Analysis and graphical representation of Navya-Nyāya expressions - *Nyāyacitradīpikā*

Arjuna S. R. *and* Amba Kulkarni

Abstract: In this paper we present a semi-automatic computational tool to represent a Navya Nyāya expressions through conceptual graphs of Sowa. This tool consists of a domain specific segmenter, a semi-automatic constituency parser and a context free parser that translates an NN Expressions into a conceptual graph.

1 Introduction

The problem of ambiguity in a Natural language was recognised long back in the Indian Grammatical Tradition. Indian logicians who were engaged in the philosophical debates realised the need of expressing communications in an unambiguous way. Their efforts culminated in a new school *Navya-Nyāya* 'Neo-Logic'. The seeds of *Navya-Nyāya* (NN) School of Philosophy are found in the Udayanācārya's work (Dravid 1996). Gaṅgeśa (12^{th} century), the founder of NN in his *Tattvacintāmaṇi* introduced the technical language of NN. This language introduces a few conceptual terms and provides a mechanism to express the underlying cognitive structure corresponding to a linguistic expression in an unambiguous way. We find the use of this language in as diverse fields as Mīmāṃsā 'exegesis' (Shastri 1991), Vyākaraṇa 'grammar' (Dwivedi 2005), Sāhitya 'literature' (B. Jha and M. Jha 1993) apart from the texts dealing with philosophical discussions. The agnostic schools such as Jaina (Sanghavi 1939) and Buddhists (Dharmakīrtı 1977) as well used this language. Bhattacharya (1990: 130) rightly observes

> "Thus this language could be used in every sphere where cognition, belief, doubt, and other epistemic and doxastic factors play an essential role. This explains why this language could be used universally in the humanities, where the epistemic factors predominate.".

Good understanding of the technical language of NN is thus necessary to understand the texts of Indian origin in various disciplines. There are two major problems we envisage in understanding the technical language of NN. The first one is understanding the semantics of the basic vocabulary and the second one is understanding the underlying syntactic structure. While Ingalls (1951), Matilal (1977), Mohanty (2000) and Shaw (1980), to name a few, worked towards understanding the concepts and comparing them with the western logic counterparts, Bhattacharya (1990) and Ganeri (2008) explored the underlying syntactic structure and the grammar as well.

Sanskrit is very rich in compound formation. This feature of Sanskrit has been utilized to its full extent by the Indian logicians in describing the cognitive structures using the technical language of NN. Such expressions are typically exceptionally long, many-a-times one compound running into pages. While the oral transmission of knowledge and all serious debates could sustain these long compounds, modern scholars not trained in oral tradition find it difficult to understand these long expressions. A two dimensional representation of such linear strings in the form of diagrams brings in more clarity thereby helping in understanding the relations between various components. A need for visual representation of the Navya-Nyāya expression (NNE) was felt earlier as well. We find the use of pictures as early as in the early 20^{th} century by Vāmācaraṇabhattāchārya (Patil 2014). In recent times V. N. Jha (1987) and Wada (2007) used diagrammatic representations in teaching. A. Kulkarni (1994) proposed a scheme to represent the NNE using the conceptual graphs of Sowa. Varakhedi (2004, 2013), and T. Kulkarni and Joshi (2013) have also used these graphs for better understanding of NNEs.

In this paper, we describe a computational tool (Nyāyacitradīpikā) that translates an NNE into a conceptual graph (CG). In the next section we describe the stages involved in parsing an NNE. The efforts in building a domain specific segmenter developed to handle NNEs are presented in the third section. The constituency parser of an NNE is described in section four. In the fifth section we present the CG notation for an NNE. The sixth section defines the Context Free Grammar to parse and translate an NNE into a CG. Next we present the tool Nyāyacitradīpikā that integrates all the modules discussed above to render an NNE as a CG. We conclude with our observations on the utility of this endeavour.

2 Parsing an NN expression

An NNE is a compound. A compound, in Sanskrit, is written as a single word without any gap or hyphen in between the components. The components are joined together following euphonic changes. Compound formation also results in the loss of case markers and the accent. This sometimes results in an ambiguous compound. Kumar, Mittal, and A. Kulkarni (2010) describe the steps involved in processing Sanskrit compounds and also discuss the associated computational complexity. The steps are

1. Splitting a compound into components.
   This involves undoing euphonic transformations.

2. Analysing its constituent structure.
   At this stage a compound is analysed showing how the components are grouped together.

3. Identifying relations between the components.
   Now the relations between the components thus grouped are made explicit.

4. Providing a paraphrase of the compound.
   Finally a paraphrase of the compound is generated.

We illustrate these steps with two examples: an English one followed by an NNE.
Example 1: Consider an English compound ‘lake water pollution reduction log’. We skip the first step, since the components here are already split.

1. Constituency analysis for this compound is
   <<<<lake-water>-pollution>-reduction>-log>

2. Relations between the components are now marked.
   <<<<lake-water>**T7**-pollution>**T6**-reduction>**T7**-log>**T6**

Here **T** stands for *Tatpuruṣa* (an endo-centric) compound and the numbers 6 and 7 indicate the genitive and the locative case markers.

3. The paraphrase of a compound is obtained by supplying the missing prepositions corresponding to the compound type. Thus the paraphrase of the above compound is
Log of the reduction in pollution of water in lake.

Example 2: Consider now an NNE which defines earth as a substance which has smell as its characteristic property.
gandhatvāvacchinnagandhaniṣṭhādheyatānirūpitādhikaraṇatāvatpṛthivī. (1)

1. After splitting the compound into its components, we get
gandhatva-avacchinna-gandha-niṣṭha-ādheyatā-nirūpita-adhikaraṇatā-vat-pṛthivī.
Here the components are separated by hyphen.

2. The constituency parse of this compound is
<<<<<gandhatva-avacchinna>-<<gandha-niṣṭha>-ādheyatā>>-nirūpita>-adhikaraṇatā> ˆvat>-pṛthivī>

3. After identifying the relations between the components, we get
<<<<<gandhatva-avacchinna>**T3**-<<gandha-niṣṭha>**T7**-ādheyatā>**K**>**K**-nirūpita>**T3**-adhikaraṇatā>**K** ˆvat>-pṛthivī>**K**
where K, $T3$, $T6$, and $T7$ stand for *karmadhāraya*, *tatpuruṣa* with instrumental case, *tatpuruṣa* with genitive case and *tatpuruṣa* with locative case suffix respectively. These are all endo-centric compounds, with a requirement of nominative, instrumental, genitive and locative case suffixes during paraphrasing.

4. Finally the paraphrase of this compound is
Sanskrit: *gandhatvena avacchinnā, gandhe niṣṭhā yā ādheyatā, tannirūpitā yā adhikaraṇatā, tadvatī pṛthivī*
Gloss: by_smell-ness delimited in_smell residing which superstratum-ness determined_by_that which substatum-ness that possessing Earth
English: Earth which has substratum-ness which is determined by the superstratum-ness that is residing in the smell and is delimited by the smell-ness.

In the following sections, we present computational modules to handle the first three stages of analysis, in the domain of Navya-Nyāya expressions.

3 Segmenter for NN expressions

Word segmentation is important for languages like Sanskrit which is so much influenced by the oral tradition that the word boundaries undergo euphonic changes resulting into a continuous string of characters. The rich productive morphology resulting into the formation of long compounds aggravate the problem. There are significant efforts in this area in the past. Huet (2006), Hyman (2009), Mittal (2010), Kumar, Mittal, and A. Kulkarni (2010), Natarajan and Charniak (2011) and Huet and Goyal (2013) have contributed efficaciously to this field. All these efforts centered around general Sanskrit texts only. For much more complex texts such as NNEs, a domain-specific segmenter is needed. An NNE is characterized by its use of long compounds consisting of a specialized vocabulary of technical terms and rich usage of secondary derivational suffixes (*taddhita*). We report below on our earlier efforts in building a segmenter for NNE, followed by the current effort resulting in the remarkable improvement in the results.

3.1 Earlier efforts

Two Sanskrit segmenters easily available for experimentation were Heritage segmenter[1] and the Saṃsādhanī segmenter[2]. We first started with the enhancement of Heritage splitter. As a first step, we manually collected NNEs from *Āloka* (Varadacharya 2007) commentary on *Tarkasaṅgraha* and from *Pañcalakṣaṇīsarvasvam* (Sastry 2005). Total 49 expressions were collected from *Āloka* commentary and 352 expressions from *Pañcalakṣaṇīsarvasvam* of *Mathurānātha*.

In order to handle NNEs, the Heritage segmenter was enhanced by adding new databanks for the inflected forms of some *taddhita* suffixes. NN technical words were included in the lexicon and a few segmenter transitions were added to manage the productive usage of *taddhita* suffixes. The recall of the Heritage segmenter after these enhancements was 91% (Arjuna and Huet 2014). The segmenter produced large number of solutions (on an

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

[2] http://sanskrit.uohyd.ernet.in/scl/sandhi_splitter/index.html

average running in thousands and sometimes even millions) bringing down the precision. The main advantages of this segmenter are its robust-ness, and a lean user interface (Huet and Goyal 2013) capable of representing exponential solutions compactly in a 2 dimensional space.

In order to study the reasons behind the large number of solutions, we used the segmenter of *Saṃsādhanī* and built a recursive greedy segmenter. The main aim of this experiment was to reduce the over-generation and prioritize the solutions bringing the most likely solution to the top. We studied the behaviour of the tool on the NNE corpus collected earlier and observed that the derivational suffixes (*taddhita*) like *ka* and primary derivational suffixes (*kṛt*) like *ṇvul* and *kta* in compound formation need special treatment because when components ending in such suffixes occur as an iic (*in initio compositi* or *samāsa-pūrvapada*) they undergo *puṃvadbhāva*[3] (resulting into a word in masculine gender). The salient features of the algorithm that reduces the over generation and brings the most appropriate solution to the top are :

1. The string is split recursively remembering the sandhi rule and the split positions to avoid splitting morphologically valid bigger chunks further. This controls the granularity.

2. NN vocabulary is preferred over the non-NN vocabulary.

3. A split without a single NN technical term is considered as an over-generation. More formal version of this constraint will be presented later.

4. Sandhi rules with their frequency noted in the Sanskrit Consortium Corpus[4] are used to get the most desired output to the top.

While we could maintain the recall with this approach to 91% as in the previous approach, we could also enhance the precision with the new approach. On an average there were only a few hundred solutions as against thousands or millions in the previous approach. This segmenter had

[3] Paṇinian rule for this application is - "*puṃvatkarmadhārayajātīyadeśīyeṣu*" *A. 6.3.42*

[4] This corpus is developed as a part of the project 'Development of Sanskrit Computational Tools and Sanskrit-Hindi Machine Translation System' sponsored by TDIL, Government of India. It contains text from various fields ranging from children stories, dramas, purāṇas to Āyurveda texts.

No of solutions	Cases	Percentage
0-5	196	55.7
6-10	56	15.9
11-100	72	20.4
101-1000	13	3.6
$>$ 1000	3	1.0
No Split	12	3.4
Total	352	100.0

Table 1
Performance of Saṃsādhanī splitter

an advantage of reducing the granularity thereby increasing the precision, without compromising the recall. The performance of this segmenter on 352 examples from *Pañcalakṣaṇīsarvasvam* are shown in Table 1. For more detailed account of both these approaches refer to Arjuna and A. Kulkarni (2014).

3.2 *Saṃsādhanī*-NN segmenter with controlled lexicon

The above two approaches, in stages, improved the precision as well as recall of the NNE segmenter. But still, the segmenter was not usable by a Naiyāyika. The Naiyāyika always wondered, if a human being can split the compound in a unique way, why does the machine find it difficult? Can we reduce further the 'ambiguities' the machine encounters? Arjuna, one of the authors, who is trained as a Naiyāyika, looked at the multiple splits produced by the segmenter and he identified the impossible splits and provided the reasons for pruning them out. The main reason was that every expression had some word in it which was not found in the lexicon, and hence the expression was split in a wrong way.

This prompted us to build a special morphological analyzer with the vocabulary from the Nyāya texts. The lexicon for the morphological analyzer was built from the high frequency words found in the Nyāya texts. This change resulted in a drastic improvement in the performance.

The performance of this segmenter over the 352 examples from *Pañcalakṣaṇīsarvasvam* are shown in Table 2. These results confirm that

No of solutions	Cases	Percentage
1	340	96.59
2	12	3.41
Total	352	100.00

Table 2

Performance of Saṃsādhanī-NN splitter

the newer algorithm prunes out all irrelevant splits. The recall is 100%, against the recall of 91% of Heritage enhanced splitter for NN and the previous version of Saṃsādhanī segmenter. At the same time the total number of splits is reduced substantially, increasing the precision. Another remarkable point is, in all the examples the correct split was always found at the first place.

4 Constituency parser for NNE

The segmented expression needs further analysis to get the underlying constituency structure. For example, a compound with three components *a-b-c* may be analysed in two different ways viz. <*a*-<*b-c*>> and <<*a-b*>-*c*>. As the number of components increase, the number of possible analyses grows fast, and is represented by a Catalan number (Huet 2006). It is the meaning compatibility (*sāmarthya*), that triggers the correct analysis. A. Kulkarni and Kumar (2011) proposed a statistical constituency parser that uses the statistical properties of a tagged corpus to model the *sāmarthya*. Due to unavailability of the tagged corpus for NN, it was not possible to follow this approach for parsing. The well defined syntax of NNEs discussed in Ganeri (2008) motivated us to look at the constituency parsing of an NNE afresh from a computational point of view.

4.1 Syntax of NN expressions

The NNE involves a small number of technical terms together with a non-logical vocabulary (Matilal 1968). Ganeri (2008) in the informal description of the NN classifies these into 6 categories.

1. **Primitive terms**
   These are the nouns such as *ghaṭa* ‘pot’, *bhūtala* ‘ground’, *gandha* ‘smell’, etc.

2. **Abstract functor**
   This is a derivational suffix ‘tva’ or ‘tā’ (-ness or -hood), that maps a noun to an abstract noun. For example, smell is mapped to smell-ness and pot to pot-ness.

3. **Relational abstract expressions**
   The relational abstract expressions are derived from relation-denoting terms by adding a ‘tva’ or ‘tā’ (-ness or -hood) suffix. For example, *pitṛ* ‘father’ is a relation-denoting term. By adding ‘tva’ suffix, it changes to *pitṛtva* ‘father-hood’, a relational abstract expression. Some other relational abstract expressions are *putratva* ‘son-hood’, *ādheyatā* ‘superstratum-ness’ and *adhikaraṇatā* ‘substratum-ness’.

4. **Conditioning operator**
   The conditioning operator *nirūpita* ‘determined by’ operates on a relational abstract expression to form a term. For example, *X-nirūpita-pitṛtva* ‘father-hood determined by X’.

5. **Sentence-forming Operator**
   The terms such as *niṣṭha* ‘resident in’ and *avacchinna* ‘delimited by’ combine a relational term with another term to form a sentence.

6. **Negation functor**
   abhāvaḥ ‘Negation/absence’.

These 6 categories are necessary to understand both the syntax as well as the semantics of the NNE. In order to represent these compounds as a conceptual graph, we just need to distinguish the concepts from the conceptual relations. Hence we classify these 6 categories into 2 types viz. the conceptual terms and conceptual relations. The *primitive terms* and the *relational abstract expressions* represent the conceptual terms. The *negation functor*, according to the Vaiśeṣika ontology also represents a conceptual term. *Conditioning operators* and *sentence-forming operators* represent the conceptual relations. The *abstract functor* ‘tva’ suffix is a morpheme which denotes a derivational suffix that maps a noun to an abstract noun. We do not represent the *‘tva’* suffix in the CG. But the resulting abstract expression is represented as a conceptual term. In addition to the *abstract functor* – the 'tva' suffix, we also need a derivational suffix 'vat' (possessing) which maps an abstract term to a noun. We represent '*vat*' as a conceptual relation.

4.2 Semi-automatic parsing

Following observations related to the syntax of NNEs were crucial in designing a constituency parser for NNEs.

1. Concepts and relations alternate in an NNE. For example, consider *gandhatva-avacchinna-gandha-niṣṭha-ādheyatā*. Here the components *gandhatva*, *gandha* and *ādheyatā* denote the concepts and the components *avacchinna* and *niṣṭha* denote the relations. In NN, every relation is binary and the two relata are called *anuyogin* and *pratiyogin*. If 'R' is a relation which connects two concepts 'a' and 'b' resulting in an expression 'a-R-b', then the term 'a' is called a *pratiyogin* and the term 'b' is called an *anuyogin*. For example, in the expression *gandha-niṣṭha-ādheyatā* the term *gandha* is the *pratiyogin* and *ādheyatā* is the *anuyogin* of the relation *niṣṭha*. Such a compound thus always will be parsed as $<< a - R > -b >$, and never as $< a- < R - b >>$. Thus this constraint rules out almost half of the possible parses. The NNE 'a-R-b' then is not ambiguous, but the one with five components 'a-R-b-S-c' where 'a', 'b', and 'c' are the concept denoting terms and 'R' and 'S' are the relation denoting terms, is ambiguous. The ambiguity is with respect to the *anuyogin* of 'R' with two possible parses being, $<< a - R > - << b - S > -c >>$ and $<<<< a - R > -b > -S > -c >$. In the first case the *anuyogin* of 'R' is 'c', while in the second, it is 'b'. It is the context that tells us which parse is correct. For example, in *samavāyasambandha-avacchinna-gandha-niṣṭha-ādheyatā*, the *anuyogin* of *avacchinna* is *ādheyatā*, while in *gandhatva-avacchinna-gandha-niṣṭha-ādheyatā*, the *anuyogin* of *avacchinna*, in one reading, can be *gandha*. So, if there are 'n' concept nodes after a relation node 'R', the *anuyogin* of 'R' potentially can be any of these 'n' concept nodes. It is the context that decides which is the correct *anuyogin*.

2. Another cue that rules out some possibilities is the use of co-relative terms in Navya-Nyāya. *Anuyogitā* and *pratiyogitā* are the co-relative

terms, similarly, *ādheyatā* and *adhikaraṇatā* are the co-relative terms. And the relation-terms *nirūpita* and *nirūpaka* always combine two co-relatives.

3. Then there is of course, a well-nested-ness constraint. The resulting constituency structure should be well-bracketed, without any crossings. In other words, if the *anuyogin* of a relation at k^{th} position is at ‘j’, then the *anuyogin* of any relation lying between ‘k’ and ‘j’ can not be beyond ‘j’.

Thus the three conditions viz. a) the *pratiyogin* is always to the immediate left of a relation node, b) *nirūpita* and *nirūpaka* always connect two co-relative terms, and c) the well-nested-ness condition, reduce the search space to a considerable degree.

Since it is not clear what other factors are responsible for the correct choice of the *anuyogin*, we involve a human being well-versed in Navya-Nyāya to mark the correct *anuyogin*s in the cases of ambiguities. We have designed an interface which takes care of the above three conditions, and dynamically reduces the search space with every choice.

For instance, the input *samavāyasambandha-avacchinna-gandhatva-avacchinna-gandha-niṣṭha-ādheyatā-nirūpita-adhikaraṇatāvat-vastu* will be parsed as shown in Figure 1.

When the user selects an *anuyogin* for the first relation *avacchinna*, then the nested parenthesis constraint removes all incompatible solutions reducing the choices of *anuyogins* for all the relations within the range. Figure 2 shows the reduced possibilities after the first choice by the user.

Once all the choices are made by the user, then one gets an unambiguous constituency parse, which may be represented either as a linear bracketed expression or as a 2 dimensional binary tree.

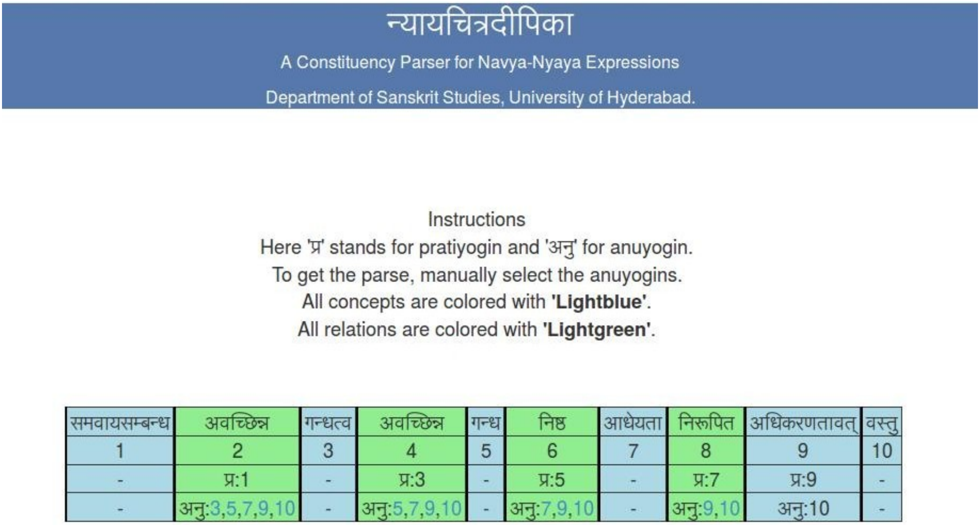
The constituency parse for the expression

gandhatva-avacchinna-gandha-niṣṭha-ādheyatā-nirūpita-adhikaraṇatāvat-vastu

as a linear bracketed expression is

<<<<<samavāyasambandha-avacchinna>-<<gandhatva-avacchinna>-<gandha-niṣṭha>-ādheyatā>>>-nirūpita>-adhikaraṇatā> ^vat-vastu>

and the binary tree representation is shown in Figure 3.



न्यायचित्रदीपिका

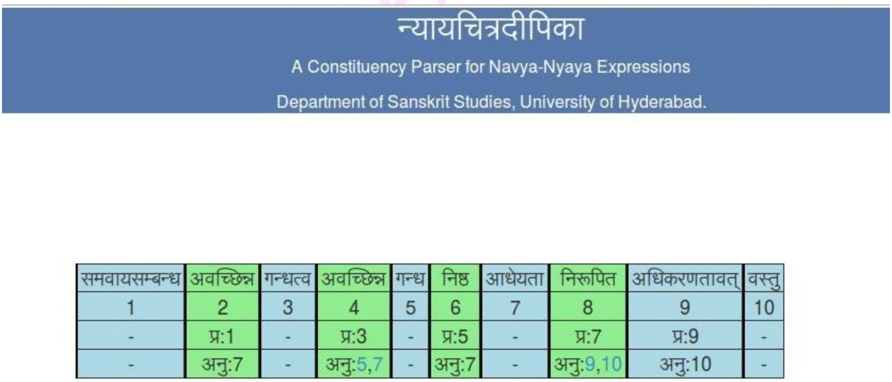
A Constituency Parser for Navya-Nyaya Expressions

Department of Sanskrit Studies, University of Hyderabad.

Instructions
Here 'प्र' stands for pratiyogin and 'अनु' for anuyogin.
To get the parse, manually select the anuyogins.
All concepts are colored with **'Lightblue'**.
All relations are colored with **'Lightgreen'**.

समवायसम्बन्ध	अवच्छिन्न	गन्धत्व	अवच्छिन्न	गन्ध	निष्ठ	आधेयता	निरूपित	अधिकरणतावत्	वस्तु
1	2	3	4	5	6	7	8	9	10
-	प्र:1	-	प्र:3	-	प्र:5	-	प्र:7	प्र:9	-
-	अनु:3,5,7,9,10	-	अनु:5,7,9,10	-	अनु:7,9,10	-	अनु:9,10	अनु:10	-

Figure 1
A screenshot of interface of NN-Parser



न्यायचित्रदीपिका

A Constituency Parser for Navya-Nyaya Expressions

Department of Sanskrit Studies, University of Hyderabad.

समवायसम्बन्ध	अवच्छिन्न	गन्धत्व	अवच्छिन्न	गन्ध	निष्ठ	आधेयता	निरूपित	अधिकरणतावत्	वस्तु
1	2	3	4	5	6	7	8	9	10
-	प्र:1	-	प्र:3	-	प्र:5	-	प्र:7	प्र:9	-
-	अनु:7	-	अनु:5,7	-	अनु:7	-	अनु:9,10	अनु:10	-

Figure 2
A screenshot of interface after user-selection

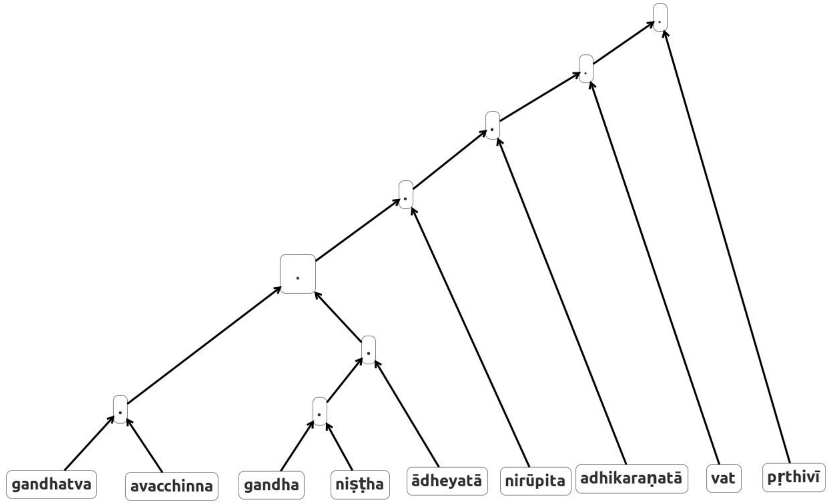


Figure 3
Constituency structure of (1)

Sometimes NNEs do not specify the relation between the concepts explicitly. For example the expression *ghaṭa-abhāva-vat-avṛttitvam* has two concepts *ghaṭa* and *abhāva* as consecutive nodes. In such cases we treat them as a compound with an un-specified relation, and produce a parse: <<<*ghaṭa-abhāva*>-*vat*>-*avṛttitvam*>.

5 Conceptual graphs for NN expression

The binary tree shown in Figure 3 may help in paraphrasing the NNE, but it does not provide any insight into the cognitive structure being described. The diagrammatic representation scheme proposed by A. Kulkarni (1994) maps such parsed structures into a conceptual graph. The conceptual graph of Sowa (1985) was originally designed as a semantic representation for natural language, and hence it is found to be more appropriate graphical representation for representing NNEs. It provides a graphical representation that is readable and at the same time formal for computational purpose. It can represent both the epistemic structure as well as the ontological structure. Further the representation scheme of conceptual graph is so general that various graphical representation methods such as parse trees, Petri nets etc. turn out to be special cases of the conceptual graph (Sowa 2006).

In a CG, the concepts are related through the conceptual relations. The concepts are represented using boxes and the relations using ovals. For instance, "A cat is on a mat" is represented in CG as in Figure 4.

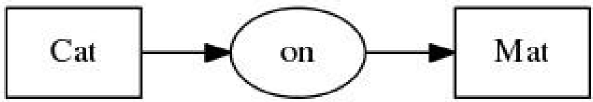


Figure 4
conceptual graph of "A cat is on a mat"

Here 'cat' and 'mat' are the concepts and are represented using boxes and the relation 'on' is represented using an oval. The canonical form in NN is 'X has Y' or 'X is Y-possessing'. Thus the canonical form for 'A cat is on a mat' is 'A mat has a cat (on it)' or 'A mat is cat-possessing'. As an another example, the canonical representation of 'Dasharatha is a father of Rama', in NN is

Dasharatha has father-hood of Rama. (2)

The preposition 'of' in this sentence being ambiguous, this is further disambiguated and expressed in NN as

Sanskrit: *Rāma-niṣṭha-putratva-nirūpita-pitṛtva-vān Daśarathaḥ* (3)
English: Dasharatha has father-hood determined by the son-hood resident in Rama.

The abstract terms father-hood (*pitṛtva*) and son-hood (*putratva*) are denoted by the concept nodes while the terms *niṣṭha, nirūpita, vat* denote the relations and hence are represented by the relation nodes. Figure 5 shows the conceptual graph for the NNE in (3). If we read this CG in Figure 5 along the directions of the arrow, we get the NNE in (3). When a conceptual node has more than one incoming arrow, there are multiple ways of producing the NN expression. To have a one-one correspondence

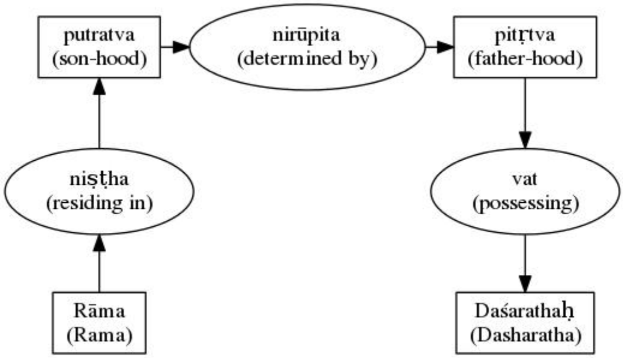


Figure 5
conceptual graph for (3)

between the NNE and the CG, we mark the position of the component in parenthesis. The modified CG for (3) is shown in Figure 6.

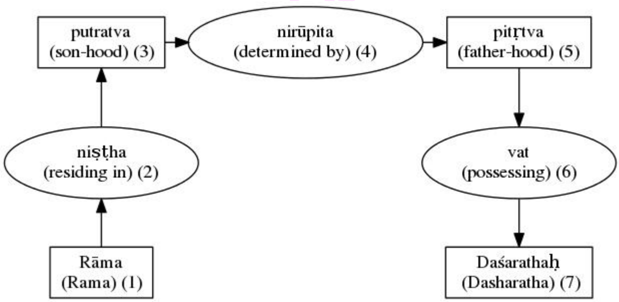


Figure 6
conceptual graph with position information for (3)

Consider now another sentence.

Sanskrit: *Rāmaḥ hastena brāhmaṇāya dhanam dadāti.* (4)

Gloss: Rama{nom.} hand{instr.} Brahmin{dat.} money{acc.} give{pres., active, 3sg}.

English: Rama gives money to a Brahmin with (his) hands.

The verbal cognition of this sentence according to the grammarian's school is

Sanskrit: *rāma-niṣṭha-kartṛtva-nirūpaka-hasta-niṣṭha-karaṇatva-nirūpaka-brāhmaṇa-niṣṭha-sampradānatva-nirūpaka-dhana-niṣṭha-karmatva-nirūpaka-dānakriyā.* (5)
English: An activity of giving characterised by the agent-hood in Rama, the instrument-ness in the hand, the recipient-ness in a Brahmin and the object-hood in money.

Figure 7 shows the rendering of this expression as a conceptual graph. The Nyāya school differs from the grammar school in terms of the chief

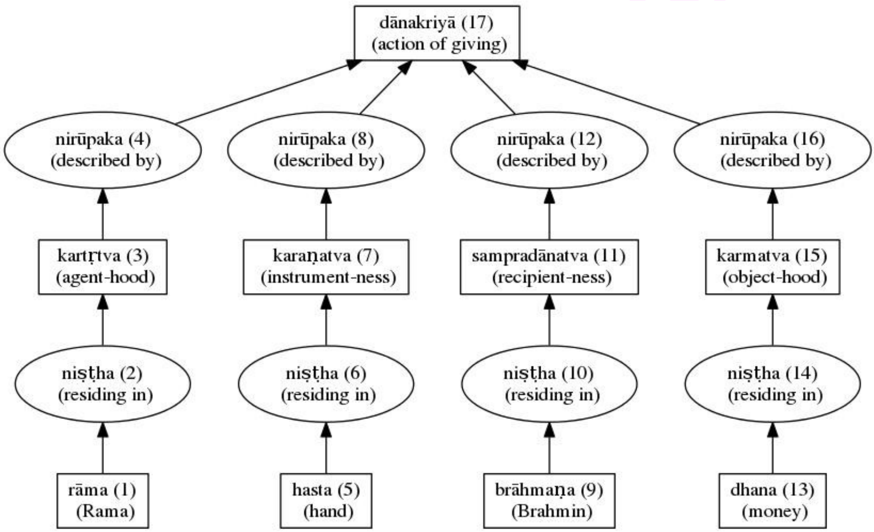


Figure 7
conceptual graph for (5)

qualificand of this cognition. While for a grammarian the activity is the chief qualificand, for a logician the term in nominative case is the chief qualificand. The verbal cognition according to the Nyāya school is

Sanskrit: *hasta-niṣṭha-karaṇatva-nirūpaka-brāhmaṇa-niṣṭha-sampradānatva-nirūpaka-dhana-niṣṭha-karmatva-nirūpaka-dānakriyā-*

anukūla-kṛti-vat-rāmaḥ.
(6)
English: Rama has the agent-hood of an activity of giving described by the instrument-ness in the hand, the recipient-ness in a Brahmin and the object-hood in money.
Figure 8 shows the the rendering of this expression as a conceptual graph.

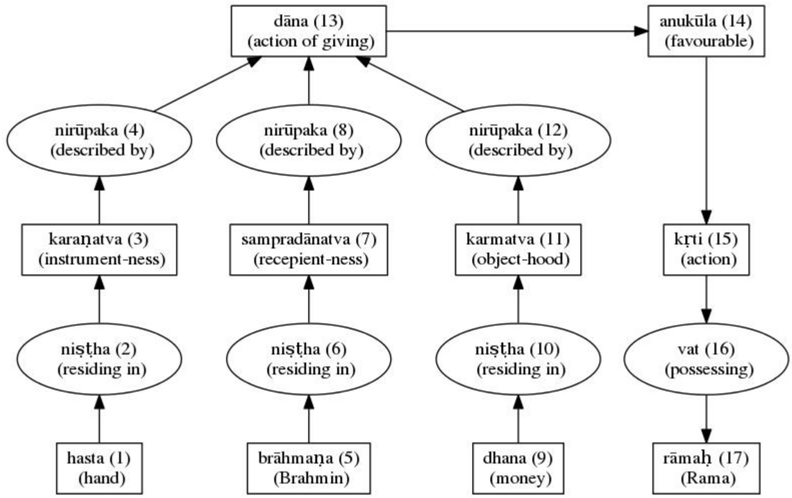


Figure 8
conceptual graph for (6)

Now let us look at the expression (1) where NNE is used to define the *pṛthivī* 'Earth'. 'Earth', according to the Indian school of ontology, is an object which has a characteristic property of having smell which differentiates it from the other objects. This is precisely expressed by (1). In this definition the components *avacchinna* 'delimited by', *niṣṭha* 'resident in', *vat/āśraya* 'possessing', and *nirūpita* 'determined by' denote conceptual relations while the components *gandhatva* 'smell-ness', *gandha* 'smell', *adhikaraṇatā* 'substratum-ness' and *ādheyatā* 'superstratum-ness' denote the concepts. The conceptual graph corresponding to this structure is shown in Figure 9. The dotted lines show the ontological reality viz. that smell-ness is the inherent property of the smell, and that the earth has smell as its characteristic property. The solid lines show the connection between the concepts through the conceptual relations expressed in the NNE (1). NNEs are used to describe the situations or events as well, in addition to

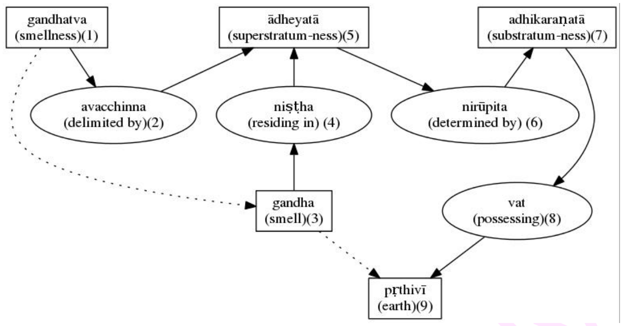


Figure 9
conceptual graph for (1)

the cognitive structures. For example, the fact 'a pot is on the ground' is described as
Sanskrit: *ghaṭa-niṣṭha-ādheyatā-nirūpita-adhikaraṇatā-vat bhūtalam.* (7)
English: The ground which has substratum-ness determined by the superstratum-ness resident in a pot.
where one cognizes the situation with ground as the chief qualificand in the cognition[5] (See Figure 10.). On the other hand if one cognises it with the pot as the chief qualificand[6], then the cognition is described as
Sanskrit: *bhūtala-niṣṭha-adhikaraṇatā-nirūpita-ādheyatā-vān ghaṭaḥ* (8)
English: The pot which has superstratum-ness determined by the substratum-ness resident in the ground.

Figure 11 gives the rendering of this expression as a conceptual graph. Note that the relations *vān* 'possessing' and *niṣṭha* 'resident in' are inverse of each other.

[5] *ghaṭavadbhūtalam* 'pot-possessing-ground'.
[6] bhūtale ghaṭaḥ 'pot on the ground'

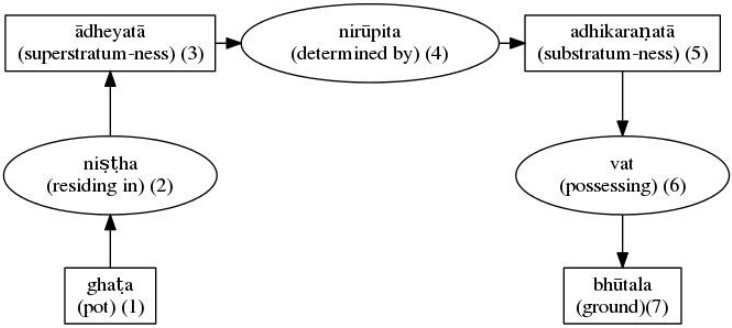


Figure 10

conceptual graph corresponding to (7)

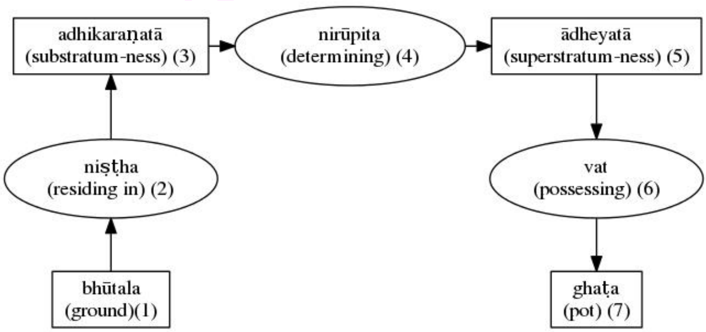


Figure 11

conceptual graph corresponding to (8)

6 Translating NN expressions into conceptual graphs

Now we define the grammar G for an NNE.

Let $G = (N, T, P, NNE)$, where
N: Set of non-terminal symbols
= {Compound_Concept, Compound_Relation, Rel_term, Concept_term},
T: Set of terminal symbols = {relation and concept},
NNE: The start symbol, and
P: Production rules as described in Table 3. The concepts are the nouns,

```
              NNE  :  Compound_Concept
                   ;
 Compound_Concept  :  '<' Compound_Relation '-' Concept_term '>'
                   ;
Compound_Relation  :  '<' Concept_term '-' Rel_term '>'
                   ;
     Concept_term  :  NNE
                   |  concept
                   ;
         Rel_term  :  relation
                   ;
```

Table 3
Production rules

relational abstract expressions, the negation functor and the terms derived with *tva* suffix from nouns. Relations are a) the sentence forming operators *niṣṭha* and *avacchinna*, b) the conditioning operator *nirūpita*, c) along with their inverse relations viz. *vṛtti* (or *āśraya*), *avacchedaka* and *nirūpaka*, respectively. In NN the relations are always binary[7]. Every relation node needs two relata. Thus, in order to draw a CG corresponding to an NN

[7] dviṣṭaḥ sambandhaḥ

relation, we need i) node labels, ii) node types, and iii) the two relata corresponding to the given relation.

We associate with each rule of this grammar a semantics in terms of an attribute grammar which then translates an NNE into a CG. The attribute grammar defining the synthesized attributes is given in Table 4.

NNE	:	Compound_Concept
		$\uparrow$*.head* $=$ $\downarrow$*.head*
	;	
Compound_Concept	:	'<' Compound_Relation '-' Concept_term '>'
		$\uparrow$*.head* $=$ *Concept_term.head*
		establish an edge between the head of the Compound_Relation to the head of the Concept_term
	;	
Compound_Relation	:	'<' Concept_term '-' Rel_term '>'
		$\uparrow$*.head* $=$ *Rel_term.position*
		draw a relation node for Rel_term.
		establish an edge between the head of the Concept_term to the relation node.
	;	
Concept_term	:	NNE
		$\uparrow$*.head* $=$ $\downarrow$*.head*
	|	CONCEPT
		$\uparrow$*.head* $=$ $\downarrow$*.position*
		draw a concept node
	;	
Rel_term	:	RELATION
		$\uparrow$*.head* $=$ $\downarrow$*.head*
	;	

Table 4
Production rules with attributes

The node labels and the node types correspond to the **intrinsic** attributes of the terminal nodes *concept* and *relation*, which are available from the lexer. The two rules in the grammar above corresponding to

Compound_Relation and *Compound_Concept* provide the links between a relation and a concept term.

6.1 An illustration

We work out an example illustrating the working of this grammar. Consider the fragment of the NNE

<<*gandha-niṣṭha*>-*ādheyatā*>

whose constituency parse, following the grammar in Table 4, is shown in Figure 12. Since no significant semantic action is associated with the node labeled as 'Concept_term' and 'Rel_term' and also with the nodes returned by the lexer, we collapse these nodes as in Figure 13 to make the graph more compact.

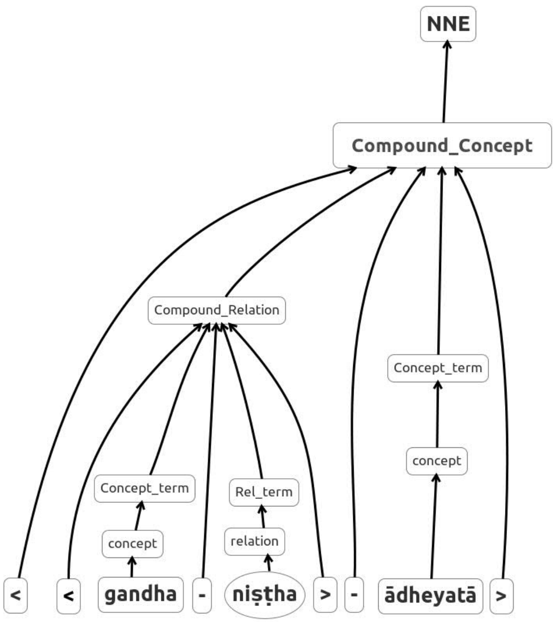


Figure 12
Constituency parse corresponding to the grammar

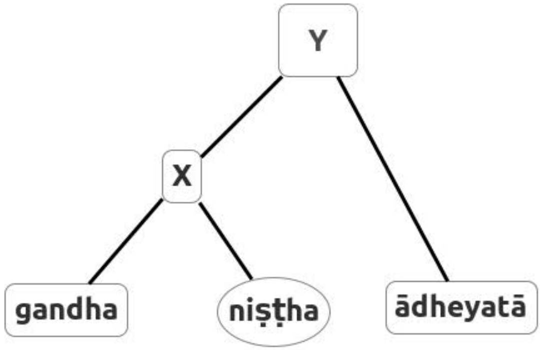


Figure 13
Compact parse

In order to generate a graph from this parse tree, we associate a 'concept-structure' with each concept and a 'relation-structure' with each relation having the relevant attributes. Figure 14 explains this.

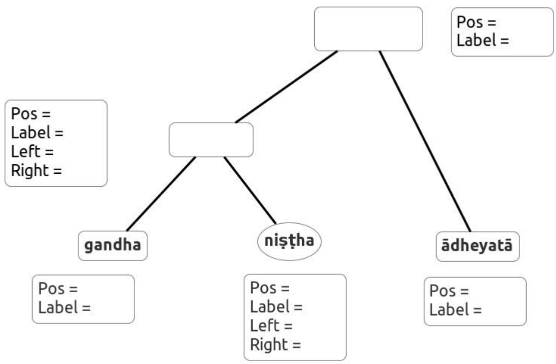


Figure 14
Concept and relation structure

Various stages in parsing are shown in the Figures 15, 16, 17 and 18. Once every relation node gets its right node position filled in, we draw the CG.

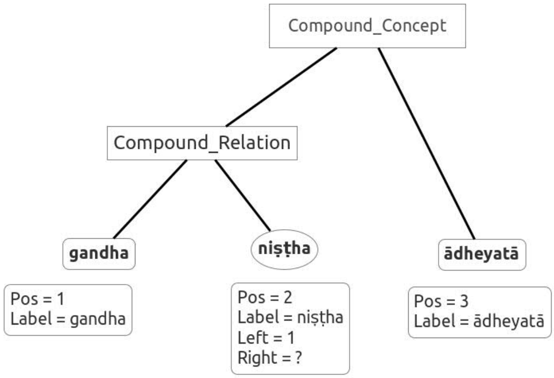


Figure 15
Intrinsic attributes from lexer

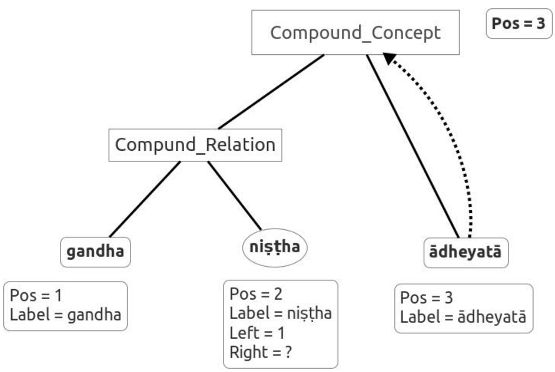


Figure 16
Compound_Concept acquires features from its child

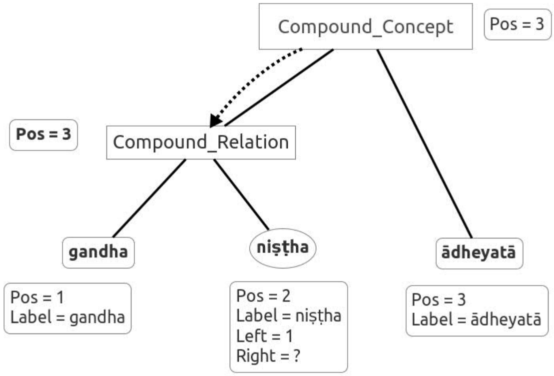


Figure 17

Compound_Relation inherits 'right' from the parent node and acquires other features from its child

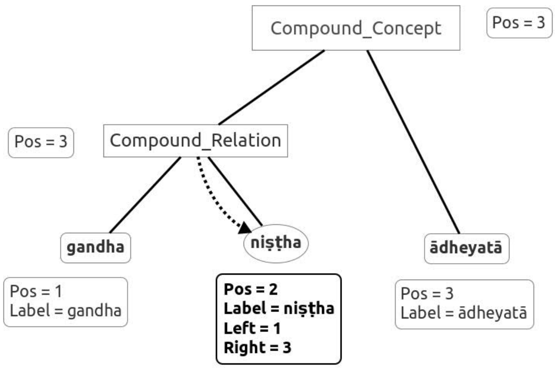


Figure 18

relation inherits the position of 2^{nd} *relata*

6.2 Modified grammar

We noticed that NN can express the cognitive structure as well as describe the physical reality. In these structures, if the compound is not ambiguous, it is not expanded with NN structure. Thus typically the NNEs are heterogeneous mixtures of classical Sanskrit and the NNEs. So there is a need to handle such heterogeneous structures as well. For example, the expression

sādhyābhāvādhikaraṇanirūpitavastu (9)

contains only one NN technical term *nirūpita* and the remaining part of the expression is an ordinary classical Sanskrit compound with 4 components *sādhya*, *abhāva*, *adhikaraṇa* and *vastu*. The grammar in Table 3, is extended further to handle these cases as well. In such cases, we establish a relation between the concept nodes, but leave the relation unspecified, marking it as 'R'. The grammar is modified in the following way to handle this. The

```
             NNE  :  Compound_Concept
                  ;
Compound_Concept  :  '<' Compound_Relation '-' Concept_term '>'
                  |  '<' Concept_term '-' Concept_term '>'
                  ;
Compound_Relation :  '<' Concept_term '-' Rel_term '>'
                  ;
    Concept_term  :  NNE
                  |  CONCEPT
                  ;
        Rel_term  :  RELATION
                  ;
```

Table 5
Modified grammar

graphical representation for (9) following the extended grammar is as shown in Figure 19.

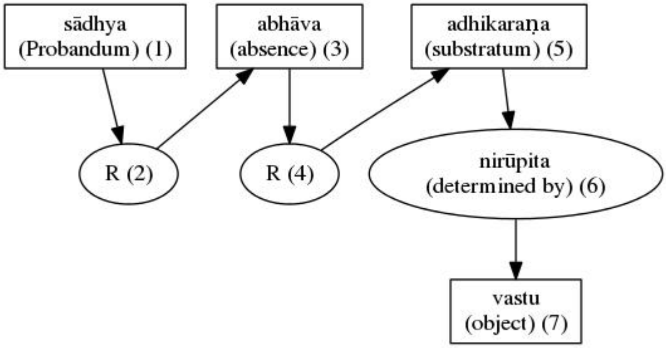


Figure 19
CG generated by modified grammar

7 Nyāyacitradīpikā

Nyāyacitradīpikā[8] combines all the three modules described above and presents a platform for a user to help him understand an NNE. We show below the outputs of these three modules for a sample input.

- **Sample input:**
  sādhyatāvacchedakasambandhāvacchinnasādhyatāvacchedakavastvava cchinnapratiyogitānirūpakābhāvavadadhikaraṇaniṣṭhādhikaraṇatvani rūpitahetutāvacchedakasambandhāvacchinnavṛttitvaniṣṭhapratiyogitā nirūpakābhāvaḥ
- **After segmentation:**
  sādhyatā-avacchedaka-sambandha-avacchinna-sādhyatā-avacchedaka-vastu-avacchinna-pratiyogitā-nirūpaka-abhāva-vat-adhikaraṇa-niṣṭha-adhikaraṇatva-nirūpita-hetutā-avacchedaka-sambandha-avacchinna-vṛttitva-niṣṭha-pratiyogitā-nirūpaka-abhāvaḥ
- **With constituency parse:**
  <<<<<<<<<<<<<<<<sādhyatā-avacchedaka>-sambandha> -avacchinna>-<<<<sādhyatā-avacchedaka>-vastu>-avacchinna>-

[8] http://sanskrit.uohyd.ernet.in/scl/NN/segmenter

pratiyogitā>>-nirūpaka>-abhāva>-vat>-adhikaraṇa>-niṣṭha>-adhikaraṇatva>-nirūpita>-<<<<hetutā-avacchedaka>-sambandha>-avacchinna>-vṛttitva>>-niṣṭha>-pratiyogitā>-nirūpaka>-abhāvaḥ>
(10)

The constituency parse needs user inputs for selection of the relata. The generated CG is shown in the Figure 20.

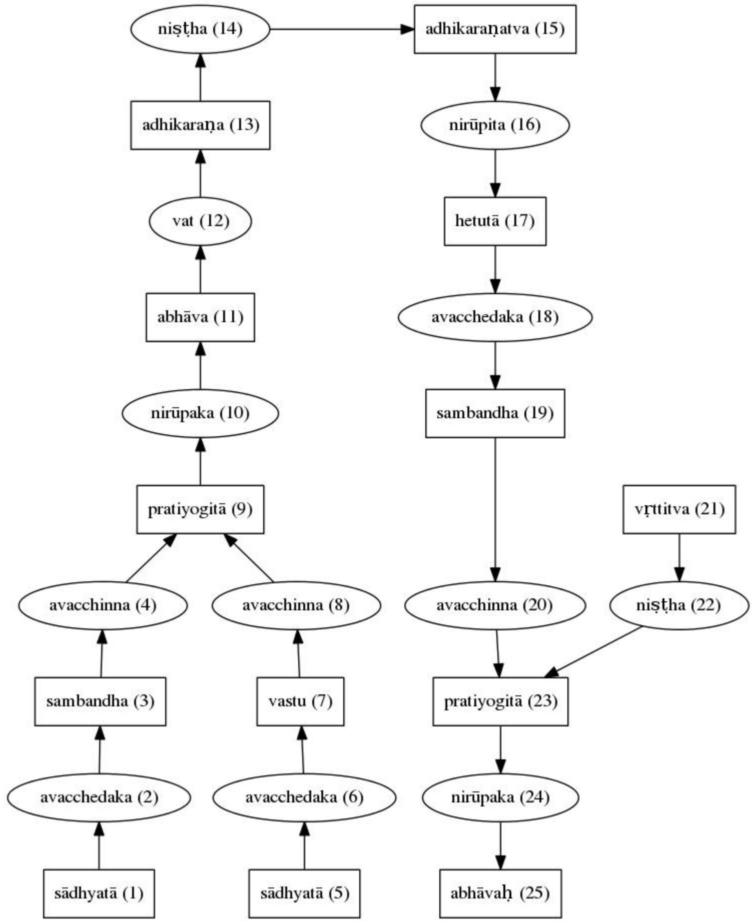


Figure 20

conceptual graph corresponding to (10)

Conclusion

Importance of the technical language of NN, especially in the field of śābdabodha (verbal cognition), has prompted us to take up the study of NNEs. We have succeeded in developing a semi-automatic tool that helps us in understanding the NNEs through CGs. Nyāyacitradīpikā combines all the three modules, viz. the segmenter, constituency parser and the conceptual graph generator into one. Human assistance is needed at the first two stages – first to choose the correct segmentation, if there are more than one, and later to choose the appropriate *anuyogin* of a relation. Use of domain specific dictionary has reduced the human interaction in segmentation to a large extent. In the case of constituency parser, use of NN syntax and well-nestedness of the parse reduces the possible solutions considerably. This semi-automatic conversion of NNEs into a conceptual graph is the first step towards understanding the complex and long NNEs. We hope the rendering into a CG would ease the process of understanding the semantics associated with these expressions.

References

Arjuna, S. R. and Gérard Huet. 2014. “Semi-automatic analysis of Navya-Nyāya compounds.” In: 30th Round table of South Asian Language Analysis: The Recent Developements in Sanskrit Computational Lingustics. (Hyderabad, University of Hyderabad, 6-9 Feb. 2014). Unpublished.

Arjuna, S. R. and Amba Kulkarni. 2014. “Segmentation of Navya-Nyāya Expressions.” In: *Proceedings of International Conference on NLP*.

Bhattacharya, Sibajiban. 1990. “Some features of the Technical Language of Navya-Nyaya.” *Philosophy East and West* 40.2: 129–49.

Dharmakīrtı. 1977. *Nyāyabindu*. Meicho-Fukyu-Kai, Japan.

Dravid, N. S. 1996. *Nyāyakusumāñjaliḥ of Udayana*. ICPR, New Delhi.

Dwivedi, Chandrika Prasad. 2005. *Vyākaraṇabhūṣaṇasāraḥ of Kaunḍabhaṭṭa*. Chowkhambha Sanskrit Prathishthan, New Delhi.

Ganeri, Jonardon. 2008. “Towards a formal regimentation of the Navya-Nyāya technical language-I.” *Logic, Navya-Nyāya & Applications*: 109–24.

Gillon, Brendan S. 2009. “Tagging Classical Sanskrit Compounds.” In: *Sanskrit Computational Linguistics 3*. Ed. by Amba Kulkarni and Gérard Huet. LNAI 5406. Springer-Verlag, pp. 98–105.

Huet, Gérard. 2005. “A Functional Toolkit for Morphological and Phonological Processing, Application to a Sanskrit Tagger.” *J. Functional Programming* 15,4: 573–614.

—. 2006. “Themes and Tasks in Old and Middle Indo-Aryan Linguistics.” In: Motilal Banarsidass, Delhi. Chap. Lexicon-directed Segmentation and Tagging of Sanskrit, pp. 307–325.

Huet, Gérard and Pawan Goyal. 2013. “Design of a lean interface for Sanskrit corpus annotation.” In: *Proceedings of International Conference on NLP*. Ed. by Dipti Mishra Sharma, Rajeev Sanghal, Karunesh Kr.Arora, and B.K.Murthy, pp. 177–86.

Hyman, Malcolm D. 2009. “From Paninian Sandhi to Finite State Calculus.” In: *Sanskrit Computational Linguistics 1 & 2*. Ed. by Amba Kulkarni Gérard Huet and Peter Scharf. LNAI 5402. Springer-Verlag, pp. 253–65.

Ingalls, Daniel H H. 1951. *Materials for the Study of Navya-Nyāya Logic.* Harvard University Press.

Jha, Badrinath and Madanmohan Jha. 1993. *Rasagaṅgādhara of Paṇḍitarāja Jagannatha.* Chaukhambha Vidyabhavan, Varanasi.

Jha, V N. 1987. *Viṣayatāvāda of Harirāma Tarkālaṇkāra.* University of Pune, Pune.

Kulkarni, Amba. 1994. "Navya-Nyāya for Scientists and Technologists: A first step," MTech dissertation. Indian Institute of Technology, Kanpur.

Kulkarni, Amba and Anil Kumar. 2011. "Statistical Constituency Parser for Sanskrit Compounds." In: *Proceedings of International Conference on NLP*. Macmillan Advanced Research Series, Macmillan Publishers India Ltd.

Kulkarni, Tirumala and Jaideep Joshi. 2013. *The language of Logic - Navyanyāya Perspectives.* Manipal University Press, Manipal, India.

Kumar, Anil and Amba Kulkarni. 2013. "Clues from Aṣṭādhyāyī for Compound type." In: *Recent Researches in Sanskrit Computational Linguistics Fifth International Symposium Proceedings.* Ed. by Malhar Kulkarni and Chaitali Dangarikar. D. K Printworld (P) Ltd, New Delhi, pp. 62–83.

Kumar, Anil, Vipul Mittal, and Amba Kulkarni. 2010. "Sanskrit Compound Processor." In: *Proceedings of the 4th International Sanskrit Computational Linguistics Symposium.* Ed. by Girish Nath Jha. LNAI 6465. Springer-Verlag, pp. 57–69.

Matilal, Bimal Krishna. 1968. *The Navya-Nyaya Doctrine of Negation.* Harvard University Press, Cambridge, Massachusetts.

—. 1976. *Nyāyasiddhāntadīpa of Śaśadhara.* Lalbhai Dalpatbhai Institute of Indology.

—. 1977. *Nyāya-Vaiśeṣika.* Harrassowitz.

—. 2005. *Epistemology, Logic and Grammar in Indian Philosophical Analysis Ed. Jonardon Ganeri.* Oxford University Press.

Mittal, Vipul. 2010. "Automatic Sanskrit Segmentizer Using Finite State Transducers." In: *Proceedings of Student Research Workshop.* Association for Computational Linguistics, pp. 85–90.

Mohanty, Jitendra Nath. 2000. *Classical Indian Philosophy.* Rowman & Littlefield Publishers Inc., Oxford.

Natarajan, Abhiram and Eugene Charniak. 2011. "S^3 - Statistical Sandhi Splitting." In: *Proceedings of IJCNLP*, pp. 301–8.

Patil, Devadatta. 2014. *Vidyādharī*. Samarth media center, Pune.

Sanghavi, S. 1939. *Pramāṇamīmāṃsā of Hemacandra*. Ahmedabad.

Sastry, Sriram. 2005. *Pañcalakṣaṇīsarvasvam*. Bharatiya Vidya Sansthan, Varanasi.

Shastri, Chinnaswami. 1991. *Mīmāṃsākoustubhaḥ of Khanḍadeva*. Chowkamba Sanskrit Series Office, Varanasi.

Shaw, J L. 1980. "The Nyāya on cognition and negation." *Journal of Indian Philosophy* 8.4: 279–302.

Shukla, Badarinatha. 1984. *Māthurī Pañcalakṣaṇī*. Rajasthan Hindi Granth Academy, Jaipur.

Sowa, John F. 1985. *Conceptual Structures*. Addison-Wesley.

—. 2006. "Semantic Networks." *Encyclopedia of Cognitive Science*.

Varadacharya. 2007. *Tarkasaṅgraha with Āloka commentary*. Arya Grantha Prakashan, Mysore.

Varakhedi, Shrinivas. 2004. "Knowledge Representation schemes of Navya-Nyāya and other Western systems." PhD thesis. Poornaprajna Samshodhana Mandiram, Bengaluru.

—. 2013. *Knowledge Representation: Navya Nyāya and Conceptual Graphs*. Chinmaya International Foundation Shodha Sansthan, Veliyanad, Kerala.

Wada, Toshihiro. 2007. *The Analytical Method of Navya-Nyāya*. Egbert Forsten, Groningen.