Chapter 9

Pāṇinian Interface for English Parsers

9.1 Introduction

Last decade has seen introduction of several parsers for English ranging from rule based to statistical. Within rule based again one sees parsers with a wide variety of formalisms such as Minipar [59] based on minimalism, enju parser [94] and LKB parser [29] based on HPSG, link parser [86] based on dependency grammar, XTAG parser [46] based on Tree Adjoining Grammar, to name a few. There are around half a dozen statistical parsers for English viz. Collins [27], Charniak [24], Stanford parser [61], re-ranking parser [25] and so on. The native output of all these parsers is naturally the grammar formalisms they follow. In case of rule based parsers, it is the grammar formalism they are based on, and in case of statistical parsers, it is the Phrase structure trees, since they are trained on the Penn Treebank which is annotated using Phrase Structure Grammar (PSG).

Recent years has also seen a growing trend towards producing dependency output in addition to the constituency trees. The dependency format is preferred over the
constituency not only from the evaluation point of view [59] but also because of its suitability [62] for a wide range of NLP tasks such as Machine Translation (MT), information extraction, question answering etc. However no two dependency output formats match with each other. There is no consensus among the dependency parser developers on the number of dependency relations and names of these relations.

Paninian Grammar (PG), the first dependency formalism, though is developed specifically for Sanskrit, has potential to provide guidelines for producing the dependency output of English sentences. Such guidelines will also be helpful in developing interfaces for the existing parsers so that one can plugin different parsers to the existing Machine Translation software.

We first summarize the issues involved with reference to English language parsing based on the dependency format output of the current English parsers. In the third section we highlight the Information theoretic viewpoint of PG, with special emphasis on English language. Fourth section contains guidelines for producing the dependency output for English, in the light of PG.

### 9.2 Dependency format output: some issues related to English

A dependency relation is an asymmetric binary relation mapping a modifier(or dependent) to the modified(or governor). The word being modified is the head. A word may have several modifiers but can modify only one word. If there are n words in a sentence, n-1 relations are necessary and sufficient to describe the parsed output.

There is a very close relationship between the dependency grammar and the link grammar [86] on which is based the link parser. The relations in link parser, however, are
not directional. The number of relations used in link parser is 106. Minipar also produces dependency format output and uses 59 relations. Carroll [23] and King [50] have proposed a set of dependency relations. Marneffe et al [62] have suggested modifications to these relations, largely based on practical considerations. The number of relations proposed by Marneffe are 47. Thus we see that there is a lot of variation among different parsed outputs with respect to the number of relations.

We looked at parsed outputs of different parsers for a wide range of sentences and recorded the phenomena where the parsed outputs differ. We also noticed certain cases where none of the parsers’ performance was acceptable. The differences in their performance could be related to the issues summarized below.

a) Whether to treat function words such as prepositions, auxiliary verbs, etc. as words indicating relations thereby avoiding relations between these words with other content words or to treat these words at par with the content words?

This will have serious effect on the number of content words and the number of relations in a sentence.

b) The basic assumption of dependency grammar is that a modifier modifies only one word. In the following sentence

\[ \text{Ram went home and slept.} \] (1)

Ram is a modifier of went as well as slept. Whether the parser should produce both the relations or only one?

Similarly in the sentences with missing wh-relativizer
I saw the man you love. (2)
The snake the mongoose attacked hissed loudly. (3)
whether the output should account for the missing wh-relativizer?

In case of subject and object control verbs such as
Ram persuaded Mohan to study well. (4)
Ram promised Mohan to study well. (5)
should the output account for the sharing of semantic roles by verbs?

(2) c) What should be the level of analysis – syntactic (specifying the subject, object relations), semantic (specifying the thematic roles), or something else?

d) Should the heads be decided semantically or syntactically? For example, in case of a cup of tea, the semantic head is tea, whereas the syntactic head is cup. In case of growth of industry, growth is both the semantic as well as syntactic head.

e) Should the sentences Ram is good. (6)
and
Ram is a doctor. (7)
be treated alike, with semantic representation as good(Ram), and doctor(Ram) respectively, or should they be analyzed differently, reflecting different underlying phrase structures?

To answer these questions, we look at English language from the ‘information coding’ point of view. We seek answers for the following questions.
i) What means does English use to code the information about relations?
ii) What is the manner of coding the information, and finally,
iii) What is the semantic content of these relations?

### 9.3 Pāñinian Grammar

According to Pāñinian Grammar (PG), a modifier may be classified into two major categories: samāṇādhikaraṇa (modifier and modified having the same locus), and vyadhikaraṇa (modifier and modified have different loci).

Examples of samāṇādhikaraṇa modifiers are:

1. a determiner modifying a noun (the boy)
2. an adjective modifying a noun (good boy)

Examples of vyadhikaraṇa modifiers are:

1. nominal expressions modifying a verbal root, also known as the kāraka relations,
2. a verb modifying another verb, etc.

Essentially, the samāṇādhikaraṇa modifier and the corresponding modified head denote the same thing, and belong to the same word group\(^1\). So this kind of relation is a ‘word-group-internal’ relation. On the other hand the vyadhikaraṇa modifier and the corresponding modified head belong to different word groups, and hence the relation involved here is ‘across-the-word-group’ relation. In short, the vyadhikaraṇa modifiers are the building blocks of the parsed structure, whereas, with the samāṇādhikaraṇa modifiers, the modifiers add flesh to this structure.

The most important vyadhikaraṇa modifiers are the kāraka relations.

---

\(^1\) Of course, there are cases where the words may belong to different word groups and still may have the same locus, as in the case of *He is a doctor.*
1. In Bharati et al [10] it has been pointed out that English codes the kāraka relations in position as well as through prepositions.

2. Languages do not code all the kāraka relations explicitly. For example, when a word has more than one kāraka role with respect to different verbs in the surface structure of a sentence, only one kāraka relation is coded and other kāraka relation needs to be inferred from the language’s grammatical rules (language conventions) or through the properties of lexical items. For example in sentence (1), it is the language convention which tells Ram is the subject of both the verbs went and slept. In sentences (2) and (3), it is the syntax of English which allows wh-drop and thereby allow sharing of more than one kāraka role by the same nominal expression. In sentence (5) the information that subject of study is Ram, and in sentence (4) it is Mohan, which is coded in the meaning of lexical items promise and persuade respectively.

3. According to PG, the kāraka relations are the relations which map nominal expressions to verbal roots. These are syntactico-semantic relations. These indicate the optimum semantic analysis one can do using the language string and the language conventions alone without appealing to the world knowledge\(^2\). Given the fact that present day computers are still not capable of handling the world knowledge, from computational point of view, it is a major milestone in the language analysis. One kāraka relation may correspond to more than one thematic role. For example, in the following sentences

\[
\begin{align*}
\text{Ram opened the lock with this key.} & \quad \text{(8)} \\
\text{This key opened the lock.} & \quad \text{(9)} \\
\text{The lock opened.} & \quad \text{(10)} \\
\text{Ram, this key and the lock are all kartā, whereas their thematic roles are viz.}
\end{align*}
\]

\(^2\text{See chapter 5 for more details.}\)
agent, instrument and goal respectively. Similarly each semantic role may get realized into more than one kāraka relation. For example, *key* in sentence (8) is kāna kāraka and in sentence (9) karta kāraka. *Lock* is the karma kāraka in sentences (8) and (9), whereas karta kāraka in sentence (10).

To summarize,

1. English codes the kāraka relations both by position as well as through the prepositions.
2. Some relations are coded explicitly and some implicitly.
3. The maximum semantics one can extract is the syntactico-semantic relations and not the thematic roles.

9.4 Guidelines for producing dependency output for English

We answer the issues raised in the second section, which will lead to the guidelines for producing the dependency output for English. Appendix G and Appendix H contain the outputs generated by the Stanford and Link parsers respectively, while the appendix I contains the dependency trees following Pāṇinian grammar.

1. In the light of earlier discussion, it is clear that we treat the prepositions connecting a noun with a verb or another noun as a relation rather than a content word. Further the auxiliary verbs together with the main verb form a ‘semantic unit’ leading to a word group with main verb as the head. Hence the auxiliary verbs should be grouped with the main verb, and there is no necessity of mentioning the internal relations.
2. Sentences (1) through (5) are all examples of kāraka sharing and implicit encoding of the unspecified kāraka relations. The implicit encodings are typically language grammar and lexicon specific and hence need to be made explicit in the parsed output.

3. On the basis of the discussion above, it is clear that, language codes only syntactico-semantic relations. So what one can extract from the language string alone is only syntactico-semantic relations and not the thematic roles. For Sanskrit we have kāraka - vibhakti mapping rules described in the Aṣṭādhyāyī. Similar rules need to be worked out for English. Till then, we describe the relations in terms of objects of prepositions or by subject and object positions. In other words, the relations will therefore be marked as subject and object, in case they are expressed by position, and by the preposition-object, such as by-obj, with-obj etc., in case they are expressed by prepositions.

4. In case of ‘of’, since it is the syntactic head expressed by the relation, we mark only the syntactic head. The determination of semantic head requires the world knowledge, and hence should be dealt with in a separate module.

5. In English two sentences may have different Phrase structures, but their semantic content may be the same. PG treats them in a uniform way, by postulating a samānādhikaraṇa relation between Ram and good, and also between Ram and doctor. This in fact is an example of samānādhikaraṇa modifier across the word groups!

9.4.1 Conclusion

In the light of above discussion the relations may be classified into three categories viz. word-group-internal relations, across-word-group-explicitly marked relations, and across-word-group-implicitly marked relations. The word-group-internal rela-
tions may be best handled by the constituency trees, whereas the across-word-group relations may best be handled by the dependency relations. Chunkers may be the reliable tools for marking the inter-word-grouping. The word grouper developed in-house performs better than the chunker on main verb-auxiliary verb grouping. Handling the implicit relations involve some heuristic rules. These need to be, therefore, marked separately.

We have shown that various parsers differ in their behaviour with respect to the issues raised above. For example Link parser treats prepositions as content words. It also treats sentences (6) and (7) differently. Stanford parser and Enju parser on the other hand try to do deeper semantic analysis leading to over-generalizations in some cases. The differences among these parsers make it difficult to compare the parsers qualitatively.

Interfaces based on the principles outlined above are being developed for various parsers. These interfaces are easy to use by a layman for understanding the ‘parsed output’ without much linguistic training [16]. It also facilitates comparison of different parsers. The initial motivation for building such interfaces was to provide a plugin facility for plugging-in different parsers.