifth word which is also not recognized by SHMT. The Parser Fine Tuner has recognized this issue and has given the correct results.

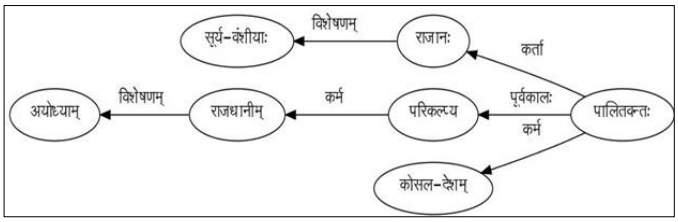

Fig 6: Graph obtained after identification of *avyaya* as an adverb

4.2.4 Identification of noun

E.g. ब्रह्मणः एतया प्रेरणया वाल्मीकिः रामायणम् रचितवान् ।

*Brahmanah etaya **preranaya** valmikih ramayanam racitavan.*

By this type of motivation by Brahma, Valmiki composed the Ramayana.

In the following sentence, the third word is an adverb which was not recognized by the SHMT. The Parser Fine Tuner will identify the unrecognized word and checks with the adjacent word, if any of the adjacent word's case ending matches with the unrecognized word, then matched words are treated as a group and the features of the adjacent word will be shared to the unrecognized word as it has been done in the case of the following sentence.

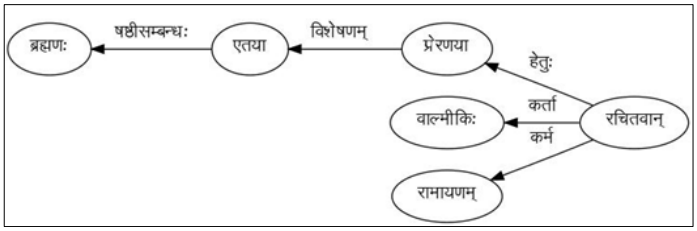

Fig 7: Graph obtained after recognizing noun

4.2.5 Wrong chunking

E.g. **सः ऐश्वर्येण कुबेरसमानः पराक्रमेण इन्द्रतुल्यः च आसीत् ।**

Sah aiSvaryeNa kuberasamanah parakrameNa indratulyah ca asit.

He was akin to Kuber in opulence and in valor to Indra. In the following sentence fourth word has to be linked to the fifth word in the same way as it was done in the case of second and third words. The third and fifth words have to be related to the conjunction *ca*. The Parser Fine Tuner has solved this issue by relating the words which are in same *vibhakti* and *vacana* into a group and shared the grammatical features to the unrecognized or wrongly chunked words.

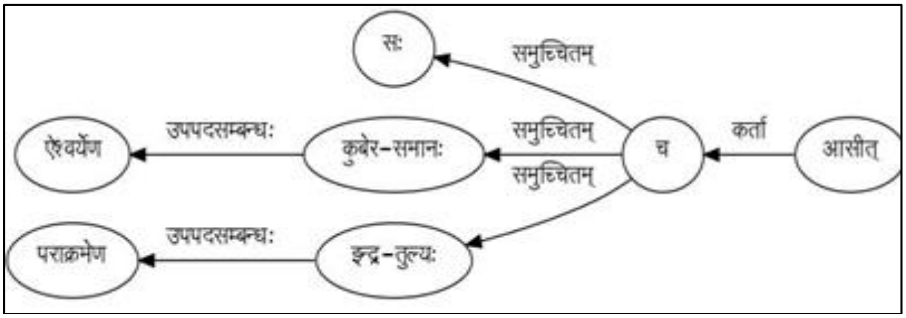

Fig 8: Graph obtained after correcting wrongly chunked sentence

4.2.6 No chunking

दृष्टिः जयन्तः जयः सिद्धार्थः अर्थसाधकः अशोकः मन्त्रपालः सुमन्त्रः च इति अष्टौ मन्त्रिणः दशरथस्य ।

DriStih jayantah jayah siddharthah arthasadhakah aSokah mantrapalah sumantrah ca iti ashtau mantriNah daSarathasya.

There were eight ministers of Dasaratha namely Drishti, Jayanta, Jaya, Siddharta, Arthasadhaka, Ashoka, Mantrapala and Sumantra.

ca and *iti* are used as conjunctions. There is no verb in the above sentence. The words from one to eight are in same *vibhakti* and *vacana* and hence they have been grouped together and related to the conjunction *ca*. The remaining words have been related according to the *akanksha* of root "अस्2".

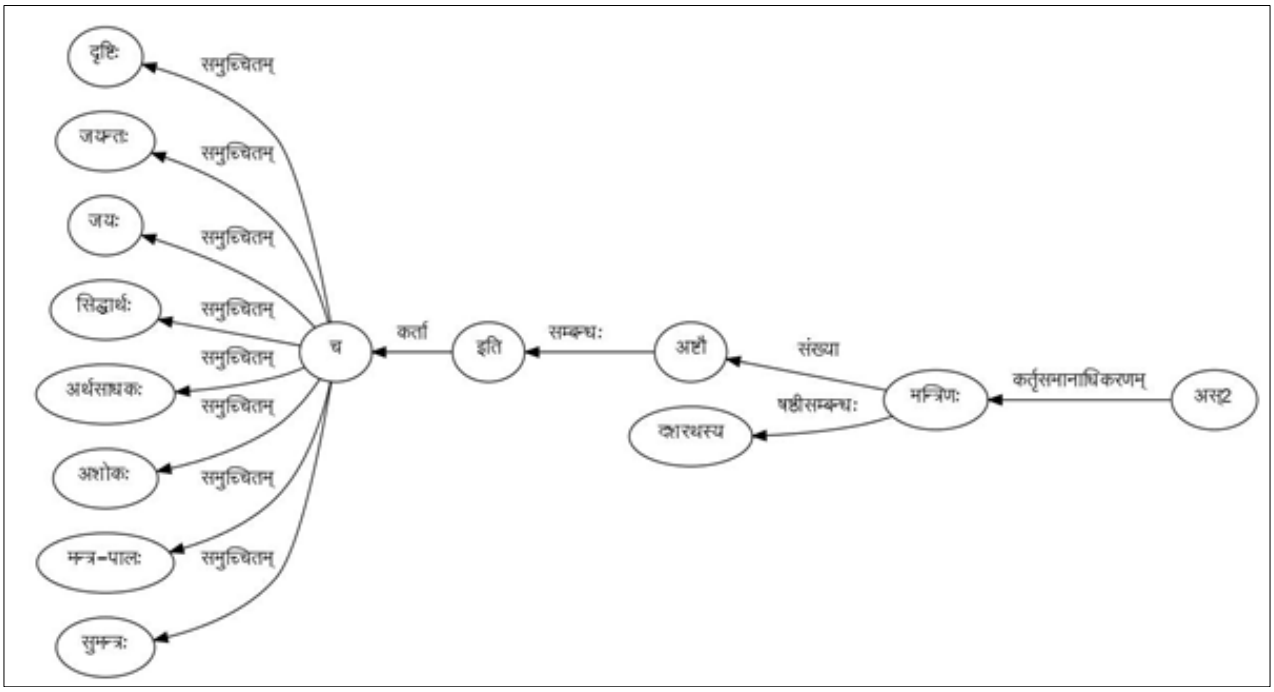

Fig 9: Graph obtained after chunking and proper analysis

4.2.7 Failure in analyzing the complex sentences

यावत् तत् स्थास्यति तावत् भवतः नाम कीर्तिः च अपि जनानाम् रसनासु विलसिष्यति ।

Yavat tat sthasyati tavat bhavatah nama kirtih ca api jananam rasanasu vilasishyati.

As long as that stands, so long your name and fame will also figure at the tongues of everybody.

The above sentence is considered as a complex sentence as it contains the conjunctions *yavat.. tavat* which are related to time. The other conjunctions like *yada.. tada, yadi.. tarhi, yat.. tat* will also functions in the same way.

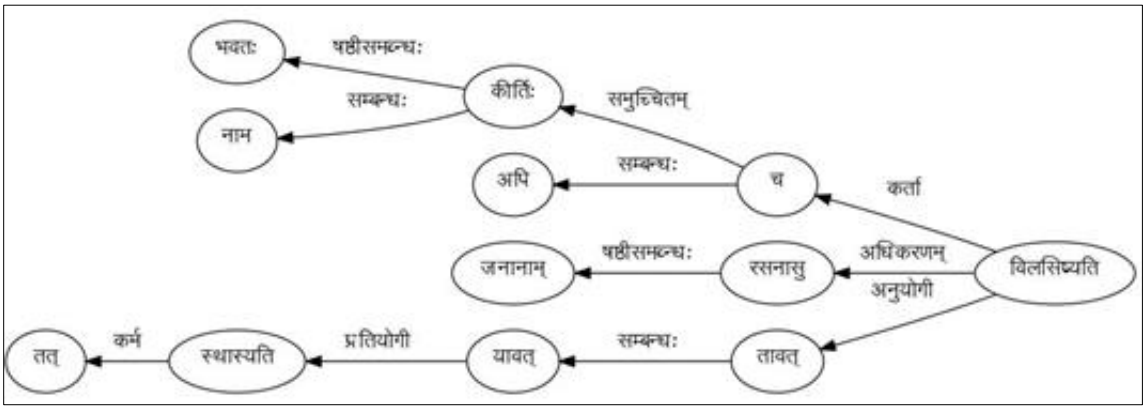

Fig 10: Output of a complex sentence

4.2.8 Erroneous semantic roles

एतत् दृष्टवतः वाल्माकेः हृदये करुणा उत्पन्नः।

Etat drishtavatah valmikeh hridaye karuNa utpanna.

Compassion emerged in the heart of Valmiki who witnessed this (incident).

SHMT has recognized the fourth word wrongly which is an adhikarana and is to be related t o the main verb in the sentence.

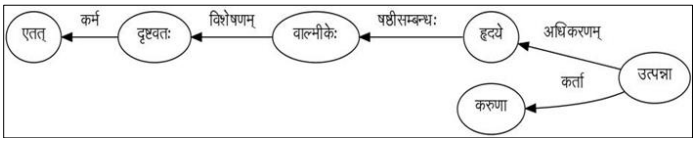

Fig 11: Output after identification of a semantic role of a word

4.2.9 Semantic roles of *Avyaya*

इति उक्त्वा ततः अन्तर्हितः जातः ।

Iti uktva tatah antarhitah jatah.

Having stated this he became invisible.

SHMT has failed to retrieve the relations of avyayas. This problem have been dealt with the help of Avyaya Info table.

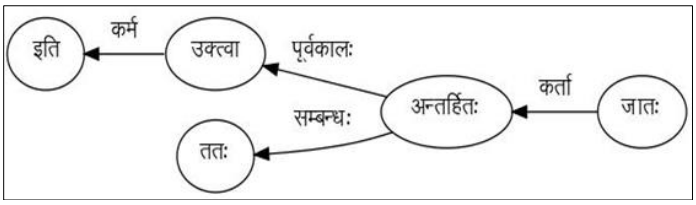

Fig 12: Output after identifying the semantic role of an avyaya

5. Evaluation of Parser Fine Tuner

Evaluation of any NLP application is essential to know its performance. For Indian languages there are no Gold Standard Data. Hence, the test data was manually parsed by the Sanskrit grammarians and then utilized to compare the results given by Parser Fine Tuner.

Sanskrit Parser (SHMT) has been tested with 100 sentences collected from children's fortnightly Sanskrit magazine called "*Chandamama*". Initially the SHMT is able to analyze 20 sentences where all words have been recognized correctly, chunked correctly and where *Karaka* relations are given correctly. The newly developed Parser Fine Tuner modules were tested with this test data. The input for the Parser Fine Tuner is the output of the SHMT. After applying the newly developed Parser Fine Tuner, 52 wrongly analyzed sentences were analyzed correctly. The results are evaluated with manually parsed data. The overall performance of the parser has been improved to 72%. The SHMT was tested with another data set of two hundred sentences, out of which only fifty three sentences were parsed correctly. The output of the SHMT was given to the Parser Fine Tuner. The results are portrayed below.

Table 5: Results of Parser Fine Tuner

Total number of sentences tested	2000
Correctly parsed sentences by SHMT	530
Correctly parsed by Parser Fine Tuner	760
Partially parsed by Parser Fine Tuner	330
Wrongly parsed by Parser Fine Tuner	380

From the above table, it is clearly evident that the Parser Fine Tuner has optimized the performance of SHMT from 26.5% to 64.5%.

6. Conclusion

Different word groups and *Karakas* have been discussed in brief with examples. SHMT has been thoroughly analyzed to ascertain how it is identifying the word groups, and the *Karaka* relations among the words in the sentence. The performance of the SHMT has been improved in the following way:

- Karaka Info table was prepared to know the functioning of each verb with or without *upasargas*, *upapadas*. Each root along with its demands has been recorded carefully.
- Certain rules like *akanksha* & *sannidhi* have been applied at different levels.
- Separate rules have been formulated to treat the *avyayas* like *ca*, *iti* etc.
- A sentence without a verb is analyzed by assuming a verbal root "*as*2".
- In case of the absence of a verb, if the *kridanta* is present in the sentence, then the *kridanta* was treated as a verb.

Tried to disambiguate a word by comparing with the adjacent words and applied the features of the matching adjacent word for the word in question.

7. References

1. Antoine R. Sanskrit Manual For High Schools Part – I, 6th Ed. Xavier Publication, Culcutta, 1968.
2. Bharati A, Sangal R. A Karaka Based Approach To Parsing Of Indian Languages. In Proceedings Of The 13th

Conference On Computational Linguistics- Association For Computational Linguistics. 1990, August;3:25-29.
3. Bharati A, Chaitanya V, Sangal R, Ramakrishnamacharyulu KV. Natural language processing: a Paninian perspective. New Delhi: Prentice-Hall of India. 1996, p. 65-106.
4. Rama Nath Sharma. Astadhyayi of Panini, 2. Munshiram Manoharlal Pblishers Pvt. Ltd., New Delhi. 2000.
5. Sag IA, Baldwin T, Bond F, Copestake A, Flickinger D. Multiword expressions: A pain in the neck for NLP. In Computational Linguistics and Intelligent Text Processing. Springer Berlin Heidelberg. 2002, p. 1-15.
6. Dias G. Multiword unit hybrid extraction. In Proceedings of the ACL 2003 workshop on Multiword expressions: analysis, acquisition and treatment-. Association for Computational Linguistics. 2003, July;18:41-48.
7. Bharati A, Rajeev Sangal, Dipti M Sharma, Sriram V, T Papi Reddy. Handling Multi-word Expressions without Explicit Linguistic Rules in an MT System. In Proceedings of Seventh International Conference on TEXT, SPEECH and DIALOGUE - 2004. Brno, Czeck Republic. 2004.
8. Katz G, Giesbrecht E. Automatic identification of non-compositional multi-word expressions using latent semantic analysis. In Proceedings of the Workshop on Multiword Expressions: Identifying and Exploiting Underlying Properties. Association for Computational Linguistics. 2006, July, p. 12-19.
9. Mishra SK. Sanskrit Karaka Analyzer for Machine Translation. Unpublished Doctoral Thesis. Jawaharlal Nehru University, New Delhi. 2007.
10. Kunchukuttan A, Damani OP. A System for Compound Noun Multiword Expression Extraction for Hindi. In Proceeding of 6th International Conference on Natural Language Processing, 2008, p. 20-29.
11. Bhadra M, Singh SK, Kumar S, Agrawal M, Chandrasekhar R, Mishra SK. Sanskrit Analysis System (SAS). In Sanskrit Computational Linguistics. Springer Berlin Heidelberg, 2009, p. 116-133.
12. Birla VK, Ahmed MN, Shukla VN. Multiword Expression Extraction–Text Processing. Proceedings of ASCNT CDAC, Noida, India, 2009, 72-77.
13. Ckakraborty T. Identification of Noun-Noun (NN) Collocations as Multi-Word Expressions in Bengali Corpus. In Student Session, International Conference of Natural Language Processing, 2010.
14. Murali N., Ramasree RJ. Kridanta Analyzer. In proceedings of Annual International Conference on Emerging Research Areas. Organized by Amal Jyothi College of Engineering, Kerala. April, 2011, p. 63-66.
15. Murali N, Ramasree RJ, Acharyulu KVRK. Avyaya Analyzer: Analysis of Indeclinables using Finite State Transducers. International Journal of Computer Applications. 2012;38(6):7-11.
16. Murali N, Ramasree RJ. Rule-based Extraction of Multi-Word Expressions for Elementary Sanskrit Texts. International Journal of Advanced Research in Computer Science and Software Engineering. 2013 November;3(11):661-667. ISSN: 2277 128X