LOCAL WORD GROUPING AND ITS RELEVANCE TO INDIAN LANGUAGES

Akshar Bharati
Vineet Chaitanya
Rajeev Sangal
Dept. of Computer Science and Engineering
Indian Institute of Technology Kanpur
Kanpur 208 016

Abstract

Indian Languages have relatively free word order; still there are units which occur in fixed order. The most important examples of these are the main verb followed by auxiliary verb sequences and nouns followed by postpositions. We term such units as verb groups and noun groups respectively.

An important point about our local verb grouping is that we do not attempt to distinguish all the fine shades of semantics associated with these verb sequences. We extract just the right information that is needed and can be extracted efficiently for further processing.

Another important point is about our strategy for grammar design. In this approach rules are arranged in several layers each forming an exception to the previous layer.

1 Introduction

Indian Languages have relatively free word order; still there are units which occur in fixed order. The most important examples of these are the main verb followed by auxiliary verb sequences and nouns followed by postpositions. We term such units as verb groups and noun groups respectively. It may be noted that verb groups and noun groups will be sub parts of what are called verb phrases and noun phrases respectively. However in our formulation we do not use the concepts of noun phrases and verb phrases because

1. the concept of verb phrase does not seem to be natural for Indian languages
2. from computational point of view recognition of noun phrases and verb phrases is neither simple nor efficient.

On the other hand noun groups and verb groups can be formed using only local/surface information and more importantly they provide sufficient information (viz 'prayoga' and 'vibhakti transformation rules') for further processing of the sentence according to Paninian kaaraka theory. So the local word grouping provides all the necessary information with minimum computational effort.

Another important point about our local verb grouping is that we do not attempt to distinguish all the fine shades of semantics associated with these verb sequences. Our experience with Hindi and Telugu data suggests that to a large extent these fine shades are conveyed by identical conventions among Indian languages and so for translation purpose we need not disambiguate them. This approach is also consistent with Indian grammatical analysis where meaning is extracted in several layers with increasing precision.

The third important point is about our strategy for grammar design. As noted by Patanjali any practical and comprehensive grammar should be written in 'utsarga apavaada' approach. In this approach rules are arranged in several layers each forming an exception to the previous layer.

2 Overview of the Parser

Before describing local word grouper, let us look at the overall parser structure. The parser is part of our machine translation system that translates isolated simple sentences from Hindi to Telugu.

Figure 1:

Function of the morphological analyzer is to take each word in the input sentence and extract its root and other associated grammatical information. This information forms the input to the local word grouper (LWG).

2.1 Local Word Grouper (LWG)

The function of this block is to form the word groups on the basis of 'local information' (i.e information based on adjacent words).

This block has been introduced to reduce the load on the core parser resulting in increased efficiency and simplicity of the overall system.

The following example illustrates the job done by the LWG. In the following sentence in Hindi:

```
ladake adhyaapak ko haar    pahanaa rahe hein.



boys   teacher  to garland garlanding

(Boys are garlanding the teacher.)
```

the output corresponding to the word 'ladake' forms one unit, words 'adhyaapak' and 'ko' form the next unit, similarly

2.2 Core Parser

The function of the core parser is to accept the input from LWG and produce an 'intermediate language' representation i.e parse structure along with the identified semantic roles) of the given source language sentence. The core parser has to perform essentially two kinds of tasks

1. assignment of semantic role for verbs
2. sense disambiguation for verbs and nouns

For the task of role assignment, the core parser uses the fundamental principle of 'aakaankshaa' (demand unit) and

The linguistic units which play the role of demand and source word groups can vary depending on the parse cycle. In the case of simple sentences, only one cycle is needed in which verb groups and some special noun groups (e.g. 'paas'(near), groups and predicative adjectives play the role of source word groups.

2.3 Grammar for Core Parser

An example illustrates the parse cycle for a verb 'jota'. The verb 'jota' in Hindi has four different meanings listed in the dictionary :

1. harness (e.g., Raam ne bail ko kolhu me jotaa, or Raam harnessed the bullock for (turning) the crusher.)
2. hitching the cart (e.g., Raam ne gaadii ko jotaa, or Ram hitched the cart.)

3. plough (e.g., Raam ne jamiindaar kaa khet jotaa, or Raam ploughed the landlord's farm.)

4. exploit (e.g., Raam ne naukar ko kaam me jota diyaa, or Ram exploited his servant by putting him to (hard) work.)

For each of the four senses, we create a kaaraka chart. A kaaraka chart specifies the mandatory kaarakas (i.e., which must be filled for the sentence to be valid), optional kaarakas, and desirable kaarakas. For each of the kaarakas, it specifies the post position marker and the semantic specification (typically in the form of semantic type) to be satisfied by the source word (group). Such a specification for a kaaraka in a kaaraka chart is called a kaaraka restriction. Thus, the kaaraka chart for the 'hitching' sense of 'jota' has two mandatory kaaraka restrictions as below:

```
restriction on kartaa kaaraka (similar to agent):

  kaaraka:               kartaa

  mandatory:             yes

  post position marker: 0

  semantic expression:  human

restriction on karma kaaraka (similar to theme):

  kaaraka:               karma

  mandatory:             yes

  post position marker: 0-or-ko

  semantic expression:  cart
```

During the parsing process, each of the source word groups is teste against each of the kaaraka restrictions in each of the kaaraka charts of the demand word groups. An appropriate data structure is created storing the source word groups and the karaka restrictions (in kaaraka charts in demand groups) they satisfy. We call each such entry as a candidate variable.

Typically, a number of source word groups will qualify for a particular demand. The job of the core parser is to make an appropriate assignment of the candidates, subject to certain constraints such as the following:

1. one candidate source word group cannot satisfy more than one demand of the same demand word.

2. every obligatory demand must be satisfied in some kaaraka chart of every demand word group.

3. every source word must have an assignment.

4. if more than one interpretation of a source word is available, then exactly one has to be selected.

(The above description suffices for this paper. The actual parser uses merged kaaraka charts and lakshan charts. That is described in Bharati et al. (1990a) and (1990b Chap. 7).)

3 Verb Groups

As described in the last section, there are typically two kinds of word groups consisting of nouns and parsargs, and main verbs and auxiliary verbs, respectively. The exact groups formed, vary from language to language. For example, in Hindi, sequence of verbs, main and auxiliaries, occur as separate words and hence have to be grouped together. In Telugu, they occur together, conjoined into a single word: as a result, the analysis is left to morphology rather than to the local word grouper.

3.1 Kriya Rupa Charts

The kriya rupa charts specify the groups to be formed out of the sequence of verbs which denote a single action. Take for example 'khaa rahaa hai' in Hindi which consists of three verbs in a sequence, but they together denote a single act of eating. The role of the auxiliary verb is to give information about tense, aspect, modality etc.

To enable the machine to do the job of grouping, the following information will be required

1. possible verbal roots in sequences
2. information about gnp agreement

Rules of gender, number, person (gnp) agreement are com- paratively complicated in Hindi. The normal case would be a verb form like 'khaataa rahtaa hai', where the gender and number of each verb coincides with the gnp of the whole sequence. But consider the following verb sequences

1. khaatii rahtii hein
2. padhate rahnaa hai

In (1), the plurality of the whole sequence is reflected only in of the individual words. (See Fig. 1.)

Sentence and (g,n) of indivdual verbs	Overall tam and (g,n) of verb group
1. ladakaa padhataa rahataa hai. (m,s) (m,s) (*,s)	_taa_raha_taa_hai (m,s)
2. ladakiyan padhatii rahatii hein. (f,s) (f,s) (*,p)	taa_raha_taa_hai (f,p)
3. ladakiyan padhatii rahatiin hein. (f,s) (f,p) (*,p)	incorrect sentence
4. ladake ko padhate rahanaa hai. (m,s) (m,s) (*,s)	taa_raha_naa_hai (*,*)
5. ladake ko padhataa rahanaa hai. (m,s) (m,s) (*,s)	incorrect sentence

Legend:

m = masculine, f=feminine, s=singular, p=plural

* = don't care (no constraint on value)

Fig. 1: Table of some verb sequences

The linguist will have to provide this information. The first task therefore is to describe the above information in some fixed format. Take the three verb-forms in Hindi,

1. khaataa rahtaa hai
2. khaatii rahtii hein
3. khaate rahnaa hai

The linguistic information for (1) and (2) will be expressed as follows:

```
        label          : taa_rah_taa_hai

   vg-gnp         : [-2, -1, -1]

   seq-agree-spec: taa [agr,agr,-] rahataa [agr,agr,-]

                           hai [-,agr,agr]

   prayog         : kartaa

  kaaraka transformation rule: normal
```

Label indicates the raw tam marker for the (entire) verb group. It consists of concatenation of the tam of the main verb, followed by the roots and raw tams of the remaining verbs, separated by '_'. In the case of Hindi, the raw tam of a verb is simply the ending of the verb. The label is unique for the verb sequence (root of the main verb not included), and can be used for getting the above specification out of all such specifications that might be there. Seq-agree-spec is a specification giving the sequence of verb roots and their tams, and the rules of agreement. vg-gnp gives the specification for filling gender, number, and person of the verb group. Prayog indicates the kind of sentence in which this sequence can occur.

In the example above, the square brackets give the specifications regarding gender, number, and person respectively. Vg-gnp indicates how the gnp of the verb group is to be arrived at from the gnp of its constituent verbs. First number in square brackets, -2 hert, indicates that gender of the verb group is to be obtained from the gender of the second last verb in the sequence, and the following -1's indicate that number and person, respectively, of verb group take their values from the last verb (i.e., -1 when counted backwards). seq-agree-spec shows that if the first (main) verb ends in taa, and is followed by rahataa and hai, the sequence is possibly grammatical. (Actually this information has already been used in constructing the label and retrieving the specification.)

In the square brackets after taa (tam of the main verb), the first agr indicates that the gender of the main verb should agree with the gender of the verb group, the second agr indicates that the number of the main verb should agree with the number of the verb group, and '-' in- dicates don't care for person of the verb.

The format can now be described.

```
<value-of-seq-agree-spec> ::= <main-verb-tam> <agree-spec>

                        <aux-verb> <agree-spec>

                                   ...

                                <aux-verb> <agree-spec>

<agree-spec> ::= [<gender-spec>, <number-spec>, <person-spec>]
```

```
<gender-spec> ::= agr | - | m | f | n

<number-spec> ::= agr | - | s | p

<person-spec> ::= agr | - | u | m | a

<value-of-vg-gnp> ::= [ <num>, <num>, <num> ]

<num> ::= -1 | -2 | ...
```

Normally, the agreement rules described above are followed. There is an exception in Hindi, however. Whenever the verb group is feminine plural, the specified agreement rules are not used. Instead the last verb must be plural and all else singular. The gender and person of the verb group is still obtained useing vg-gnp.

The above suggests the method or the algorithm to be followed by the local word grouper. Given a sequence of verbs, V1 to Vn, the label is formed by taking the tam of V1 and concatenating with the roots and tams of the verbs V2 to Vn. Using the label, the specification is obtained. Vg-gnp is now used to obtain the gnp of the verb group. If an exception condition occurs (e.g., feminine plural in Hindi), the agreement rules associated with the exception are tested. Otherwise, the agreement rules with the specification (associated with the label) are applied. In case of success, the verb group is formed and processing contin- ues at the word after Vn. In case of failure of agreement rules, the above process is repeated with V1 to Vn-1. The process of forming a verb group starting from V1 terminates when either a verb group is formed using more than one verb besides V1, or a verb group consisting of single verb V1 is formed. In the latter case, gnp of the verb group is the same as that of V1. (More specialized methods can be used to avoid forming labels and checking for those sequences which cannot possibly occur. Some such methods are being used for Hindi. For example, intermediate verb groups contain at most two verbs; so we test for them directly. These can again be generalized after more data is available from other Indian languages.)

3.2 Kaaraka Chart Transformation

Both root as well as the tam in the verb group decides what kaaraka roles are mandatory, what are optional, what vibhakti they take, etc. For example, consider the following sentences with

```
Ram  ko  gaadi jotani  padi.


Ram -ko  cart  hitch   had-to

Ram had to hitch the cart.


Ram  se  gaadi nahi joti  gayi.


Ram -se  cart  not  hitch could

Ram could not hitch  the cart.
```

The vibhakti marker with Ram is 'ko' in the first sentence and 'se' in the second sentence. The vibhakti that the kartaa takes is determined by tam: naa_padaa in the first sentence and 0_gayaa in the second sentence.

The above is achieved by kaaraka chart transformation. As described earlier in core parsing (Sec. 2.3), root of the verb group is used for selecting the appropriate kaaraka chart. The kaaraka chart is transformed using tam in the verb group, and the tranformed kaaraka chart is what is actually used in core parsing. In the example here , the default kaaraka chart for jota (Sec. 2.3) shows the post-position marker as 0. It would get transformed using tam in the first sentence to 'ko', and to 'se' in the second sentence.

3.3 Semantics in Stages

Another significant aspect of our approach is that we do not try to get the full semantics immediately, rather it is extracted in stages depending on when it is most appropriate to do so. For example, in verb grouping only tam label is created, detailed time and modality information is not extracted. In fact, examples suggest that such information is rather complex and requires further processing and context etc. A particular tam like "taa_hai" (written as "taa hai" , "tii hai" depending on gnp) captures various shades of meanings as shown below

1. Ram chaay piitaa hai.
   Ram drinks tea.
   (Ram has no objection in drinking tea)
2. suurya puurva me nikaltaa hai.
   The sun rises in the east.
   (refers to periodicity in a natural law like sunrise)
3. prithvii suurya ke caaro taraf ghuumtii hai.
   Earth revolves around the sun.
   (refers to continuity in a natural law like the earth
   moving round the sun)
4. uske baad Ram ghar jaataa hai aur use ghar band miltaa hai.
   After that Ram goes home and he finds the house locked.
   (in the context of story telling, it indicates one
   instance of the activity of going and finding)
5. yadi vah aataa hai to tum jaanaa.
   If he comes, you go.
   (in a conditional sentence it signifies a one time
   activity)
6. Ram ko tairnaa aataa hai.
   Ram knows swimming.
   (indicates capability)

7. bicchuu dank maartaa hai.

   Scorpion stings.

   (shows the scorpion's inherent nature which is situation

   dependent)

8. Ram is samay duudh piitaa hai.

   At this time, Ram drinks milk.

   (shows habit)

9. Ram in dino duudh piitaa hai.

   These days Ram drinks milk.

   (signifies an activity which is localised in recent time)

In machine translation for Indian languages, it would be sufficient to keep tam labels without doing any semantic analysis provided the mapping of tam label from source language to tam label in the target language covers the senses spanned. Preliminary analysis among Hindi , Telugu, Kannada and Tamil indicates that it is indeed so.

4 Noun Groups

Noun groups are formed out of nouns and parsargs. In Hindi, each of the parsargs gets grouped with the preceding nouns. For example, the parsargs in each of the lines below, get grouped with the preceding noun ladake (boy).

```
ladake ne

ladake ke liye

ladake kii
```

For the noun to participate in the grouping, it must be in oblique form. For example, ladake has two possible lexical entries, one corresponding to singular-oblique and the other to plural-direct. (The oblique case for Hindi implies that noun takes a parsarg.) Only the singular-oblique lexical entry takes part in grouping with the parsarg. The number of the noun group is set to singular.

The actual parsarg encountered (possibly empty) is stored in the noun group as vibhakti. In morphologically rich languages, the noun itself gets declined depending on its relationship with the verb. In languages less rich in morphology, the parsarg serves a similar purpose. By incorporating both these in the group and calling it vibhakti, we are able to deal with it in a uniform way during core parsing.

```
In the sentence,

ladake phal   khaa rahe hein.



boys   fruits eat -ing  are

boys are eating fruits.
```

the first noun group is formed out of ladake alone. Only the plural-direct lexical entry takes part, and the number of resulting noun group is plural.

5 Strategy for Grammar Development

While the approach oulined above works quite well for most of the sentences, there are some problem cases. The problems usually arises due to ambiguity in lexical category of words, which produces conflict in word grouping. Depending on what lexical category is actually present, it results in a different word group.

To handle the problem cases we need a flexible approach which can make use of special rules and possibly case by case analysis. We follow the approach in which grammar rules are arranged in layers, each layer containing rules and forming an exception to the higher layer. This approach, called utsarga-apavaada (default-exception), is advocated by Patanjali (1880) for writing a practical and comprehensive grammar.

Let us first look at some of the conflicts and special rules (for some of the conflicts). Later we will look at the approach.

1. Conflict between taa form of verbs and the corresponding nouns : Sotaa is a verb (meaning 'to sleep') as well as a noun (meaning underground water). Similarly khaataa verb and khaataa noun (ledger or account).

2. Conflict between yaa form of verb and the corresponding nouns: Diyaa in Hindi can occur as a verb (to give) or as a noun (lamp).

3. Conflict between naa form of verb and nouns: Sonaa can occur as a verb (to sleep), a verbal noun (the act of sleeping), karma of verb, or as a noun unrelated to sleep (gold). The first three senses are related and the conflicts are systematic. Similar conflicts are likely to occur in many other cases. The last conflict for the sense (gold) is 'random' however. So also are the conflicts for cases (1) and (2) above. It should be possible to deal with systematic conflicts by general rules discovered by linguists; for the random conflicts, however, no rules are expected to work and a case by case treatment is necessary.

4. Conflict of verbal root with noun: Samajh can occur as a verb (to understand) as well as a noun (the result of understanding).

5. Conflicts of parsargs with nouns, verbs, relational words, etc . For example, se, kii, ke liye, and par normally occur as parsargs, but sometime they act as other category of words.

The above kind of conflicts can potentially be taken care of by the utsarga-apavaada approach in which there are multiple layers of rules. Normal rules are in layer 1. The problem cases, like the ones outlined above, would be declared as exceptions and there would be rules in layer 2 for them. Here is a sketch of the layers:

```
Layer 1 (General conflict resolver):

(1) If word is followed by pure parsarg  =>  noun

(where '=>' shows 'resolves to')

(2) If word is followed by verb sequence at the end of the sentence

=> final verb goup (most probably).



Layer 2 (specific exceptions):

In case of tam taa or yaa:
```

1. if followed by hua =¿ verb (most probably).

   Exception to be handled at layer 3: for khaataa as mentioned

2. if reduplication (e.g., khaata khaate) =¿ verb

3. if immediate previous word is kaa =¿ noun (most probably).

   Exception to be handled at layer 3: for sotaa as mentioned above.

Note that exceptions that lead to layer 3 are identified, but suitable rules at that layer have not been worked out. That task remains for the future. There is a need to take up systematic studies on this and related issues.

6 Some Open Problems

We list below some problems relating to conflicts between parsargs and other lexical categories, for which the solution in terms of suitable rules remains to be worked out.

```
(1) se can be used as a comparision word. For example,

 Phool  se   komal Ram ne dhanush tod diya.



 flower like soft  Ram -ne bow    broke

 Ram who was as soft as a flower broke the bow.
```

Perhaps when se is used as a comparision word, there are only a ' limited number of words that can follow it. Linguists can make catalog of such words.

```
(2) kii can occur as a verb. For example,

 Ram ne Mohan se baat kii.



 Ram -ne Mohan -se talk -ed.

 Ram talked to Mohan.
```

It might appear that one way to distinguish between the two usages is that the verb occurs at the end of the sentence unlike the parsarg. When kii occurs at the end, it can only be a verb. But would the following sentence be considered a incorrect sentence:

```
 Ram ne  kitaab padhi Mohan kii.



 Ram -ne book   read  Mohan -kii

 Ram read the book of Mohan.

(3) Par can occur as a noun. For example,
```

```
   Chitrakaar par   se    chitra  banaataa hai.


   artist     feather -se picture make -s

   Artist makes a picture with a feather.
```

Note that par se together can act as parsarg. For example,

```
            Vayuyan   sir   par se gujara.


   aeroplane head -par -se passed

   The aeroplane passed over the head.
```

(4) ke liye can occur as a verb. For example,

```
   meine ye   aam     das rupaya ke  liye.


   I   these mangoes ten  rupees -ke bought

   I bought these magoes for ten rupees.
```

In section 5, we have mentioned the conflict in taa forms of sotaa and khaataa verbs which also correspond to nouns. The LWG must make the right choice. Normally, depending on whether a postposition marker or an auxiliary verb follows khaataa (or Sotaa, for that matter) it can be decided whether it is used in the verb sense or the noun sense. However, there are some problem cases. In the sentence below, for example, a verb 'hua' follows 'khaataa' which occurs in a noun sense.

```
  Jab   se   khaataa  hua  hai,  neend nahi aati.


  Since when account  being,     sleep not  come

  Ever since I have an account, I cannot sleep.
```

7 Conclusions

In this paper, we have argued for the need for local word groups in Indian languages to handle those units in which word order is important. These word groups seem to be at the right level because firstly, they allow us to deal with issues of kaaraka role assignment (in core parsing) uniformly (hopefully) for all Indian languages. Secondly, they make exactly the right kind of information available to the core parse. We also outline the utsargs-apavaada approach to grammar design that is particularly suited for grammar development.

References

- Bharati, Akshar, Vineet Chaitanya, and Rajeev Sangal, 1990. 'A Computational Grammar for

Indian Languages Processing', Tech. Report TRCS-90-96, Dept of Computer Sc. & Engg, I.I.T. Kanpur, 1990a.

- Bharati, Akshar, Vineet Chaitanya, and Rajeev Sangal, 1990. 'A Computational Framework for Indian Languages', Tech. Report TRCS-90-100, Dept of Computer Sc. & Engg, I.I.T. Kanpur, 1990b.
- Bharati, Akshar, and Rajeev Sangal, "A Karaka Based Approach to parsing of Indian Languages," COLING-90: Int. Conf. on Computational Linguistics, Assoc. of Computational Linguistics, 1990c (forthcoming).
- Kielhorn, F., (ed.), The Vyakarana-Mahabhashya of Patanjali, Otto Zeller Verlag Osnabruck, 1970, (Reprint of edition 1880).p. 6 (m 1.1.1).