Designing a Constraint Based Parser for Sanskrit

Amba Kulkarni, Sheetal Pokar, and Devanand Shukl

Department of Sanskrit Studies,
University of Hyderabad,
Hyderabad
apksh@uohyd.ernet.in,{sjpokar,dev.shukl}@gmail.com

Abstract. Verbal understanding (śābdabodha) of any utterance requires the knowledge of how words in that utterance are related to each other. Such knowledge is usually available in the form of cognition of grammatical relations. Generative grammars describe how a language codes these relations. Thus the knowledge of what information various grammatical relations convey is available from the generation point of view and not the analysis point of view. In order to develop a parser based on any grammar one should then know precisely the semantic content of the grammatical relations expressed in a language string, the clues for extracting these relations and finally whether these relations are expressed explicitly or implicitly. Based on the design principles that emerge from this knowledge, we model the parser as finding a directed Tree, given a graph with nodes representing the words and edges representing the possible relations between them. Further, we also use the Mīmāṁsā constraint of ākāṅkṣā (expectancy) to rule out non-solutions and sannidhi (proximity) to prioritize the solutions. We have implemented a parser based on these principles and its performance was found to be satisfactory giving us a confidence to extend its functionality to handle the complex sentences.[1]

Key Words: Sanskrit, Constraint Based Parser, Information coding, *ākāṅkṣā*, *sannidhi*.

1 Introduction

Śābdabodha is the understanding that arises from a linguistic utterance. The three schools of Śābdabodha viz. *Vyākaraṇa*, *Nyāya* and *Mīmāmsā* mainly differ in the chief qualificand of the Śābdabodha. Nevertheless to begin with, all these three schools need an analysis of an utterance. This analysis expresses the relations between different meaningful units involved in an utterance. The utterance may be as small as a single word or as big as a complete novel. In what follows, however, we take a sentence[2] as a unit, and as such we discuss

[1] Thanks to Gérard Huet and Peter Scharf for their valuable remarks.

[2] roughly *ekatiṅ vākyam (vārttika on tvāmau dvitīyāyāḥ 8.1.23, halantapuṁlliṅgaprakaraṇam).*

only the relations of words within a sentence and do not deal with the discourse analysis.

A generative grammar of any language provides rules for generation. For analysis, we require a mechanism by which we can use these rules in a reverse way. The reversal in some cases is easy and also deterministic. For example, subtraction is an inverse operation of addition and is deterministic. The reversal may not always be deterministic. Let us see a simple example of non-deterministic reversal with which all of us are familiar. The multiplication tables or simple method of repetitive addition provides a mechanical way for multiplication. Given a product, to find its factors is a reverse process. Multiplication of two numbers, say, 4 and 3 produces a unique number 12. But its decomposition into two factors is not unique. 12 may be decomposed into two factors as either {6,2} or {4,3} in addition to a trivial decomposition {12,1}. Thus the inverse process may at times involve non-determinism. Depending upon the context, if one factor is known, the other factor gets fixed. For example, if you are interested in distributing 12 apples among 2 children, then one of the factors being 2, the other factor, viz. 6, is determined uniquely.

This is true of a generative grammar as well. To give an example, look at the following two sūtras of Pāṇini.

- anabhihite (2.3.1)
- kartṛkaraṇayos tṛtīyā (2.3.18)

These two sūtras together, in case of a passive voice (karmaṇi prayogaḥ), assign third case[3] to both the kartā as well as karaṇam kāraka as in

(1) *rāmeṇa bāṇena vāliḥ hanyate.*

Now, when a hearer (who knows Sanskrit grammar) listens to this utterance, he notices two words ending in the third case suffix and that the construction is in passive voice. But unless he knows that *rāma* is the name of a person and *bāṇa* is used as an instrument, he fails to get the correct reading. In the absence of such an 'extra-linguistic' knowledge, there are two possible interpretations viz. either *rāma* is kartā and *bāṇa* is karaṇam, or *bāṇa* is kartā and *rāma* is karaṇam leading to a non-determinism.[4]

The process of analysing a sequence of words to determine the underlying grammatical structure with respect to a grammar is parsing. There are two distinct ways of developing a parser for a language. The first method which has gained recent predominance is to use statistical machine learning techniques to learn from a manually annotated corpus. This requires a large human annotated

[3] the word 'case' is used for *vibhakti*.

[4] There are two more possibilities, since both have the same gender, number, and vibhakti, one can be an adjective of the other.

corpus. Second method is to use the grammar rules of generation to 'guess' the possible solutions and apply constraints to rule out obvious non-solutions. There have been notable efforts in developing parsers by both the statistical methods as well as grammar based methods for various languages (Lin,1998; Marneffe, 2006; Sleator,1993). A parser based on Pāṇinian Grammar Formalism for modern Indian languages is described in Bharati, et. al. (1995; 85-100). This parser is modeled as a bipartite graph matching problem. A statistical parser for analysing Sanskrit is described in Hellwig (2008). The shallow parser of Huet (2006, 2009) uses bare minimum information of transitivity of a verb as a sub-categorisation frame and models it as a graph-matching algorithm. The main purpose of this shallow parser is to filter out non-sensical interpretations. It is therefore natural for Huet to develop small tools such as 'ca' handler with more priority to rule out non-grammatical solutions (rather than to develop a full-fledged parser)

While designing a grammar based parser, two major design issues[5] one has to address are: a) what should be the level of semantic analysis, and b) which relations to represent in the parsed output. In order to decide on these issues, in what follows, we first look at the Sanskrit grammar to see what kind of semantic relations can be extracted from a language string, precisely where is the information about these relations coded, and whether the extracted relations are from primary sources or secondary. Later we discuss the issues the mechanical processing throws up, and the possible ways to handle them. Based on these observations, we decide various design parameters. The next section discusses mathematical formulation of the problem, its implementation and finally its performance analysis.

2 Encoding of grammatical relations in Sanskrit

Parsing unfolds a linear string of words into a structure which shows explicitly the relations between words. For example, the parse of

(2) *rājā viprāya gāṁ dadāti.*

may be described as in Figure 1.

The task of a parser involves identifying various relations between the words. So the parser developer should decide on the nature of relations and the means to identify the relations. Sanskrit has the unique privilege of having an extant grammar in the form of Aṣṭādhyāyī. It has been demonstrated (Bharati, forthcoming) that Pāṇini had given utmost importance to the information coding and the dynamics of information flow in a language string. In what follows we

[5] The issues in the development of a statistical parser are totally different. They are related to the size of the annotated corpus, the number of annotated tags used, their fine-grained-ness, etc.

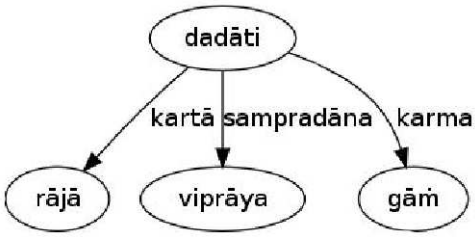

Fig. 1. semantic relations

look at the information coding in Sanskrit from the point of view of designing a parser.

2.1 Semantic content of the relations

Though the correspondence between the semantic relations and the kāraka relations is duly stated in the grammar, what is encoded in words is only the kāraka relations. There is no one-to-one relation between thematic and kāraka relations. One kāraka relation may correspond to more than one thematic relation and one thematic relation may be realised by more than one kāraka relations (Kiparsky, 2009: 49). What can be extracted from a language string alone without using any extra-linguistic information are the syntactico-semantic relations or the kāraka relations and not the pure semantic relations. We give below some examples in our support.

Svatantraḥ kartā The vārttikas under Pāṇini's sūtra *kārake* (1.4.23) go like this[6]

> In the sentence *devadattaḥ pacati*, the activity of cooking refers to the activity of devadattaḥ viz. putting a vessel on the stove, pouring water in it, adding rice, supplying the fuel etc. and this activity refers to the activity of the pradhāna kartā. In the sentence *sthālī pacati*, the cooking activity refers to holding the rice and water till the rice cooks and this activity is that of a vessel. In the sentence *edhāḥ pakṣyanti*, the cooking activity refers to the supply of sufficient heat by a piece of firewood and thus refers to the activity of an instrument.

[6] adhiśrayaṇodakāsecanataṇḍulāvapanaidho'pakarṣaṇakriyāḥ pradhānasya kartuḥ pākaḥ ||(ma. bhā. 1.4.23. vā 8) ||
droṇaṁ pacatyādhakaṁ pacatīti saṁbhavanakriyā dhāraṇakriyā cādhikaraṇasya pākaḥ ||(ma. bhā. 1.4.23.vā 9)||
edhāḥ pakṣyantyā viklitter jvaliṣyantīti jvalanakriyā karaṇasya pākaḥ ||(ma. bhā. 1.4.23.vā 10) ||

In real world, *devadattaḥ*, *sthālī* and *edhāḥ* are the agent, locus and the instrument respectively. But what is expressed by these language strings is just the *kartṛtva* of the pradhāna kartā, adhikaraṇam and karaṇam respectively and NOT the agent, locus and instrument.

śeṣe Similarly the relation between *vṛkṣa* and *śākhā*, *pitṛ* and *putra*, and *rājan* and *puruṣa* in the phrases *vṛkṣasya śākhā*, *pituḥ putra.h* and *rājñaḥ puruṣaḥ* is marked by the genitive case suffix, and Pāṇini groups all of them under the sūtra *śaṣṭhī śeṣe* (2.3.50). Semantically however the first is avayava-avayavī-bhāva (part-whole-relation), the second one is janya-janaka-bhāva (parent-child-relation), and the third one is sva-svāmi-bhāva (owner-possession-relation).

***adhiśīṅsthāsāṁ karma* (1.4.46)** In the sentences

(3) *hariḥ vaikuṇṭham adhiśete.*
(4) *muniḥ śilāpaṭṭam adhitiṣṭhati.*
(5) *sādhuḥ parvatam adhyāste.*

vaikuṇṭha, *śilāpaṭṭa* and *parvata* are in the second case, and Pāṇini assigns them a karma role. However, semantically, all of them are the loci of the activities of the associated verbs viz. adhi-śīṅ, adhi-sthā, and adhi-ās. Hence the naiyāyikas, who want to map the 'world of words' to the real world, find it difficult to accept the karmatva of these words and they qualify this karmatva on the second case ending as *ādhārasya anuśāsanika-karmatva* (Dash, 1991;141). Thus, there is a deviation between the real world and what is expressed through the words.

sahayukte 'apradhāne (2.3.19) In the sentence,

(6) *mātrā saha bālakaḥ āgacchati.*

the agreement of the verb is with *bālakaḥ*, and not with *mātarā*. According to the sūtra (2.3.19), 'saha' is used with the apradhāna (sub-ordinate) kāraka. Thus in this example, *mātā* is sub-ordinate and *bālaka* is the main kartā. However, at another level of semantic analysis, the situation is reversed. It is *mātā* who carries the child in her arms and thus *bālaka* is apradhāna and *mātā* is the pradhāna kāraka. Thus again there is a mismatch between the reality and what sentence actually codes in terms of grammatical relations.

From all the above examples, it is clear that the world of words (śabda-jagat) is different from the real world. To match the extracted relations with the experience of the real world, extra-linguistic information is needed. Since the extra-linguistic information is not easily accessible, and is open ended, we would extract only syntactico-semantic relations that depend solely upon the linguistic

/ grammatical information in a sentence.

2.2 Clues for extracting the relations

Sanskrit being inflectionally rich, we know that suffixes mark the relation between words. Similarly certain indeclinables mark some grammatical relations. Agreement between the words also indicate certain grammatical relations. We discuss below these cases with examples.

1. Abhihitatva
   The Pāṇinian sūtra 'anabhihite' (2.3.1) (if not already expressed) is an important sūtra that governs the vibhakti assignment to the nominals. The vārttika[7] on this sūtra explains abhihita as the one which is expressed either by tiṅ (a finite verbal suffix), kṛt (a non-finite verbal suffix), taddhita (derivational nominal suffix) or samāsa (compound). E.g. in the sentence

   (7) *rāmaḥ vanaṁ gacchati.*

   the verb being in the active voice (kartari prayogaḥ), the verbal suffix 'ti' expresses the kartā, while in the following sentence in passive voice (karmaṇi prayogaḥ)

   (8) *rāmeṇa vanaṁ gamyate.*

   the karma is expressed by the verbal suffix. As such, in both cases, the one which is expressed (kartā and karma respectively) is in the nominative case and shows number and person agreement with the verb form.

   Some of the kṛt suffixes also express the kārakas. For example, in

   (9) *dhāvan aśvaḥ.*

   the kṛt suffix in 'dhāvan' expresses the relation of kartā (kartari kṛt (3.4.68)).

2. Vibhakti
   The verbal as well as nominal suffixes in Sanskrit are termed vibhaktis. We have already seen that verbal suffixes (*tiṅ*), through abhihitatva, mark the relations between words. Now we consider the nominal suffixes. They fall under two categories.
   (a) vibhakti indicating a kāraka relation
   This marks a relation between a noun and a verb known as a kāraka relation. Sanskrit uses seven case suffixes to mark six kāraka

[7] tiṅkṛttaddhitasamāsaiḥ parisaṁkhyānam (ma. bhā. 2.3.1. vā.)

relations viz. *kartā*, *karma*, *karaṇa*m, *sampradāna*m, *apādāna*m and *adhikaraṇa*m. The genitive suffix, in addition to marking a kāraka relation[8], is predominantly used to mark the noun-noun relation. There is no one-to-one mapping between the case suffixes and the kāraka relations, which makes it difficult to determine the relation on the basis of vibhakti alone.

(b) upapada vibhakti
In addition to the noun-noun relations expressed by the sixth case, there are certain words, most of them indeclinables called upapadas, which also mark a special kind of noun-noun relation. These indeclinables, mark a relation of a noun with an another noun, and in turn demand a special case suffix for the preceding noun. For example, the upapada 'saha' demands a third case suffix for the preceding noun as in

(10) *rāmeṇa saha sītā vanaṁ gacchati.*

3. Indeclinables (avyaya)
The indeclinables mark various kinds of relations such as negation, adverbial(manner adverbs only), co-ordination, etc. Sometimes they also provide information about interrogation, emphasis, etc. We distinguish the upapadas from the avyayas, mainly because, though most of the upapadas are also indeclinables, they demand a special case suffix on the preceding word, whereas it is not so with indeclinables.

For example, the relation of 'na' with 'gacchati' in the sentence

(11) *rāmaḥ gṛhaṁ na gacchati.*

is that of 'negation(niṣedha)'. Similarly, the relation of 'mandam' with 'calati' in the sentence

(12) *rāmaḥ mandaṁ calati.*

is that of 'adverbial(kriyāviśeṣaṇa)'. The relation of 'eva' with 'rāma' in the sentence

(13) *rāmaḥ eva tatra upaviṣṭati.*
is that of 'emphasis(avadhāraṇa)'.

4. Samānādhikaraṇa
Agreement in gender, number and case suffix marks *samānādhikaraṇa* (having the same locus), or the modifier-modified relation between two nouns as in

[8] kartṛkarmaṇoḥ kṛti (2.3.65)

(14) *śvetaḥ aśvaḥ dhāvati.*
(15) *aśvaḥ śvetaḥ asti.*

In (14) as well as (15), the words *aśvaḥ* and *śvetaḥ* have the same gender, number and vibhakti indicating samānādhikaraṇa. However, there is a slight difference between the information being conveyed. In (15), the word *śvetaḥ* is a predicative adjective (vidheya viśeṣaṇa), while in (14) it is an attributive adjective.

2.3 Explicit Versus Implicit relations

Relations need not always be encoded directly through suffixes or morphemes. Sometimes the information is coded in the 'Language Convention'. The sūtra

samānakartṛkayoḥ pūrvakāle (3.4.21)

states that the suffix *ktvā* is used to denote the preceding of two actions that share the same kartā. Then the question is what relation does *ktvā* suffix mark? - the relation of kartṛtva or the relation of pūrvakālīnatva? or both?

Bhartṛhari in vākyapadīyam states (3.7.81-82),

pradhānetayoryatra dravyasya kriyayoḥ pṛthak
śaktirguṇāśrayā tatra pradhānamanurudhyate 3.7.81

pradhānaviṣayā śaktiḥ pratyayenābhidhīyate
yadā guṇe tadā tadvad anuktāpi prakāśate. 3.7.82

i.e., in case X is an argument of both the main verb as well as the subordinate verb, it is the main verb which assigns the case and the relation of X to the sub-ordinate verb gets manifested even without any other marking.

From the sentences

(16) *rāmaḥ dugdhaṁ pītvā śālām gacchati.*
(17) *rāmeṇa dugdhaṁ pītvā śālā gamyate.*

it is clear that the vibhakti of *rāma* is governed by the main verb *gam*. And hence, the information that *rāma* is also the *kartā* of the verb *pā* is not expressed through any of the suffixes. The ktvā suffix expresses only the precedence relation (pūrvakālīnatva).

Similarly the sūtra

samānakartṛkeṣu (icchārtheṣu) tumun (3.3.158)

states that in case of verbs expressing desire, the infinitive verb in the subordinate clause will have the same kartā as that of the verb it modifies. Here also the primary information available from the non-finite verbal suffix *tumun* is the relation of purpose.[9]

The sharing in case of *ktvā* and *tumun* suffixes is the result of the pre-conditions *samānakartṛkayoḥ* or *samānakartṛkeṣu* in 3.4.21 and 3.3.158 respectively which act as Language Conventions.

3 Factors useful for Śābdabodha

As mentioned above, the generation problem is a direct problem, and the analysis problem is a reverse problem, and is non-deterministic. This problem was well recognised by the mīmāṁsakas who proposed four conditions viz. ākāṅkṣā (expectancy), yogyatā (mutual compatibility), sannidhi (proximity) and tātparya (intention of the speaker) as necessary conditions for proper verbal cognition. With the help of examples, we explain below, how the first three factors play an important role in the rejection of non-solutions from among the several possibilities. We have not discussed the importance of the fourth factor, since the kind of analysis it involves is out of the scope of the present discussion.

3.1 *Ākāṅkṣā* (Expectancy)

In the sentence,
(18) *rāmaḥ vanaṁ gacchati.*

each of the 3 words in this sentence has multiple morphological analyses.
rāmaḥ = rāma {gender=m, case=1, number=sg},
= rā[10] {lakāra=laṭ, person=1, number=pl, voice=active, parasmaipadī}.

vanaṁ = vana {gender=n, case=1, number=sg},
= vana {gender=n ,case=2, number=sg}.

gacchati = gam {lakāra=laṭ, person=3, number=sg, voice=active, parasmaipadī},
= gacchat (gam śatṛ) {gender=m, case=1, number=sg},
= gacchat (gam śatṛ) {gender=n, case=1, number=sg}.

This may lead to the following two possible sentential analysis:

- *rāma* = kartā of the action indicated by *gam*,
  vana = karma of the action indicated by *gam*.

[9] tumunṇvulau kriyāyām kriyārthyāyām (3.3.10)
[10] *rā* in the sense of *dāne* from the second (*adādi*) gaṇaḥ

– *vana* = karma of the action indicated by *rā*,
gacchati = simultaneity of the actions indicated by *rā* and *gam*,
vayam = kartā of the action indicated by the verb *rā* (not expressed explicitly, but through the verbal suffix).[11]

Of these two analysis, the second analysis can be ruled out on the basis of non-fulfilment of kartā and karma expectancies of the verb *gam*, and the sampradānam expectancy of the verb *rā*. The first analysis being complete in itself, it is preferred over the second one.

3.2 *Yogyatā* (Compatibility)

Consider the sentence,
(19) *śakaṭaṁ vanaṁ gacchati.*
The possible morphological analyses of each of the three words are given below.
śakaṭam = śakaṭa {gender=n, case=1, number=sg},
= śakaṭa {gender=n, case=2, number=sg}.

vanaṁ = vana {gender=n, case=1, number=sg},
= vana {gender=n, case=2, number=sg}.

gacchati = gam {person=3, lakāra=laṭ, number=eka, voice=active, parasmaipadī},
= gacchat (gam+śatṛ) {gender=m, case=1, number=sg},
= gacchat (gam+śatṛ) {gender=n, case=1, number=sg}.

Now, more than one word can't have the same kāraka role unless it is already expressed (abhihita). This leads to the following possible sentential analyses[12]:

– *śakaṭa* = kartā of the action indicated by *gam*,
vana = karma of the action indicated by *gam*.

– *vana* = kartā of the action indicated by *gam*,
śakaṭa = karma of action indicated by *gam*.

– *vana* = kartā of the action indicated by *gam*,
śakaṭa = modifier of *vana*.

– *vana* = karma of the action indicated by *gam*,
śakaṭa = modifier of *vana*.

[11] The sentence is interpreted as - (tasmin) gacchati (sati), vayaṁ vanaṁ rāmaḥ
As (he) goes, let us give the forest (to somebody).
[12] Assuming that the modifier is to the left, which need not be true in case of poetry.

Out of these, the last two do not fulfill all the mandatory expectancies of a verb. Among the first two, the first one is preferable over the second one, since *śakaṭa* has an ability to move while *vana* can not move. Hence *śakaṭa* is preferable as a kartā of the verb *gam* than *vana*. Thus the yogyatā or the competency of the nouns to be eligible candidates for the kāraka roles plays an important role here. However, the context may overrule the condition of yogyatā. It is possible to have a reading where, all the residents of vana are going to see the new śakaṭa, and thus *vana* qualifies to be a kartā. The yogyatā and the context thus compete with each other and hence one needs discourse analysis to prune some of the possibilities.

3.3 *Sannidhi* (Proximity)

Consider,
(20) *rāmaḥ dugdham pītvā śālām gacchati.*
Here the possible analyses are:

- *rāma* = kartā of *gam*,
  dugdha = karma of *pā*,
  śālām = karma of *gam*,
  pā = preceding action with respect to *gam*.

- *rāma* = kartā of *gam*,
  dugdha = karma of *gam*,
  śālām = karma of *pā*,
  pā = preceding action with respect to *gam*.

A competent speaker rules out the second solution on account of non-compatibility of the arguments viz. *dugdha* and *śālā* do not have semantic competence to be the karma of *gam* and *pā* respectively.

The arguments in the correct solution are closer. We mark the words by their positions, and define the proximity measure of a relation as the distance between its two arguments, and the proximity measure of a solution as the sum of the proximity measures of the various relations in the parse. The proximity measure of the above two parses is

- *rāma* = kartā of *gam*
  (dist = position of *gam* - position of *rāma* = 5 -1 = 4)
  dugdha = karma of *pā* (dist = 3-2 = 1)
  śālām = karma of *gam* (dist = 5-4 = 1)
  pā = preceding action with respect to *gam* (dist = 5-3 = 2)
  Thus the total distance = 4 + 1 + 1 + 2 = 8

– *rāma* = kartā of *gam* (dist = 5-1 = 4)
dugdha = karma of *gam* (dist = 5-2 = 3)
śālām = karma of *pā* (dist = 4-3 = 1)
pā = preceding action with respect to *gam* (dist = 5-3 = 2)
Thus the total distance = 4 + 3 + 1 + 2 = 10

The one with greater proximity (or smaller distance) is preferred as a solution. Though Sanskrit is a free-word-order language, the following sentence with exchange of the karmas is not acceptable.

(21) **rāmaḥ śālām pītvā dugdhaṁ gacchati.*

Equally unacceptable prose orders are

(22) **rāmaḥ pītvā śālām dugdhaṁ gacchati.*
(23) **rāmaḥ dugdham śālām pītvā gacchati.*

which involve crossing of links expressing the relations. A small pilot study of anvaya of Saṁkṣepa Rāmāyaṇa (Kutumbashastri, 2002) sentences show no evidence of crossing of links.

It is worth exploring the Calder mobile model suggested by Staal (1967) and further worked out by Gillon (1993) in the light of the mīmāṁsā principle of sannidhi. It may result in a better computational criterion for sannidhi.

4 Design Principles

The foregoing discussions lead to the following design principles for the constraint-based parser.

1. The relations will be marked as kāraka relations.
   [Using these kāraka relations and extra-linguistic knowledge, the semantic analysis may be carried out in the next level of processing.]
2. Only those relations that are marked directly by the morphemes will be extracted.
   [No relations that require some post-processing, or are based on secondary information will be extracted in the first step. The next level of processing will use this information to mark the unspecified or shared relations, if any.]
3. To prioritize the solutions, only the conditions of ākāṅkṣā and sannidhi will be used.
   [The condition of yogyatā will be used as and when the information is available in machine usable form, with the understanding that this knowledge may not be relied on completely.]
4. While dealing with prose, it will be assumed that there is no cross-linking of the relations between the words.

5 Mathematical Model

Let each word in a sentence be represented as a node in a graph, and the nodes be connected by the directed labelled edges. Then the problem of parsing a sentence may be modelled as

> Given a Graph G with n nodes, the task is to find a sub-graph T which is a directed Tree.[13]

Assuming that the words can be partitioned into two classes viz. the words which have an expectancy called demand words and the words which satisfy the demand called source words, Bharati et. al. reduced the parsing problem to matching a bipartite graph (Bharati,1995; 96). But in reality, the words can not be partitioned into two classes. We come across words which can be demand words in some context and source words in some other context, or in the same context a kṛdanta (primary derivative), e.g. can be both a demand word as well as a source word. Bharati et. al. (1995; 91) also needed the requirement of kārakas and their optionality for each verb. But then, a parser based on such information will fail to parse sentences with ellipsis, or the real corpus where we come across sentences with incomplete information.

With a robust parser, that produces at least partial solution in case of ellipsis, as an aim, we relax the above conditions. So we give away the constraint that a word can be exclusively either a demand or a source word. Further we treat all kārakas at the same level, irrespective of whether they are mandatory or optional, and assign penalty to lower the priority of those solutions which do not satisfy the mandatory expectancies.

We divide the problem into three parts:

1. For a given sentence, draw all possible labeled directed edges among the nodes.
2. Identify a sub-graph T of G such that T is a directed Tree which satisfies the given constriants.
3. Prioritize the solutions, in case there is more than one possible directed Tree.

In what follows we describe our model.

A matrix is a convenient way of representing the graphs for computing purpose. In our case, each word represents a node of a graph, and with each pair of nodes is associated zero or more labels, indicating the possible relations between these nodes. The strong constraint on these relations is that there can be at the most one label associated with a pair of nodes. This then naturally suggests a 3D matrix representation, whose elements are either 0 or 1, where the 3 dimensions represent two nodes and a relation label. Further, each word has one

[13] A tree is a graph in which any two vertices are connected by exactly one simple path.

or more morphological analyses. Hence, corresponding to each node there exists a record with one or more cells, each cell representing one morphological analysis of the word. Let the j^{th} analysis of the i^{th} node be represented by $[i, j]$. Thus the address of a typical element of the 3D matrix is $([i, j], R, [l, m])$. The first pair of letters i and j correspond to the source word analysis, while the second pair of letters l and m represent the demand word analysis. R is the name of the relation of the l^{th} word to the i^{th} word. j indicates the morphological analysis of the i^{th} word responsible for this relation, and m indicates the morphological analysis of the l^{th} word that triggers this relation. In short the tuple $([i, j], R, [l, m])$ represents a relation R due to the m^{th} morphological analysis of the l^{th} word to i^{th} word due to its j^{th} morphological analysis. For ease of representation, we represent the tuple as (i, j, R, l, m). Thus, the initial graph with all possible relations between various nodes is represented as 5D matrix C such that
$C[i, j, R, l, m] = 1$, if such a relation exists,
$= 0$, otherwise.

Task 1: Based on the available information in a given sentence in the form of abhihitatva, vibhakti, sāmānādhikaraṇya, and the expectancies the matrix C is populated with 0s and 1s.

Here are sample rules (just enough to illustrate an example), expressed in English.

Rule 1:
If the sentence has
a noun(say 's') in prathamā vibhkati,
a verb(say 't') in kartari prayogaḥ, in 3rd person, and
's' and 't' agree in number,
then 's' is possibly a kartā of 't'.

Rule 2:
If the sentence has
a noun(say 's') in dvitīyā vibhkati,
a verb(say 't') in kartari prayogaḥ, and is sakarmaka (roughly transitive)
then 's' is possibly a karma of 't'.

Rule 3:
If the sentence has
a noun(say 's') in saptamī vibhkati, and
a verb(say 't'),
then 's' is possibly an adhikaraṇa of 't'.

Now consider the sentence
(24) *rāmaḥ vanaṁ gacchati.*

The analyses of various words are numbered as follows:
[1, 1]: rāma {gender=m, case=1, number=sg},
[1, 2]: rā {gaṇaḥ=*adādi*, lakāra=laṭ, person=1, number=pl, prayogaḥ=kartari, parasmaipadī}.

[2, 1]: vana {gender=n, case=1, number=sg},
[2, 2]: vana {gender=n ,case=2, number=sg}.

[3, 1]: gam {lakāra=laṭ, person=3, number=sg, voice=active, parasmaipadī},
[3, 2]: gacchat (gam śatṛ) {gender=m, case=1, number=sg},
[3, 3]: gacchat (gam śatṛ) {gender=n, case=1, number=sg}.

The above 3 rules with this input then produce the following output showing all possible relations between various analses:
[2, 2] is a possible karma of [3, 2]
[2, 2] is a possible karma of [3, 3]
[2, 2] is a possible karma of [1, 2]
[2, 2] is a possible karma of [3, 1]
[2, 1] is a possible kartā of [3, 1]
[1, 1] is a possible kartā of [3, 1]

The resulting graph is shown in Figure 2.

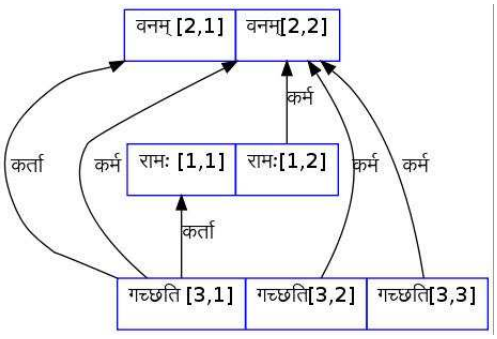

Fig. 2. Graph showing all possible relations

Task 2: In order to get a Tree from this graph, we impose the following constraints.

1. A morpheme(vibhakti) marks only one relation.
   I.e., a node can have one and only one incoming arrow.

$\sum_{j,R,k,l} C[i,j,R,k,l] = 1, \forall i.$

2. Each kāraka relation is marked by a single morpheme.
   There can not be more than one outgoing arrow with the same label from the same cell, if the relation marks a kāraka relation,[14] i.e. there can not be two words satisfying the same kāraka role of the same verb.
   $\sum_{i,j} C[i,j,R,k,l] = 1$, for each tuple (R,k,l).

3. A morpheme does not mark a relation to itself.
   A word can't satisfy its own expectancy. i.e. a word can't be linked to itself[15]. Or there can not be self loops in a graph.
   $\sum_{j,R,k} C[i,j,R,i,k] = 0, \forall i.$

4. Only one valid analysis of every word per solution
   (a) If a word has both an incoming arrow as well as an outgoing arrow, they should be through the same cell.
   $\forall i \forall j \sum_{R,l,n} C[i,j,R,l,n] + \sum_{a,b,R,k!=j} C[a,b,R,i,k] \leq 1.$

   (b) If there is more than one outgoing arrow through a node, then it should be through the same cell.
   if, for some i,j,R,l,m C[i,j,R,l,m] = 1,
   then $\forall a \forall b \forall R \sum_{a,b,R,k!=j} C[a,b,R,l,k] = 0.$
5. All the words in a sentence should be connected.[16]

6. There are no crossing of links
   If all the nodes are plotted in a straight line, then they should not intersect each other. i.e.,
   if $C[i,j,R,k,l] = 1$, then
   $\forall v \forall y C[u,v,w,x,y] = 0$, if i < x < k and u < i or u > k.

The resultant graph is a Tree provided:

1. It is connected[17].
2. It has n-1 edges.
   The fact that only sup / tiṅ suffix in every word marks a relation with some other word in a sentence, and abhihita kāraka is not expressed by any sup suffix, it is guaranteed that there are exactly n-1 edges.

---

[14] adhikaraṇam is treated as an exception since one can have more than one adhikaraṇam as in
rāmaḥ adya pañca vādane gṛham agacchat.

[15] in case of some of the taddhita suffixes which are in svārtha, there will be self loops. But we do not consider the meaning of taddhita suffixes in the first step, and thus can avoid the self loops

[16] This condition is not yet implemented.

[17] Since, this condition is not yet implemented, the resulting graph need not be a Tree.

Task 3: The solutions are prioritized using the conditions specified below.
For each of the solutions, the cost is calculated as
Cost = $\sum_{i,R,j} c_{iRj}$, where
i) $c_{iRj} = |j - i| * wt_R$, if $C[i, a, R, j, b] = 1$ for some a and b.
= 0 otherwise.
ii) $wt_R = rank(R)$, if R is a kāraka relation (appendix I shows the ranking)
= 100, otherwise.
This cost ensures the following:

1. ākāṅkṣā (kāraka relation) is preferred over other relations (rank[18] of the relations takes care of this.).
2. The ranking of the solutions on the basis of distance-based weights takes care of sannidhiḥ.

6 Implementation

The first task demands the inputs from grammar, whereas the second and the third tasks are purely mathematical ones, which can be handled by a constraint solver. The separation of tasks into three sub-tasks makes it not only modular, but also easy for a grammarian to test his/her rules independently. For the first task, an expert shell CLIPS is being used, whereas for the second task, a constraint solver MINION is being used. The system is available at

`http://sanskrit.uohyd.ernet.in/~anusaaraka/sanskrit/MT/test_skt.html`

There is no specific reason behind using these special software tools except the familiarity and the availability under the General Public License. No special efforts were put in towards the efficiency of the system since the main purpose of this exercise is to have a proof of the concept.

7 Performance

The current system allows only *padaccheda-sahita-eka-tiṅ-gadya-vākyam*. To measure the performance of this parser, we used hand tagged data. Around 110 sentences with single finite verb were selected from a school book (see appendix A for a sample). These sentences were tagged manually showing the relation of each word in the context. The sentences being simple, each sentence had a single possible parse in the context. There were 525 token words. The average length of the sentences was approximately 5, with a maximum length of 14 words. Morphological analyser is a pre-requisite for a parser. In order to avoid the cascading effect of errors due to non-availability of the morphological analysis, before running the parser, we ensured that the correct morphological analysis of all the words is being produced. Thus, given all possible correct analyses of the words, the task of the parser was to come up with a correct

[18] Better ranking scheme needs to be developed to take care of default word order.

parse. Though the parser produces multiple parses, for the evaluation purpose, we chose only the first parse. Among the 113 sentences, 97 (86%) sentences had the first parse correct and 16 (14%) sentences had one relation wrong. Out of these 16, 10 relations had wrong label, 3 had wrong attachments and 3 went wrong in both the label as well as attachments.

The analysis of wrong results showed that most of the wrong relations were due to non-availability of appropriate knowledge to make the fine-grained distinction. For example, manually tagged corpus makes a distinction between kāla-adhikaraṇa and deśa-adhikaraṇa, gauṇa and mukhya karma in case of dvi-karmaka (di-transitive) verbs, hetu and karaṇam, to name a few. Another cause of ambiguity was the verbs in the curādi (10^{th}) gaṇa. For most of the verbs in this class, the causative and non-causative forms are the same. This then leads to a wrong parse, since we also allow elipsis. In case there are n (> 1) adjectives, there can be more than one possible way these adjectives can group with the following noun. But we produce a single parse where the adjectives are linked as a chain with the rightmost adjective qualifying the noun directly. This chain just indicates a chunk, and the internal grouping of these adjectives and also their relation with the head noun is left to the user for interpretation.

A sample output of a sentence

bālyakāle rāmaḥ daśarathasya ājñayā viśvāmitrasya yajñam rākṣasebhyaḥ rakṣitum vanam agacchat.

is produced in Figure 3.

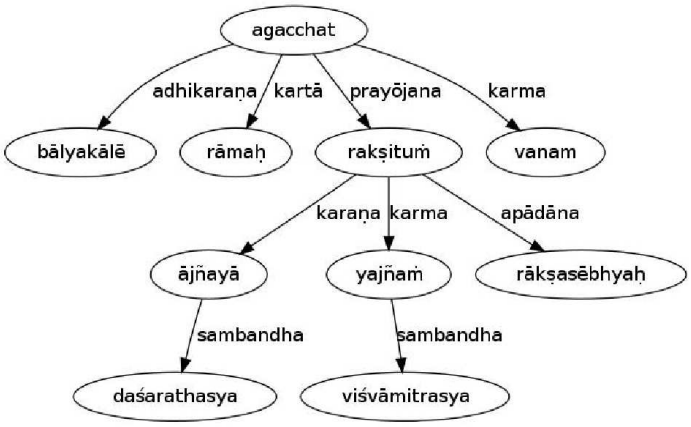

Fig. 3. Sample parse output

8 Challenges

The result with limited test cases is encouraging. The real corpus, even with small children's stories involves much more complex constructions, not necessarily confining to 'eka tiṅ vākyam'. The constructions involve co-ordination between two or more verbs, sentence connectives such as 'yadā-tadā, yathā-tathā, atha, tasmāt', etc. Thus, even at the level of simple texts, one can not do away with discourse analysis.

Another important problem that needs to be addressed is to handle a little more semantics than can be handled with syntactico-semantic relations. For example, it would be desirable to distinguish between hetu and karaṇa at least, though not between mukhya karma and gauṇa karma.

Third problem is regarding the upapadas. Upapada acts more like a function word (dyotaka) than a content word (vācaka). So in case of upapadas, it would be desirable to group the upapada together with the content word in the vibhakti it demands and then mark its relation with other content word. Thus e.g. in the sentence *rāmaḥ muninā saha vanaṁ agacchat*, it is desirable to parse it as in figure 4 than as in figure 5. This means a upapada should be treated as a function word, and as such should not be represented by a node.

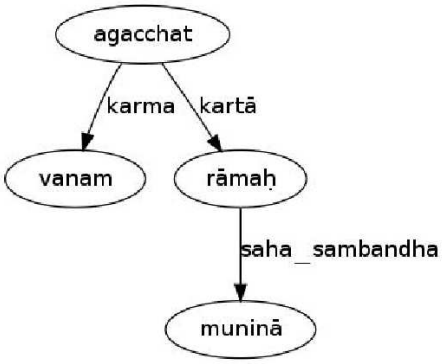

Fig. 4. saha-function

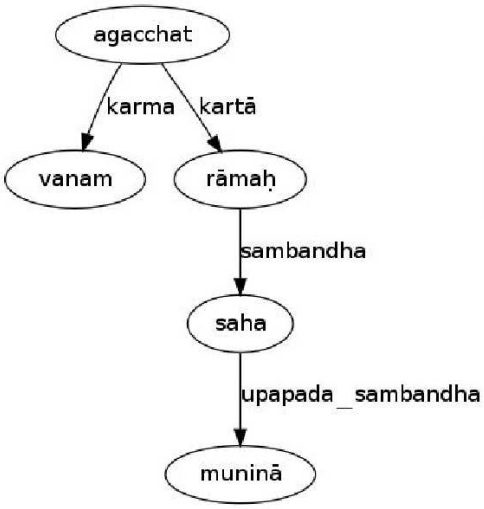

Fig. 5. saha-content

The vibhaktis, as we know, denote more than one meaning. For example, the second case suffix denotes the meaning of *kriyāviśeṣaṇa* (manner), *kāla* (time)

or *adhvan* (path) in addition to the *karma*. To decide an appropriate role, now what one requires is the knowledge of yogyatā. In other words, our e-dictionaries should be rich with semantic properties of the words such as whether it denotes time, path or the manner, etc.

Since the parser does the analysis 'mechanically', it detects the problems of 'violation' of the rules more easily. We give just one example (more examples can be found in Gillon, 2002) from the anvaya of 'Saṁkṣepa Rāmāyaṇam'.

> guhena lakṣmaṇena sītayā ca sahitaḥ rāmaḥ vanena vanaṁ gatvā bahūdakāḥ nadīḥ tīrtvā bharadvājasya śāsanāt citrakūṭam anuprāpya vane ramyam āvasathaṁ kṛtvā devagandharvasaṅkāśāḥ te trayaḥ ramamāṇāḥ sukhaṁ nyavasan. (Śloka 30-32)

This sentence poses the following problems:
a) Whom does the phrase 'te trayaḥ' refer to?
b) *rāmaḥ* does not agree with the finite verb *nyavasan*. Is it not a violation of *samānakartṛkayoḥ pūrvakāle*?
c) Does *gatvā* precede *tīrtvā* or *nyavasan*?
d) In case of *vanena vanaṁ* what should be the meaning of the third case?

In spite of these problems, this parser can act as a tool to discover various kinds of semantic knowledge necessary to build a semantic parser.

9 Acknowledgement

This work is a part of the Sanskrit Consortium project entitled 'Development of Sanskrit computational tools and Sanskrit-Hindi Machine Translation system' sponsored by the Government of India.

References

1. Bharati, Akshar and Sangal, Rajeev, *A Karaka Based Approach to Parsing of Indian Languages*, In *COLING90: Proc. of Int. Conf. on Computational Linguistics (Vo l. 3), Helsinki*, Association for Computational Linguistics, NY, August 1990.
2. Bharati, Akshar, Chaitanya, Vineet and Sangal, Rajeev, *NLP A Paninian Perspective*, Prentice Hall of India, Delhi, 1994.
3. Cardona George, *Pāṇini and Pāṇinīyas on śeṣa Relations,* Kunjunni Raja Academy of Indological Research, Kochi, 2007.
4. Dash Achyutanand, *The syntactic role of adhi in the Pāninian Kāraka system* in Pāṇinian Studies Prof. S. D. Joshi Felicitation volume, ed. Deshpande Madhav M, and Bhate Saroja, Center for South and Southeast Asian Studies, University of Michigan, U.S.A., 1991.
5. Gent, Ian P., Jefferson, Chris and Miguel, Ian. *MINION: A Fast, Scalable, Constraint Solver,* The European Conference on Artificial Intelligence 2006 (ECAI 06).

6. Gillon Brendan S. *Word Order in Classical Sanskrit* Indian Linguistics, v.57, n.1, pp. 1-35, 1996.
7. Gillon Brendan S. *Bhartṛhari's rule for unexpressed kārakas: The problem of control in Classical Sanskrit* Indian Linguistic Studies, Festschrift in Honor of George Cardona, Ed. Deshpande, Hook, Motilal Banarasidass, Delhi, 2002.
8. Hellwig, Oliver, *Extracting Dependency Trees from the Sanskrit Texts* Proceedings of the Sanskrit Computational Linguistics Symposium, Ed. Kulkarni and Huet, LNAI 5406, Springer Verlag, 2009.
9. Huet, Gérard, Formal Structure of Sanskrit Text: Requirements Analysis for a Mechanical Sanskrit Processor Proceedings of the Sanskrit Computational Linguistics Symposium, Ed. Huet, Kulkarni and Sharf, LNAI 5402, Springer Verlag, 2009.
10. Huet, Gérard, Shallow syntax analysis in Sanskrit guided by semantic nets constraints Proceedings of International Workshop on Research Issues in Digital Libraries, Ed. Majumder, Mitra and Parui, ACM Digital Library, Dec 2006.
11. Jigyasu, Brahmadatt, 1979. *Ashtadhyayi (Bhashya) Prathamavrtti, three volumes*, Ramlal Kapoor Trust Bahalgadh, (Sonepat, Haryana, In dia) (In Hindi)
12. Joshi, S.D. (editor) 1968. *Patanjali's Vyakarana Mahabhashya*, (several volumes), Univ. of Poona, Pune, India.
13. Joshi, S.D. and Roodebergen J.A.F., 1998. *The Aśṭādhyāyīof Pāṇini* (several volumes), Sahitya Akademi, Delhi, India.
14. Kiparsky, P. *On the Architecture of Panini's Grammar*, Proceedings of the Sanskrit Computational Linguistics Symposium, Ed. Huet, Kulkarni and Scharf, LNAI 5402, Springer Verlag, 2009.
15. Kutumbashastri, V. *Saṁkṣepa Ramāyaṇam*, Teach Yourself Samskrit series, ed., Rashtriya Sanskrit Samsthanam, New Delhi, 2002.
16. Lin D. *Dependency-based evaluation of MINIPAR.* In Workshop on the evaluation of Parsing Systems, Granada, Spain, 1998.
17. Marneffe M., MacCartney B. and Manning C. D. *Generating Typed Dependency Parses from Phrase Structure Parses* The fifth international conference on Language Resources and Evaluation, L REC 2006, Italy.
18. Pande, Gopal Dutt *Vaiyākaraṇa Siddhāntakaumudī of Bhattojidikshita* (Text only), Reprint Edition. Varanasi: Chowkhamba Vidyabhavan, 2000.
19. Ramakrishnamacharyulu, K.V. *Annotating Sanskrit Texts based on Śābdabodha systems,* Proceedings of the Sanskrit Computational Linguistics Symposium, Ed. Kulkarni and Huet, LNAI 5406, Springer Verlag, 2009.
20. Ramanujatatacharya, N.S. *Śābdabodha Mīmāṁsā* Institute Francis De Pondicherry, 2005.
21. Sharma, Raghunath *Vākyapadīyam, Part III* With commentary Prakāśa by Helaraja and Ambakartri Varanaseya Sanskrit Visvavidyalaya, Varanasi, 1974.
22. SK: Siddhāntakaumudī See Pande
23. Staal, J.F. *Word Order in Sanskrit and Universal Grammar* Reidal, Dordercht (Foundations of Language, Supplementary series: v.5), 1967.
24. Sleator D. D., Temperley D. *Parsing English with a link grammar* In third international Workshop on Parsing Technologies, 1993.

A Sample story

nadyāḥ taṭe ekaḥ vṛkṣaḥ asti| vṛkṣasya samīpam ekā śilā asti| vṛkṣasya śākāsu nīḍāḥ santi| nīḍeṣu vihagāḥ vasanti| nīḍāḥ vihagān rakṣanti| vṛkṣasya

adhaḥ vānarāḥ santi| kapayaḥ gṛham na racayanti| te sarvadā itastataḥ bhramanti| ekasmin divase śītam tān pīḍayati| te śītāt trāṇāya agnim icchanti| kutrāpi te agnim na vindanti| ekaḥ guñjāyāḥ phalāni paśyati| guñjāyāḥ phalāni raktāni santi| saḥ agneḥ sadṛśāni guñjāyāḥ phalāni ānayati| tāni guñjāyāḥ phalāni śilāyām saṁharati| te sarve guñjā-phalam paritaḥ upaviśanti| agneḥ icchayā te mukhaiḥ tāni dhamanti| te agnim na vindanti| te vānarāḥ analāya vṛthā āyāsam kurvanti| teṣām śītam na naśyati| kapayaḥ mūrkāḥ santi|

B Relations

The relations used, along with their ranks are given in Table 1.

(0)	upapada vibhakti	(12)	kāla-adhikaraṇaṁ
(1)	kartā	(13)	viṣaya-adhikaraṇaṁ
(2)	prayojaka kartā	(14)	kartā-samānādhikaraṇaṁ
(3)	prayojya kartā	(15)	viśeṣaṇaṁ
(4)	karma	(16)	kriyā-viśeṣaṇaṁ
(5)	reserverd for gauṇakarma	(17)	tādarthya
(6)	reserverd for mukhyakarma	(18)	pūrvakālīna
(7)	karaṇaṁ	(19)	sambandha
(8)	sampradānaṁ	(20)	kāraka_ṣaṣṭhī
(9)	apādānaṁ	(21)	niṣedha
(10)	adhikaraṇaṁ	(22)	sambodhana
(11)	deśa-adhikaraṇaṁ		

Table 1. Relations