सार्थक
An Integrated Discourse Semantic Model for Bilingual Corpora

Deepak Sharma K Vikram Manav Ratan Mital
Amitabha Mukerjee & Achla M Raina
IIT Kanpur INDIA
{deepaks,vikram,manavrm,amit,achla}@iitk.ac.in

Abstract

This paper reports our work on Saarthaka, an integrated discourse semantic model for bilingual corpora. The first component of the model is a bilingual parser, which operates on input strings from English and Hindi, two of the structurally most disparate languages used in India. The parses are based on partial grammars of English and Hindi that have been developed using the available HPSG formalism. The multiple syntactic parses yielded by the parser are subjected to a Word Sense Disambiguator (WSD) to arrive at an ordered list of sense combinations using the WordNet. The sense combinations are processed to generate first order predicate structures, which are subsequently used for spatial analysis and discourse representation. An Acquired Discourse Information Structure (ADIS) is instantiated with the objective of processing bilingual text inputs. The ADIS draws from a knowledge base consisting of a-priori domain information and a set of inference rules to construct a semantic model of the input discourse. Results are demonstrated for English and Hindi parsing, and for English-only integration with ADIS.

Keywords : Bilingual, Semantics, Natural Language, HPSG, Discourse, ADIS, Disambiguation, Wordnet, Ontology, Representation

1 Introduction

This paper outlines our ongoing work on an integrated discourse semantic model for Hindi and English, two structurally disparate languages used in the Indian multilingual context. This work proposes innovations at several levels:

- *Multilingual Parsing* Most of the available work on natural language understanding systems focuses on devices that operate on a single language ⟦7⟧. No serious effort has been made to devise generalized approaches that would work on more than one languages simultaneously, although some success on natural language parsing has been reported on structurally related languages such as English and French, or English and German. We describe a bilingual parser which operates on input strings from English and Hindi. The parses are based on partial grammars of English and Hindi that have been developed using the available HPSG ⟦12⟧ formalism.

- *Sentence level Processing* The multiple syntactic parses yielded by the parser are subjected to a Word Sense Disambiguator (WSD) based on WordNet ⟦8⟧ to arrive at a prioritized list of sense combinations. The most credible sense combinations are processed to generate first order predicate structures, which are subsequently used for spatial analysis and discourse representation.

- *Discourse Processing* The Acquired Discourse Information Structure (ADIS) draws from a-priori domain information and a set of inference rules to construct a semantic model of the input discourse. The model relies on WordNet availability in several languages along with synset equivalence maps to provide a natural language understanding system that operates on bilingual/multilingual text inputs within limited contextual domains.

The larger aim of the effort is to provide multilingual tools for animating stories created by children and semi-literate adults ⟦10⟧. Figure 1 shows a discourse representation schema common to texts in Hindi and English.

2 Theoretical Framework

सार्थक uses the HPSG framework for grammar representation and processing. Head-driven Phrase Structure Grammar ⟦12⟧ provides a framework which is especially suitable both for purely linguistic research and for NLP applications. It rests on a meticulous and coherent formalism - the logic of typed feature structures - which is well-suited for computational applications, providing reversibility, declarativeness and the use of partial descriptions. Further, HPSG provides a formalism which is amenable to semantic treatment

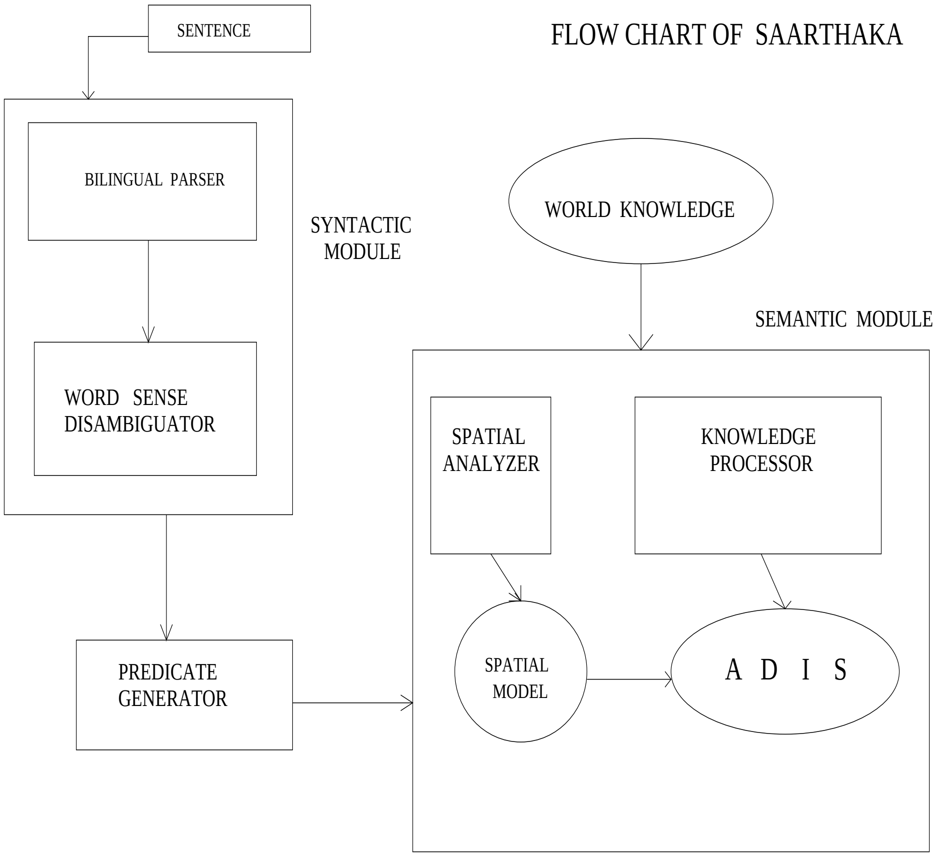

Figure 1: सार्थक model for discourse comprehension.

of linguistic input. Apart from English, which is the most comprehensively described, the HPSG framework has been adopted in studies of various other languages, e.g. German, Korean, Italian, Japanese, etc. As the second stage of its syntactic processing, it applies a word sense disambiguation ⟦4⟧ to the parsed sentences of the discourse to arrive at a list of ordered word sense combinations. This is done using HPSG and WordNet ⟦3⟧.

From the set of word sense combinations thus obtained, we generate a set of predicate structures to model the discourse semantics using first order predicate calculus. The classical technique of resolution and unification can be then applied to this discourse representation to build up what we call the Acquired Discourse Information Structure or the ADIS. This whole approach is entirely independent of whether the discourse language is Hindi or English.

2.1 Domain

At present, the work is restricted to a domain of about 400 words, which capture a typical family environment. There are enough concepts in the domain to generate rich and meaningful stories. The senses for words have been chosen using Wordnet 1.7. The Devanag transliteration scheme has been used for representing the Hindi text.

3 Parsing

Parsing involves searching for all possible phrase combinations which form a given sentence. In HPSG terms, it essentially boils down to finding the heads of all constituent phrases in a given sentence and the phrases that saturate the sub-categories of these heads. With sentences that are ambiguous all possible parses should be generated. Our system does not always assume a grammatically well-formed input; it is capable of recognizing sentences which are ungrammatical.

We apply a dynamic programming approach to the problem of parsing which essentially means that the main problem is broken down into sub problems. These sub problems are solved ensuring that the overlapping sub problems do not get re-solved. This procedure is called 'Chart Parsing' in which a chart is maintained which stores intermediate parses for constituent phrases. The procedure of parsing can be enunciated in the following steps:

1. If n is the number of tokens in a sentence, a chart of size n x n is prepared. In this chart, the position (i,j) would, at the end of the execution of the algorithm, contain all possible parses of the phrase formed by the words from i to j , both inclusive.

2. The entire procedure of parsing consists of many iterations. In each iteration, every entry of all the tokens in the sentence is taken. Its 'sub-category' object is examined to figure out if appropriate neighbouring phrases in the chart can saturate it. In case it can, a new 'Phrase' object is generated and inserted at the appropriate location in the chart.

3. In every iteration, this process is repeated. Before inserting a 'Phrase', the presence of previous instances of the same phrase is checked for. In case a previous instance exists, there is no sense in re-inserting the

'Phrase' object. Equality of phrases can easily be checked in any programming language with the concept of 'references'.

4. In a particular iteration, if no new phrases have been inserted, the program terminates at the end of that iteration.

5. It can be easily proved that this procedure will terminate. At the end of program execution, the cells in the chart contain complete phrases. Possible structures for the complete sentence can be found at location (n,n).

3.1 Grammar Generation

Grammar generation involves building lexicons and descriptions of the languages under study. English is one of the most comprehensively studied languages within the HPSG framework, and several lexicons and descriptions of the language are already available. Hindi, however, remains relatively unexplored. Therefore we have developed a lexicon of Hindi and a partial grammar based on this lexicon ranging over simple sentences and complex sentences involving embeddings and adjuncts. Even for English, we have constructed a restricted lexicon of our own, primarily because the available lexicons are too large to be integrated into a semantic treatment that is crucial to our work. Nevertheless, all of this work on Hindi and English lexicons is scalable, the sizes of the lexicons being roughly of the order of the size of the domain. Extending the system such that it works on larger domains is a task that can be accomplished simply by adding entries to the existing lexicons. We have also built interfaces that makes it easy for a user, who is not habituated to the notations that we are using, to add entries to our database. Our system affords the user the freedom of using whatever notations and features he may choose, provided he uses them consistently throughout the lexicon. This has been done in order to achieve some flexibility in the system at the user end.

The entries in the lexicon are 'Head-driven' in the spirit of HPSG, which implies that all the properties of a phrase are a function of the head of that phrase. We have also added constraints on the order of phrases/words in the sentence.

3.2 Lexicons

These lexicons are made to handle many complex constructions in the languages. In the lexicons, entries are kept such that sentences containing relative clauses, participial adjuncts, gerundival adjuncts etc. are also expected to get parsed. The system has been tested on variety of comple sentences and results were as expected.

The sample entry of the word 'dog' in our English lexicon could be something like this

`dog:dog` N(-,S,-,C) { D(S) | *J | ! }

The first field is the description of the word itself, in terms of its features. The second field is the set of features which can exist in the sub-category of this head word. The entry here means that the word `dog`, preceded necessarily by one and only one determiner(singular), and any number of adjectives would form a phrase with `dog` as the head. Similarly the word `'dogs'` would have

`dogs:dog` N(-,P,-,C) { ~D(P) | *J | ! }

The word following the colon is the root word of the one preceding the colon. This root word field is used during the semantic processing. Here one can see that the determiner is not compulsory. We have used certain such notations like '~' and '*' for the sake of convenience, so that the lexicon is easily manageable. '~' means that the feature occurring after it can occur either once or not at all and '*' means that the feature can occur any number of times. Again, the entry for `'eats'` would be

`eats:eat` V(P,S) { N(N,S,-,-) | ! | N(A,-,-,-) }

A sample entry of the word 'घर' from the Hindi lexicon would look like

घर:घरN(N,3,S,M,-,V) { ~D(S,M) | ~J(S,M) | ~k(S,M) | ! | }

The description of Hindi nouns are given with the help of six features as compared to four in English. Similarly, the subcategorization also differs at few parameters which are unique to each of the language.

Considering the entry for the word 'खाता' which would be as

खाता:खा T(-,-,S,M,1,V) { / N(N,-,S,M,1,V) | [N(A,-,-,-,1,V) | a(V)] | ~i | ~p | ~s | ~l | ~Y \ | ! }

The notations '/' and '\' imply that all features that occur inside them can occur in any order in the sentence. Such a notation is necessary for a language like Hindi where word order is not as important as in English. The notation '[' and ']' mean the only one of all the features that occur inside these markers can occur in the sub-category.

4 Word Sense Disambiguation

For word sense disambiguation, we use the head-subcat relationship, available in an HPSG parsed output [[12]], to represent combinatorial possibilities of the various senses of different words taken from the WordNet.

The semantic proximity between the HPSG head and its subcats constitutes a sense combination which is captured in the WSD lexicon as a *restriction* on the word-sense usage, and for which a credibility is assigned. The WSD lexicon is a list of *restrictions* for each sense combination involving the target word. For instance the entry for a particular sense of `bark` is

```
bark
V
(HEAD:#V:bark_767337_1)(SUB:N:person5145)=0.5
!the person barks...
```

Each term here has a three tuple representing the restriction on the usage of the head word `bark`. The notation '#V' indicates that the target word (here `bark`) is being used as a verb, and the string `bark_767337_1` means the word 'bark' with synset 767337 and sense 1. The restriction above holds on the subject and one subcategory stating that the subject of this verb should belong to the ontological class person corresponding to WordNet synset 5145. The letters V, N, A or P indicate the part of speech. Finally, the term `=0.5` reflects a "credibility" measure for this restriction (between 0 and 1). Note that the character '!' comments a line in the lexicon with usage notes.

These entries represent the co-occurance *possibilities* of the various senses of the word bark with other entities, which are represented in the form of subcats and even subcats of subcats of the head bark. Consider the sentence:

`the person barked at the baby`

considering five possible senses for bark and three for baby, results in 15 possibilities, of which the WSD routine picks out the two most credible sense combinations:

- bark = *speaking in unfriendly tone* , and
- baby may be either *a young offspring* or *the youngest member of a group*.

In the example above, we talked about an ontological class, `person5145`. These ontological tree are derived from the one in hyponymy and hypernymy relations present in the WordNet. For the words of interest to us, we have added further classes to the evolving WordNet ontology.

The procedure for the ordering of the sense combinations is as follows. One by one, each possible sense of each word in the sentence is taken and all possible combinations are considered. If the combination does not match a restriction in the WSD lexicon, it is rejected. Otherwise a credibility value is assigned. For words not in the WSD lexicon, they are assigned a default credibility.

While matching subcategorial phrases, which are maintained in the lexicon as ontological classes, matching is considered successful when the ontological class mentioned in the entry is an ancestor of the ontological class of the node in the ontological tree. If it matches, then the *credibility* that was attached to that entry is copied over to the putative node. At the end of the search, when all the *heads* in the parse tree have been visited, the *credibilities* of all the nodes are combined using a simple addition heuristic – so a particular sense combination is assigned the sum of the *credibilities* of its individual words as its overall *credibility*. Finally the list of sense combinations is ordered using this *credibility*.

5 Acquired Discourse Information Structure (ADIS)

Numerous systems for representation of world knowledge have been proposed ⟦2⟧, one of the most extensive being CYC ⟦5⟧. Extraction of semantics from a discourse in Natural Language is still regarded as a challenging job, and much remains to be learned from the processes of the human mind ⟦11⟧.

ADIS is our name for the discourse-level semantic processing tool. The ADIS maintains a formal descriptive model in First Order Logic ⟦6⟧ of nouns in the discourse in terms of attributes, states, and positional and relational properties. There are a large number of world attributes that are stored with each ontological class ⟦1⟧. ADIS also incorporates an inference engine which is able to make inferences from the F.O.L descriptors in ADIS. ADIS takes its input in predicate form (provided a-priori in the Knowledge Processor module 5.2) and updates its database. Any inferences made about the properties or attributes of objects in the discourse are also stored in the ADIS under that object. e.g. for the sentence

`Aditya saw Sharada in the room`

the attributes `saw (Aditya,in)` is stored in the object `Aditya` along with a reference to `in` which is the headword acting as a pointer to the object `Sharada` as `in(room)`. Hence, the locations and relations are clearly specified.

5.1 Predicate generation

The Predicate generation module transforms the sense combinations into first order predicate structures. The predicates in these structures are unambigous because the corresponding words in the original sentences have been tagged with their WordNet senses by the Word Sense Disambiguator. The mechanism of generation of these predicates is domain dependent and deploy several heuristics, e.g., the simple input sentence

`Aditya killed Sharada`

is `kill_980806[Aditya_99999990, Sharada_99999991]` where the numbers denote the synset numbers of the attached words. Note that proper nouns as in Aditya and Sharada above are provided with our own synset numbers. In subsequent discussion we will not make any more references to the synset numbers and assume them to be present wherever necessary.

5.2 Knowledge Processor and Representation

At the stage of parsing, the various noun objects in the discourse are identified. The ADIS is built up as the relationships between the noun objects and their states (e.g. a person object may have states hunger, alive, bleeding, etc with associated fuzzy values). The state information changes as a consequence of verb predicates defined on the objects. An example could be, that `kill[shyam, aditya]` changes the `alive` state of `aditya` to 0. We are deliberately using an overly simplistic model of inference, which would benefit by adopting non-monotonic models in the future ⟦13⟧.

The Knowledge Processor formulates the notion of a-priori knowledge to be considered before building a semantic representation of the discourse. The various "equivalents" and "implications" of certain linguistic facts expressed in first order predicate formalism are set in the database of the Knowledge Processor. These propositional facts are then unified with the discourse information to yield extra information about the discourse.

The relationships that have been considered are spatial, kinship, and action. There is an ontological tree in which every object of the discourse occupies a particular position. Depending upon that position, the object inherits various attributes. *e.g.*, `Sharada` is a `woman` and hence is a `living thing` and

an `organism`. Hence, she has the state `alive` set to 1 and possesses two hands, two legs, is capable of motion, etc. A temporal analysis module (currently very simplistic) associates a unique timestamp with each predicate structure. This helps maintain a temporal relationship between events. The current implementation merely assigns events timestamps in their order of appearance in the discourse, and devices such as flashback will only be handled in future versions. The timestamp is important in predicate generation and anaphora disambiguation.

5.3 Spatial Analysis

The system deploys an extensive spatial analysis of the discourse. The spatial modelling is done using a system of equations that model *potential field energy* for the various prepositional relationships between objects in an open space ⟦9⟧. Each prepositional relationship generates a potential field, e.g. behind(cupboard, chair), would generate a potential field behind the chair where the cupboard may be placed; the minima of this field and other attributes are decided based on the total space available and relative sizes of the cupboard, chair, etc. Additional constraints on the cupboard such as near(cupboard, window) would generate other field all of which are superimposed (potentials are added) to obtain an overall potential for positioning the cupboard.

The precise mechanism of generation of spatial descriptions is as follows: we first define an open space for the whole framework which treated as a NULL entity. Then the first noun object encountered in the discourse is placed in the center of the co-ordinate space. Subsequently, all the objects that come up in the discourse are placed at minima of the resultant potential field in the space which is obtained by the superposition of the various individual spatial potential fields. The prepositional relationship `in` is treated as a special case, it serves to changes the framework of an object (and its spatially related objects) into the object specified by the relationship.

For the purpose of spatial positioning default sizes of the various objects have been kept in the database by treating the objects as simple cuboids. Complex shape information has not been handled. This information has been maintained in the ontological tree itself and the default values can be changed whenever adjectives come up in the discourse.

6 Results

The testing of the system involves testing both the syntactic and the semantic modules.

6.1 Parser Results

The syntactic module, bilingual parser, has been successfully tested on various discourses both in English and Hindi for the purpose of parsing. The system generates multiple parses wherever possible. A story based on a well-known Indian epic Raamayana containing sentences of sufficient complexity was successfully parsed. The story may be input as follows `Raama was the son of Dasharatha. Dasharatha had three wives. Raama was affectionate to Kaikayii who was the wife of Dasharatha. Mantharaa who was Kaikayii's maid-servant instigated her against Raama. Kaikayii asked Dasharatha to send Raama to the forest. Raama, Siitaa and Lakshamana went to the forest. They lived in a hut in the forest. Raavana who was the king of Lankaa heard about the beauty of Siitaa. He went to their hut. He saw Siitaa sitting inside the hut. He abducted Siitaa carrying her to Lankaa. Raama decided to fight Raavana. Hanumaana prepared troops for Raama. Having killed Raavana in the war, Raama was pleased to meet Siitaa. Raama, Siitaa and Lakshamana returned to Ayoddhyaa. Everybody was happy.`

Let's consider the sentence He saw Siitaa sitting inside the hut. The sentence can be interpreted in two different ways. The system produces both the parses corresponding to these senses, which are shown as follows.

1.
```
+-He saw Siitaa sitting inside the hut
   +-He N(N,S,M,C)
   +-saw Siitaa sitting inside the hut
      +-saw V(S,-)
      +-Siitaa N(-,S,F,P)
      +-sitting inside the hut
         +-sitting G
         +-inside the hut
            +-inside P(N)
            +-the hut
               +-the D(-)
               +-hut N(A,S,-,C)
```
2.
```
+-He saw Siitaa sitting inside the hut
   +-He N(N,S,M,C)
   +-saw Siitaa sitting inside the hut
      +-saw V(S,-)
      +-Siitaa sitting inside the hut
```

```
+-siitaa N(-,S,F,P)
+-sitting inside the hut
    +-sitting G
    +-inside the hut
        +-inside P(N)
        +-the hut
            +-the D(-)
            +-hut N(A,S,-,C)
```

A Hindi traslated version of the story on Raamayan was tested on the system and the results were as expected. The Hindi story is as follows.

राम राजा दशरथ का पुत्र था। राजा दशरथ की तीन पत्नियाँ थी। उनकी पत्नी कैकयी राम को बहुत प्यार करती थी। कैकयी की एक दासी मंथरा थी। मंथरा ने कैकयी को राम के लिये भड़काया। कैकयी ने राजा दशरथ से राम को वनवास पर भेजने के लिये वर माँगा। राम उनकी पत्नी सीता और उनका भाई लक्ष्मण वन में गये। उन्होनें वन जाते हुए अनेक राक्षसों को मारा। वे वन में एक कुटिया में रहते थे। लंका के राजा रावन ने सीता के सौन्दर्य की बातें सुनी। रावन उनकी कुटिया में गया। उसने सीता को बैठे हुए देखा। रावन सीता का अपहरन कर लंका ले आया। राम ने रावन से युद्ध का निश्चय किया। हनुमान ने राम के लिये सेना बनाई। राम ने युद्ध में रावन को मारा। राम हँसते हुए सीता से मिला। राम सीता और लक्ष्मण अयोध्या आये। सभी प्रसन्न थे।

Consider the sentence
उन्होनें वन जाते हुए अनेक राक्षसों को मारा।
The sentence clearly has two different readings and the system is able to produce both the results showing accurate parses. The parses are as following. 1.

```
+-उन्होनें वन जाते हुए अनेक राक्षसों को मारा
    +-उन्होनेंn(V)
    +-वन जाते हुए अनेक राक्षसों को
    |   +-वन जाते हुए अनेक राक्षसों
    |   |   +-वन जाते हुए
    |   |   |   +-वन जाते
    |   |   |   |   +-वन N(A,3,S,M,-,V)
    |   |   |   |   +-जाते F(-,-,P,M,1,-)
    |   |   |   +-हुए S(-,-,P,M,1,V)
    |   |   +-अनेक D(-,P)
    |   |   +-राक्षसों N(A,3,P,M,0,V)
    |   +-को a(V)
    +-मारा A(-,-,S,M,-,V)
```

2.

```
+-उन्होनें वन जाते हुए अनेक राक्षसों को मारा
   +-उन्होनें वन जाते हुए
   |    +-उन्होनें n(V)
   |    +-वन जाते हुए
   |    |    +-वन जाते
   |    |    |    +-वन N(A,3,S,M,-,V)
   |    |    |    +-जाते F(-,-,P,M,1,-)
   |    +-हुए S(-,-,P,M,1,V)
   +-अनेक राक्षसों को
   |    +-अनेक राक्षसों
   |    |    +-अनेक D(-,P)
   |    |    +-राक्षसों N(A,3,P,M,0,V)
   |    +-को a(V)
   +-मारा A(-,-,S,M,-,V)
```

As noted earlier, the rest of the system, the WSD and the ADIS works only for English but may be adapted to work on Hindi after the availibility of a stable version of the Hindi WordNet.

6.2 Word Sense Disambiguator

The word sense disambiguator is also quite a successful module, being able to generate correct word sense combinations in case of quite a large number of cases. An example of the case that it handles is as follows: the sentence

`the dog barked at the person`

after passing through the parser and subsequently the word sense disambiguator is parsed and sense tagged as follows:

```
+-bark dog the at person the
+-bark_767468
   +-dog the
   — +-dog_1702084
   — +-the_0
   +-at person the
   +-at_0
   +-person the
   +-person_5145
   +-the_0
```

As is clear from the example, based on the context information, the system is clearly able to figure out which sense of the word bark is being referred to over here.

6.3 ADIS

The semantic module handles simple discourses such as:

`The thief was in the park near the house. He entered the house. The dog saw the thief. The dog barked at him.`

First, words such as park, bark etc have to be disambiguated. and the specific sense and its associated predicate identified. Anaphora such as `him` have to be resolved. Then, the above discourse results in the following predicate structures:

- `in[thief, park]`,
- `near[park, house]`,
- `enter[thief, house]`,
- `see[dog, thief]`,
- `bark[dog, thief]` .

The spatial model places the thief in a park whose default size is (80,80,2) at co-ordinates (40, 40, 0) (the units are in metres). The park itself is placed at (150,150,0) in NULL. The house is of size (40, 40, 30) and is placed at (150, 219, 0). Then the thief is placed in the house at (40, 42, 0) (in the frame of reference of the house) and the dog is placed at (40, 40, 0). This type of detailed reasoning is used in graphics object generation for animating the discourse.

The ontological category of the thief and dog is known and this establishes the first few facts in the ADIS about the discourse. The information is set in the ADIS under the headers corresponding to `thief` and `dog`. There are two entries created for `park` and `house` as well. There is one simple information inthe Knowledge Processor database for the information at hand - $enter[\#, \%] \& A[\%, is(container)] > P[in[\#, \%]]$ which changes the space of the thief from park to house in ADIS when the second sentence is encountered. ADIS is tested in a simple question answering module - e.g. the question `Who saw the thief?` it produces the answer `dog`.

7 Future Work

The first part of the system, *viz.* the bilingual parser, has been tested on a large number of English and Hindi sentences with a very high degree of accuracy. Now, we are working towards the possibility of extending it to a more general multilingual domain by trying to make it work for other Indian languages as well. The work from Word Sense Disambiguator onwards has been worked out for English only. This work needs to be extended to Hindi once the Hindi WordNet is available. The word sense disambiguator lexicon which can incorporate fairly complex structures, needs to be extended to cover greater vocabulary. The spatial analyzer is quite general. The semantic module has been tested on about 10 to 12 stories collected from voluntary contributors, and has worked well on them. Now, precise formulation of world knowledge in generalized contexts needs to be done. Also, a better mechanism for generation of predicate structures in general cases would be worked out.

References

[[**1**]] Ying Ding. The enabler for the semantic web. *Journal of Information Science*, 27, 2001.

[[**2**]] Scott E. Fahlman. *NETL, A System for Representing and Using Real World Knowledge.* MIT Press, 1980.

[[**3**]] Christiane Fellbaum. *WordNet: An Electronic Lexical Database.* The MIT Press, Cambridge, 1998.

[[**4**]] W. Gale, K. Chuch, and D. Yarowsky. A method for disambiguating word senses in a large corpus. *Computers and the Humanities*, 26:415–439.

[[**5**]] R. V. Guha and D. B. Lenat. Cyc: A mid term report. *AI Magazine*, 11, 1990.

[[**6**]] Michel Hess. Recent developments in discourse representation theory. *Communication with Men and Machines*, 1991.

[[**7**]] Jochen L. Leidner. Bibliography of combinatorial linguistics and natural language engineering, 1996.

[[**8**]] Xiaobin Li, Stan Szpakowicz, and Stan Matwin. A wordnet-based algorithm for word sense disambiguation. In *IJCAI*, pages 1368–1374, 1995. citeseer.nj.nec.com/li95wordnetbased.html.

[[**9**]] Amitabha Mukerjee, Kshitij Gupta, Siddharth Nautiyal, Mukesh P. Singh, and Neelkanth Mishra. Conceptual description of visual scenes from linguistic models. *Journal of Image and Vision Computing*, 18, 2000.

[[**10**]] Amitabha Mukerjee, Mahim Mishra, and Rajat Bhattacharjee. A natural language interface for generating video animation. Technical report, Indian Institute of Technology Kanpur, 1999.

[[**11**]] Makato Nagao. *Language and Artificial Intelligence.* Elsvier Science Publishers, 1987. Discusses the problem of 'Language and Mental Mechanisms'.

[[**12**]] Carl Pollard and Ivan E. Sag. *Head-Driven Phrase Structure Grammar.* University of Chicago Press, 1992. A classic text on HPSG and its computational implementation for parsing purposes,.

[[**13**]] Teodor C. Przymusinski. Three-valued non-monotonic formalisms and semantics of logic programs. In *Logic Programming and Non-monotonic Reasoning*, pages 103–106, 1990.