Chunker and Shallow Parser for Free Word Order Languages: An Approach based on Valency Theory and Feature Structures

Dipanjan Das and **Monojit Choudhury**
Department of Computer Science and Engineering
Indian Institute of Technology Kharagpur
West Bengal, INDIA 721302
Email: *{dipanjan, monojit}@mla.iitkgp.ernet.in*

Abstract

Free word order languages have relatively unrestricted local word group or phrase structures that make the problem of chunking quite challenging. On the other hand, a robust chunker can drastically reduce the complexity of a parser that follows. We present here a computational framework for chunking of free word order languages based on a generalization of the *valency theory*. Every word has certain valency constraints that allow it to form local word groups with the adjacent words. The groups so formed have their own valency constraints and thus can recursively participate in the grouping process. This is implemented through feature structure unification. Sentence level constraints on the groups help the chunker reject invalid groupings and detect errors in POS tags provided by a POS-tagger that precedes the chunker. Based on this approach, a Bengali chunker has been implemented with a reasonably good accuracy. The paper also describes how this method can be generalized to develop a complete parser for free word order languages by incorporating semantic information as well as probabilistic models.

Keywords: valency theory, feature structure, chunking, local word grouping, parsing, free word order languages, part-of-speech tagging, morphological analysis

Chunker and Shallow Parser for Free Word Order Languages: An Approach based on Valency Theory and Feature Structures

Abstract

Free word order languages have relatively unrestricted local word group or phrase structures that make the problem of chunking quite challenging. On the other hand, a robust chunker can drastically reduce the complexity of a parser that follows. We present here a computational framework for chunking of free word order languages based on a generalization of the *valency theory*. Every word has certain valency constraints that allow it to form local word groups with the adjacent words. The groups so formed have their own valency constraints and thus can recursively participate in the grouping process. This is implemented through feature structure unification. Sentence level constraints on the groups help the chunker reject invalid groupings and detect errors in POS tags provided by a POS-tagger that precedes the chunker. Based on this approach, a Bengali chunker has been implemented with a reasonably good accuracy. The paper also describes how this method can be generalized to develop a complete parser for free word order languages by incorporating semantic information as well as probabilistic models.

1. Introduction

Chunking or *local word grouping* refers to the identification of syntactically correlated parts of words in a sentence and is usually the first step towards parsing of a natural language sentence [Abney, 1991]. Statistical and supervised learning methods for chunking have been explored during the past decade and are quite popular since they are language independent, robust and readily applicable to a new domain or language as long as there is enough data to train the system [Buchholz *et al,* 1999 ; Ramshaw and Marcus, 1995].

However, the problem of chunking takes a new dimension for free-word order languages, where the internal structure of the chunks or local word groups (LWG) is relatively unrestricted. Often sentence level clues or constraints become necessary for identification of the LWGs, but at the same time, a robust and efficient chunker for a free-word order language can drastically reduce the complexity of the parser. We propose here a computational framework for chunking of free-word order languages and describe the implementation of a Bengali chunker in this framework. Due to the lack of machine-readable linguistic resources in Bengali (tagged and chunked corpus to be more precise), the rules for the system have been manually designed. However, given a sufficiently large tagged corpus, where the LWG boundaries are marked, the rules can be statistically learnt as well.

The approach described here is motivated by the valency theory [Herbst, 1999] and feature-structure unification methods used in Phrase Structure Grammars [Carpenter, 1992]. We propose a generalization of the valency theory, where every word has a valency structure represented in form of feature structures. Local word groups are formed when unification is possible between the feature structures of two adjacent words. As a result, every local word group has its own feature structure and valency constraints. There are also sentence level constraints over the local word groups, which can identify ill-formed groups or errors in the POS tags of the words.

This paper is organized into six sections. Section 2 defines the problem and gives a brief introduction to features structures and the valency theory in the context of chunking and parsing of free-word order languages. Section 3 describes the feature structure and the valency constraints used for Bengali chunker. Section 4 elucidates the algorithm for unification and grouping of words followed by a section on the implementation details and the results observed. The concluding section summarizes the paper and describes how this method can be modified for development of a parser for free-word order languages.

In this paper, Bengali script has been written in italicized Roman fonts following the ITRANS notation [Chopde, 2002].

2. The Problem

Chunking or local word grouping has been traditionally defined as the process forming groups of words based on local information [Bharati *et al*, 1995]. Abney [1991] viewed chunks or local word groups to be connected subgraphs of the parse tree of a sentence. The distinction between function and content words were used to design a chunker based on non-deterministic LR – parsing.

Recent techniques in chunking use statistical and machine learning approaches. This has two advantages – first, it obviates manual creation of chunking rules and grammars; second, given a sufficiently large corpus of a language, where the chunks are marked, the algorithm can readily adapt itself to that language. *Transformation-based learning* techniques have been able to achieve approximately 92% precision and recall rates in identifying noun phrases in English sentences and around 88% for other complicated chunks [Ramshaw and Marcus, 1995]. Other works report more than 90% precision and recall rates for identification of English noun, verb, prepositional, adjective and adverbial phrases [Buchholz, et al., 1999; Zhang et al, 2002].

Indo-Aryan languages being relatively free word ordered are difficult to tackle using a generative grammar approach. Moreover, unavailability of chunked corpora precludes the use of available statistical approaches. The problem of dividing a Hindi sentence into word groups has been explored by Bharati *et al* [1995]. The output of the grouper served as an input to a computational Paninian parser. However, in their work, they made a distinction between local word groups and phrases. They assert "from a computational point, the recognition of noun phrases and verb phrases is neither simple nor efficient". Therefore, the scope of the grouper was limited and much of the disambiguation task was left to the parser.

The problem of phrase identification in Hindi has been explored by Ray *et al* [2003] using a rewrite rule based approach. But their technique gave only one possible grouping of a particular sentence, did not consider sentence level constraints, which at times generated ill-formed groups and the case of noun-noun groups has been completely ignored. In the next subsection we take a look at the structural property of phrases in Bengali, which being a free word order language reflects the challenges involved in chunking of such languages.

2.1 Structure of Phrases and LWG in Bengali

The word order of Bengali is not as rigid as English. To illustrate this, consider the following sentence.

ekaTA sAdA gho.DA dekhalAma sei lAla girjAra sAmane.
one white horse saw[first person] that red church[of] in front
(I saw a white horse in front of that red church.)

Table 1 shows the different ways in which one can permute the words of this sentence without changing the meaning. However, not all permutations are allowed. For example, the sentence

sei lAla gho.DA dekhalAma ekaTA sAdA girjAra sAmane .
That red horse saw[first person] one white church[of] in front .
(I saw that red horse in front of a white church.)

conveys a different sense. Some of the other permutations of the sentence can be ungrammatical as well. This can be explained by the fact that certain groups of words have fixed order, which we define as the local word groups. The groups can be permuted without restriction, but the words within a group must occur contiguously. In Table 1, the fragments that occur in a chunk are the LWGs.

LWGs illustrated in the table, however allow certain degree of permutation of the words within themselves. For example, the LWG "ekaTA sAdA gho.DA" can also be stated as "sAdA ekaTA gho.DA" without changing the sense. However, the sub-group "girajAra sAmane" is completely fixed. Therefore, we define the concepts of *strong* and *weak* LWGs.

ekaTA sAdA gho.DA	*dekhalAma*	*sei lAla girjAra sAmane .*
dekhalAma	*ekaTA sAdA gho.DA*	*sei lAla girjAra sAmane .*
dekhalAma	*sei lAla girjAra sAmane*	*ekaTA sAdA gho.DA .*
sei lAla girjAra sAmane	*dekhalAma*	*ekaTA sAdA gho.DA .*
sei lAla girjAra sAmane	*ekaTA sAdA gho.DA*	*dekhalAma .*
ekaTA sAdA gho.DA	*sei lAla girjAra sAmane*	*dekhalAma .*

Table 1. Different arrangements of words in a sentence.

Definition 1: A *strong word group* is one that has an internal rigid word order and any permutation of the constituent words either changes the sense of the group or is grammatically incorrect.

Definition 2: A *weak word group* is composed of more than one *strong word groups*, and there is negligible change in the sense of the word group, when the individual strong groups are permuted among themselves, but the constituent strong groups may not be placed beyond the weak group boundary.

It may be mentioned here that in [Bharati *et al*, 1995] only strong groups have been identified, where as in [Ray *et al*, 2003] some of the weak groups were also considered. The identification of weak word groups is a step ahead towards parsing as they capture phrases and sometimes even clauses in a sentence. An example sentence that portrays both *strong* and *weak* word groups can be like:

{ (ja~Ngala diYe) (yete yete) } {(ekaTA) (chhoTa) (nadi) } (chokhe) (pa.Dala) .
forest through go[participle] go[participle] one small brook eyes[in] fell .
(While travelling through the forest I noticed a small brook .)

Here, (ja~Ngala diYe) and (yete yete) form two strong word groups that in turn form a larger weak group, which can be considered to be a adverbial phrase. Although, *chokhe* and *pa.Dala* are two disjoint strong word groups and can surface anywhere in the sentence, together they form a multi-word expression. Identification of such free order multiword expressions is a very difficult problem and is beyond the scope of the present work.

Another syntactic feature of Bengali is the deletion of be-verbs in present tense. This is also referred to as *hidden copula*. For example, in the sentence

ei baiTA AmAra.
this book mine
(This book is mine.)

the 'is' has been dropped. Here, the groups are *ei baiTA* and *AmAra*. However, a sentence like "*ei baiTA AmAra ba.Dai bhAlo*" (this book of mine is very nice), is comprised of two weak groups – *ei baiTA AmAra* and *ba.Dai bhAlo*. This clearly illustrates the need for sentence level constraints during local word grouping of free word order languages.

2.2 Valency Theory and Feature Structure

Valency theory takes an approach towards an analysis of sentences that focuses on the role of certain key words in a sentence. For example, certain verbs allow only certain kinds of predicates some of which are compulsory while some others may be optional. The arguments that a verb can take are defined in terms of its valency. It was introduced by French structuralist Lucien Tesnière in dependency grammar formalism. Numerous theoretical publications [Helbig, 1992; Welke, 1995] have established it as arguably the most widely used model of complementation in languages like German, French, Romanian and Latin. Valency theoretic description of English has also been explored [Allerton, 1982; Herbst, 1988; Herbst, 1999]. Traditionally, valency theory has focused on the syntactic and semantic valencies of verbs and occasionally of nouns.

We propose here a generalization of the valency theory, where each syntactic unit of a sentence at every level of analysis has a certain number of valencies associated with it. For the purpose of chunking, where only adjacent words or groups can be attached to form larger groups, the units (word or word groups) have a maximum of two valencies associated with them – the right and the left. The right valency defines the affinity of the unit to form groups with the units that lie to its right in a sentence. The left valency can be defined likewise.

However, the affinity of a unit to form groups also depends on the syntactic attributes of the adjacent units. For example, a noun can readily form a group with an adjective to its left, but never with a finite verb. Therefore, it is not enough to specify whether a unit can combine with other units to its right or left; the syntactic attributes of the adjacent units are also important. This can be aptly captured by the concept of feature structure (FS). Feature structures are widely used in phrase structured grammars to capture the syntactic dependencies between the words. We define our own FS to incorporate the notion of valency. Successful unification between adjacent FS implies formation of a valid group with its own FS, i.e. the valency constraints.

The parallelism between the valency theory and the computational Paninian parser [Bharati *et al*, 1995] needs a mention here. In the Paninian approach the *akanksha* of a verb and *yogyata* of a predicate or a phrase is comparable to the concept of valency.

3 Overview of the Framework

Figure 1 shows the general framework proposed in this work. There are three basic units – a POS tagger, the basic grouper and a global constraint checker. The focus of this work is on the basic grouper and the constraint checker.

The POS tagger takes an input sentence and tags each word with its POS category and other morphological attributes valid for that category. The grouper does not presuppose hundred percent accuracy of the POS tagger. It tries to formulate valid groups based on the POS tags and the morphological attributed provided by the tagger. Nonetheless, if it finds some anomaly in the groups so formed at the sentence level, it identifies possible locations where the POS tagger might have gone wrong and gets back to the POS tagger with the identified errors. The POS tagger, in turn tries to rectify those errors by providing alternative tags. Thus the whole process runs in an interactive manner and finally arrives at a correct grouping.

Sections 4 and 5 look at the basic grouping procedure. Here we take a look at the global constraint checker. Global constraints can be either syntactic or semantic. A syntactic constraint can be only on the POS categories of the words, for example in the following sentence *Ara*, which is here an adverb, has been wrongly tagged by the POS tagger as a conjunction.

Sentence	*Ami*	*Ara*	*khAba*	*nA*
POS tags	pronoun	*conjunction	finite verb	negative particle
Transliteration	I	more	eat [future]	no
Translation	I will not eat any more.			

The algorithm groups *khAba nA* as a finite verb group and makes *Ami* to be a pronominal phrase. At the end of the process, local constraints on the conjunction leaves *Ara* ungrouped since the categories of the two word groups on the left and right of *Ara* cannot be different. After the basic grouping process, the global constraint checker finds a conjunction group *Ara* none of whose valencies has been satisfied. Since, the minimum valency of a conjunction should be two, the global constraint checker reports a possible error in the POS tag of the word *Ara*.

Checking of semantic constraints is beyond the scope of the present work. Given a correct POS tagging, local word grouping can proceed successfully without them. But with errors in POS tags, semantic constraints may become essential for parsing purposes.

4 Feature Structures for Bengali

Although the concept of feature structure as well as valency theory is independent of any language, the specific features and the attributes that the features can take are overtly language specific. Therefore a complete definition of the feature structure requires a detailed linguistic analysis of the language, especially the dependencies between the words in strong and weak LWGs and between the groups themselves. The contents of a feature structure are shown in Fig 2.

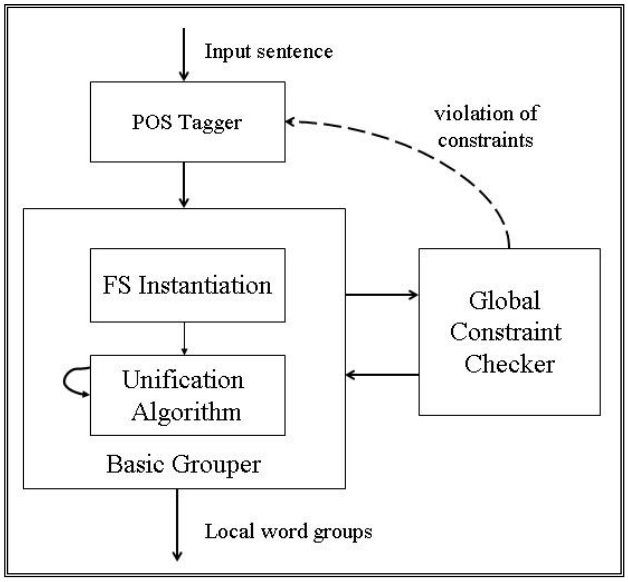

Figure 1. The Overview of the Framework

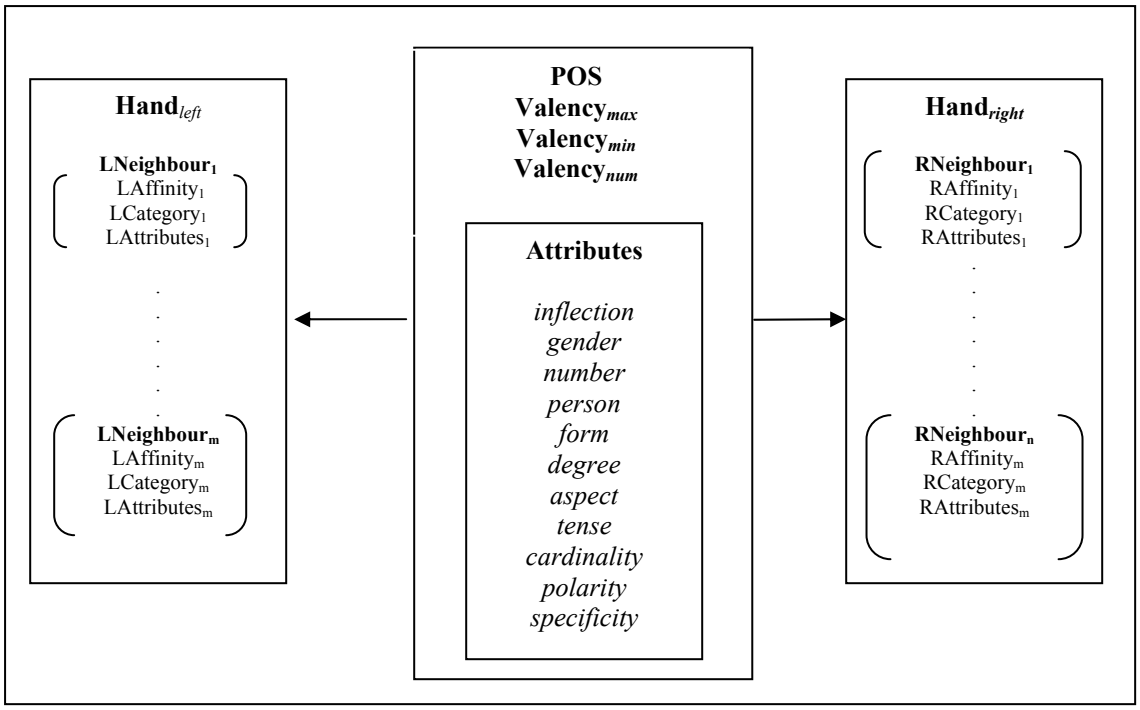

Figure 2. Contents of a Feature Structure

The **POS** category and **Attributes** of the FS are self explanatory. We explain below the other features:

- **Valency$_{max}$**: The maximum number of adjacent units that the object can combine with to form groups.
- **Valency$_{min}$** : The minimum number of adjacent units that the object can combine with to form groups.
- **Valency$_{num}$** : The number of valency values that should be satisfied by the object for a valid grouping. For example, if the possible valency values of a particular object are 1 and 2. Then $Valency_{max} = 2$, $Valency_{min} = 1$ and $Valency_{num} = 2$. Please note that $1 \leq Valency_{num} \leq 3$.
- The ***hands*** : Each FS has two *hands* – the right and the left. The right (left) hand specifies the constraints on the units to the right (left) of an object that can be grouped with it. The constraints are specified as a list of 3-tuples <*Category, Attributes, Affinity*>. *Category* and *Attributes* stand for the allowable POS category and the attributes of the category (such as inflection) respectively. The *Affinity* can be either *weak* or *strong,* depending on whether the group formed will be a weak group or strong group.

Usually, a POS tagger tags a sentence at the most specific level. For example, if there are different tags for proper nouns (NP) and common nouns (NN), the POS tagger tags a given noun as either NN or NP. However, at the local word group level, both NN and NP behave identically. This can be modelled using a hierarchical classification of POS tags, where the leaves are the tags provided by the tagger, but FS can specify constraints at any level of abstraction. Thus, a postposition can have a Noun to its left, it is immaterial whether the noun is an NP or NN. Figure 3 shows the classification for Bengali POS tags used during this work.

5 Grouping Algorithm

The algorithm for grouping proceeds in two distinct steps. In the first, for each word of the input sentence to be chunked, the corresponding feature structures are instantiated. In the next, the feature structures are unified using specific unification rules.

5.1 Instantiation of Feature Structures

The sentence to be processed is first fed into a POS tagger that tags each word with its respective POS tag and its different syntactic attributes derived from its morphology. This string of tagged words acts as the input to the instantiation phase of feature structures. The major steps of instantiation are as follows:

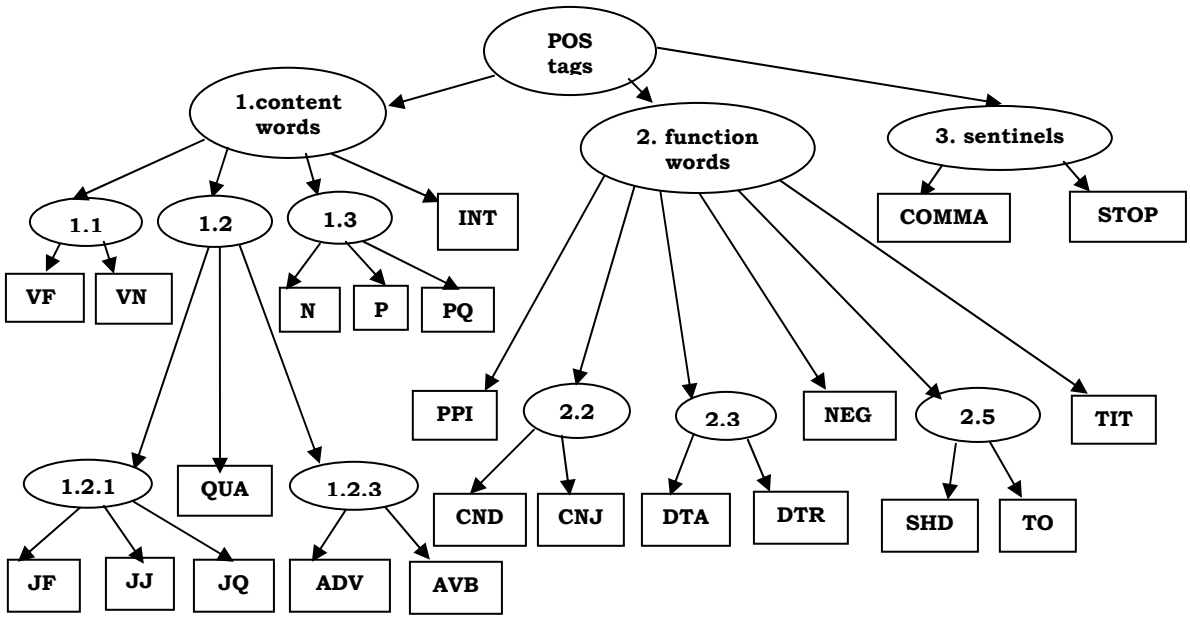

Figure 3 Classification of Bengali POS

1. Each word of the sentence is extracted one after another.
2. A rule file which assists in the process of instantiation consists of the template for FS for a POS category or a group of POS categories (that have similar attributes, left and right hand structures).
3. For a particular word, its POS tag and the attribute information supplied by the POS tagger is noted. For the POS category, the template for FS is searched for in the rule file.
4. The concerned template is selected and with the attributes, a new instance for the FS is created.
5. Steps 3 and 4 are repeated until all words of the sentence are exhausted.

5.2 Grouping process

Following the instantiation of feature structures for each word of the sentence, the string of feature structures serves as the input to the grouper where a two tier grouping process takes place. The strong groups are identified during the first phase of grouping followed by identification of all possible weak groups. The algorithm is as follows :

Phase 1

1.1. Grouping can proceed from either the left or right extremity of the string of feature structures. From any extremity, each FS is considered one after another. The FS under consideration is termed as the *seed.*
1.2. Let for the seed *x*, *r* and *l* be the right and the left neighboring FSs respectively.
1.3. If the right (left) hand of *x* contains a 3-tuple <cat_i, att_i, strong> and the category and attributes of *r* (*l*) are cat_i and att_i respectively , then *x* may form a strong word group with *r* (*l*).
1.4 If the right (left) hand of *x* contains a matching 3-tuple, then a local constraint checking (described below) is performed on *x* and *r* (*l*). If they satisfy the constraints, the *x* forms a strong word group *s* with *r* (*l*).
1.5 The rules for unification of the feature structures are read from a repository and the FS for the strong word group *s* formed in Step 1.4 is accordingly instantiated.
1.6 Steps 1.2-1.5 are repeated with *s* as the new *seed*.
1.7 In step 1.3, if the grouper cannot find a matching 3-tuple, the FS to the right (left) of *x* is chosen as the new seed. Iteration continues from step 1.2.
1.8 If sentence boundary is encountered to the right (left), the grouper stops.

Phase 2

2.1. A stack *S* of partial groupings of a sentence (i.e. a string of FSs) and the corresponding seed for each grouping is defined. It is initialized with the output of phase I (i.e. the string of strong groups).The seed may be chosen as either the rightmost or the leftmost group.
2.2. While the stack *S* is not empty, it is popped to get the partial grouping *P* with seed *x*.
2.3 Let *r* and *l* be the right and the left neighboring FSs of *x* respectively.
2.4. There are three possibilities –

a. *x* cannot form a weak group with *r* (*l*). This is the case when no 3-tuple in the right (left) *hand* of *x* allows unification with *r* (*l*).
b. *x* definitely forms a weak group will *r* (*l*). This is the case when a 3-tuple in the right (left) *hand* of *x* can be unified with *r* (*l*) and the valency constraints (described later) forces *x* to form the group.

c. *x* may or may not form a weak group with *r* (*l*). This is the case when a 3-tuple in the right (left) *hand* of *x* can be unified with *r* (*l*), but the valency constraints do not force *x* to form the group.

2.5. If *a* holds, then the FS to the right (left) of *x* is considered to be the new *seed* and the process of grouping continues from step 2.3.

2.6 If *b* holds, then the unification procedure takes place and a weak group is formed which becomes the new seed. Grouping process continues from step 2.3 .

2.7 If *c* holds, there are two partial groupings, one in which the unification takes place and the new weak group is the seed (as in 2.6) or no group is formed and the seed is *r* (*l*), as in 2.5. Both of these partial groupings are pushed into the stack and the process continues from step 2.2

2.8 If a sentence boundary is encountered to the right (left), the current grouping is sent to the solution set after *global constraint checking*. The grouper continues from step 2.2.

Local Constraint Checking

Local Constraints are basically constraints on the valencies of a FS.

1. In a few cases (e.g. conjunctions), when an FS *x* is forming a group with an FS *r* (*l*) to its right (left), it becomes important to view the feature structures *r* and *l* simultaneously with certain constraints. If these constraints are not satisfied, grouping cannot proceed successfully.
2. When a strong group is formed, it is always essential to check whether the valency constraints of *x* are satisfied. The valency constraints are the values of $Valency_{max}$, $Valency_{min}$ and $Valency_{num}$ which provides information about the number of components *x* can combine with.

6 Implementation and Results

A chunker for Bengali based on the framework described here has been implemented. The inherent object oriented nature of the feature structures has prompted us to implement the chunker in Java. The chunker also depends on a morphological analyzer (MA) and a part-of-speech (POS) tagger, which have not been described in this paper. However for the sake of completeness, a few words must be mentioned about them. Both of these tools have been developed in house. The POS tagger has been designed with a combination of both supervised and unsupervised stochastic learning techniques. Its accuracy is around 80%. The MA can provide information about the inflections, gender, number, person, tense, aspect, polarity, specificity etc. for a word. It has been implemented using Directed Acyclic Word Graph methods. The accuracy of the MA is around 95%.

The salient features of the chunker for Bengali have been summarized below:

- The POS tags a sentence using 43 tags, which are considered as the basic POS categories of the feature structures.
- The rules for instantiation of the feature structures are specified separately in a file. For a given POS category, the file contains its maximum and minimum valencies, the number of valencies and the information about its left and right hands. There are around 25 such rules.

- The rules for unification are specified in another file that provides the information about the new group formed after unification of two FSs. For example, an FS of category qualifier and another of category adjective unify to form an adjective group, that is an FS of category adjective.
- The implemented chunker can form strong word groups as well as weak ones. But it fails to recognize multi-word expressions.
- Multiple word groupings for the same sentence are also recognized by the chunker.
- Errors in POS tags by the POS tagger are identified to certain extent using local and global constraint checks, but rectification of those errors though active interaction with the POS tagger has not yet been implemented.
- The chunker cannot handle cases where semantic issues are involved.

The chunker has been tested on a set of 50 randomly selected sentences. The strong word groups are identified with 100 % accuracy provided the POS tags are correct. Weak groups are identified with around 80% accuracy for correct POS tags. For a few cases, it could also identify the errors in the POS tags. A detailed testing of the system is underway.

7 Conclusion

In this paper, we have described a new framework for chunking of free word order languages based on a generalization of the valency theory implemented in form of feature structures. We defined the concepts of strong and weak local word groups. Weak local word groups may correspond to phrases or clauses in a sentence and therefore identification of such groups can be considered to be a significant step towards parsing and hence the system can be referred to as a shallow parser.

This formalism can be extended to complete parsing of free word order languages. This can be accomplished through the inclusion of *semantic valencies* and *non-adjacent syntactic valencies*. The idea is similar to that of *karaka* or traditional valencies of verbs. Future research can also focus on extending the concept of affinity from strong, weak or null to a probability value between 0 and 1, where 0 denotes no affinity and 1 denotes a strong affinity. Weak affinities can be something around 0.5. These affinities can be learnt from a chunked corpus.

References

[Abney, 1991] S. Abney. *Parsing by chunks* In Berwick, Abney, and Tenny, editors,*Principle-Based Parsing.* Kluwer Academic Publishers.

[Allerton, 1982] D. Allerton. *Valency and the English Verb*, London/New York: Academic Press, 1982

[Bharati *et al*, 1995] A. Bharati, V.Chaitanya, and R.Sangal. *Natural Language Processing A PaninianPerspective*; Prentice Hall India, 1995.

[Buchholz *et al,* 1999] S. Buchholz, , J. Veenstra and W. Daelemans. "Cascaded Grammatical Relation Assignment". *Proceedings of EMNLP/VLC-99*, University of Maryland, USA, 1999.

[Carpenter, 1992] R. L. Carpenter. *The Logic of Typed Feature Structures: With Applications to Unification Grammars, Logic Programs and Constraint Resolution*, Cambridge University Press, UK, 1992

[Chopde, 2002] A. Chopde. *ITRANS*, version 5.30, July 2002, http://www.aczone.com/itrans/

[Helbig, 1992] G. Helbig. *Probleme der Valenz- und Kasustheorie,* Tübingen: Niemeyer, 1992

[Herbst, 1988] T. Herbst. *A valency model for nouns in English*. Journal of Linguistics. 265-301, 1988

[Herbst, 1999] T. Herbst, *English Valency Structures – A first sketch*. Technical report EESE 2/99, 1999. http://webdoc.sub.gwdg.de/edoc/ia/eese/artic99/herbst/main1.html#0

[Ramshaw and Marcus, 1995] L. A. Ramshaw, and M. P. Marcus. *Text Chunking using Tansformation-Based Learning*; ACL Third Workshop on Very Large Corpora, 1995.

[Ray *et al,* 2003] P.R. Ray, V. Harish, S. Sarkar, and A. Basu. *Part of Speech Tagging and Local Word Grouping Techniques for Natural Language Parsing in Hindi*, Proceedings of International Conference on Natural Language Processing (ICON 2003), Mysore 2003.

[Welke, 1995] K. Welke. *Dependenz, Valenz und Konstituenz* in Ludwig M. Eichinger and Hans-Werner Eroms (eds.): *Dependenz und Valenz*, Hamburg: Buske, 163-175., 1995

[Zhang *et al*, 2002] T. Zhang, F. Damerau, and D. Johnson. *Text Chunking based on a Generalization of Winnow* In Journal of Machine Learning Research, volume 2 (March), 2002.