Hindi PropBank Annotation Guidelines

Draft, December 2010

Contents

1. Propbank Annotation Goals
2. Using the tools
3. Annotation of numbered arguments
4. Annotating of modifiers
5. Causatives
6. Unaccusatives (to be added)
7. Complex predicates
8. Empty categories
9. Relatives and Correlatives
10. Passives
11. Questions
12. Special cases of topicalization
13. Span of annotation
14. Using Sanchay

Hindi PropBank Guidelines

1. PropBank Annotation Goals

PropBank is a large annotated corpus consisting of information regarding the argument structure of predicates. PropBanking involves creating a semantic layer of annotation that adds predicate argument structure to syntactic representations (Palmer et al, 2005). The Hindi PropBank annotations are done on top of the Hindi dependency treebank (Sharma et al, 19xy). For each verb, PropBank represents the information about the arguments that appear with the verb in its corresponding **frame file**. The arguments of the verbs are labeled using a small set of numbered arguments, e.g. Arg0, Arg1, Arg2, etc. The following table shows the frame file for the verb *pii* 'drink' which has two arguments: Arg0 and Arg1.

'pii'	'to drink', Transitive
raam ne sharaab pii	
'Ram drank liquor'	
Arg0	drinker: *raam*
Arg1	liquid: *sharaab*

Table 1: A frame file for Hindi

In the frame files for each verb, the numbered argument labels are associated with fine-grained verb-specific descriptions. For instance, in the case of the *pii* 'drink', Arg0 is the 'agent' who performs the action of drinking (the 'drinker') and Arg1 is the entity that is affected by the action of the agent (the 'liquid'). Thus typically, when the verb *pii* appears in a sentence, it will have two arguments which have the semantic roles indicated in its frame file.

Additionally, the sentences in a corpus occur with modifiers that are not part of the semantic specifications of the verb. E.g. the verb *pii* 'drink' implies a 'drinker' and a liquid that is drunk. But the verb itself does not specify *when* the drinking happened or *where* or *how*, so this kind of information is not provided in the frame file for the verb. But if one comes across a sentence such as 'Ram drank liquor in a bar yesterday', the expressions 'in a bar' and 'yesterday' that provide additional temporal, spatial (or manner) information about the situation have to be annotated. Since this information is not provided in the frame file, the annotator will use one of a set of special labels to

indicate the kind of information provided by these modifiers. Typically, these modifiers are annotated using functional tags such as ArgM-LOC, ArgM-TMP, ArgM-MNR.

Hindi-Urdu is a language that allows the speaker to freely omit arguments of the verb in discourse-pragmatically licensed contexts. To take the example above, one can say 'Ram drank liquor', but if 'Ram' has been talked about before, or is otherwise salient in the context, one could say 'drank liquor' without overtly mentioning the subject, 'Ram'. In a corpus, one come across many such sentences where the arguments of the verb are "missing" although they can be retrieved from the context. Although PropBank annotation does not typically involve adding empty arguments to syntactic trees, in the case of Hindi-Urdu we have taken a somewhat different approach. We insert the core empty arguments of the verb (subject, object or indirect object) and then go ahead and assign them semantic role labels just as we do for the overt arguments. The information contained in the verb frame files can act as a valuable resource in allowing for recovery of the different kinds of empty arguments before we annotate them using the semantic role labels.

PropBank annotation is carried out on data that has already been parsed syntactically, or treebanked. In English, this has been done on the Penn Treebank. In the case of Hindi, it will be carried out on a Hindi dependency treebank. Below we mention four goals we consider important for the Hindi PropBank.

(i) As mentioned above an important goal is to provide semantic role labels to the arguments. These are in the form of numbered arguments, e.g Arg0, Arg1,Arg2 etc. They are numbered in order to be more generic and theory neutral (Palmer et. al, 2010). The same numbered argument should be consistent in terms of its semantic role across different syntactic realizations of the same argument of the verb. For example, *darvAzA* 'door' recieves Arg1 in both the sentences (1a) where it is the grammatical object of the verb and (1b) where it is the grammatical subject.[1] The reason it receives the same numbered argument label is because the argument *darvAzA* bears the same semantic role in both sentences – it is the object that undergoes motion.

(1) a. *[$_{ARG0}$ rAm ne] [$_{ARG1}$ **darvAzA]** KolA*

1 The transliteration conventions in this document are not yet consistent. Eventually, all Hindi examples will be transliterated using the wx format. For a mapping between Devnagari script and the WX characters, please see /data/home/verbs/shared/cleardata/hindi/wx-chart.jpg.

Raam Erg door opened
'Raam opened the door.'

b. ***[ARG1 darvAzA]*** *KulA*
door opened
'The door opened'

The consistency in semantic role labelling is also helpful in the training of machine learning systems such as automatic semantic role labellers. However consistency in semantic role labelling for the arguments of the same verb does not mean that only one set of semantic role (SR) labels are available for a specific verb. PropBank also takes into account the different senses of the same verb while annotating the semantic roles. It is possible that two senses of the same verb have a different set of SR labels. For example, the verb *KilnA* takes an Arg1 only in (2a) with sense1 'to bloom', but it takes an Arg1 and Arg2_LOC in (2b) with sense2 'to look good'.

(2) a. ***[ARG1 kaliyAM]*** *KilIM*
buds bloomed
'The buds bloomed'

b. ***[ARG1 poSAk]*** ***[ARG2_LOC Ap par]*** *Kil rahI hE*
dress you on *bloom PROG are.*
'The dress is looking good on you.'

(ii) The second goal of the PropBank annotation involves assigning functional tags to all modifiers of the verb, such as manner (MNR), locative (LOC), temporal (TMP).

(3) *BuS us se* ***guruvAr ko*** ***White House mEM*** ***akele mEM*** *mile*
Bush him with Thursday on White House in alone in met
'Bush met with him privately in the White House on Thursday.'

Rel: *mile*
Arg0: BuS
Arg1: us se
ArgM_TMP: guruvAr ko

ArgM_LOC: White House mEM

ArgM_MNR: akele mEM

(iii) A third goal of Hindi PropBank is to identify the mappings between the PropBank semantic role and the *karaka* and non-*karaka* dependency relations used in Dependency Structure. As a first step, this is done by including information about the Dependency Structure grammatical relations in the frame file(s) for each verb.

[*following taken from the DS guidelines*] In Dependency Structure, all *karaka* relations have been labeled starting with a 'k' followed by a number. Paninian grammar talks about six *karaka* relations. Although the basic number of *karakas* is six, there are a number of relations which are either finer types of *karakas* (such as k2p, k2g etc) or are in some way or the other related to a *karaka* (such as k1s, k2s, k1u, k2u etc). The labels for dependency relations other than *karakas* start with an 'r'.

A sample verb entry is provided below:

aa **sense1: come**

Roles:

Paninian	**Theta Roles**	**Propbank**
k1 *(karta)*	**Agent**	**Arg1**:*entity in motion / 'comer'*
k2p *(goal)*	**Goal**	**Arg4**:*end point*

Necessity of the Argument: k1, k2p → m (mandatory)

Eg1: (destination)

saal bhar yahaa shraddhaalu aate rahate hE
year long here devotees come.Imp AUX be.Pres

'Devotees come here the whole year.'

REL:	*A*	'come'
k1 →Arg1:	*shraddhaalu*	'devotees'
k2p →Arg4:	*yahaa*	'here'
k7t → ArgM-TMP:	*saal bhar*	'year long' → o (optional) (*kaladhikaran*)

(iv) Finally, Hindi PropBank annotation involves annotation of null arguments in the context of a particular verb sense, e.g. it is possible to say *calaa gayaa* '(S/he) left' in Hindi without overtly specifying the argument of the verb *cal* 'leave' if the argument is recoverable from the discourse-pragmatic context. We also identify arguments of the verb that are obligatorily null: (a) the null subject of complement clauses of verbs such as *caah* 'want' (*mohan NULL ghar jaanaa caahtaa hae* 'Mohan wants [NULL to go home]'); (b) the null subject of adjunct clauses (*ghar jaakar mohan khaanaa khaaegaa* '[(After) NULL going home], Mohan will eat food'); (c) the gapped argument in participial modifiers (*khile phuul* '[NULL] blossomed flowers'); (d) the omitted arguments in coordinate constructions (*mohan ghar jaaegaa aur khaanaa khaaegaa* 'Mohan will go home and [NULL] eat food'). An example where we insert a null element in a sentence with a complement clause (so-called 'PRO' in generative grammar) is shown below.

(4) *mEMne [PRO ghar jAnA] cAhA*
I-Erg NULL home to-go wanted
'I wanted to go home.'
Rel: *jAnA*
Arg0: NULL

The subject of the verb *jaanaa* 'to go' in this example is inserted as an empty category PRO at the PropBank stage and eventually it is annotated like any other overt argument.

These four tasks of PropBank annotation: argument labeling, annotation of modifiers, identifying the mapping between Dependency Structure and PropBank, and insertion and annotation of empty categories are discussed in detail below. Besides these three tasks, Hindi PropBank annotation is also concerned with the annotation of complex

predicates, (discussed in section 3) and with distinguishing unaccusative vs. unergative verbs (discussed in section xy). But first in section 2, we describe how various tools, such as cornerstone and Sanchay, are used for these tasks. In section 4, we will discuss various issues and constructions which would aid the process of empty category insertion and PropBank annotation of arguments (overt elements or empty categories), modifiers and complex predicates.

2. ***Using the tools***

2.1. Framefiles

Frame files are an important source of information for annotators (In PropBank, the annotation will be carried out verb-by-verb). The frame files provide verb-specific description of all possible semantic roles, as well as illustrate these roles by examples. If a verb has more than one sense, then this is also represented in the frame, with illustrative examples. In Hindi, a verb can have more than one morphological realization e.g *cala* 'walk', *calA* 'make walk', *calvA* 'make someone walk'. These different forms are also part of the verb frame for *cala*. A verb may also be combined with a noun or adjective to form a complex predicate e.g *PEslA karnA* 'to make a decision' (lit. decision make-Infin). See Part 4 for more details.

2.2. Frame files creation and use of cornerstone: The procedure used in creating Hindi verb Frame files is described in this section. As mentioned above, the issues that arise in this context are discussed in section 4 below (also discussed in the literature on syntax-semantics mappings in South Asian languages, e.g. Mohanan 1994, Butt 1995).

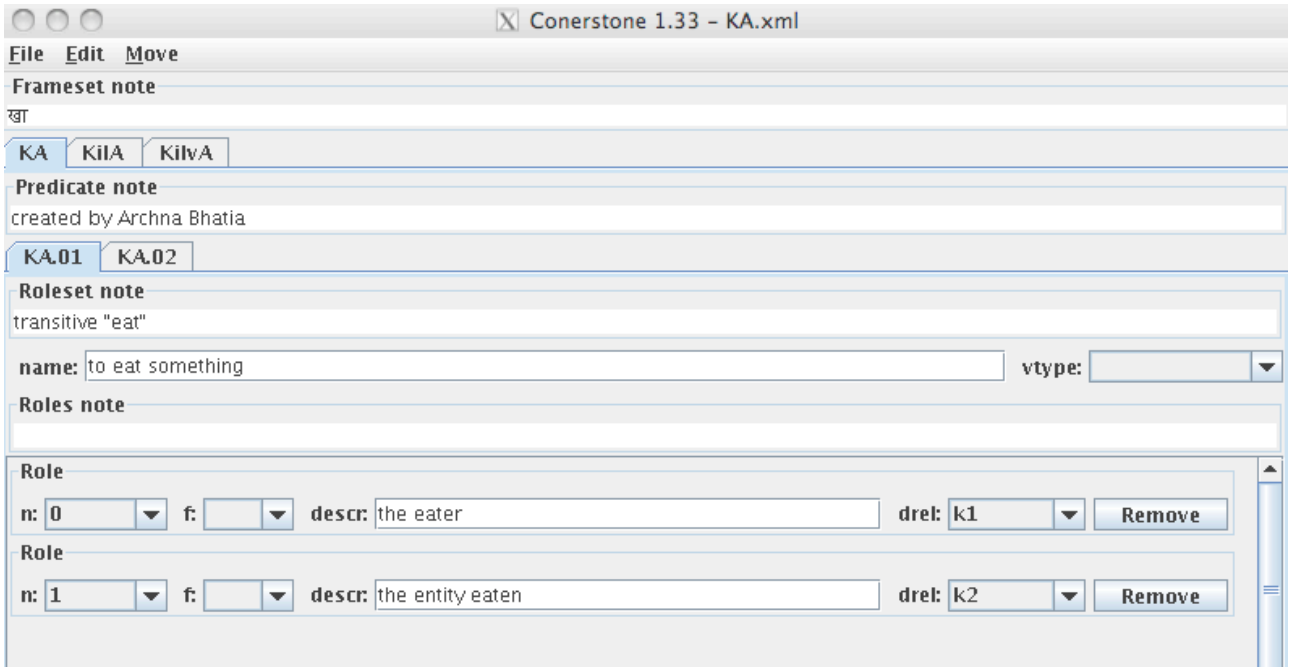
2.2.1 Verb selection

We begin creating framefiles, first starting with the most frequent verbs in the corpus and then going on to the lower-frequency vebrs. When frames are created for the verbs, we make sure that for the primary sense of the verb, there is a frameset and an example taken from the corpus. If a verb has a different sense and the numbered arguments (Arg0-4) that it takes differ significantly from another sense then, that should be included as another ‘roleset’ which will be numbered differently.

2.2.2 Annotation using Cornerstone

The screenshot below shows the tool Cornerstone which is used to create a frame file.

Cornerstone may be accessed at /home/verbs/shared/propbank/cornerstone. Make sure that you use version 1.33.



Below, we outline the format for entering values in each of the fields seen in Cornerstone above:

Frameset note: Enter the LEMMA in Devnagari. The lemma is the citation form of the verb (in Hindi, this is the bare form of the verb, e.g. *KA* 'eat', and not its inflected forms, e.g. *KayA* 'ate').
Frameset tabs: Enter different forms of the verb's lemma, if it can be transitivized or causativized. In the example above, we can see that *KA* becomes *KilA* 'eat.CAUS' ('feed') and *KilavA* 'feed.CAUS'. Each of these will get a separate frameset. If the verb is a support verb in a complex predicate, e.g. the verb *kara* in the complex predicate *snaan kar* 'bath do' then we will enter *kara_LV* as one of the framesets (see section xx for more details).

Predicate note: ["created by" …]

Roleset tabs with Roleset Ids:
Roleset note: some identification information, e.g. transitive/causative/unaccusative. If the verb is unaccusative, it might be useful to

mention which of the unaccusative tests it passes.[2] Moreover, if there are exceptional cases among unaccusatives like the motion verbs which may display variable behaviour with respect to the unaccusative diagnostics, then this information should be specified in the roleset note.

Name: give meaning or sense of the verb (in English)

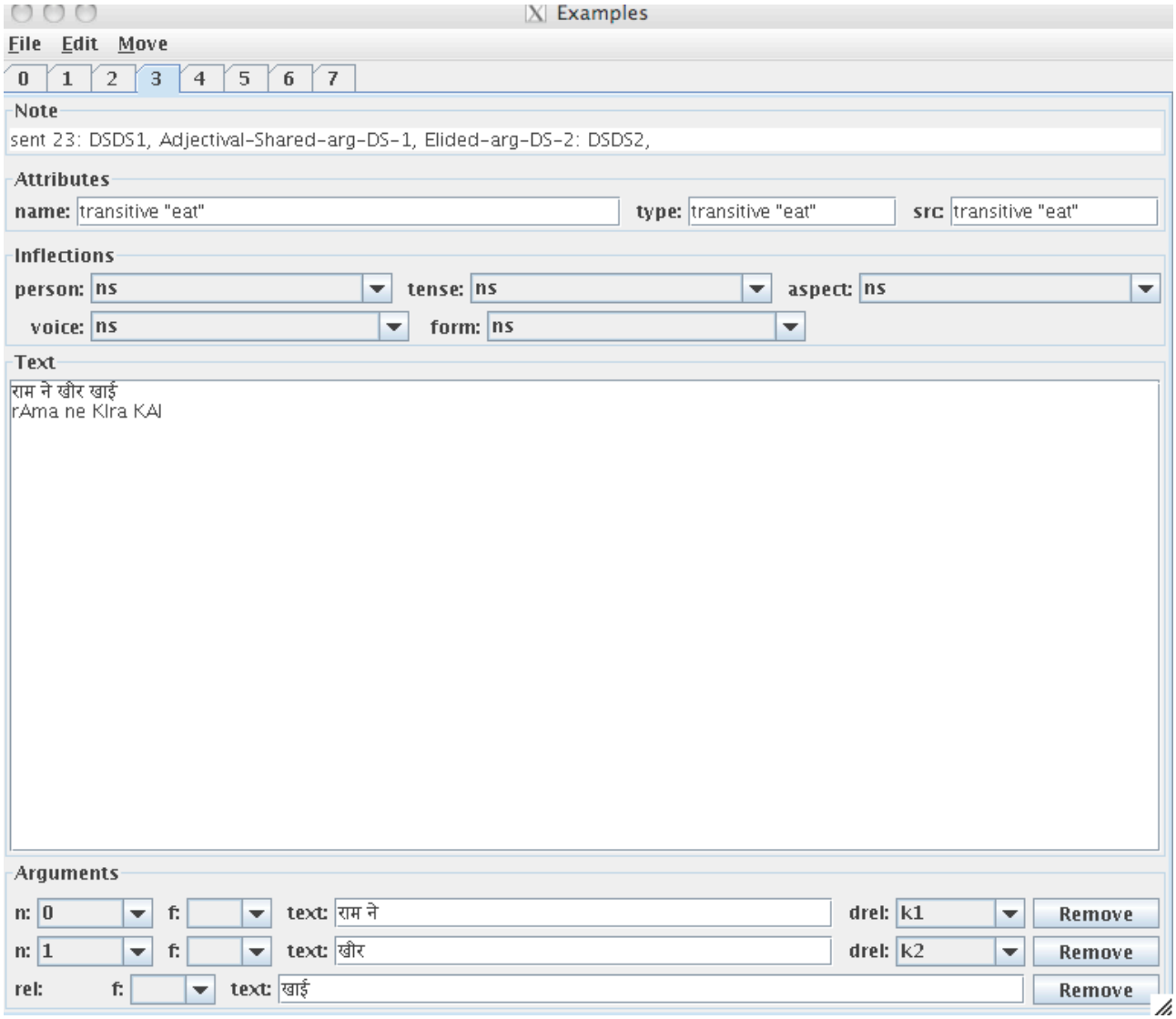
vtype: Type of intrans V (Unacc/Unerg)

Roles note: may enter information to disambiguate roles/ syntactic information relevant to roles

Role: n:Arg0, Arg1, …

descr: description about that thing, e.g. the eater

drel: karaka label k1/k2 mapping for the PropBank role



[2] For each intransitive verb in the corpus, the frame-file creator determines whether it is unaccusative or unergative by using the verb in a number of diagnostic test frames.

The above shows a screenshot of the example screen. For each roleset, it is important to include an example of every roleset that is included in the frame.

Example

Note: Source of sentence, e.g. corpus or web. The above example shows the actual sentence number. It is not necessary to include the number, a note about the source should be enough.

Name: brief identifier of the example/meaning …
src: title of corpus that was the source (optional)

Text: actual example in Devnagari
Include relevant indices, use bracketing if required
Arguments:

N: Arg0, Arg1, … f: TMP, LOC, … text: relevant argument chunk from the example *drel*: k1, k2 that maps to the PB roles

Use more than one example if necessary to illustrate the point.

2.3. Using the Frame files for PropBank annotation:

Here is an example of a Framefile.

kara.01 ; sense 'do'
Arg0: doer
Arg1: thing done

E.g *rAma ne kAma kiyA* 'Ram Erg work did' "Ram worked"

This is one sense of the verb represented by the frameset numbered as 01, 02 etc. Verbs can be quite polysemous but it is difficult to provide a set of roles for all the different senses of the verb. Instead, we only differentiate between two senses of the verb when it takes a different set of arguments. For example, the verb *baca* has two

senses below:

baca.01 ‘remain’

Arg1: thing remaining
Arg2: benefactive, entity getting the remaining thing

E.g. *mere paas 10 rUpaye bace hE*
me-obl with 10 rupees remain be
‘I have 10 rupees left with me’

Arg1: *10 rUpaye*
Arg2: *mere pAsa*

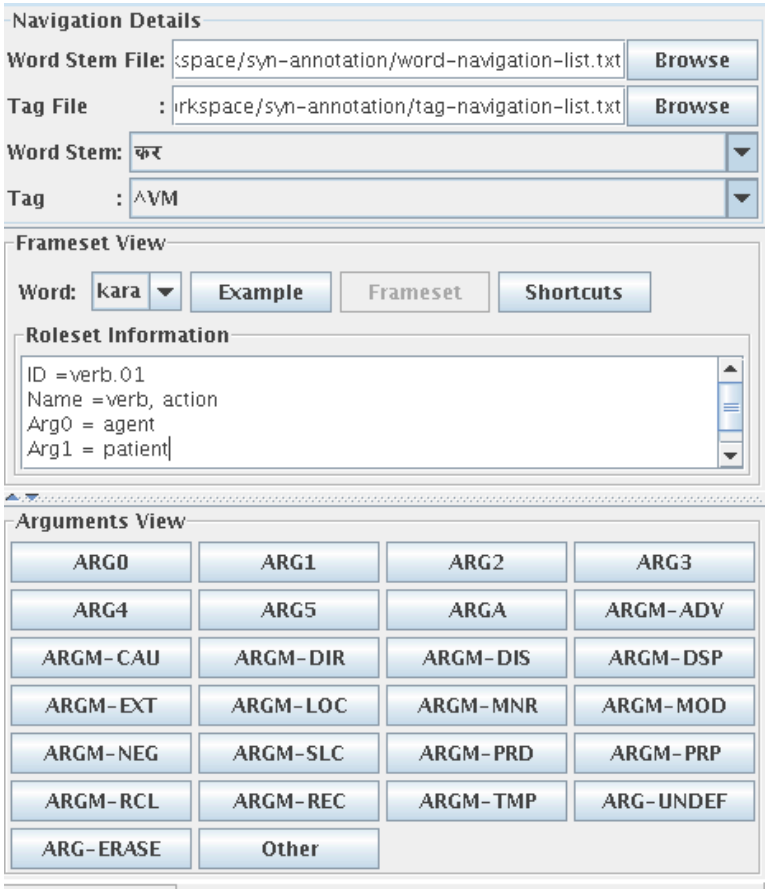
baca.02 ‘avoid’

Arg0: avoider
Arg1: thing avoided

E.g. *yaha SarAba se bacawA hE*
He liquor loc avoid be
‘He avoids (drinking) liquor’

Arg0: *yaha*
Arg1: *SarAba*

In this case, we have two framesets indicating the difference in the senses for the verb *baca*. When annotating, it is necessary to look at all the different senses specified in the frame file before annotating. The annotation tool- Jubilee will load the frame file in the PropBank annotation pane:



3. Annotating numbered arguments

The arguments of a verb are annotated using numbered arguments. We use 10 numbered arguments. However, some of these are numbered along with their respective function tags, for example Arg2_GOL, Arg2_DIR, etc. The list of numbered arguments are as follows:

Numbered Arguments	Description
1. Arg0	Prototypical agent, actor, experiencer, causee
2. Arg1	Prototypical patient, theme
3. Arg2	Beneficiary
4. Arg2-GOL	Goal, destination
5. Arg2-SOU	Source
6. Arg2-ATTR	Attribute
7. Arg2-LOC	Location
8. Arg2-DIR	Direction
9. Arg3	Instrument

10. ArgA	Intermediate causer
11. ArgC	Causer

ARG0

In general, the numbered argument Arg0 corresponds to the prototypical agent of a verb. The verb itself can be transitive, intransitive, or ditransitive. For example, in a sentence with a simple transitive verb like *KA* 'eat' the entity performing the act of eating will be the agent and hence, Arg0. In the example *rAma KIra KAwA hE* 'Ram eats rice pudding', the noun phrase *rAma* is labeled Arg0.

Arg0 is also used for the agentive participant of ditransitive verbs that are derived from transitive verbs using the transitivizing *–A* morpheme in Hindi (e.g. *KilA* 'cause to eat (feed)' which is derived from *KA* 'eat' or *sikhaa* 'to cause to learn (teach)' which is derived from *siikh* 'learn'). The *-A* is also used to derive transitive verbs from intransitive verbs (e.g. *rulA* 'irritate/cause to cry' which is derived from *ro* 'cry'; or *giraa* 'make fall' which is derived from *gir* 'fall'). The agentive participant of such derived transitive verbs also receive the Arg0 label. Some examples are shown below:

5. ***[rAma ne*** $_{ARG0}$*] sIwA ko kAnA KilAya*
   Rama erg Sita dat food fed '
   Ram fed Sita the food'

6. ***[Mohan ne***$_{ARG0}$*] bacce ko siKAyA*
   Mohan erg child dat taught
   'Mohan taught the child'

7. ***[Mohan ne***$_{ARG0}$*] rAma ko rulAyA*
   Mohan erg rama dat made cry
   'Mohan made Rama cry'

Among intransitive verbs, we typically find Arg0 in those verbs that take an animate subject, for example *naac* (dance) in *sIwA nAca rahI hE* 'Sita is dancing'. The verbs *naac* 'dance' and *dauR* 'run' are examples of intransitive verbs that require an agent

subject because the actions involved in these verbs involve a participant that is intentional and controls the action described by the verb. Inanimate subjects of intransitive verbs can also be Arg0 when they display strong agentivity. For instance, natural forces such as *Andhi* ‘storm’ or *tUfAna* ‘typhoon’ are labeled with Arg0 in sentences such as *tUfAna ne KhidkiyoM ko woda dAlA* ‘(The) typhoon broke the windows’. Metaphorical extensions such as *bAriSa ne PuloM ko naya jIvana xiyA* ‘(The) rain gave the flowers new life’ involve an Arg0 for *bAriSa ne* ‘rain Erg’. Look at the verb and see if it typically takes an Agent subject (e.g. *jiivan denaa* ‘give life’).

Arg0 is also seen in cases where the sentence has an experiencer subject. In Hindi, the experiencer subject can be found in sentences like *mujhko chaand dikha* ‘I glimpsed the moon’ where *mujhko* ‘I Dat(ive)’ does not have agent like properties, because it is not controlling the action, but nevertheless the act of seeing is an internally caused reaction to the moon. The experiencer subject is found with certain verbs like *dikhnaa* ‘glimpse’*, milnaa* ‘find’*, lagnaa* ‘feel’*, suujhnaa* ‘be struck (with an idea)’. In such cases, the frame file will indicate the ‘non-prototypicality’ of the agents by specifying that the Arg0 is an ‘experiencer’ in the description of the role in the frame file.

We also assign the Arg0 label for possessor subjects as in the following example. As with experiencer subjects, the frame file will indicate that such arguments lack the degree of agentivity associated with ‘prototypical’ transitive verbs such as *toR* ‘break’ or *maar* ‘hit’:

8. *raam ke ek beTii hai*
Ram gen one daughter is
‘Ram has one daughter’

For passivized subjects, although syntactically, the subject has been demoted, it is still the doer of the action e.g *raam dvaaraa kheer khaayii gayii* ‘(The) rice pudding was eaten by Raam’ and hence *raam* will get Arg0.

In Hindi, the morpheme –*ne* is often indicative of an agent, and as Arg0 is mostly associated with agentivity, this can sometimes provide a clue about the identity of Arg0. However, this is not always the case. E.g. a verb such as *nahaa* ‘bathe’ can have an agent with a dative case marker *ko*, e.g. *Atif ko nahaanaa padZaa* ‘Atif had to bathe’

where the modal *padZaa* assigns Dative case to the subject *Atif*. Similarly a light verb such as *jaa* ‘go’ can also cause an agentive participant to occur without the *–ne* morpheme when it co-occurs with a transitive main verb, e.g. *atif saaraa khaanaa khaa gayaa* ‘Atif ate up all the food’. However, for PropBanking, we will analyze the verb’s event and its participants irrespective of the modal or light verb that changes the case on the noun.

ARG1

In contrast to Arg0 which is the prototypical agent, Arg1 is the label that is assigned to objects that are acted upon by another participant and are affected by the action described in the verb.

Hence, *rotii* ‘bread’ in *raam rotii khaataa hai* ‘raam eats bread’ which is the object of the transitive verb gets the label Arg1. The notion ‘affected by the action described by the verb’ is however quite broadly contrued, e.g. with the verb *sunaa* ‘listen’ we have an example like *pradhaan mantrii ne neta kii maang sunii* ‘(the) prime minister heard the leader’s appeal’ where *netaa kii maang* ‘(the) leader’s appeal’ gets the Arg1 label even if an ‘appeal’ is not an entity that is acted upon by another participant and affected by an action in a concrete way.

The Arg1 label typically occurs as the ‘affected’ entity of verbs that are transitive or ditransitive. But within the intransitive verbs, there is a special class of intransitives called **unaccusative** verbs. These have a syntactic subject which does not have any agent like properties (contrast with the case of the intransitive verb *naachnaa* ‘dance; which needs an agentive subject). Verbs such as *khulnaa* in *darvaazaa khulaa* (The door opened) do not require an agent subject and hence *darvaaza* will get the Arg1 label. The Arg1 label here is thus assigned to the ‘theme’ argument, typically defined as the entity that is at rest or undergoing motion or change of state.

For example, in the following sentence, we have the verb *jala* ‘to burn’, which is unaccusative and hence, *chaatron ke haath* gets the Arg1 label

9. *chaatron ke haath jala gaye*
Students gen hands burn past
‘The students burnt their hands’

Arg1 is also found in verbs like *honA* ‘to be/become’ where the subject is a “theme” argument with some attribute. It is also described as a linking verb (Kachru, 2006), which establishes a relationship between the subject and a complement. The complement could be an attribute like *achcha* ’good’ or a location *ghar mein* ‘at home’ or an identity e.g `doctor’. In the following example, *rAma* will get Arg1:-

10. ***[rAma***$_{ARG1}$***]*** *acCA hE*
ram good is
‘Ram is good’

11. ***[saDak***$_{ARG1}$***]*** *cauDii huii*
road wide became
‘(The) road became wide’

ARGC

In Hindi, it is possible to add the causative morpheme *–vA* to a transitive or ditransitive verb in order to get the meaning: to cause someone to do X (for further details see section xx on causatives). For example, *biknA* (to sell) becomes *bikvAnA* (to cause someone to sell something). In such constructions, the causer has a special status in the sentence as it denotes the person who is causing the agent to actually perform the action. Hence, it gets the label ArgC. For example, in the sentence below, *rAma* is the ArgC:

12. ***[rAma ne***$_{ARGC}$ *] slwA se bacce ko KAnA KilvAyA*
Raam erg Sita instr child dat food (made) feed
‘Ram made Sita feed the child food’

13. ***[sitaa ne***$_{ARGC}$***]*** *mohan se ram ko girvaaya*
Sitaa erg Mohan instr ram ac made fall
‘Sita caused Mohan to make Ram fall’

14. ***[sitaa ne***$_{ARGC}$***]*** *mohan se raam ko rulvaaya*
sitaa erg mohan instr Ram acc made cry
‘Sita caused Mohan to make Ram cry’

ARGA

The *–vA* morpheme allows one or more intermediate causers as well, for example *slwA se* in the example below (for further details see section xx on causatives). The labels for intermediate causers is ArgA.

15. *[rAma ne$_{ARGC}$]* ***slwA se*** *bacce ko KAnA KilvAyA*
    Raam erg Sita instr child dat food (made) feed
    'Ram made Sita feed the child food'

It is also possible to have more than one intermediate causer:

16. *[rAma ne$_{ARGC}$]* ***slwA se rlwa dvaara*** *bacce ko KanA KilvAyA*
    Ram erg Sita inst Rita by child dat food feed-cause
    'Ram made Sita, who via Rita made the child eat food.'

ARG2

The Arg2 label is used to denote the beneficiary of the action of the verb. For example, in the sentence:

17. *rAma ne [**SyAma ko**$_{ARG2}$] kiwAba xl*
    rama erg Shyam dat book give
    'Ram gave Shyam a book'

Arg2 is the beneficiary of the action of *xenA* (give). Arg2 is found in verbs that need a recipient argument e.g. *parosa* 'serve', *laa* 'bring'. *bataa* 'tell', etc.

Arg2 will also be the label we use for other kinds of participants such as the causee in the case of causative verbs, e.g. the argument *bacce ko* 'child Dative' in the sentence *siita ne aayaa se bacce ko khaanaa khilvaayaa* 'Sita had the maid feed the child food'. In this sentence, we use the Arg2 label for *baccaa* 'child' which is the causee. Although the child is performing the act of eating, it does not get the Arg0 label in this construction because it does not initiate the act of eating and is hence seen as less controlling. Another motivation for using the Arg2 label for the causee in such cases comes from the

fact that we can regard 'indirect causative' verbs such as *khilvaa* 'cause to feed' as being derived from a ditransitive verb *khilaa* 'feed'. The recipient of the action denoted by ditransitive verbs such as *khilaa* 'feed' receives the Arg2 label (just as the recipient of the action described by underived ditransitive verbs such as *de* 'give). The same Arg2 label is retained when the ditransitive *khilaa* is further causativized to yield the verb *khilvaa* 'cause to feed' (for further details see section xx on causatives).

ARG2-GOL

The Arg2-GOL is the destination or goal argument for the verb. It is useful to think of Arg2-GOL as a motion leading to an end point. For example, a verb like *pahuMcnA* 'to reach' takes a destination argument:

18. *cAcA [**dilli**$_{ARG2\text{-}GOL}$] pahuMca rahe hai*
    uncle delhi reach prog be
    'Uncle is going to reach Delhi'

lOtnA 'to return', *BejanA* 'to send', *AnA* 'to come', *jAnA* 'to go', *calanA* 'to walk/go', *GusanA* 'to push one's way into' *Coda* 'to drop something/someone' are some of the verbs that take the Arg2-GOL label.

19. *Ram [**ghar**$_{ARG2\text{-}GOL}$] gayaa*
    Ram home went
    'Ram went home'

ARG2-SOU

This label is applied to those arguments with the meaning 'from' or 'move away from'. For example,

20. *raam [**Cawa se**$_{ARG2\text{-}SOU}$] nIce giraa*
    Ram roof abl down fell
    'Ram fell down from the roof'

21. *sIwArAma [**dusre prawiniXI se**$_{ARG2\text{-}SOU}$] Age nikal gayA hai*
    Sitaram other candidate abl ahead remove LV be
    'Sitaram has moved ahead of the other candidate'

22. *maine apne [**bhagwaan se**$_{ARG2\text{-}SOU}$] aashirwaad mAMgA*
I-erg my god abl blessings asked ‘
I asked for blessings from my god’

This label can also be used as a starting point for any action. For example:

23. *kishorI [**haridvAra se**$_{ARG2\text{-}SOU}$] xillI AyI WI*
Kishori Haridvar loc Delhi came past
‘Kishori had come from Haridvar’

This label can also be found in verbs of transfer e.g *Karixa* ‘to buy’, *mAMga* ‘to demand’ For example:

24. *KariSma ne [**sanjay se**$_{ARG2\text{-}SOU}$] xo laKa rupaye mAMge Te*
Karishma erg Sanjay goal two lakh rupees demand past
‘Karishma had demanded two lakh rupees from Sanjay’

ARG2-ATTR

This label is applied to those arguments that describe some property of another argument, which is often (but not always) the subject. The most common example is the predicative element that occurs with *honA* (to be). For example:

25. *raam [**buddhimaan**$_{ARG2\text{-}ATTR}$] hai*
Ram intelligent be
‘Ram is intelligent’

Arg2-ATTR can also be thought of as an identity marking argument with Arg1 in the case of a verb like *bana* ‘to become’;

26. *Ram [dukaan ka **maalik**$_{\textbf{ARG2}\text{-}ATTR}$] ban gayaa*
Ram shop gen owner become go.perf
‘Ram became the owner of the shop’

For verbs like *maanana* ‘to believe/consider’ we give the following analysis:

27. *ve log gandhiji ko **[bapU**$_{ARG2\text{-}ATTR}$] mAnawe hai*
those people Gandhi-hon dat bapu consider be
‘Those people consider Gandhiji as Bapu’

In both the above cases, the highlighted argument is the ARG2-ATTR

However, verbs of cognition like *mAnanA*, *samajhanA* may not always get Arg2-ATTR analysis, e.g many times these are verbs with three components- a subject, a direct object and a complement referring back to the direct object. They don’t always occur like this, but sometimes they can. In such a case, we would have to give both the direct object and complement Arg1 and then make an ICH link. So this analysis for maananaa might need to be re-done.

ARG2-LOC

The Arg2-LOC label is given to locations that are neither destinations nor sources but seem to be essential to understanding the verb event. For example *kaatanaa* ‘to spend time’ gets Arg2-LOC:

28. *loga yahan sadakon par [raat*$_{ARG2\text{-}LOC}$*] kaat rahe hai*
People here streets on night spend prog be
‘People are spending the night on the street’

29. *raam [mere ghar mein* $_{ARG2\text{-}LOC}$*] rahtaa hai*
ram my house in stay be
‘Ram is staying at my house’

30. *billi [per mein*$_{ARG2\text{-}LOC}$*] phas gayi hai*
cat tree loc stuck go be
‘The cat is stuck in the tree’

The three examples above indicate that the verb’s action requires a location, although it may sometimes be metaphorical. It is usually without any direction like the Arg2-SOU

and Arg2-GOL. Other examples are *Guma* ‘to roam’, Kisaka ‘to slip’, *guzara* ‘to pass by’, *duuba* ‘to drown’.

ARG2-DIR

The Arg2-DIR label is given to directions that do not terminate in a goal (directed paths). E.g.:

31. *vo piiche khiskaa*
    he backwards scooted.
    'he scooted backwards'

Here 'backwards' specifies the direction of motion although no endpoint/goal is specified.

ARG3

Arg3 is applied to those arguments that are instruments i.e. they enable the action to take place. Usually, they are artifacts rather than persons. We can use the test “*X kaa upyoga*” (using X) to check whether an argument can be assigned Arg3. In the following sentence, *caaku se* ‘with a knife’ gets Arg3:

32. *raam ne [**cAku se**ARG3] Ama kAta*
    Ram erg knife instr mango cut
    ‘Ram cut the mango with a knife’

It is possible to rewrite the above sentence as : *raam ne cAku ka upyog karke Am kAtA* ‘Ram used a knife to cut the mango’. As this is possible, we assign *cAku* the label Arg3. The verb *kAta* ‘to cut’ is an example of a verb that needs an Arg3 label.

However the “*X kaa upyoga”* test is quite stringent and includes only cases of prototypical instruments. There are other cases where the means whereby an action is performed can be labeled using the Arg3 label. An interesting case is the verb *bharnA* ‘to fill’ where the argument *paanii* ‘water’ gets *–se* case-marking (typically used with instruments) and invites a construal whereby the pot is filled BY MEANS of using water:

33. *sIwA ne* ***[paanii se****ARG3]* *ghade ko* *BarA*
Sita erg water instr pot dat fill
'Sita filled the pot with water'

However, one might argue that *paanii* 'water' is a theme argument (it describes an entity in motion) and should get an Arg1 label. The fact that one can add another argument that functions as an instrument suggests that the Arg1 analysis might be the appropriate one: *sIwA ne cammac se ghade ko paanii se BarA* 'Sita filled the pot with water with a spoon'.

We also have the problem of *se*-marked arguments which might need a different label, i.e. something like ARG3-COM (comitative)

34. *(shikshaa kshetra se*$_{ARG3\text{-}COM?}$*) anek dharmsansthain judii hai*
education field with many religious institutions connected are
'Many religious institutions are connected to the field of education'

4. Annotating modifiers

The modifiers of a verb are annotated using the semantic role tags beginning with ArgM. The following types of modifiers are being used in PropBank:

1. DIR: Directionals
2. LOC: Locatives
3. MNR: Manner
4. ARGM-MEANS/PATH
5. GOL: Goal
6. EXT: Extent
7. TMP: Temporal
8. REC: Reciprocals
9. PRD: Secondary Predication
10. PNC: Purpose
11. CAU: cause
12. DIS: discourse
13. ADV: adverbials
14. DSP: direct speech

15. MOD: modals
16. LV: light verbs
17. NEG: negation

ARGM-DIR

Directional modifiers show motion along some path. Both "source" and "goal" are grouped under "direction." On the other hand, if there is no clear path being followed a "location" marker should be used instead. Thus, *ghar kii or bhaagnaa* ‘run towards home" involves a directional, but *khetoN mein phirnaa* ‘roam in the fields’ involves a location.

35. *mohan **udhar** dauRaa.*
Mohan there ran.
‘Mohan ran there’

ARGM-LOC

Locative modifiers indicate where some action takes place. The notion of a locative is not restricted to physical locations, but abstract locations are being marked as LOC as well:

*36. raam ne **bazaar mein** ravi ko dekhaa.*
Raam erg market in ravi dat saw
‘Raam saw Ravi in the market’

37. ***apne bhaashan mein**, mohan ne neta kii khuub tariif kii.*
self lecture in Mohan erg leader gen much praise did.
‘In his lecture, Mohan praised the leader highly’

ARGM-MNR

Manner adverbs specify how an action is performed. For example, *duusroN ke saath kaam acchaa kartaa hae* ‘he works well with others’ is a manner. Manner tags should be used when an adverb is an answer to a question starting with *kaise* ‘how’?

38. *vaha bahuta **wejZa** bolataa hai*
he very fast talk.Imp be
'He talks very fast'

ARGM-MEANS/PATH

This is like an ARGM equivalent to the instrumental ARG3. It functions to accommodate cases such as the following:

39. *Yanaa ko turanta [ek kaar se$_{ARGM\text{-}MEANS}$] aspataal le jaaya gayaa*
Yanaa dat immediately one car with hospital take go AUX
'Yanaa was immediately taken to the hospital by car'

40. *Raj [paarti dvaara $_{ARGM\text{-}MEANS}$] BJP se sambandh toda lene ke paksh mein hai*
Raj party by BJP with connection break LV gen side in is
'Raj is in favour of breaking connection with the BJP via the party.

ARGM-GOL

This tag is for the goal of the action of the verb. This includes beneficiaries and the final destination of some motion verbs. 'Goal' would be used for modifiers that indicate that the action of the verb was done for someone or something, or on their behalf:

41. *mohan ne **ravii keliye** caay banaaii.*
Mohan erg Ravi for tea made.
'Mohan made (some) tea for Ravi'

ARGM-EXT

ArgM-EXT indicates the amount of change occurring from an action, and are used mostly for (a) numerical adjuncts like *aaluu kii kiimat **10 pratishat** baRh gaii hae* 'the price of potatoes has increased **by 10%**' (b) quantifiers such as *bahut* 'a lot' and (c) comparatives such as '(he worked) **more than she did**':

42. *1998 se 2008 tak usne mumbai mein kaam kiyaa*

1998 from 2008 until he-erg Mumbai in work did
'He worked in Mumbai from 1998 to 2008'

In comparative constructions such as the following:

43. *mohan ne* ***siitaa se*** *tez bhaagaa.*
Mohan erg Sita abl fast ran.
'Mohan ran faster than Sita did'

44. *raadhaa* ***miiraa kii tulnaa meiM*** *adhik suMdar hai*
Radha Mira gen comparison in more beautiful is
'Radha is more beautiful than Radha'

ARGM-TMP

Temporal ArgMs show when an action took place, such as *1987 mein* 'in 1987', *pichle hafte* 'last week', *turant* 'immediately'. Also included in this category are adverbs of frequency (eg. *aksar* 'often', *hameshaa* 'always', *kabhii-kabhii* 'sometimes' (with the exception of *kabhii nahii* 'never', see NEG below), adverbs of duration (*ek saal keliye/ek saal mein* 'for a year/in an year'), order (e.g. *pehlaa* 'first'), and repetition (eg. *baar-baar, phir se* 'again'):

45. ***kala*** *paanii barasaa thaa*
yesterday water rained past
'it rained yesterday'

ARGM_REC

These include reflexives and reciprocals such as *khud, apne aap* 'him/her self'*, saath(-saath)/ek saath* 'together'*, ek duusre* 'each other'*, dono 'both'*, which refer back to one of the other arguments. Often, these arguments serve as the Arg 1 of the relation. In these cases, the argument should be annotated as the numbered argument as opposed to the reciprocal modifier.

46. *Mohan aur siitaa ne ek saath kaam kiyaa.*
Mohan and Siitaa erg together work did
'Mohan and siitaa worked together'

47. *mohan ne apne aap kaam kiyaa*.
Mohan erg by-himelf work did
'Mohan worked by himself'

Note that the reciprocal is **not** an ARGM_REC in the following sentence, since they function as an argument.

48. *Mohan aur raam ne ek duusre ko dekhaa.*
Mohan and Raam erg one another Dat looked.
'Mohan and Ram looked at each other'

ARGM-PRD: ***markers of secondary predication (PRD)***

These are used to show that an adjunct of a predicate is in itself capable of carrying some
predicate structure. In Hindi, secondary predication does not appear to be as productive as it is in English. A typical example might include:

49. *bacce ne seb ko* ***choTe-choTe TukRon mein*** *kaaTaa*
child erg apple dat small-small pieces in cut.
'The child cut the apple **in little pieces**'

Note: should we be treating some instances of resultative secondary predication as involving small clauses that are licensed by the verb? How to treat depictives? – will have to look at some corpus examples.

ARGM-PRP

Purpose clauses are used to show the motivation for some action. Clauses beginning with "in order to" are canonical purpose clauses.

50. *maine **dillii jaane ke liye** Tikata khariida.*
I-erg Delhi go-inf for ticket bought
'I bought tickets in order to go to Delhi.'

ARGM-CAU

Similar to "Purpose clauses", these indicate the reason for an action. Clauses beginning with "because" or "as a result of" are canonical cause clauses. Also questions starting with 'why':

51. *[**muKya gavAha ke is bayAna se**$_{ARGM\text{-}CAU}$] sarabjIwa kI rihAI kI ASA badha gayI hE*
primary witness gen this testimony instr Sarabjit gen acquittal gen hope increase go be
'By the primary witness' testimony, the hope for Sarabjit's acquittal has increased'

52. *maine **mohana kii bhuul kii vajaha se** kitaab kho dii.*
I-erg Mohan gen mistake gen because inst book lose LV.
'I lost the book because of Mohan's mistake'

ARGM_DIS

These are markers which connect a sentence to a preceding sentence. Examples of discourse markers are: *lekin/parantu* 'but', *aur* 'and', *iske alaavaa* 'instead', *yaa* 'or', *yaanii* 'namely', etc. Note that conjunctions corresponding to 'but', 'or', 'and' in Hindi are only marked in the beginning of the sentence. Do not mark 'and', 'or', 'but', when they connect two clauses in the same sentence.

53. ***lekin** kal hamaare ghar kaun aane waalaa hae?*
But tomorrow our house who come-inf one is?
'But who is going to come to our house tomorrow?'

54. ***isake alaava**, maovaadi ke raambacana yaadav ko giraftaar kara liyaa gayaa*

this-gen moreover, Maoist gen Rambachan Yadav dat arrest do LV AUX.
'In addition to this/Moreover, Maoists' Rambachan Yadav was arrested'

Another type of discourse markers includes vocatives and interjections:

55. ***maa****, mujhe kal dilli jaana hai*
mother, I-dat tomorrow Delhi go-inf be
'Mother, I want to go to Delhi tomorrow'

56. ***he bhagwaan****, mujhe maafii do.*
Oh God, I-dat forgiveness give.Imp.
'Oh God, forgive me'

ARGM_ADV (Adverbials)

These are used for syntactic elements which clearly modify the event structure of the verb in question, but which do not fall under any of the headings above.

a) Temporally related (modifiers of events): ***mohan kii pratiiskhaa mein*** *siitaa wahiin khaRii rahii* 'Sita remained standing there, **in anticipation of Mohan'**
b) Intensional (modifiers of propositions): *shaayad* 'probably, possibly'
c) Focus-sensitive: *sirf/keval* 'only', *bhii* 'also, even'
d) Sentential (evaluative, attitudinal, viewpoint, performatives): *soubhaagya se* 'fortunately',
e) *asal mein* 'in actuality', *kaanoon ke anusaar* 'legally', *ke baavjuud* 'despite', *agar* 'if', etc.

As opposed to ArgM-MNR, which modify the verb, ARGM-ADVs usually modify the entire sentence.

57. ***shaayad*** *siitaa khaanaa khaayegii*
maybe Sita food will eat
'Maybe Sita will eat food'

58. ***keval*** *do laRkO ne saaraa kaam kiyaa*.
Only two boys erg all work did
'Only two boys did all the work'

ARGM_DSP Direct Speech

Note: We may not need this label since it appears to be used in English Pbank simply as a device to join together discontinuous constituents of directly quoted speech in the PS tree. The DS tree for Hindi will put such discontinuous constituents under one node, thus obviating the need for this label for Hindi Pbank.

ARGM_MOD (modals)

Modal constructions in Hindi convey notions such as ability, desire, obligation, permission, etc. In Pbank, we will annotate the following cases using the ARGM-Mod label.

59. *mohan kaam kar* ***sakegaa****.*
Mohan work do able.fut
'Mohan will be able to do the work'

60. *mohan kaam kar* ***paaegaa****.*
Mohan work do able.fut
'Mohan will be able to do the work'

61. *mohan ko ghar jaanaa* ***caahiye****.*
Mohan dat home go-inf ought
'Mohan ought to go home'

62. *mohan ko kaam karnaa* ***padZaa****.*
Mohan dat work do-inf had
'Mohan had to work'

63. *mohan ko kulfii khaanii* ***hai****.*
Mohan dat icecream eat-inf is
'Mohan has/wants to eat icecream'

64. *raam ne mohan ko duudh piine* ***diyaa****.*
Raam erg Mohan dat milk drink-inf gave.
'Ram allowed Mohan to drink millk'

[Note: additionally we may provide frame files for the individual modal verbs (e.g. the copula *ho* 'be' conveys the notion of obligation/desire only when it occurs in conjunction with an infinitive verb, and it has the effect of assigning Dative case to the subject). Also discuss what we do with *vaalaa*, e.g. in the sentence *mohan kal kaam karne* ***vaalaa*** *hae* 'mohan is going to work tomorrow']

ARGM_LV (light verbs)

Light verbs are semantically bleached verbs that combine with the bare form of the verb to convey different meanings. Typically these meanings are aspectual, but may also involve suddenness, inception, surprise, etc.

65. *mohan ro* ***paRaa***
Mohan cry lie
'Mohan burst out crying'

66. *siitaa saaraa khaanaa khaa* ***gayii***
siitaa all food eat went
'Siitaa ate up all the food'.

67. *siitaa saaraa khaanaa khaa* ***cukii*** *hae*
siitaa all food eat complete.prf is
'Siitaa has eaten all the food'.

67. *saritaa ne saRii khariid* ***lii****.*
Sarita erg sarii buy took
'Saritaa has bought the sarii'

68. *bacce ne mAA ko gend de* ***diyaa****.*
Boy erg mother dat ball give gave.
'(The) boy has given his mother the ball'

Typically, the light verb is not negated independently of the main verb, nor can they be scrambled or have an adverbial inserted between the main verb and the light verb. Hence they will not be annotated as independent verbs.

ARGM-NEG *Negation*

This tag is used for elements such as *nahii* 'not', *kabhii nahii* 'never, *naa* 'not' and other markers of negative sentences. Negation is an important notion for Propbank annotation; therefore, all markers which indicate negation should be marked as NEG. For example, when annotating adverbials like *kabhii nahii* 'never,, which could be marked as either TMP or NEG, the NEG tag should be used.

69. *laRkii ghar **nahII** gayii*
girl home not went
'(The) girl did not go home'

70. *laRkii kaam **kabhii nahII** karegii*.
Girl work never do.fut
'(The) girl will never do work'.

Be careful to distinguish these from conjunctions such as *naa hii* 'not only,' which does not actually indicate that the verb is negative and should not be annotated because it is a conjunction.
*Include a note on how ArgM-Neg and ArgM-Mod are going to be annotated. This would have to be done after the chunks are expanded.

5. Causatives

As mentioned earlier, in Hindi it is possible to add the causative morpheme *–vA* to a transitive or ditransitive verb in order to get the meaning: to cause someone to do X. The treatment of causatives in Hindi PropBank is quite uniform in terms of the assignment of numbered semantic roles to the arguments of verbs with the *–vA*. As discussed above, the outermost argument (the causer) which is annotated with the ArgC label. Any intermediary causers (typically marked with the *–se* case-maker or a

postpositional phrase such as *–ke dvaaraa*) are annotated with the ArgA label. The remaining arguments are marked using Arg2 and/or Arg1. That is, if there is a recipient or beneficiary of the action, it gets the Arg2 label. The 'affected' or theme argument gets the Arg1 label. We provide examples of these different possibilities below:

(a) 'Quadritransitive' verb (*Kil-vA* 'cause to feed') derived from a ditransitive verb (*KilA* 'feed'):

71. *[rAma ne$_{ARGC}$] [sIwA se$_{ArgA}$] [bacce ko$_{Arg2}$] [KanA$_{Arg1}$] Kil-vAyA*
Raam erg Sita instr child dat foo feed-CAUS
'Ram had/made Sita feed the child food'

(b) 'Ditransitive' verb (*rul-vA* 'cause to make cry'; *kat-vA* 'cause to cut') derived from a transitive verb (*rulA* 'make cry'; *kAt* 'cut'):

72. *[Sita ne $_{ARGC}$] [mohan se$_{ArgA}$] [ram ko$_{Arg1}$] rulvAyA*
Sita erg mohan instr ram acc make-cry-CAUS
'Sita had/made Mohan make Ram cry'

73. *[Mohan ne $_{ARGC}$] [raam se$_{ArgA}$] [peRa$_{Arg1}$] katvAyA*
Mohan erg raam inst tree cut-CAUS
'Mohan made/had Raam cut the tree'

Having adopted a more-or-less uniform analysis of *–vA* causatives as far as semantic role labeling is concerned, it also desirable to capture some of the fine-grained differences among different types of causatives at the verb-specific level. This can be achieved by adding a more detailed description of the role in the frame file of the verb. For instance, although the outermost argument, labeled with the ArgC role, is uniformly the 'causer', the arguments marked with the ArgA label (the 'intermediary causers') can be differentiated based on the role they play in the action described by the verb. For instance, the participant *Raam* in the sentence *mohan ne raam se peRa katvAyA* 'Mohan had Raam cut the tree' can be construed as the agent of cutting the tree rather than as an intermediary causer who gets someone else to cut the tree (and in fact, must be construed as such for some speakers of Hindi). Such an interpretation occurs with causatives that are derived from transitives that are themselves derived from

unaccusatives (it is also possible that the causatives are directly derived from the intransitive verbs). So the participant marked with *–se* can be interpreted as the agent of the transitive action when it occurs with verbs such as *girvaa* 'cause (someone) to make (something) fall', *khulvaa* 'cause (someone) to make (something) open', *tuRvaa* 'cause (someone) to break (something)', *bikvaa* 'cause to sell' and so on.

6. Unaccusatives

[Insert section on flow diagram used to diagnose unaccusativity + diagnostics used]

7. Complex Predicates

The annotation of complex predicates will be done as follows:

1. The nominal hosts of complex predicates marked with POF by DS will be automatically annotated with ARGM-PRX label by PropBank
2. During the annotation of simple verbs, we will not look at those cases of the verb where ARGM-PRX exists.
3. As the second pass, we will employ diagnostics to classify the "pof" cases into classes of complex predicates (weak, strong, etc.).
4. We will then have to create frames for the nominals that take the POF label
5. All the complex predicates will then be annotated using nominal verb frames, but common argument structure (in accordance with English and Arabic)

The annotation of complex predicates will be done in two passes. In the first instance, the support verb in a complex predicate such as *corii karnaa* 'theft do' will be annotated as a support verb, and the internal argument(s) with the label ARG-PRX.

If the light verb is intransitive (e.g. *paese corii ho gae* 'the money got stolen' where the support verb is the intransitive verb *jaa* 'go') the nominal *corii* and *paese* respectively will be annotated with ARG-PRX. If the support verb is transitive (e.g. *mohan ne paese corii kiye* 'mohan stole the money'), the subject argument *mohan ne* will be annotated with the appropriate label (Arg0 in this case) and the nominal and its argument (*corii* and *paese*) will be annotated with ARG-PRX. In a second pass, the nominal and the support verb will merged into a single predicate.

8. Empty Categories

[to be revised pending discussion]

8.1 Insertion and annotation

Hindi-Urdu is a language that allows the speaker to freely omit arguments of the verb in discourse-pragmatically licensed contexts. For instance, one can say 'Ram drank liquor', but if 'Ram' has been talked about before, or is otherwise salient in the context, one could say 'drank liquor' without overtly mentioning the subject, 'Ram'. In a corpus, one come across many such sentences where the arguments of the verb are "missing" although they can be retrieved from the context. Although PropBank annotation does not typically involve adding empty arguments to syntactic trees, in the case of Hindi-Urdu we have taken a somewhat different approach. We insert empty categories corresponding to the core arguments of the verb including subjects, direct and indirect objects (so-called "little" *pro*, marked as *pro*), using the context to determine which sense of the verb is relevant. We also use the verb frame file to determine the number and semantic roles of the arguments that are not overtly realized (and that must be inserted).

For instance, in the following example, the subject argument (Arg0) of the transitive verb *paRh* 'read' can be elided, e.g. when it is recoverable from the prior discourse or situational context. In the second sentence, the object argument (Arg1) is missing.

74. **pro* kitaab paRh-egii*
NULL book read-fut
'(She) will read the book'.

75. *kis ne darwaazaa khol-aa? mohan ne *pro* khol-aa*
*w*ho erg door open-perf? Mohan erg *pro* khol-aa
'Who opened the door? Mohan opened (it)'.

In addition, three other kinds of obligatorily non-overt categories are inserted in PropBank:

- empty subject arguments occurring in nonfinite complement and adjunct clauses ("big" *PRO* marked as *PRO*);

- empty arguments in relative clauses (labeled as *RELPRO*)
- empty arguments in coordination and gapping constructions (labeled as *GAP-pro*) NOTE: we will not be actually distinguishing between GAP-pro and pro when we insert null categories. Rather they will be uniformly labeled as pro when they are inserted. However, we will be distinguishing between the two types of empty categories in a postprocessing step. The 'pro' that we insert in coordination and gapping constructions will be coindexed to its antecedent using a special coindexation label (e.g. 'gapref'). The uses of 'pro' elsewhere will not be coindexed (since the antecedent may be found only by looking outside the sentence, in prior discourse), except for any 'pro' instances found in relatives or correlatives. See the section on coreference for further details.

In all these cases, the empty argument is controlled by an antecedent within the same sentence. (Please note that we distinguish between "big" *PRO* represented in caps as *PRO*, and "little" *pro* represented as *pro*).

Since the environments in which *PRO* and *RELPRO* occur can be identified deterministically, these labels will be inserted automatically during a preprocessing step. The null elements *GAP-pro* and *pro* are inserted manually.

For instance, in the following example, the empty subject of the nonfinite complement of the verb *chaah* 'want' is labeled with *pro*. Note that the empty subject is controlled by the subject of the matrix clause (*mohan ne* 'mohan erg'):

76. *mohan ne$_i$ [[*PRO*]$_i$ kitaab paRh-nii] chaah-ii*
Mohan erg NULL book read-Inf want-perf
'Mohan wanted to read the book.'

The category *RELPRO* in the following example represents gaps in participial relative clauses that are used as pronominal modifiers of noun phrases:

77. *zyaadaatar [*RELPRO*] kal khul-e] darvaaze*
most-of-the NULL yesterday open-perf doors
'Most of the doors that opened yesterday'

The category *GAP-pro* in the following examples represents gaps in coordination constructions (where clauses are linked together using conjunctions such as *aur* ‘and’, *lekin* ‘but’) and gapping constructions (where the verb is missing along with one or more of its arguments):

78. *mohan-ne kitaab$_i$ paRh-ii aur [*GAP-pro*]$_i$ so ga-yaa*
M.-Erg book read-Perf and NULL sleep go-Perf
'Mohan read the book and slept.’

79. *John-ne seb$_i$ khaa-yaa aur Mary-ne bhii [*GAP-pro]$_i$ (khaayaa)*
J.-Erg apple eat-Perf and M.Erg also NULL
'John ate an apple and Mary too.'

Below is a table that summarizes the kinds of null elements that will be inserted and annotated by PropBank.

Description	**Examples**	**Label**
Empty relative pronoun	*jyaadaatar [*RELPRO*] kal khul-e] darwaaze* most-of-the NULL yesterday open-Perf doors 'most of the yesterday opened (by themselves) doors' Examples of other cases: 1. *RELPRO dillii jaane waalaa laRkaa* ‘RELPRO Delhi going one boy’ 2. *RELPRO piine kaa paanii* ‘RELPRO drinking of water’ ***3. [RELPRO_i t_i khaaye gaye] phal*** **‘**RELPRO gotten eaten fruit’ 4. *[RELPRO roTii khaane waalaa] laRkaa* ‘RELPRO bread eating one boy’ 5. *[RELPRO pro khaane waalaa] laRkaa* RELPRO pro eating one boy’ 6. *[PRO RELPRO khaane waalii] roTii* PRO RELPRO eating one bread’	*RELPRO*

Empty arguments of the verb (regular pro-drop)	**pro* kitaab paRh-egii* NULL book read-fut '(She) will read the book'. *siitaa-ne john-ko kitaab dii aur mary-ne *pro* magazine dii* S.-Erg J.-Dat book give.Perf and M.Erg NULL magazine give.Perf 'Sita gave John a book and Mary a magazine.'	*pro*
Empty arguments in coordination constructions.	*mohan-ne kitaab$_i$ paRh-ii aur [*GAP-pro*]$_i$ so ga-yaa* M.-Erg book read-Perf and NULL sleep go-Perf 'Mohan read the book and slept.' *kitaab$_i$ mohan-ne likh-ii aur [*GAP-pro*]$_i$ raam-ne paRh-ii.* Book Mohan-erg write-Perf and NULL raam-erg read-Perf. The book Mohan wrote and Ram read.	*GAP-pro*
Empty k1 Argument (control)	*mohan-ne$_i$ [*PRO*]$_i$ kitaab paRh-nii] chaah-ii* M.-Erg NULL book read-Inf want-Perf 'Mohan wanted to read the book.'	*PRO*
Missing arguments in gapping constructions	*John-ne seb$_i$ khaa-yaa aur Mary-ne bhii [*GAP-pro*]$_i$ (khaayaa)* J.-Erg apple eat-Perf and M.Erg also NULL (ate) 'John ate an apple and Mary too.'	*GAP-pro*

Null elements corresponding to 'pro' are automatically inserted at the beginning of the VGF chunk in the order Arg0, Arg2/Arg2 Arg1.

Special cases (to be filled in)

8.2. Coreference

We will co-index two types of empty categories: *PRO* and *GAP-pro*. In the case of PRO, the coindexation type is **'coref'** and the PRO is linked with its antecedent in the matrix clause.

80. *mohan-ne $_i$ [[*PRO*]$_i$ kitaab paRh-nii] chaah-ii*
M.-Erg NULL book read-Inf want-Perf
'Mohan wanted to read the book.'

We do not co-index all cases of PRO. In those sentences containing the so-called "PROarb", where the antecedent is not to be found in the same clause, no coindexation is provided for the PRO. For instance;

81. *yahAA [[*PRO*] dhuumrapaan karnaa] manaa hae.*
Here NULL spitting do forbidden is.
'It is forbidden to spit here'

We also insert indices to indicate the link between the empty category and its antecedent in coordination constructions and gapping constructions. The type of coindexation here is different from that provided for the PRO cases because here we are simply "copying" lexical material from the initial conjunct to the second conjunct. Therefore we will use a different coindexation convention here, annotating such cases as "**gapref".**

82. *mohan-ne kitaab$_i$ paRh-ii aur [*GAP-pro*]$_i$ so ga-yaa*
M.-Erg book read-Perf and NULL sleep go-Perf
'Mohan read the book and slept.'

83. *kitaab$_i$ mohan-ne likh-ii aur [*GAP-pro*]$_i$ raam-ne paRh-ii.*
Book Mohan-erg write-Perf and NULL raam-erg read-Perf.
'The book Mohan wrote and Ram read'

10. Relative Clauses

The English PropBank analysis of relative clauses used a ArgM-Link SLC label on the relative pronoun which is then linked to the head noun using a * (star) link. However, relative clauses in English also have a trace, which is what gets the actual PropBank label. Since we do not have a corresponding trace in the DS tree structure, annotators will annotate the relative pronoun with the numbered argument label directly. Second, PropBank annotators will add a link from the relative pronoun to the NP it is associated with. The relative pronoun will be annotated as ArgM-LINK-SLC (selectional constraint link). Then the annotator will select the appropriate NP node and create the link. (This is done by clicking Argument on the Jubilee menu bar followed by clicking Functions. From the options therein, select '*' (shortcut: Ctrl+Shift-8). At this point, the linked annotation should appear on the
selected NP node.)

Note: The above procedure involves annotating the relative pronoun with a numbered argument label + an ArgM-LINK-SLC label. *Currently Jubilee will not allow double-annotation of this kind*, so either the procedure or Jubilee will have to be modified. One option is that, as the relative pronoun already has a coreference link to the head noun in the DS structure, PropBank may not have to redundantly add another link. Or PBank can automatically insert the link in a postprocessing step.

The linking procedure discussed above will indicate the relationship between a noun phrase and its clausal modifier in correlative and relative clause constructions:

84. ***merii behen$_i$ [jo$_i$ dillii mein rehti hai]***, *kal aa rahi hai.*
My sister who Delhi in stay.Imp is tomorrow come PROG is.
[My sister]$_i$ [who$_i$ lives in Delhi] is coming tomorrow.

PropBank Annotation
Rel: *reh* 'stay'
Arg1: *jo* 'who'
LINK-SLC: [*jo* 'who'] * [*merii behen* 'my sister']
Arg2-Loc: *dillii mein* 'in Delhi'

85. *mai* ***[vo kitaab]$_i$*** *khariidungaa,* ***[jo$_i$ sale-pe hai]***

I that book buy.fut that sale on is

I [that book]$_i$ will buy, [which$_i$ is on sale]

PropBank Annotation

Rel: *ho* 'be'

Arg1: *jo* 'who'

LINK-SLC: [*jo* 'who'] * [*vo kitaab* 'that book']

Arg2-Loc: *sale-pe* 'on sale'

86. ***[jo kitaab$_i$ sale-pe hai]****, mai* ***[vo kitaab]$_i$*** *khariidungaa*

which book sale on is, I that book buy.fut

[which book$_i$ is on sale], I will buy [that book]$_i$.

PropBank Annotation

Rel: *ho* 'be'

Arg1: *jo kitaab* 'which book'

LINK-SLC: [*jo kitaab* 'which book'] * [*vo kitaab* 'that book']

Arg2-loc: *sale-pe* 'on sale'

87. ***[jis-ne$_i$ jo$_j$ caahaa]****,* ***us-ne$_i$ vo$_j$*** *kiyaa*

who-erg what wanted, he-erg that did

'whoever what wanted, s/he that did'

PropBank Annotation

Rel: *caah* 'want'

Arg1: *jis-ne* 'who'

LINK-SLC: [*jo kitaab* 'which book'] * [*vo kitaab* 'that book']

Arg1: *sale-pe* 'on sale'

88. *[jis-ne$_i$ jo$_j$ chaahaa], pro$_i$ pro$_j$ kiyaa*

who-erg what wanted, NULL NULL did.

'Whoever what wanted, NULL NULL did'

PropBank Annotation

Rel: *caah* 'want'

Arg1: *jis-ne* ‘who’

LINK-SLC: [*jis-ne* ‘who’] * [*pro*]

Arg2: *jo* ‘what’

LINK-SLC: [*jo* ‘what’] * [*pro*]

Note: we may need a separate mechanism to show that the argument of the matrix verb is actually the noun phrase + the relative clause; that is, in the sentence ‘the boy who is tall runs everyday’, the argument of ‘run’ is ‘the boy who is tall’ (and not just ‘the boy’). Can we assume as a default, that all the subtrees emanating from the node labeled with the numbered argument label in the DS structure will be included as part of the constituent that receives the label?

10. Passives

Sentences can be either active (The executive committee approved the new policy) or passive (The new policy was approved by the executive committee). In active sentences, the subject is the agent or a do-er of the action, marked as Arg0 in Propbank. In passive sentences, the subject of the sentence is acted upon by some other agent or by something unnamed, and is being marked as Arg1 in Propbank.

In Hindi Propbank, the demoted subject of passives is labeled as Arg0 if it is overtly realized.

11. Questions

Wh-phrases in questions are in-situ in Hindi, and are simply annotated with the argument role that would be assigned to it by the verb if it had been a regular argument. E.g. in the following example *mohan* would be annotated with Arg0.

89. *Mohan **kyaa** khaataa hae?*
Mohan what eat.Imp is
‘what does Mohan eat’

12. Special cases of topicalization

These cases involve instances where a constituent is topicalized and is then repeated, often in the form of a pronoun, and the two constituents are not already indexed in the tree. For example, ***gandhiijii, unho-ne*** *hamaare desh kii sevaa mE sabkuch balidaan kiyaa* 'Gandhiijii, he sacrificed everything in the service of our nation.'

If the rel to be annotated is *balidaan kar* 'sacrifice do' the annotator would first have to annotate the pronoun *unho-ne* 'he-erg' as ARG-1, then provide a coreference link to the pronoun's referent, *gandhiijii*. Select and annotate the pronoun in the argument position, then select the topicalized node and click Argument on the Jubilee menu bar, followed by clicking Functions. From the options therein, select '*' (shortcut: Ctrl+Shift-8). The linked annotation should appear in the TreeBank view and in the annotation view at the top of the screen.

13. Span of Annotation

13.1 Boundaries of annotation

For the purposes of PropBank annotation, annotators should only assign arguments within a certain syntactic span surrounding the rel. The structure of the tree reflects which constituents in an utterance are truly arguments of a particular predicate; thus, even when annotators feel that a constituent outside of this span has some semantic bearing on the rel, it should not be annotated. Rather, the syntactic span of annotation should be respected: everything within that span should be encompassed by an argument label (with exceptions described below), and nothing outside of that span should be annotated (with exception of linking annotation, such as that of relative clauses).

Do not tag noun modifiers (labeled ? in DS) or conjunctions (labeled ?? in DS), unless these begin the sentence and are being used in a discourse function. Do not tag auxiliary verbs or modals or light verbs. These verbs will come up for annotation and at that point the appropriate will be selected without further annotation.

Note: Are the subtrees of the node labeled with a *karaka* label always within the annotation span for PBanking purposes?

13.2 Where to place tags

[Note: Insert Jubilee-specific instructions here]

14. Using Sanchay

SSF format and its use in Sanchay

The annotated data in the treebank is stored in SSF format (Bharati et al., 2007). This format is used for representing dependency trees. The SSF is a four column format in which the first column is for address, the second column is for the token, the third column is for the category of the node and the fourth column has other features.

Address	Token	Category	Attribute-value pair
1	((	NP	<fs af=',,,,,,,' name='NP' drel='k1:VGF'>
1.1	खुशनूदा	NNP	<fs af='खुशनूदा,n,f,sg,3,o,0,0' posn='10' name='खुशनूदा'>
1.2	ने	PSP	<fs af='ने,psp,,,,,,' posn='20' name='ने'>
	))		
2	((	NP	<fs af=',,,,,,,' name='NP2' drel='k2:VGF'>
2.1	उसे	PRP	<fs af='वह,pn,any,sg,3,o,को,ko' posn='30' name='उसे'>
	))		
3	((	JJP	<fs af=',,,,,,,' name='JJP' drel='adv:VGF'>
3.1	काफी	QF	<fs af='काफी,avy,,,,,,' posn='40' name='काफी'>
	))		
4	((	VGF	<fs af=',,,,,,,' name='VGF' stype='declarative' voicetype='active'>
4.1	समझाया	VM	<fs af='समझा,v,m,sg,any,,या,yA' posn='50' name='समझाया'>
4.2	।	SYM	<fs af='।,punc,,,,,,' posn='60' name='।'>
	))		

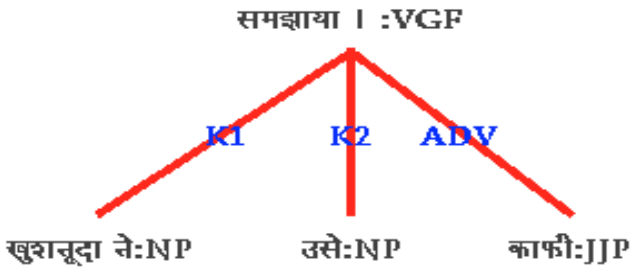
As seen from the example above, the first column, *address* consists of numbers, the second *token* shows the words in the sentence as well as some brackets. The brackets denote the boundaries of the word groups or chunks in the sentence. For instance 'खुशनूदा ने' forms one chunk and is demarcated by the brackets from other chunks 'उसे'

'काफी' and 'समझाया'.

Each of these chunks have labels, which are mentioned in the third column *category.* The chunk labels are in line with the open brackets in the second column,e.g. 'खुशनूदा ने' has a chunk label 'NP' that stands for Noun Phrase. The category column also has part of speech tags for each token in the sentence. Hence, inside the chunk 'NP', 'खुशनूदा' has the part of speech tag NNP and 'ने' has the part of speech tag PSP.

The final column, i.e. the *attribute-value pair* is most important from the PropBank point of view. In this column, there is a lot of information that is contained within angled brackets < >. The 'fs af' signals the beginning of the feature structure.

Below, we see a visual representation of the same sentence shown above in SSF format. Now, if we look at the representation in this tree, the verb समझाया is at the head of the tree and its arguments, including the adverb are its dependents. Note that the chunk 'खुशनूदा ने' is a dependent and not just 'खुशनूदा'.



This is an important point to note, relationships are shown between chunks and not individual words. In order to understand the marking of dependency relationships and also PropBank relations, this should be kept in mind.

Now, we return to the feature structure column in the SSF tree. We find that there are two different types of feature structures: at the chunk level and at the word level. Let us

examine the chunk level feature structure first.

Address Token Category Attribute-value pair
2 ((NP <fs af=',,,,,,,' name='NP2' drel='k2:VGF'>

In the token column, we see an open bracket which signals the beginning of the chunk. The chunk is labelled NP in the category column. In the Attribute-value pair column, we see name='NP2' and drel='k2:VGF'

Both these are pairs, joined together with the '=' sign. Now these pairs are made up of an *attribute*: 'name' and 'drel' each of which have some *value.* In this case the values are 'NP2' and 'k2:VGF' respectively. The attribute by itself is meaningless without a value and vice versa.

Attribute Value Attribute Value
name='NP2' drel= 'k2:VGF

Now let us look at the two attributes. The 'name' attribute is found at every node in the SSF tree. For this particular NP, its name in the tree is NP2. The drel attribute stands for 'dependency relation' and it has the value 'k2:VGF'. This means that the chunk with name 'NP2' has a relation of k2 with the chunk that has the name VGF.

Going back to the SSF tree, look at the chunk that contains the verb 'समझाया' (address 4) and it will have the name='VGF'. We show the dependency relation between the head 'समझाया' and its dependent 'उसे' by means of the attribute-value pair 'drel=k2:VGF'.

Similarly, if we wish to show PropBank relations, we will add another attribute to the feature structure at the *chunk level.* Now the feature structure will look like this:

<fs af=',,,,,,,' name='NP2' drel='k2:VGF' **pbrel='Arg1:VGF**'>

The feature structure at the *token level* looks different- it will never have an attribute like 'drel'. For example:

2.1 उसे PRP <fs af='वह,pn,any,sg,3,o,को,ko' posn='30' name='उसे'>

Here, the information enclosed in the brackets contains the word level information such as the root, number, gender, person and case. The feature structure contains the 'name' attribute but not the 'drel' or 'pbrel' attributes

For PropBank annotators, two kinds of attribute labels will be used: **pbrel** and **pbmrel**. We have seen an example of **pbrel** above and **pbmrel** is used for the null arguments that are inserted in PropBank, viz. pro, PRO and RELPRO. The attribute-value pair will be added in the same way into Sanchay. For example, the chunk label for the null arguments will be NULL__NP and the attribute label will be **pbmrel**. It can have values as pro-k1, RELPRO-k1, RELPRO-k2 and PRO-k1, GAP-pro.

8 ((NULL__NP <fs name='NULL__NP' pbmrel='pro-k1:VGF2'>

Annotation without the right values for 'pbrel' or 'pbmrel' will result in an incorrect tree with wrong relations, hence it is important to know exactly the right values for each pair in the SSF tree.

References

Semantic Role Labeling (eBook), Martha Palmer, Daniel Gildea, Nianwen Xue, In Synthesis Lectures on Human Language Technologies , ed., Graeme Hirst, Morgan & Claypool, 2010

PropBank Annotation guidelines for English, Claire Bonial, Jena Hwang, Olga Babko-Malaya, Jinho D Choi, Martha Palmer, 2009

A. Bharati, R. Sangal and D. M. Sharma. 2007. SSF: Shakti Standard Format Guide. LTRC Technical Report-33

Resources used in creating the Hindi framesets

- Unified Verb Index (http://verbs.colorado.edu/verb-index/)
- Hindi WordNet (http://www.cfilt.iitb.ac.in/wordnet/webhwn/)
- DS annotation guidelines (http://ltrc.iiit.ac.in/MachineTrans/research/tb/final_guidelines-ver2.pdf)
- Hindi Treebank (IIIT-Hyderabad database)