Insights into Non-projectivity in Hindi

Prashanth Mannem, Himani Chaudhry, Akshar Bharati
Language Technologies Research Center,
International Institute of Information Technology,
Gachibowli, Hyderabad, India - 500032
{prashanth,himani}@research.iiit.ac.in

Abstract

Large scale efforts are underway to create dependency treebanks and parsers for Hindi and other Indian languages. Hindi, being a morphologically rich, flexible word order language, brings challenges such as handling non-projectivity in parsing. In this work, we look at non-projectivity in Hyderabad Dependency Treebank (HyDT) for Hindi. Non-projectivity has been analysed from two perspectives: graph properties that restrict non-projectivity and linguistic phenomenon behind non-projectivity in HyDT. Since Hindi has ample instances of non-projectivity (14% of all structures in HyDT are non-projective), it presents a case for an in depth study of this phenomenon for a better insight, from both of these perspectives.

We have looked at graph constriants like planarity, gap degree, edge degree and well-nestedness on structures in HyDT. We also analyse non-projectivity in Hindi in terms of various linguistic parameters such as the causes of non-projectivity, its *rigidity* (possibility of reordering) and whether the reordered construction is the *natural* one.

1 Introduction

Non-projectivity occurs when dependents do not either immediately follow or precede their heads in a sentence (Tesnire, 1959). These dependents may be spread out over a discontinuous region of the sentence. It is well known that this poses problems for both theoretical grammar formalisms as well as parsing systems. (Kuhlmann and Möhl, 2007; McDonald and Nivre, 2007; Nivre et al., 2007)

Hindi is a verb final, flexible word order language and therefore, has frequent occurrences of non-projectivity in its dependency structures. Bharati et al. (2008a) showed that a major chunk of errors in their parser is due to non-projectivity. So, there is a need to analyse non-projectivity in Hindi for a better insight into such constructions. We would like to say here, that as far as we are aware, there hasn't been any attempt to study non-projectivity in Hindi before this work. Our work is a step forward in this direction.

Non-projectivity can be analysed from two aspects. a) In terms of graph properties which restrict non-projectivity and b) in terms of linguistic phenomenon giving rise to non-projectivity. While a) gives an idea of the kind of grammar formalisms and parsing algorithms required to handle non-projective cases in a language, b) gives an insight into the linguistic cues necessary to identify non-projective sentences in a language.

Parsing systems can explore algorithms and make approximations based on the coverage of these graph properties on the treebank and linguistic cues can be used as features to restrict the generation of non-projective constructions (Shen and Joshi, 2008). Similarly, the analyses based on these aspects can also be used to come up with broad coverage grammar formalisms for the language.

Graph constraints such as *projectivity*, *planarity*, *gap degree*, *edge degree* and *well-nestedness* have been used in previous works to look at non-projective constructions in treebanks like PDT and DDT (Kuhlmann and Nivre, 2006; Nivre, 2006). We employ these constraints in our work too. Apart from these graph constraints, we also look at non-projective constructions in terms of various parameters like factors leading to non-projectivity, its rigidity (see Section 4), its approximate projective construction and whether its the natural one.

In this paper, we analyse dependency structures in Hyderabad Dependency Treebank (HyDT). HyDT is a pilot treebank containing dependency annotations for 1865 Hindi sentences. It uses the annotation scheme proposed by Begum et al. (2008), based on the Paninian grammar formalism.

This paper is organised as follows: In section 2, we give an overview of HyDT and the annotation scheme used. Section 3 discusses the graph properties that are used in our analysis and section 4 reports the experimental results on the coverage of these properties on HyDT. The linguistic analysis of non-projective constructions is discussed case by case in Section 5. The conclusions of this work are presented in section 6. Section 7 gives directions for future works on non-projectivity for Hindi.

2 Hyderabad Dependency Treebank (HyDT)

HyDT is a dependency annotated treebank for Hindi. The annotation scheme used for HyDT is based on the Paninian framework (Begum et al., 2008). The dependency relations in the treebank are syntactico-semantic in nature where the main verb is the central binding element of the sentence. The arguments including the adjuncts are annotated taking the meaning of the verb into consideration. The participants in an action are labeled with *karaka* relations (Bharati et al., 1995). Syntactic cues like case-endings and markers such as post-positions and verbal inflections, help in identifying appropriate *karakas*.

The dependency tagset in the annotation scheme has 28 relations in it. These include six basic karaka relations (adhikarana [*location*], apaadaan [*source*], sampradaan [*recipient*], karana [*instrument*], karma [*theme*] and karta [*agent*]). The rest of the labels are non-karaka labels like vmod, adv, nmod, rbmod, jjmod etc...[1] The tagset also includes special labels like *pof* and *ccof*, which are not dependency relations in the strict sense. They are used to handle special constructions like conjunct verbs (ex:- *prashna kiyaa* (`question did`)), coordinating conjunctions and ellipses.

In the annotation scheme used for HyDT, relations are marked between chunks instead of words. A chunk (with boundaries marked) in HyDT, by definition, represents a set of adjacent words which are in dependency relation with each other, and are connected to the rest of the words by a single incoming dependency arc. The relations among the words in a chunk are not marked. Thus, in a dependency tree in HyDT, each node is a chunk and the edge represents the relations between the connected nodes labeled with the karaka or other relations. All the modifier-modified relations between the heads of the chunks (inter-chunk relations) are marked in this manner. The annotation is done using Sanchay[2] mark up tool in Shakti Standard Format (SSF) (Bharati et al., 2005). For the work in this paper, to get the complete dependency tree, we used an automatic rule based intra-chunk relation identifier. The rules mark these intra-chunk relations with an accuracy of 99.5%, when evaluated on a test set.

The treebank has 1865 sentences with a total of 16620 chunks and 35787 words. Among these, 14% of the sentences have non-projective structures and 1.87% of the inter-chunk relations are non-projective. This figure drops to 0.87% if we consider the intra-chunk relations too (as all intra-chunk relations are projective). In comparison, treebanks of other flexible word order languages like Czech and Danish have non-projectivity in 23% (out of 73088 sentences) and 15% (out of 4393 sentences) respectively (Kuhlmann and Nivre, 2006; Nivre et al., 2007).

3 Non projectivity and graph properties

In this section, we define dependency graph formally and discuss standard propertiess uch as single headedness, acyclicity and projectivity. We then look at complex graph constraints like gap degree, edge degree, planarity and well-nestedness which can be used to restrict non-projectivity in graphs.

In what follows, a dependency graph for an input sequence of words $x_1 \cdots x_n$ is an unlabeled directed graph $D = (X, Y)$ where X is a set of nodes and Y is a set of directed edges on these nodes. $x_i \rightarrow x_j$ denotes an edge from x_i to x_j, $(x_i, x_j) \in Y$. $\rightarrow^*$ is used to denote the reflexive and transitive closure of the relation. $x_i \rightarrow^* x_j$ means that the node x_i *dominates* the node x_j, i.e., there is a (possibly empty) path from x_i to x_j. $x_i \leftrightarrow x_j$ denotes an edge from x_i to x_j or vice

[1]The entire dependency tagset can be found at http://ltrc.deptagset.googlepages.com/k1.htm

[2]http://sourceforge.net/projects/nlp-sanchay

versa. For a given node x_i, the set of nodes dominated by x_i is the *projection* of x_i. We use $\pi(x_i)$ to refer to the projection of x_i arranged in ascending order.

Every dependency graph satisfies two constraints: acyclicity and single head. *Acyclicity* refers to there being no cycles in the graph. *Single head* refers to each node in the graph D having exactly one incoming edge (except the one which is at the root). While acyclicity and single head constraints are satisfied by dependency graphs in almost all dependency theories. Projectivity is a stricter constraint used and helps in reducing parsing complexities.

Projectivity: If node x_k depends on node x_i, then all nodes between x_i and x_k are also subordinate to x_i (i.e dominated by x_i) (Nivre, 2006).

$$x_i \to x_k \;\; \Rightarrow \;\; x_i \to^* x_j$$

$$\forall x_j \in X : (x_i < x_j < x_k \;\; \vee \;\; x_i > x_j > x_k)$$

Any graph which doesn't satisfy this constraint is *non-projective*. Unlike acyclicity and the single head constraints, which impose restrictions on the dependency relation as such, projectivity constrains the interaction between the dependency relations and the order of the nodes in the sentence (Kuhlmann and Nivre, 2006)..

Graph properties like *planarity*, *gap degree*, *edge degree* and *well-nestedness* have been proposed in the literature to constrain grammar formalisms and parsing algorithms from looking at unrestricted non-projectivity. We define these properties formally here.

Planarity: A dependency graph is *planar* if edges do not cross when drawn above the sentence (Sleator and Temperley, 1993). It is similar to projectivity except that the arc from dummy node at the beginning (or the end) to the root node is not considered.

$$\forall (x_i, x_j, x_k, x_l) \in X,$$

$$\neg((x_i \leftrightarrow x_k \wedge x_j \leftrightarrow x_l) \wedge \; (x_i < x_j < x_k < x_l))$$

Gap degree: The gap degree of a node is the number of gaps in the projection of a node. A gap is a pair of nodes ($\pi(x_i)_k$, $\pi(x_i)_{k+1}$) adjacent in $\pi(x_i)$ but not adjacent in sentence. The gap degree of node $Gd(x_i)$ is the number of such gaps in its projection. The gap degree of a sentence is the maximum among gap degrees of nodes in $D(X,Y)$ (Kuhlmann, 2007).

Edge degree: The number of connected components in the span of an edge which are not dominated by the outgoing node in the edge. Span $span(x_i \to x_j)$ = $(min(i,j), max(i,j))$. $Ed(x_i \to x_j)$ is the number of connected componenets in the span $span(x_i \to x_j)$ whose parent is not in the projection of x_i. The edge degree of a sentence is the maximum among edge degrees of edges in $D(X,Y)$. (Nivre, 2006) defines it as degree of non-projectivity. Following (Kuhlmann and Nivre, 2006), we call this edge degree to avoid confusion.

Well-nested: A dependency graph is well-nested if no two disjoint subgraphs *interleave* (Bodirsky et al., 2005). Two subgraphs are disjoint if neither of their roots dominates the other. Two subtrees S_i,S_j interleave if there are nodes $x_l, x_m \in S_i$ and $x_n, x_o \in S_j$ such that $l < m < n < o$ (Kuhlmann and Nivre, 2006).

The gap degree and the edge degree provide a quantitative measure for the non-projectivity of dependency structures. Well-nestedness is a qualitative property: it constrains the relative positions of disjoint subtrees.

4 Experiments on HyDT

Property	*Count*	*Percentage*
All structures	1865	
Gap degree		
Gd(0)	1603	85.9%
Gd(1)	259	13.89%
Gd(2)	0	0%
Gd(3)	3	0.0016%
Edge degree		
Ed(0)	1603	85.9%
Ed(1)	254	13.6%
Ed(2)	6	0.0032%
Ed(3)	1	0.0005%
Ed(4)	1	0.0005%
Projective	1603	85.9%
Planar	1639	87.9%
Non-projective & planar	36	1.93%
Well-nested	1865	100%

Table 1: Results on HyDT

In this section, we present an experimental evaluation of the graph constraints mentioned in the previous section on the dependency structures in

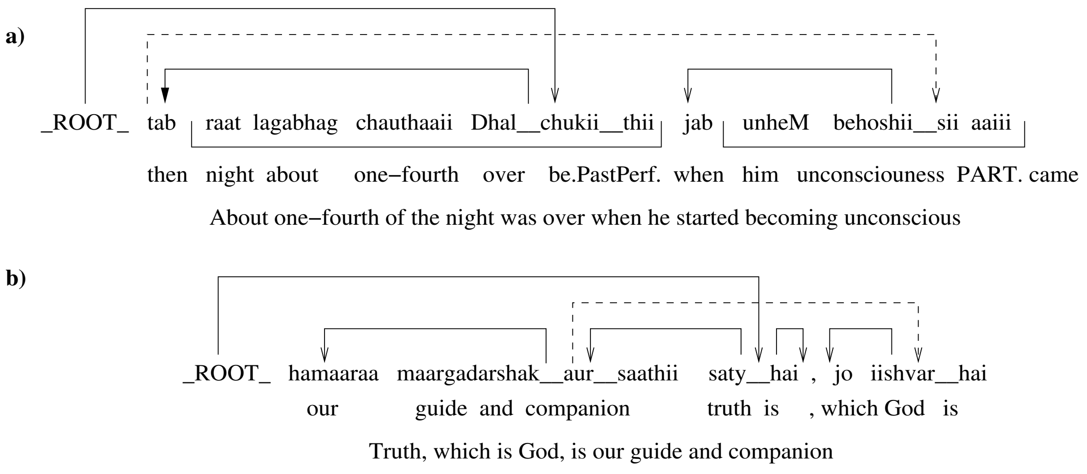

a)
ROOT tab raat lagabhag chauthaaii Dhal__chukii__thii jab unheM behoshii__sii aaiii
then night about one−fourth over be.PastPerf. when him unconsciouness PART. came
About one−fourth of the night was over when he started becoming unconscious
b)
ROOT hamaaraa maargadarshak__aur__saathii saty__hai , jo iishvar__hai
our guide and companion truth is , which God is
Truth, which is God, is our guide and companion

Figure 1: a) Relative co-relative construction, b) Extraposed relative clause construction

HyDT. Since HyDT is a small corpus and is still under construction, these results might not be the exact reflection of naturally occurring sentences in real-world. Nevertheless, we hope these results will give an idea of the kind of structures one can expect in Hindi.

We report the percentage of structures that satisfy various graph properties in table 1. In HyDT, we see that 14% of all structures are non-projective. The highest gap degree for structures in HyDT is 3 and in case of edge degree, it is 4. Only 3 structures (1.5% approx.) have gap degree of more than 1 in a total of 262 non-projective sentences. When it comes to edge degree, only 8 structures (3%) have edge degree more than 1.

The difference in the coverage of gap degree 1 & 2 (and the fact that gap degree 1 accounts for 13.9% of the structures) shows that a parser should handle non-projective constructions at least till gap degree 1 for good coverage. The same can be said about edge degree.

5 Cases of non-projectivity in HyDT

We have carried out a study of the instances of non-projectivity that HyDT brought forth. In this section, we classify these instances based on factors leading to non-projectivity and present our analysis of them. For each of these classes, we look at the *rigidity* of these non-projective constructions and their best projective approximation possible by reordering. Rigidity here is the reorderability of the constructions retaining the gross meaning. *Gross meaning* refers to the meaning of the sentence not taking the discourse and topic-focus into consideration, which is how parsing is typically done.
e.g., the non-projective construction in figure 1b, `yadi rupayoM kii zaruurat thii to mujh ko bataanaa chaahiye thaa`[3] can be reordered to form a projective construction `mujh ko bataanaa chaahiye thaa yadi rupayoM kii zaruurat thii to`. Therefore, this sentence is not rigid.

Study of rigidity is important from natural language generation perspective. Sentence generation from projective structures is easier and more efficient than from non-projective ones. Non-projectivity in constructions that are non-rigid can be effectively dealt with through projectivisation.

Further, we see if these approximations are more *natural* compared to the non-projective ones as this impacts sentence generation quality. A natural construction is the one most preferred by native speakers of that language. Also, it more or less abides by the well established rules and patterns of the language.

We observed that non-projectivity is caused in Hindi, due to various linguistic phenomena manifested in the language, such as relative co-relative constructions, paired connectives, complex coordinating structures, interventions in verbal arguments by non-verbal modifiers, shared arguments in non-finite clauses, movement of modifiers, ellipsis etc. Also, non-projectivity in Hindi can occur within a clause (*intra-clausal*) as well as between elements across clauses (*inter-clausal*).

We now discuss some of these linguistic phenomena causing non-projectivity.

[3]The glosses for the sentences in this section are listed in the corresponding figures and are not repeated to save space.

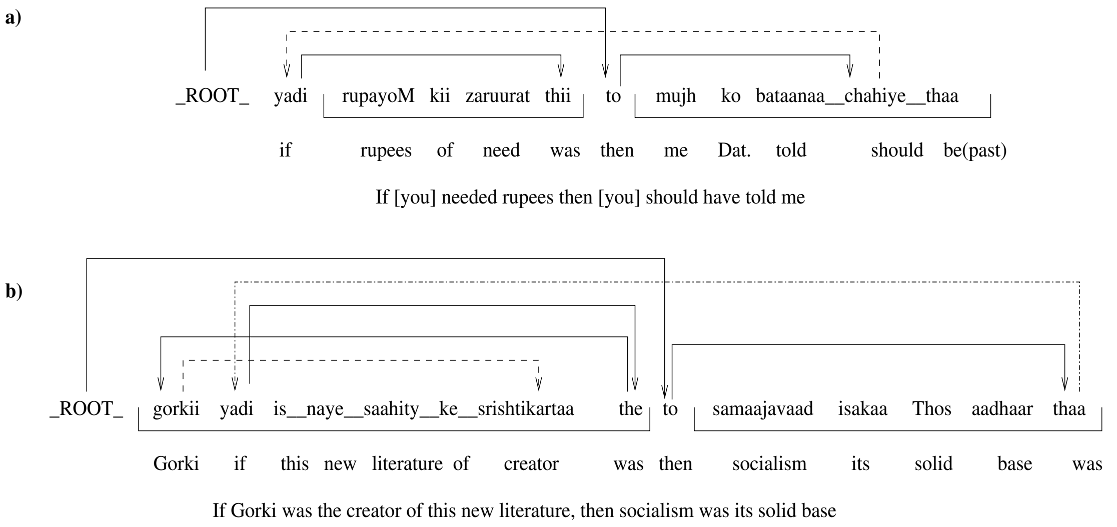

a)
ROOT
yadi
rupayoM kii zaruurat thii
to
mujh ko bataanaa__chahiye__thaa
if rupees of need was then me Dat. told should be(past)
If [you] needed rupees then [you] should have told me
b)
ROOT
gorkii yadi is__naye__saahity__ke__srishtikartaa the
to
samaajavaad isakaa Thos aadhaar thaa
Gorki if this new literature of creator was then socialism its solid base was
If Gorki was the creator of this new literature, then socialism was its solid base

Figure 2: a) Paired connectives construction, b) Construction with non-projectivity within a clause

5.1 Relative co-relative constructions

The pattern in co-relatives is that a demonstrative pronoun, which also functions as determiner in Hindi, such as `vo` (*that*), always occurs in correlation with a relative pronoun, `jo` (*which*). In fact, the language employs a series of such pronouns : e.g., `jis-us` '*which-that*', `jahaaM-vahaaM` '*where-there*', `jidhar-udhar` '*where-there*', `jab-tab` '*when-then*', `aise-jaise` (Butt et al., 2007).

Non-projectivity is seen to occur in relative co-relative constructions with pairs such as `jab-tab`, if the clause beginning with the `tab` precedes the `jab` clause as seen in figure 1a. If the clause with the relative pronoun comes before the clause with the demonstrative pronoun, non-projectivity can be ruled out. So, this class of non-projective constructions is not rigid since projective structures can be obtained by reordering without any loss of meaning. The projective case is relatively more natural than the non-projective one. This is reaffirmed in the corpus where the projective relative co-relative structures are more frequent than the non-projective sentences.

In the example in figure 1a, the sentence can be reordered by moving the `tab` clause to the right of the `jab` clause, to remove non-projectivity.

`jab unheM behoshii sii aaii tab raat lagabhag chauthaaii Dhal chukii thii` – *when he started becoming unconscious, about one-fourth of the night was over*

5.2 Extraposed relative clause constructions

If the relative clause modifying a noun phrase (NP) occurs after the verb group (VP), it leads to non-projectivity.

In the sentence in figure 1b, non-projectivity occurs because `jo iishvar hai`, the relative clause modifying the NP `hamaaraa maargadarshak aur saathii` is extraposed after the VP `saty hai`.

This class of constructions is not rigid as the extraposed relative clause can be moved next to the noun phrase, making it projective. However, the resulting projective construction is less natural than the original non-projective one.

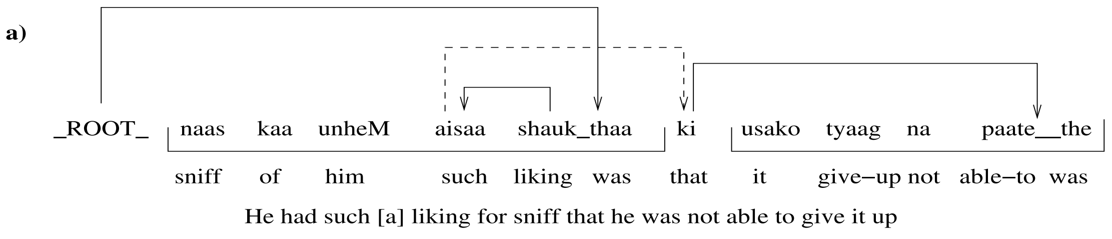
The reordered projective construction for the example sentence is `hamaaraa maargadarshak aur saathii, jo iishvar hai, saty hai` – *Our guide and companion which is God is truth*

This class of non-projective constructions accounts for approximately half of the total non-projective sentences in the treebank.

5.3 Intra-clausal non-projectivity

In this case, the modifier of the NP is a non-relative clause and is different from the class 5.2.

In the example in figure 2b, the NP `gorkii` and the phrase modifying it `is naye saahity ke srishtikartaa` are separated by `yadi`, a modifier of `to` clause. Intra-clausal non-projectivity here is within the clause `gorkii yadi is naye saahity ke srishtikartaa the`.


a)
ROOT
naas kaa unheM aisaa shauk_thaa
ki
usako tyaag na paate__the
sniff of him such liking was that it give−up not able−to was
He had such [a] liking for sniff that he was not able to give it up

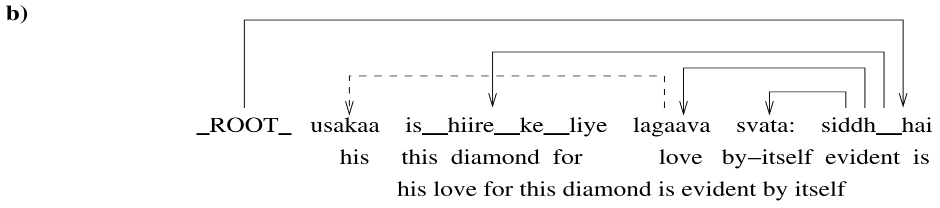

b)
ROOT usakaa is__hiire__ke__liye lagaava svata: siddh__hai
his this diamond for love by−itself evident is
his love for this diamond is evident by itself

Figure 3: a) `ki` complement clause, b) Genetive relation split by a verb modifier

To remove non-projectivity, reordering of such sentences is possible by moving the non-modifier, so that it no more separates them. Here, moving `yadi` to the left of `gorkii` takes care of non-projectivity thus making this class not rigid. The reordered projective construction is more natural.

`yadi gorkii is naye saahity ke srishtikartaa the to samaajavaad isakaa Thos aadhaar thaa`

5.4 Paired connectives

Paired connectives (such as `agar-to` *’if-then’*, `yadi-to` *’if-then’*) give rise to non-projectivity in HyDT on account of the annotation scheme used.

As shown in figure 2a, the `to` clause is modified by the `yadi` clause in such constructions. Most of these sentences can be reordered while still retaining the meaning of the sentence: the phrase that comes after `to`, followed by `yadi` clause, and then `to`. Here mentioning `to` is optional.

This sentence can be reordered and is not rigid. However, the resulting projective construction is not a natural one. `mujh ko bataanaa chaahiye thaa yadi rupayoM kii zaruurat thii [to]` – *(you) should have told me if (you) needed rupees*

Connectives like `yadi` can also give rise to intra-clausal non-projectivity apart from inter-clausal non-projectivity as discussed. This happens when the connective moves away from the beginning of the sentence (see figure 2b).

5.5 `ki` complement clause

A phrase (including a VP in it) appears between the `ki` (*that*) clause and the word it modifies (such as `yaha` (*this*), `asiaa` (*such*), `is tarah` (*such*), `itana` (*this much*)), resulting in non-projectivity in the `ki` complement constructions. The verb in this verb group is generally copular. Since Hindi is a verb final language, the complementiser clause (`ki` clause) occurs after the verb of the main clause, while its referent lies before the verb in the main clause. This leads to non-projectivity in such constructions. The `yaha-ki` constructions follow the pattern: `yaha`-*its property*-VP-`ki` clause.

E.g. `yaha-rahasya-hai-ki shukl jii pratham shreNii ke kavi kyoM the.`

This class of constructions are rigid and non-projectivity can’t be removed from such sentences. In cases where the VP has a transitive verb, the `ki` clause and its referent, both modify the verb, making the construction projective. For ex. In `usane yaha kahaa ki vaha nahin aayegaa`, `yaha` and the `ki` clause both modify the verb `kahaa`.

In figure 3a, the phrase `shauk thaa` separates `aisaa` and the `ki` clause, resulting in non-projectivity.

5.6 A genetive relation split by a verb modifier

This is also a case of intra-clausal non-projectivity. In such constructions, the verb has its modifier embedded within the genetive construction.

In the example in figure 3b, the components of the genetive relation, `usakaa` and `lagaav` are separated by the phrase `is hiire ke liye`.

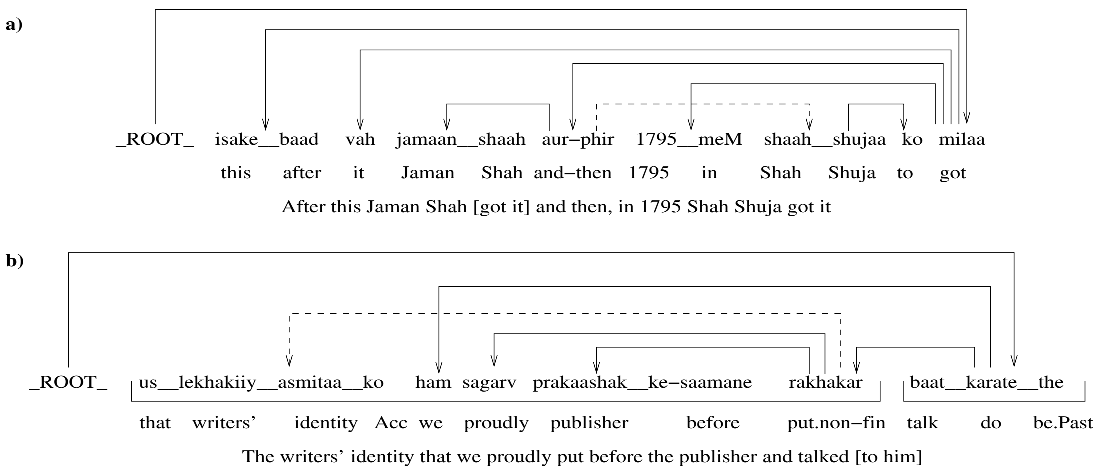

a)
ROOT isake__baad vah jamaan__shaah aur−phir 1795__meM shaah__shujaa ko milaa
this after it Jaman Shah and−then 1795 in Shah Shuja to got
After this Jaman Shah [got it] and then, in 1795 Shah Shuja got it
b)
ROOT us__lekhakiiy__asmitaa__ko ham sagarv prakaashak__ke−saamane rakhakar baat__karate__the
that writers' identity Acc we proudly publisher before put.non−fin talk do be.Past
The writers' identity that we proudly put before the publisher and talked [to him]

Figure 4: a) A phrase splitting a co-ordinating structure, b) Shared argument splitting the non finite clause

The sentence is not rigid and can be reordered to a projective construction by moving the phrase `is hiire ke liye` to the left of `usakaa`. It retains the meaning of the original construction and is also, a more natural one.

`is hiire ke liye usakaa lagaav svata: siddh hai` – *his love for this diamond is evident by itself*

5.7 A phrase splitting a co-ordinating structure

As seen in figure 4a, non-projectivity is caused in the sentence because, embedding of the phrase `1795 meM` splits the co-ordinating structure `jamaan shaah aur-phir shaah shujaa`. These kinds of constructions can be reordered. So, they are not rigid. The projective constructions are more natural.

`isake baad vah jamaan shaah ko aur-phir shaah shujaa ko 1795 meM milaa`

Non-projective Class	*Count*	%
Relative co-relatives constructions	18	6.8 %
Extraposed realtive clause constructions	101	38.0 %
Intra-clausal non-projectivity	12	4.5 %
Paired connectives	33	12.4 %
`ki` complement clauses	52	19.5 %
Genetive relation split by a verb modifier	10	3.8 %
Phrase splitting a co-ordinating structure	4	1.5 %
Shared argument splits the non-finite clause	10	3.8 %
Others	26	9.8 %

Table 2: Non-projectivity class distribution in HyDT

5.8 Shared argument splits the non finite clause

In the example in 4b, `hama` is annotated as the argument of the main verb `baawa karate the`. It also is the shared argument of the non finite verb `rakhakara` (but isn't marked explicitly in the treebank). It splits the non finite clause `us lekhakiiya asmitaa ko` **`ham`** `sagarv prakaashak ke saamane rakhakara`

Through reordering, this sentence can easily be made into a projective construction, which is also the more natural construction for it.

`ham us lekhakiiy asmitaa ko sagarv prakaashak ke-saamane rakhakar baat karate the`

5.9 Others

There are a few non-projective constructions in HyDT which haven't been classified and discussed in the eight categories above. This is because they are single occurences in HyDT and seem to be rare phenomenon. There are also a few instances of inconsistent *NULL* placement and errors in chunk boundary marking or annotation.

6 Conclusion

Our study of HyDT shows that non-projectivity in Hindi is more or less confined to the classes discussed in this paper. There might be more types of non-projective structures in Hindi which may not have occurred in the treebank.

Recent experiments on Hindi dependency parsing have shown that non-projective structures form a major chunk of parsing errors (Bharati et al., 2008a). In spite of using state-of-art parsers which handle non-projectivity, experiments show that the types of non-projectivity discussed in this paper are not handled effectively.

The knowledge of such non-projective classes could possibly be used to enhance the performance of a parser. This work further corroborates Kuhlmann's work on Czech (PDT) for Hindi (Kuhlmann and Nivre, 2006). Specifically, as discussed in section 4, the non-projective structures in HyDT satisfy the constraints (gap degree ≤ 2 and well-nestedness) to be called as mildly non-projective.

7 Future Work

We propose to use the analysis in this paper to come up with non-projective parsers for Hindi. This can be done in more than one ways, such as:

The constraint based dependency parser for Hindi proposed in (Bharati et al., 2008b) can be extended to incorporate graph properties discussed in section 3 as constraints.

Further, linguistic insights into non-projectivity can be used in parsing to identify when to generate the non-projective arcs. The parser can have specialised machinery to handle non-projectivity only when linguistic cues belonging to these classes are active. The advantage of this is that one need not come up with formal complex parsing algorithms which give unrestricted non-projective structures.

As the HyDT grows, we are bound to come across more instances as well as more types of non-projective constructions that could bring forth interesting phenomenon. We propose to look into these for further insights.

References

R. Begum, S. Husain, A. Dhwaj, D. Sharma, L. Bai, and R. Sangal. 2008. Dependency annotation scheme for indian languages. In *In Proceedings of The Third International Joint Conference on Natural Language Processing (IJCNLP)*, Hyderabad, India.

Akshar Bharati, Vineet Chaitanya, and Rajeev Sangal. 1995. *Natural Language Processing: A Paninian Perspective*. Prentice-Hall of India.

Akshar Bharati, Rajeev Sangal, and Dipti Sharma. 2005. Shakti analyser: Ssf representation. Technical report, International Institute of Information Technology, Hyderabad, India.

Akshar Bharati, Samar Husain, Bharat Ambati, Sambhav Jain, Dipti Sharma, and Rajeev Sangal. 2008a. Two semantic features make all the difference in parsing accuracy. In *Proceedings of the 6th International Conference on Natural Language Processing (ICON-08)*, Pune, India.

Akshar Bharati, Samar Husain, Dipti Sharma, and Rajeev Sangal. 2008b. A two-stage constraint based dependency parser for free word order languages. In *Proceedings of the COLIPS International Conference on Asian Language Processing 2008 (IALP)*, Chiang Mai, Thailand.

Manuel Bodirsky, Marco Kuhlmann, and Mathias Mhl. 2005. Well-nested drawings as models of syntactic structure. In *In Tenth Conference on Formal Grammar and Ninth Meeting on Mathematics of Language*, pages 88–1. University Press.

M. Butt, T. H. King, and S. Roth. 2007. Urdu correlatives: Theoretical and implementational issues. In *Online Proceedings of the LFG07 Conference*, pages 87–106. CSLI Publications.

Marco Kuhlmann and Mathias Möhl. 2007. Mildly context-sensitive dependency languages. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 160–167, Prague, Czech Republic, June. Association for Computational Linguistics.

Marco Kuhlmann and Joakim Nivre. 2006. Mildly non-projective dependency structures. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, pages 507–514, Sydney, Australia, July. Association for Computational Linguistics.

Marco Kuhlmann. 2007. *Dependency Structures and Lexicalized Grammars*. Ph.D. thesis, Saarland University.

Ryan McDonald and Joakim Nivre. 2007. Characterizing the errors of data-driven dependency parsing models. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 122–131, Prague, Czech Republic, June. Association for Computational Linguistics.

Joakim Nivre, Johan Hall, Sandra Kübler, Ryan McDonald, Jens Nilsson, Sebastian Riedel, and Deniz Yuret. 2007. The CoNLL 2007 shared task on dependency parsing. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*, pages 915–932, Prague, Czech Republic, June. Association for Computational Linguistics.

Joakim Nivre. 2006. Constraints on non-projective dependency parsing. In *In Proceedings of European Association of Computational Linguistics (EACL)*, pages 73–80.

Libin Shen and Aravind Joshi. 2008. LTAG dependency parsing with bidirectional incremental construction. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 495–504, Honolulu, Hawaii, October. Association for Computational Linguistics.

Daniel Sleator and Davy Temperley. 1993. Parsing english with a link grammar. In *In Third International Workshop on Parsing Technologies*.

L. Tesnire. 1959. *lments de Syntaxe Structurale*. Libraire C. Klincksieck, Paris.