Towards a psycholinguistically motivated dependency grammar for Hindi

Samar Husain
Department of Linguistics
Universität Potsdam
Germany
husain@uni-potsdam.de

Rajesh Bhatt
University of Massachusetts
Amherst, MA , USA
bhatt@linguist.umass.edu

Shravan Vasishth
Department of Linguistics
Universität Potsdam
Germany
vasishth@uni-potsdam.de

Abstract

The overall goal of our work is to build a dependency grammar-based human sentence processor for Hindi. As a first step towards this end, in this paper we present a dependency grammar that is motivated by psycholinguistic concerns. We describe the components of the grammar that have been automatically induced using a Hindi dependency treebank. We relate some aspects of the grammar to relevant ideas in the psycholinguistics literature. In the process, we also extract statistics and patterns for phenomena that are interesting from a processing perspective. We finally present an outline of a dependency grammar-based human sentence processor for Hindi.

1 Introduction

Human sentence processing proposals and modeling works are overwhelmingly based on phrase-structure parsing and constituent based representation. This is because most modern linguistic theories (Chomsky, 1965), (Chomsky, 1981), (Chomsky, 1995), (Bresnan and Kaplan, 1982), (Sag et al., 2003) use phrase structure representation to analyze a sentence. There is, however, an alternative approach to sentential syntactic representation, known as dependency representation, that is quite popular in Computational Linguistics (CL). Unlike phrase structures where the actual words of the sentence appear as leaves, and the internal nodes are phrases, in a dependency grammar (Mel'čuk, 1988), (Bharati et al., 1995), (Hudson, 2010) a syntactic tree comprises of sentential words as nodes. These words/nodes are connected to each other with edges/arcs. The edges can be labeled to show the type of relation between a pair of node. For example, in the sentence *John kissed Mary*, 'John' and 'Mary' are connected via arcs to 'kissed'; the former arc bears the label 'subject' and the latter arc the label 'object'. Taken together, these nodes with their connections form a tree. Figure 1 shows the dependency and the phrase structure trees for the above sentence.

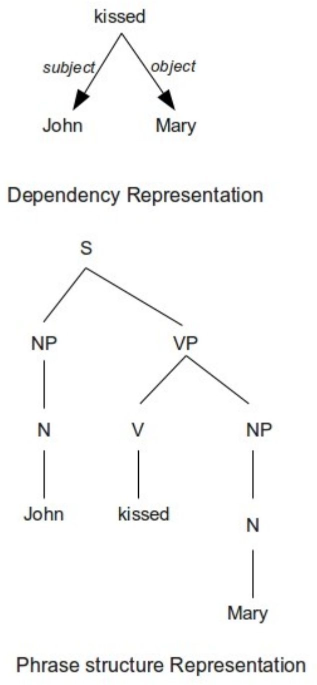

Figure 1: Dependency tree and Phrase structure tree

There have been some previous attempts to use lexicalized grammars such as LTAG, CCG, etc. in psycholinguistics. These lexicalized grammars have been independently shown to be related to dependency grammar (Kuhlmann, 2007). For example, Pickering and Barry (1991) used categorial grammar to handle processing of empty categories. Similarly, Pickering (1994) used dependency categorial grammar to process both local and non-local dependencies. More recenly, Tabor and Hutchins (2004) used a lexical grammar based parser and Kim et al. (1998) used lexicalized tree-adjoining grammar (LTAG) to model certain processing results. Demberg (2010) has recently proposed a psycholinguistically motivated LTAG (P-LTAG). Despite the success of dependency paradigm in CL, it has remained unexplored in psycholinguistics. To our knowledge, the work by Boston and colleagues (Boston et al., 2011), (Boston et al., 2008) is the only such attempt. There are some very interesting open questions with respect to using dependency representation and dependency parsers while building a human sentence processing system. Can a processing model based on dependency parsing paradigm account for classic psycholinguistic phenomena? Can one adapt a high performance dependency parser for psycholinguistic research? If yes, then how? How will the differences in different dependency parsing paradigms affect the predictive capacity of the models based on them?

This paper is arranged as follows, in Section 2 we mention some experimental works that have motivated the grammar design. Section 3 discusses the grammar induction process and lists out the main components of the grammar. In Section 4 we present statistics of Hindi word order variations as found in the treebank and point out some patterns that are interesting from a processing perspective. We also talk about prediction rules that are an important component in the grammar. Section 5 then presents a proposal for developing a human sentence processing system by adapting graph-based dependency parsing. Finally in Section 6 we discuss some issues and challenges in using dependency grammar paradigm for human sentence processing. We conclude in Section 7.

2 Motivation: Some relevant experimental work

Some crucial design decisions in our research have been inspired by psycholinguistic experimental work. In this section we will mention these works. But before that, we will briefly discuss Hindi, the language that we are working with.

Hindi is one of the official languages of India. Hindi is the fourth most widely spoken language in the world[1]. It is a free-word order language and is head final. It has relatively rich morphology with verb-subject[2], noun-adjective agreement. Examples (2) to (6) below show some of the possible word order variations possible with (1). These permutations are not exhaustive - in fact all 4! permutations are possible.

(1)	malaya	ne	abhiisheka	ko	kitaaba	dii
	Malaya	ERG	Abhishek	DAT	book	gave

‘Malaya gave a book to Abhishek.’ (S-IO-O-V)

(2) malaya ne kitaaba abhiisheka ko dii (S-O-IO-V)

(3) abhiisheka ko malaya ne kitaaba dii (IO-S-O-V)

(4) abhiisheka ko kitaaba malaya ne dii (IO-O-S-V)

(5) kitaaba abhiisheka ko malaya ne dii (O-IO-S-V)

(6) kitaaba malaya ne abhiisheka ko dii (O-S-IO-V)

A great deal of experimental research has shown that working-memory limitations play a major role in sentence comprehension difficulty (e.g., Lewis and Vasishth (2005)). We find numerous instances in natural language where a word needs to be temporarily retained in memory before it can be integrated as part of a larger structure. Because of limited working-memory, retaining a word for a longer period can make sentence processing difficult. Abstracting away from details, on this view, one way in which processing complexity can be formulated is by using metrics that can incorporate dependent-head distance (Gibson, 2000), (Grodner and Gibson, 2005). This idea manifests itself in various forms in the psycholinguistics literature. For example, Gibson (2000) proposes integration cost and storage cost to account for processing complexity. Lewis and Vasishth (2005) have proposed a working memory-based theory that uses the notion of decay as one determinant of memory retrieval difficulty. Elements that exists in memory without being retrieved for a long time will decay more, compared to elements that have been retrieved recently or elements that are recent. In addition to decay, the theory also incorporates the notion of interference. Memory retrievals are feature based, and feature overlap during retrieval, in addition to decay, will cause difficulty.

As opposed to locality-based accounts mentioned above, expectation-based theories appeal to the predictive nature of sentence processor. On this view, processing becomes difficult if the upcoming sentential material is less predictable. Surprisal (Hale, 2001), (Levy, 2008) is one such account. Informally, surprisal increases when a

[1](http://www.ethnologue.com/ethno_docs/distribution.asp?by=size).

[2]This is the default agreement pattern. The complete agreement system is much more complex.

parser is required to build some low-probability structure.

Expectation-based theories can successfully account for so called anti-locality effects. It has been noted that in some language phenomena, increasing the distance between the dependent and its head speeds up reading time at the head (see Konieczny (2000) for such effects in German, and Vasishth and Lewis (2006) for Hindi). This is, of course, contrary to the predictions made by locality-based accounts where such an increase should cause slowdown at the head. There is considerable empirical evidence supporting the idea of predictive parsing in language comprehension (e.g. Staub and Clifton (2006)). There is also some evidence that shows that the nature of predictive processing can be contingent on language specific characteristics. For example, Vasishth et al. (2010) argue that the verb-final nature of German subordinate clauses leads to the parser maintaining future predictions more effectively as compared to English. As Hindi is a verb-final language, these experimental results become pertinent for this paper. Locality-based results are generally formalized using the limited working memory model. Such a model enforces certain resource limitations within which human sentence processing system operates. On the other hand, expectation/prediction-based results have to be accounted by appealing to the nature of the processing system itself.

Hindi being a free word order language, experimental work that deal with the processing cost of word-order variation is also important to us. Experimental work points to the fact that human sentence processing is sensitive to word order variation (e.g. Bader and Meng (1999), Kaiser and Trueswell (2004)). However, it is still not clear as to why/how word order variation influences processing costs. Processing costs could be due to variety of reasons (such as, syntactic complexity, frequency, information structure, prosody, memory constraints, etc).

So, there are three streams of experimental research that are relevant for us: (a) locality effects, (b) anti-locality/expectation effects, (c) word-order variation effects. In section 3 and 4 we will discuss how insights from (b) and (c) inform some of our design decisions. Later in section 5 while discussing human sentence processing, we will touch upon the notion of locality in our parsing approach.

3 Inducing a grammar

To develop a dependency grammar we will make use of an already existing Hindi dependency treebank (Bhatt et al., 2009). The treebank data is a collection of news articles from a Hindi newspaper and has 400k words. The task of automatic induction of grammar from a treebank can be thought of as making explicit the implicit grammar present in the treebank. This approach can be beneficial for a variety of tasks, such as, complementing traditional hand-written grammars, comparing grammars of different languages, building parsers, etc. (Xia and Palmer, 2001), (Kolachina et al., 2010). Our task is much more focused, we want to bootstrap a grammar that can be used for a dependency-based human sentence processor for Hindi.

3.1 Lexicon

The lexicon comprises of syntactic properties of various heads (e.g. verbs). Based on *a priori* selection of certain argument relations (subject, object, indirect object, experiencer verb subject, goal, noun complements of subject for copula verbs) we formed around 13 verb clusters[3]. These clusters were then merged into 6 super-clusters based on the previously mentioned relations (this time acting as discriminators[4]). These clusters correspond to, (1) intransitive verbs (e.g. *so* 'sleep', *gira* 'fall'), (2) transitive verbs (e.g. *khaa* 'eat'), (3) ditransitives (e.g. *de* 'give'), (4) experiencer verbs (e.g. *dikha* 'to appear'), (5) copula (e.g. *hai* 'is'), (6) goal verbs (e.g. *jaa* 'go').

These 6 verb classes can be thought of as tree-templates and can be associated with various class specific constraints such as, number of mandatory arguments, part of speech, category of the arguments, canonical order of the arguments, relative position of the argument with respect to the verb, agreement constraints, etc. Figure 2 shows a simplified transitive template that can be associated with all the transitive verbs in the lexicon. Its

[3]Clustering originally gave us 31 classes that were manually checked to give us 13 correct classes. Although this filtering was done manually, most of the remaining 18 clusters could have also been identified automatically. The proportion of verbs associated with them is very low. These clusters are mainly due to annotation errors.

[4]Each of these clusters were then compared to each other in order to remove common verbs.

first argument (subject) is represented by a variable *i*, and its second argument (object) is represented by a variable *j*. The verb itself is shown as variable *x*. These variables (*i*, *j*, *x*) will be instantiated by those lexical items that are of a particular type. For example, *x* can only be instantiated by transitive verb class members[5]. Similarly, only those items that satisfy the dependent constraints can be instantiated as subject or object. These constraints can be of various kinds, such as the part of speech (POS) category, semantic type, etc. The tree-template in figure 2 also encodes the arity of the verbal head, as well as the canonical word order of its dependents (cf. figure 3). Lastly, the tree-template shows that adjuncts are not mandatory. Figure 3 shows a template for a transitive argument structure but with object fronting. In the figure, object is indexed with *j*, and its canonical position is shown by $\emptyset_j$. Note that the arc coming into the empty node ($\emptyset_j$) is not a dependency relation; the arc and the node are just a representational strategy to show word order variation in the tree-template.

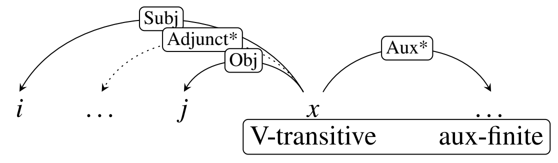

Figure 2: A simplified transitive tree-template. aux = Auxiliaries. * signifies 0 or more instances.

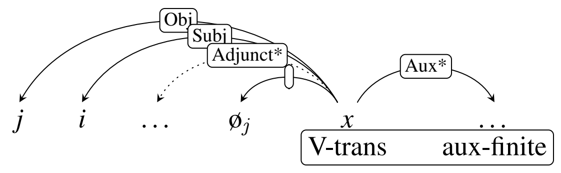

Figure 3: A simplified transitive tree template showing object fronting. trans = Transitive, aux = Auxiliaries, $\emptyset_j$ = canonical position of object *j*

3.2 Frame variations

The tree-templates we saw in section 3.1 have been induced automatically using the finite verb occurrences in the treebank. While inducing the clusters we neglected the differences in tense, aspect and modality (TAM) of the verbs (e.g. perfective, obligational) that sometime leads to different case-markings on the arguments. This is because we are focusing on the number of arguments, not the case-markings on the arguments. But, finite-templates cannot be used for non-finite verbs. This is because the surface requirements of non-finite verbs are different from that of finite verbs[6]. For example, when *khaa* 'eat' occurs as *khaakara* 'having eaten', its original requirement for a subject changes to mandatorily not taking any visible subject. In addition, it requires another finite or a non-finite verb as its head.

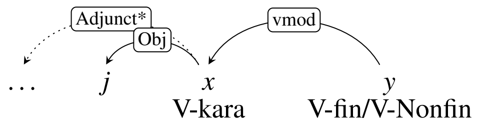

Figure 4: A -kara tree-template. fin = Finite.

One way to think about *-kara* template is that it is an outcome of transforming the transitive argument structure (Bharati et al., 1995). The transformation will perform a series of add/modify/delete operations on the transitive-template leading to the template shown in figure 4.

Another way to think about this would be basically a non-transformational account, where we assume that the non-finite templates are obtained independently of the transitive templates. This would amount to looking at only the non-finite verbs in the treebank and identifying some generalization about various non-finite instances, i.e. creating classes based on the inflections such *kara* (encodes causality/sequentiality), *e (hii)* (sequentiality), *taa huaa* (signifies co-occurring event), *naa* (gerundive form), etc.

From a grammar extraction perspective, the method that is used to get the non-finite templates is not that important. As long as one can extract the correct requirements of the non-finite verbs, it should suffice. On the other hand, such a distinction could, in fact, be very relevant from a lexical processing perspective. Do we build the *kara* template on the fly or do we just access its information from a stored entry? Such a design decision will have to await future investigation.

[5]Only finite verbs were considered to form the verb clusters. Syntactically, non-finite verbs in Hindi behave differently than their finite counterpart. This is not only reflected in the inflection, but also in the number of visible arguments. We discuss such cases in Section 3.2.

[6]This also holds true for passives.

3.3 Probability distribution of dependency types

Each verb class (*i*) is associated with a probability distribution of its dependents (j=1..n). The probability of a node (*x*) being a dependent of *i* with relation *r*, is computed as:

$$\mathrm{P}_{x,r,i} = \frac{\lambda_{(x,r,i)}}{\sum_{j=1}^{n} \lambda_{(j,i)}} \quad (1)$$

where $\lambda_{(x,r,i)}$ is the count of i $\rightarrow$ x (with dependency relation *r*), and the denominator signifies the total count of all the dependents of *i*.

Such probabilities can be used to score a dependency tree at any given point during the parsing process. This is of course a simplification and as we will note in section 5, there are other ways to induce the probability model that can be used to score a dependency tree.

The dependency grammar that we have been building will be used to model a human sentence parser. The incrementality of the parser and the observation that many nouns can appear without any post-position makes it necessary to identify if the two nouns appearing together are collocations or independent arguments.

One way to compute the collocational strength between words x' and y' is by using pointwise mutual information (Manning and Schütze, 1999):

$$\mathrm{I}(x', y') = log\frac{p(x', y')}{p(x')p(y')} \quad (2)$$

Again, as we will see in section 5, there might be other ways in order to incorporate this knowledge for parsing purposes.

3.4 Prediction rules

As each incoming word is incorporated into the existing structure, predictions are made about upcoming words based on current information. In order for the parser to proceed in such a fashion it must have ready access to such information. The grammar that we propose provides this information in the form of prediction rules. The kind of information that the (implemented) parser utilizes to make such predictions can be influenced by various theoretical assumptions (and/or experimental results). For illustrative purpose, while gathering statistics to formulate predictions shown in this section, we consider only verbal arguments. The presence of adjuncts has been neglected.

We begin with one simple cue, case of the arguments. Considering the occurrence of the first verbal argument in a sentence and its case, we tried to predict the verb class using the data in the Hindi dependency treebank. Here are some predictions based on frequent case-markings: *ne* (ERG) $\rightarrow$ transitive, *ko* (ACC) $\rightarrow$ transitive, *se* (INST) $\rightarrow$ ditransitive, *0* (NOM) $\rightarrow$ intransitive.

The verb classes that we get for *ne*, *se* and *0* reflect the default distribution of ERG, INST and NOM case-markers vis-à-vis the type of verbs they tend to occur with. Of course, predictions will become more precise as more words are processed. When the first two argument case-markers are considered we get:

0 0:	copula
0 -:	intransitive
0 se:	transitive
0 ko:	transitive
0 ne:	transitive
ne 0:	transitive
ne ko:	transitive/ditransitive
ne se:	ditransitive
ko 0:	ditransitive
ko ko:	ditransitive
ko se:	ditransitive
ko ne:	transitive
ko -:	transitive
se 0:	ditransitive
se ne:	ditransitive

And as we get more information, we might have to revise our previous predictions and make necessary structural changes (or rerank multiple structures). For example, the first *ko* occurrence predicts a transitive-template, but that is later revised to a ditransitive if we happen to see a *0* or a *ko* case-marker.

The prediction rules shown above have been automatically extracted from the dependency treebank. Other than the case-marker, one can also use other features to make our predictions more realistic. For example, we could use features such as sentence position, animacy feature (using a resource such as WordNet), etc.

4 Processing concerns

Having discussed the main components of the grammar, in this section, we will raise some processing concerns based on the statistics gathered from the treebank.

4.1 Canonical and non-canonical structures

4.1.1 Argument structure variation

As mentioned previously, the tree-template for each verb class also encodes the canonical order in which its argument should appear. For example, in a transitive class, it is expected that the subject should precede the object, and that the object should immediately precede the verb. Such word order information can be extracted from the treebank.

Reflecting each verb class, following word orders were extracted from the treebank:

Transitive
- *Subj Obj*
- *Subj Obj-cm* (cm=case-marker; *ko* (ACC), *se* (INST))

Ditransitive
- *Subj Indirect-Obj Obj*
- *Subj Indirect-Obj Obj-cm*
- *Subj Indirect-Obj Obj-LOC*
- *Indirect-Obj Obj* (missing Subj)
- *Indirect-Obj Obj-LOC* (missing Subj)
- *Subj Indirect-Obj* (missing Obj)

Experiencer verbs
- *DAT NOM*

Copula
- *Subj Noun-complement*: Copula

Others
- *Subj Obj-LOC*: Verbs such as *jaa* 'go'
- *Object* verb collocation (whether the object appears immediately before the verb)
- *Object* verb order (the relative position of the object with respect to the verb; left or right)

Based on the above patterns, following are some main trends:

- Close to 11% of argument structures are non-canonical,
- Non-canonical order for "Subj Obj", "Subj Noun-complement" (for copula verbs) is rare; "Obj Subj" = 6.7% out of all "Subj Obj" instances, while "Noun-complement Subj" = 3.6% out of all "Subj Noun-complement" instances,
- When Obj is not case-marked it is more likely to appear next to the verb (82.2%), than when it is case-marked (47.2%)
- Total number of Obj appearing after the verb is extremely rare (.35% of all object verb instances)[7] (this is not considering clausal objects that occur with verbs such as *kaha* 'say').

The canonical order is encoded in the tree-template and non-canonical word order is therefore reflected in a separate tree. We can see this in the representations in figure 2 and figure 3. Figure 2 represents the canonical order and figure 3 the order with object fronting. To reflect that this is a non-canonical word order, a null node is also shown in object's canonical pre-verbal position and indexed to the fronted object.

As just noted, non-canonical word order due to changes in argument structure order is not so frequent. The occurrence of non-canonical word order has been attributed to information structure constraints. For example, (Butt and King, 1996) have argued for the following word position - discourse function mapping for Urdu/Hindi: Sentence initial → TOPIC, Pre-verbal (immediate) → FOCUS, Post-verb → BACKGROUND INFORMATION, Pre-verbal (others) → COMPLETIVE INFORMATION.

Other related work on Hindi word order (e.g. Kidwai (2000)[8], Gambhir (1981), Kachru (2006)) also have a discourse-centric explanation. There is some work that has investigated word order variation effects during sentence processing. Vasishth et al. (2012), Vasishth (2004) investigated the effect of discourse on word order variation, while Patil et al. (2008) looked at effects of word order and information structure on intonation.

4.1.2 Non-projective dependencies

A dependent and its head can sometimes be discontiguous; the constraint that the head-dependent pair is contiguous is called the projectivity constraint. More formally, an arc i → j is projective if, for node *i* with *j* as its child, any node *k*, such that i < k < j (or i > k > j) is dominated by i (i →* k) (Nivre and Nilsson, 2005).

The dependency treebank being used has 3755 non-projective arcs. Amongst these, intra-clausal dependencies related to verbal heads account for 16.2% of non-projective arcs and 5% of such dependencies are due to intra-clausal dependencies with nominal head. While relative clauses (RCs)

[7]We note here that this pattern and the fact that non-canonical structures are rare, might be because our corpus is a written corpus. Spoken Hindi has more postverbal material and non-canonical structures than written Hindi.

[8]Kidwai (2000) has a syntactic explanation but her syntactic features have discourse motivations (topic, focus etc.).

account for close to 25% of all non-projective arcs. For a more detailed classification of non-projective dependencies in Hindi, see Mannem et al. (2009).

Embedded relative clauses and correlatives with canonical word order lead to projective structures, while right extraposed relative clauses such as the one shown in figure 5 are non-projective. A right extraposed relative clause can also be projective, this can happen in certain non-canonical configurations and is quite rare (see Table 1).

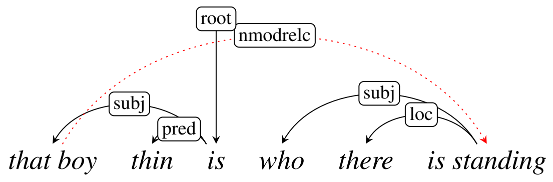

Figure 5: Right extraposed relative clause (non-projective).

Type	Projective	Non-projective	Total
Embedded	2.4	.2	2.6
Correlative	17.3	.5	17.8
Right extraposed	2.5	76.7	79.2

Table 1: Relative clause types (Occurrence in %). Total RC count = 1198.

Recently, Levy et al. (2012) have shown that extraposed RC structures in English are difficult to process. Such structures in English are non-projective and are quite rare[9]. As opposed to this, right-extraposed RCs in Hindi are the most frequently occurring structures amongst all relative clause types (cf. table 1). Like English, these structures are also non-projective in Hindi. The question then arises as to whether these non-projective structure are also difficult to process in Hindi, and if yes, why Hindi speakers prefer such configurations, as opposed to a projective structure of embedded relative clause or correlatives?[10]

At this point in time, we can pose the following questions:
- What is the difference between the processing of a canonical structure and its non-canonical counterpart in Hindi?
- How can we quantify this difference?

To answer these questions one needs to conduct targeted experiments. We need to investigate these questions because it will help us make more informed decisions to implement the grammar and the sentence processor. For instance, if it turns out that processing non-projective relative clause structures in Hindi is easy, then what does it say about the parser adaptability based on specific language patterns? And how will that knowledge affect the design of the parser?

4.2 Prediction rules

Given the observation that predictions made by the parser will go wrong and the parser will have to make revisions (or rerank), we need to ask:

- What is predicted?
- What are the different cues that are pooled to make a prediction?
- What is the processing cost when a prediction is incorrect?
- How can we quantify this cost?
- How does the prediction system interact with other aspects of the comprehension process?

Table 2 shows how our predictions based on case-markers (when the first two arguments have been seen) can vary in terms of correctness in word order and verb class. For example, after the 1st *ko* is seen, it predicts a canonical transtive-template, this prediction changes to non-canonical transitive template in case *ne* happens to be the next case-marker; on the other hand if a *0* case-marker was encountered instead, the parser revises its prediction to a canonical ditransitive-template. These predictions have been made before arriving at the verb, this means, sometimes these predictions could be incorrect. In such a case, the verb (or additional arguments) itself will eventually help make the final revision.

So, based on the above discussion, there are 2 factors that will influence the processing cost of a prediction:

- Correct/Incorrect verb-template prediction,
- Correct/Incorrect word-order prediction

[9]Table 1 in Levy et al. (2012), P(extraposedRC|context) is 0.00004, while P(RC|context) is 0.00561.

[10]Recently, Kothari (2010) in a judgement study found no effect of discourse in Hindi native speakers' preference for correlatives over right extraposed RCs. In another judgement study involving non-restrictive RCs, Kothari shows that Hindi native speakers prefer embeddings for short RCs while right extrapositions are preferred when the RC is long.

Prediction	CO	NCO
Correct prediction	Predicted → *0 0*: copula Correct → *0 0*: copula	Predicted → *ko ne*: transitive Correct → *ko ne*: transitive
Incorrect prediction (incorrect class)	Predicted → *0 0*: copula Correct → *0 0*: transitive	Predicted → *ko ne*: transitive Correct → *ko ne*: ditransitive
Incorrect prediction (incorrect word order)	Predicted → *ko ko*: ditransitive Correct → *ko ko*: ditransitive (NCO)	Predicted → ? Correct → ?
Incorrect prediction (incorrect class and word order)	Predicted → *ko 0*: ditransitive Correct → *ko 0*: transitive (NCO)	Predicted → ? Correct → ?

Table 2: Different prediction scenarios. Canonical order: CO, Non-canonical order: NCO

Based on the presented grammar design, the processing hypothesis about the cost of such a prediction is:

Correct prediction < Incorrect prediction (argstr order or verb class) < Incorrect prediction (argstr and class)

This hypothesis will of course need to be evaluated experimentally.

5 An outline of human sentence processing using dependency parsing

We will adapt the graph-based dependency parsing paradigm (Kübler et al., 2009) to model human sentence processing. The parser will be used to compute certain cognitive measures (such as surprisal, retrieval cost; cf. Boston et al. (2008), Demberg and Keller (2008)) that will in turn be used to predict processing difficulty.

Graph-based parsing data-driven models parameterizes directly on subtrees. Arc-factored models that only exploit single head-child node pair will be implemented. The parsing algorithm comprises of finding a maximal spanning tree (MST) out of a complete graph using the arc parameters. Note here that this formulation of the parsing algorithm (McDonald et al., 2005a), (McDonald et al., 2005b) needs to be modified in order to adapt it for the goals of this paper. In particular, the algorithm needs to be incremental. It is easy to see how this can be done. Instead of starting with the complete sentence, one needs to form complete graphs out of all the available words. If the length of the sentence is n, this will involve extracting an MST n-1 times, i.e., after hearing/reading each word. By doing so, the worst case complexity of the algorithm remains unchanged. Another modification that needs to be incorporated is the use of prediction rules within the parsing process; this will involve forming complete graphs using unlexicalized tree-template (that will be predicted by already seen tokens), and extracting MST out of it. The other important task after implementing the parser will be to use the parser to compute certain measures (such as surprisal, locality-based costs). These measures can then be used to predict processing difficulty. Within the graph-based parsing paradigm, a probability model can be induced using the method proposed by McDonald et al. (2005a), McDonald et al. (2005b)[11]. Once we have such a probability model, surprisal and locality-based costs can then be computed.

To the best of our knowledge the work by Boston and colleagues (Boston et al., 2008), (Boston et al., 2011) is the only other work that has employed dependency parsing to model human sentence processing difficulty. Unlike what has been proposed here, they used a transition-based dependency parsing model (Nivre, 2003). However, their parser will not be able to correctly analyse crossing/discontiguous dependencies. In addition, they have no notion of prediction explicitly built into their system.

The other work that bears similarity with our work is that of Demberg (2010). Demberg (2010), unlike us, used a variant of lexicalized tree-adjoining grammar (a psycholinguistically motivated LTAG called P-LTAG). And although, LTAG is related to dependency grammars (Kuhlmann, 2007), the choice of grammar has a considerable impact on the parsing system that one employs to model processing difficulty. Our predicted tree template looks similar to the prediction tree of Demberg (2010). But again, the operations and mechanisms that will be employed by us to construct the syntactic structure will be influenced by the constraints put in by the properties of dependency grammar and the graph-based parsing algorithm, and will be significantly different from the P-LTAG based operations such as substitution, adjunction (and verification).

6 Issues and Challenges

Our dependency grammar based human sentence processing system presents itself as an attractive alternative to phrase structure based models currently dominant in the psycholinguistic literature. This is because of its representational simplicity and availability of efficient dependency

[11]Such a probability model can be used as an alternative to the one mentioned in section 3.3

parsing paradigms. It seems to be well suited to model expectation-based psycholinguistic theories. However, there are certain issues that will need to be eventually addressed in order for the dependency model to have comprehensive coverage.

The first issue is related to the representational aspect of dependency structures. There is considerable evidence that while processing some dependencies, for example, filler-gap dependencies, anaphora resolution, etc., human sentence comprehension system uses certain grammatical constraints (Phillips et al., 2011). These constraints (e.g. c-command) have been traditionally formalized using phrase-structure representation. If it is true that the parser does employ configurational constraints such as c-command then it will be imperative to formulate a functionally equivalent definition of c-command within the dependency framework.

The second issue is related to parser adaptation. Adapting the graph-based dependency parser in order to effectively compute the cognitive measures will be the most challenging task of this work. In particular, the same parser has to be conceptualized to compute both locality-based as well as expectation-based measures (Boston et al., 2008). In addition, the prediction system needs to be seamlessly integrated within the parsing process.

7 Conclusion

In this paper we introduced our work towards building a psycholinguistically motivated dependency grammar for Hindi. We outlined the main components of such a dependency grammar that was automatically induced using a Hindi dependency treebank. We discussed certain language patterns that were interesting psycholinguistically. We sketched how a graph-based dependency parser can be used to model sentence processing difficulty. We finally mentioned some issues with using a dependency based human sentence processing model.

References

M. Bader and M. Meng. 1999. Subject-Object Ambiguities in German Embedded Clauses: An Across-the-Board Comparison. *Journal of Psycholinguistic Research.*, 28(2):121–143.

A. Bharati, V. Chaitanya, and R. Sangal. 1995. *Natural Language Processing: A Paninian Perspective*. Prentice-Hall of India, New Delhi.

R. Bhatt, B. Narasimhan, M. Palmer, O. Rambow, D. Sharma, and F. Xia. 2009. A Multi-Representational and Multi-Layered Treebank for Hindi/Urdu. In *Proceedings of the Third LAW*, ACL-IJCNLP '09, pages 186–189.

M. F. Boston, John T. Hale, U. Patil, R. Kliegl, and S. Vasishth. 2008. Parsing costs as predictors of reading difficulty: An evaluation using the Potsdam Sentence Corpus. *Journal of Eye Movement Research*, 2(1):1–12.

M. F. Boston, J. T. Hale, S. Vasishth, and R. Kliegl. 2011. Parallel processing and sentence comprehension difficulty. *Language and Cognitive Processes*, 26(3):301–349.

J. Bresnan and R. M. Kaplan. 1982. *The Mental Representation of Grammatical Relations.* The MIT Press, Cambridge, MA.

M. Butt and T. C. King. 1996. Structural topic and focus without movement. In *M. Butt and T. H. King, eds., The First LFG Conference. CSLI Publications.*

N. Chomsky. 1965. *Aspects of the theory of syntax*. MIT Press.

N. Chomsky. 1981. *Lectures on government and binding*. Dordrecht: Foris.

N. Chomsky. 1995. *The Minimalist Program*. The MIT Press.

V. Demberg and F. Keller. 2008. Eye-tracking corpora as evidence for theories of syntactic processing complexity. In *Cognition*, volume 109, pages 193–210.

V. Demberg. 2010. *A Broad-Coverage Model of Prediction in Human Sentence Processing*. Ph.D. thesis, The University of Edinburgh.

V. Gambhir. 1981. *Syntactic restrictions and discourse functions of word order in standard Hindi*. Ph.D. thesis, University of Pennsylvania, Philadelphia.

E. Gibson. 2000. Dependency locality theory: A distance-based theory of linguistic complexity. In *A. Marantz, Y. Miyashita W. O'Neil (Eds.), Image, language, brain: Papers from the first mind articulation project symposium. Cambridge, MA: MIT Press.*

D. Grodner and E. Gibson. 2005. Consequences of the serial nature of linguistic input for sentenial complexity. *Cognitive Science*, 29(2):261–290.

J. Hale. 2001. A probabilistic earley parser as a psycholinguistic model. In *Proceedings of the NAACL*, NAACL '01, pages 1–8.

R. Hudson. 2010. *An Introduction to Word Grammar*. Cambridge University Press.

Y. Kachru. 2006. *Hindi*. John Benjamins Publishing Company, Philadelphia.

E. Kaiser and J. C. Trueswell. 2004. The role of discourse context in the processing of a flexible word-order language. *Cognition*, 94:113–147.

A. Kidwai. 2000. *XP-Adjunction in universal grammar: Scrambling and binding in Hindi- Urdu*. Oxford University Press, New York.

A. Kim, B. Srinivas, and J. Trueswell. 1998. The convergence of lexicalist perspectives in psycholinguistics and computational linguistics. In *P. Merlo and S. Stevenson (eds.), Papers from the Special Section on the Lexicalist Basis of Syntactic Processing, CUNY Conference*.

P. Kolachina, S. Kolachina, A. K. Singh, V. Naidu, S. Husain, R. Sangal, and A. Bharati. 2010. Grammar Extraction from Treebanks for Hindi and Telugu. In *Proceedings of The 7th International Conference on Language Resources and Evaluation.*

L. Konieczny. 2000. Locality and parsing complexity. *Journal of Psycholinguistic Research*, 29(6):627–645.

A. Kothari. 2010. *Processing Constraints And Word Order Variation In Hindi Relative Clauses*. Ph.D. thesis, Stanford University.

S. Kübler, R. McDonald, and J. Nivre. 2009. *Dependency Parsing*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers.

M. Kuhlmann. 2007. *Dependency Structures and Lexicalized Grammars.* Ph.D. thesis, Saarland University, Saarbrücken, Germany.

R. Levy, E. Fedorenko, M. Breen, and E. Gibson. 2012. The processing of extraposed structures in english. *Cognition*, 122(1):12–36.

R. Levy. 2008. Expectation-based syntactic comprehension. *Cognition*, 106(3):1126 – 1177.

R. L. Lewis and S. Vasishth. 2005. An activation-based model of sentence processing as skilled memory retrieval. *Cognitive Science*, 29:1–45.

P. Mannem, H. Chaudhry, and A. Bharati. 2009. Insights into non-projectivity in Hindi. In *ACL-IJCNLP Student Research Workshop*.

C. D. Manning and H. Schütze. 1999. *Foundations of Statistical Natural Language Processing*. MIT Press, june.

R. McDonald, K. Crammer, and F. Pereira. 2005a. Online large-margin training of dependency parsers. In *Proceedings of the 43rd ACL*, ACL '05.

R. McDonald, F. Pereira, K. Ribarov, and J. Hajič. 2005b. Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of HLT/EMNLP*.

I. Mel'čuk. 1988. *Dependency Syntax: Theory and Practice*. State University of New York Press.

J. Nivre and J. Nilsson. 2005. Pseudo-projective dependency parsing. In *Proc. Of ACL 2005*.

J. Nivre. 2003. An efficient algorithm for projective dependency parsing. In *8th International Workshop on Parsing Technologies (IWPT)*, pages 149–160.

U. Patil, G. Kentner, A. Gollrad, F. Kuegler, C. Fery, and S. Vasishth. 2008. Focus, word order and intonation in Hindi. *Journal of South Asian Linguistics*, 1(1):55–72.

C. Phillips, M. W. Wagers, and E. F. Lau. 2011. Grammatical illusions and selective fallibility in real-time language comprehension. In *J. Runner (ed.), Experiments at the Interfaces, Syntax and Semantics*, volume 37, pages 153–186.

M. Pickering and G. Barry. 1991. Sentence Processing without Empty Categories. *Language and Cognitive Processes*, 6(3):229–259.

M. Pickering. 1994. Processing Local and Unbounded Dependencies: A Unified Account. *Journal of Psycholinguistic Research*, 23(4):323–352.

I. A. Sag, T. Wasow, and E. M. Bender. 2003. *Syntactic Theory: A Formal Introduction, 2nd edition.* CSLI.

A. Staub and C. Clifton. 2006. Syntactic prediction in language comprehension: Evidence from either ... or. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 32:425–436.

W. Tabor and S. Hutchins. 2004. Evidence for Self-Organized Sentence Processing: Digging-In Effects. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 30(2):431–450.

S. Vasishth and R. L. Lewis. 2006. Argument-head distance and processing complexity: Explaining both locality and antilocality effects. *Language*, 82(4):767–794.

S. Vasishth, K. Suckow, R. L. Lewis, and S. Kern. 2010. Short-term forgetting in sentence comprehension: Crosslinguistic evidence from head-final structures. *Language and Cognitive Processes*, 25(4):533–567.

S. Vasishth, R. Shaher, and N. Srinivasan. 2012. The role of clefting, word order and given-new ordering in sentence comprehension: Evidence from Hindi. In *Journal of South Asian Linguistics*, volume 5.

S. Vasishth. 2004. Discourse Context and Word Order Preferences in Hindi. In *R. Singh (ed.), The Yearbook of South Asian Languages and Linguistics*, pages 113–127.

F. Xia and M. Palmer. 2001. Converting dependency structures to phrase structures. In *Proceedings of the First International Conference on Human Language Technology Research*.