Multilingual Information Processing Through Universal Networking Language

Pushpak Bhattacharyya
Department of Computer Science and Engineering,
Indian Institute of Technology,
Bombay.
pb@cse.iitb.ernet.in

In this report, we describe the multilingual information processing activity going on in the Computer Science and Engineering Department at Indian Institute of Technology Bombay. The basis of all this activity is the recently proposed inter lingua called the *Universal Networking Language (UNL).* We do sentence level encoding of English, Hindi and Marathi into the UNL form and decode this information into Hindi and Marathi. Thus effectively we create a way of semi automated translation from English to Hindi and Marathi and also between Hindi and Marathi.

The encoded UNL is used for not only MT but also for various other document processing activities. The encoding process can be looked upon as the process of knowledge extraction. The extracted knowledge is used for automatic hyper linking, summarizing and categorizing of documents.

1 Introduction

The internet today has to face the complexity of dealing with *multi linguality.* People speak different languages and the number of natural languages along with their dialects is estimated to be close to 4000. Of the top 100 languages in the world, English occupies the top position, with Hindi coming fifth and Marathi fourteenth.

The Universal Networking Language [1] been introduced as a digital meta language for describing, summarizing, refining, storing and disseminating information in a machine-independent and human-language-neutral form. The UNL represents information, i.e., meaning, sentence by sentence. Sentence information is represented as a hyper-graph having concepts as nodes and relations as arcs. This hyper-graph is also represented as a set of directed binary relations, each between two of the concepts present in the sentence. Concepts are represented as character-strings called *Universal Words (UWs).* UNL vocabulary consists of:

1. Universal Words (UWs): Labels that represent word meaning.
2. **Relation Labels:** Tags that represent the relationship between Universal Words.
3. Attribute Labels: Express additional information about the Universal Words that appear in a sentence.

An UNL Expression can be seen as a UNL graph. For example,

jhaona jaao kMpnal ka AQyaxa hO]sanao Apnao inavaasasqaana maoM ek AiQavaoSana Aayaaoijat ikyaa hO.

John jo company mein adhyaksh hai usane apane nivaasasthaan mein ek adhiveshan aayojit kiyaa hai.
John, who is the chairman of the company, has arranged a meeting at his residence.

UNL for the sentence is,

```
;======================= UNL ======================
; John jo company mein adhyaksh hai usane apane nivaasasthaan mein ek
adhiveshan aayojit kiyaa hai.
[S]
mod(chairman(icl>post):01.@present.@def,company(icl>institution):02.@def)
aoj(chairman(icl>post):01.@present.@def,          John(icl>person):00)
agt(arrange(icl>do):03.@entry.@present.@complete.@pred,John(icl>person):00)
pos(residence(icl>shelter):04,          John(icl>person):00)
obj(arrange(icl>do):03.@entry.@present.@complete.@pred,meeting(icl>conference
):05.@indef)
plc(arrange(icl>do):03.@entry.@present.@complete.@pred,residence(icl>shelter):0
4)
[/S]
;==============================================
```

The UNL graph for the sentence is given in figure 1.

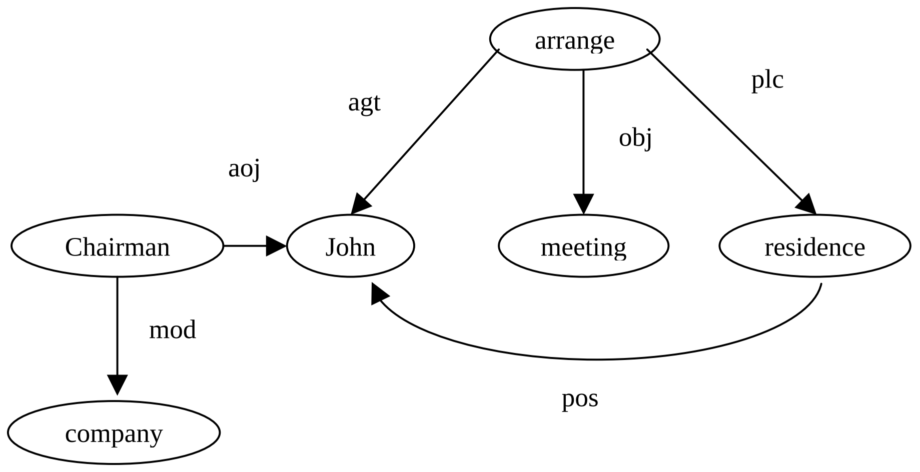

Figure 1: UNL graph

In the above, *agt* means the *agent*, *obj* the *object*, *plc* the *place*, *aoj* the *attributed object* and *mod* the *modifier.* The detailed list of such relations can be found in the reference cited above. Also the *icl* construct helps restrict the meaning of the word to single meaning.

It may be understood from the discussion above that the UNL could be a very effective vehicle for developing multilingual web based applications. The UNL expressions provide the *meaning content* of the text and search could be carried out on this meaning base instead of the text. This of course means developing a novel kind of search engine technology. The merit of such a system is that the information in one language need not be stored in multiple languages.

We mention here the Indian effort at language analysis and generation- especially of Hindi. Many systems have been developed for translation to and from Indian languages. The Anusaaraka system is based on Paninian Grammar [2] and renders text from one Indian language into another. Anusaaraka analyses the source language text and presents the information in a language, which has a flavour of the source language. It tries to preserve information from the input to the output text. For this task, grammaticality is relaxed and special purpose notation is devised wherever necessary. The aim of this system is to allow language access and not machine translation.

IIT Kanpur is involved in designing translation support systems called *Anglabharati* and *Anubharati* [3] from English to Indian languages and vice-versa and among Indian languages. The approach is based on word expert model utilizing *Karaka* theory, pattern directed rule base and hybrid example base.

A project called MaTra [4] , a human-aided translation system for English to Hindi is going on at NCST, Mumbai. The focus is on the innovative use of man-machine synergy. The system breaks an English sentence into chunks, analyses the structure and displays it using an intuitive browser-like representation, which the user can verify and correct, after which the system generates the Hindi. Ambiguities are resolved with user help as are the words not present in the dictionary.

Structured semantic knowledge can be used for a variety of applications. Clark, Thompson, Holmback and Duncan [5] have implemented a system for Boeing which is based on word relationships and uses semantic nets to increase precision of search engine query results. Senniappan and Bhattacharyya [6] have proposed the use of UNL in automatic hyperlink generation. Green [7] has also demonstrated automatic hyperlink generation using semantic similarity.

2 The Analyser System: EnConverter

Encoding into UNL is first of all a parsing problem [8]. We have used the The EnConverter [9] system (henceforth called *EnCo)* in the UNL Project, Institute for Advanced Studies, United Nations University, Tokyo for our task. EnCo is a language independent parser which provides a framework for morphological, syntactic and semantic analysis synchronously. It analyses sentences by accessing a knowledge rich **lexicon** and interpreting the **Analysis Rules**. The process of formulating the rules is in fact programming a sophisticated symbol processing machine**.** Thus the process of enconversion from natural language sentences involves constructing analysis rules and building a knowledge rich lexicon linking the language words with UWs covering the extremely varied language phenomena and concepts.

EnCo operates on the nodes of the Node-list through its windows. There are two types of windows, namely *Analysis Window* and *Condition Window*. Two current focused windows called *Analysis Windows (AW)* are circumscribed by the windows called *Condition Windows (CW).* Figure 2 shows the structure of EnCo.

EnCo uses the Condition Windows for checking the neighbouring nodes on both sides of the Analysis Windows in order to judge whether the neighbouring nodes satisfy the conditions for applying an Analysis Rule or not. The Analysis Windows are used to check two adjacent nodes in order to apply one of the Analysis Rules. If there is an applicable rule, EnCo adds Lexical attributes to or deletes Lexical attributes from these nodes, and/or creates a partial syntactic tree and/or UNL network according to the type of the rule.

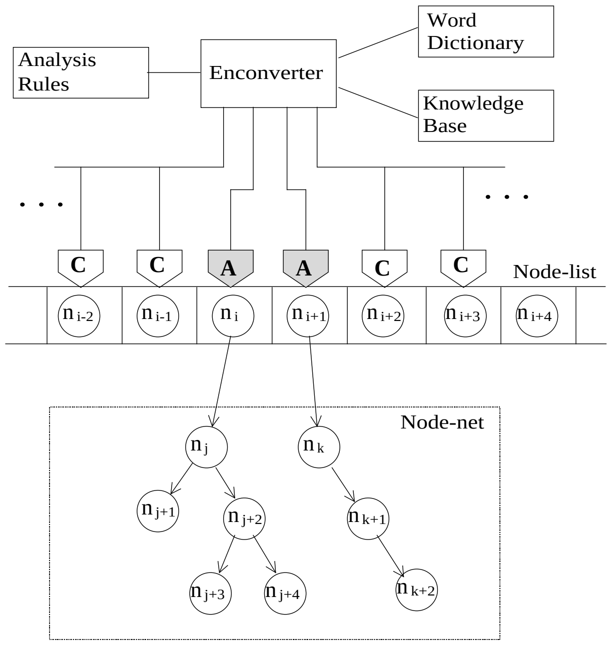

Figure 2: Structure of EnConverter
"A" indicates an Analysis Window, "C" indicates a Condition Window,
and "n_n" indicates an Analysis Node

Lexicon

The lexicon used for the system consists of mapping of Hindi words to Universal Words and lexical-semantic attributes describing the words. Any entry in the Dictionary is put in the following format.

`[HW] {ID} "UW" (ATTRIB1, ATTRIB2, ...) <FLG,FRE,PRI>;`

where,

```
HW = Head Word
ID = Identification of Head Word (Omitable)
UW = Universal Word
ATTRIB = Attribute
FLG = Language Flag (e.g., E for English)
FRE = Frequency of Headword
PRI = Priority of Headword
```

Some examples of Dictionary entries for Hindi are given below:

`[rama] {} "Ram(icl>person)" (N,P,MALE,ANI,Na,3SG) <H,0,0>;`

```
[saundr]{} "beautiful(icl>state)"(ADJ) <H,0,0>;
```

The attributes in the lexicon are collectively called *Lexical Attributes* (both semantic and syntactic attributes). The syntactic attributes include the word category- noun, verb, adjectives, *etc.* and attributes like *person* and *number* for nouns and *tense* information for verbs. The semantic attributes are derived from an Ontology DAG.

Predicate Preserving Parser

As EnCo scans the input sentence from left to right the following actions are taken at every step: *Morphological Analysis and Decision Making.* Morphology is concerned with the ways in which words are formed from basic sequences of morphemes. It acts as the crossroads between phonology, lexicon, syntax, semantics, and context. Two types are distinguished [10]:

- Inflectional Morphology
- Derivational Morphology

Inflectional Morphology defines possible variations on a root form, which in traditional grammars were given as *paradigms*, for example, calaa [calaa](walked) , cala rha hO [cal rahaa hai] (is walking), calaogaa [calegaa] (will walk), *etc.* Our approach for implementing Inflectional Morphology is different from the conventional approach. The UWs defined in the lexicon have word roots specified as Universal Words followed by appropriate restrictions while the headword has the *Longest Common Lexical Unit* (LCLU) of all the possible transformations of the word. For example, the dictionary entry for the verb calanaa [calanaa](to walk) is:

```
[cal]{}”walk(icl>do)”(List of Semantic and Syntactic
                              Attribute)<H,0,0>;
```

Where, *cal* is the LCLU for *calaa, calegee, calakara, etc.* Word suffixes for nouns, verbs and adjectives are kept in the lexicon. For example,

```
[aa]{}”aa”(VMOR)<H,0,0>;
```

EnCo selects the longest matched entry from the lexicon, starting from the first character. Thus, when EnCo consults the dictionary for a particular morpheme say *calaa*, it will be able to retrieve the LCLU (*cal* here) present in the dictionary. Then, the rules, which look ahead and signal that there is a verb suffix ahead, take control and complete the morphological analysis.

In case of decision making, according to the lexical attributes of the nodes under the two Analysis Windows, the parser decides whether they are to be combined into a single headword or a relation is to be set up between them or an UNL attribute is to be generated. While combining or modifying the two nodes, one of the nodes is deleted from the node-list. Rules ensure that the predicate or the entry node of the sentence is never deleted until the end of the analysis.

As in any parsing situation, the major decision is whether to *Shift* or to *Reduce*. Here, *Reduce* refers to deletion of a node from the Node-list after it is no longer required. Like other parsers, PPP also gives greater priority to *Shift* rules. The rule types performing the *Shift* process are **L** and **R,** while the rules performing *Reduce* are the combination (+ or -) and modification rules (< or >). There are several *Reduce* decisions:

- Generation of UNL Attributes
- Generation of Dynamic Lexical Attributes
- Generation of Relations

Generation of relations is the most important decision. According to the Lexical Attributes (both Static and Dynamic) of the nodes under Analysis Windows, the semantic relation between the nodes is decided. For example, if there is a noun under the LAW with semantic attributes- PLACE (as above) and verb under the RAW and, we decide that the relation **plt** needs to be resolved:

```
lnode:[dillee]{}“Delhi(icl>place)"(N,P,PLACE,CASEADD,PLT)<H,0,0>;
rnode:[jaa rahaa hai ]{}  "go(icl>event)" (V,MORADD,BLKINSERT) <H,0,0>;
```

The rule is:

```
   >{N,PLACE,PLT::plt:}{V,^plt:plt::}P20;
```

The rule deletes the noun, as the verb is the entry node of the sentence.

3 The Generator System: Deconverter

Deconverter (Henceforth called **DeCo)** [11] of the UNL system converts information described in UNL expression to a sentence of any target language. It is a language generation system which uses *word dictionary, co-occurrence dictionary and the generation rules* of the target language. The methodology followed for generation is as follows:

The UNL expressions, which are binary in nature, are converted into a *hyper graph*. This hyper graph structure is called the NODE-NET. The root node of the NODE-NET is the entry node, which represents the main predicate. The start and end of generation is triggered by two symbols known as the SHEAD (Sentence Head) and STAIL (Sentence Tail). Each node in the node net is related with another node by UNL relational label. When a rule is applied for solving a relational label, the UW, which was in the node of NODE-NET previously, is included in the NODE-LIST and thus generated. It means that word is included in the sentence.

The NODE-LIST will contain the words, which have been included in the final sentence by application of rules. Thus, at each step, the NODE-LIST shows which words have been added by resolving the Relational Labels. The NODE-LIST contains those words to which rules have been applied and have been included in the generation process. The following table shows the content of a node. Node information includes the headword, which is the target language equivalent of the given UW, UW itself and the grammatical attributes which correspond to the UW. For example, the UW *he* has the following structure.

Hindi equivalent	**UW**	**Grammatical Attributes**
vah	He	PRON,MALE,3SG,ANI,HPRON

PRON - pronoun
HPRON - human pronoun
MALE - gender is male
3SG - third person singular
ANI - animate

Prior to the application of rules, the system looks up the word dictionary entries which corresponds to the UW. When more than two words are found corresponding to a unique UW, the system retrieves all of them. For example, the word *work* may have two forms:

a. Noun form - kama (kAma)
b. Verb form - kama kr (kAma kar)

The generation rules are applied to the nodes in the node-list. The system applies generation rules according to the information in the node-list and inserts each node of the node-net into the node-list and then replaces each node in the node-list with the corresponding word to get the final result. This process is based the concept of *Generation Windows* (GW) and *Condition windows* (CW). The latter are used to check the neighboring nodes on both sides of generation windows to judge whether the neighboring node satisfies the condition to apply the generation rule. The generation window is used to check the applicability of one of the generation rules and take the necessary action.

4 An Example: Encoding a Hindi Sentence

Here we give an example of the anlysis of a siple assertive senetce. Assertive simple sentences have only one main clause. The analysis of such sentences is explained below with an example:

kanapur maoM [na idnaaoM bahut zMD hO.

kaanapur mein in dino bahut thand hai.

Kanpur-in these days very cold-is

It is very cold in Kanpur these days.

The Node-list is shown within “<<” and “>>”. The Analysis Windows are denoted within “[“ and “]”. Steps related to morphological analysis are not shown here. The nodes delimited by “/” are those explored and fixed by the system.

/<</[kaanapur]/[mein]/"in dino bahut thand hai"/>>/

The noun of type *PLACE* and case marker *mein* are combined and an attribute *PLC* is added to the noun to indicate that *plc* relation can be formed between the main predicate of the sentence and this noun.

/<</[kaanapur mein]/[in]/"dino bahut thand hai"/>>/

Here we right shift to put the demonstrative pronoun *in* with the succeeding noun. This is based on the observation that a noun always succeeds a demonstrative pronoun.

/<</kaanapur mein /[in]/[dino]/"bahut thand hai"/>>/

At this step, the *mod* relation is resolved between the demonstrative pronoun *in* and the noun d*ino.*

```
/<</kaanapur mein /[dino ]/[bahut ]/"thand hai"/>>/
```

The system right shifts here because there is no combination or modification rule between a noun and an adverb.

```
/<</kaanapur mein /dino /[bahut ]/[thand hai]/>>/
```

The system recognises *thand hai* as the predicate of the sentence because of the combination with *hai*. So it generates *man* relation between the adverb *bahut* and the predicate *thand hai.*

```
/<</kaanapur mein /[dino ]/[thand hai]/>>/
```

By looking at the *TIME* attribute of the noun *dino*, the system resolves *tim* relation between *dino* and the predicate.

```
/<</[kaanapur mein ]/[thand hai]/>>/
```

Similarly by using the *PLACE* attribute of the noun *kaanapur,* the system generates *plc* relation between the noun *kaanapur* and the predicate.

```
/<</thand hai/[>>]/
```

A right shift at this point brings the Sentence Tail (STAIL) under the LAW and thus signals the end of analysis. This right shift rule also attaches the attribute *@entry* to the last word left in the Node-list and thus the predicate *thand hai* is preserved till the end. The output of the system is as under:

```
;============================== UNL ============================
; kaanapur mein in dino bahut thand hai
[S]
mod(day(icl>period):0H.@pl,        these:0D)
man(cold(icl>state):0U.@entry.@present, very(icl>intensifier):0N)
tim(cold(icl>state):0U.@entry.@present,  day(icl>period):0H.@pl)
plc(cold(icl>state):0U.@entry.@present,  Kanpur(icl>place):00)
[/S]
;==============================================================
=
```

In the analyser system we have dealt with complex language phenomena like interrogation, exclamation, clauses, compounding and so on [12].

5 Some statistics

We have constructed analysers for Hindi and English and the generators for Hindi and Marathi. This needed linking English, Hindi and Marathi language strings with the Universal Words. Also the Analysis and Generation rules for these languages had to be made. We give below some quantitative information:

Number of Entries in the Hindi-UW dictionary: 65000

Number of Entries in the Marathi-UW dictionary: 5000

Number of Analysis Rules for English: ~5000

Number of Analysis Rules for Hindi: ~6000

Number of Generation Rules for Hindi: ~6500

Number of Generation Rules for Marathi: ~3000

6 Other Applications using UNL

Since the UNL expressions can be looked upon as the extracted knowledge of the documents, we have carried out research on how to use these for various cocumnet processing tasks. Notable among them are automatic hyper linking and text clustering. In the former the keywords- as candidates for setting up links from- are obtained from the UNL graphs. Heavily linked words are possible candidates for keywords. Similarly the linkage and relation label information in the UNL graphs are used for constructing the document vectors in the semantic dimension. These vectors are then processed with clustering algorithms. The experimental results are promising.

7 Conclusions

Universal Networking Language has been found to be very useful for various multi lingual information tasks as well as document processing applications. The UNL graph is looked upon as the extracted knowledge from the documents. This knowledge is used for semi automatic MT, automatic hyper linking, text clustering and such other applications.

References:

1. Uchida H., Zhu M., The Universal Networking Language (UNL) specifications version 3.0, *1998*. Technical Report, United Nations University, Tokyo, 1998.

   http://www.unl.unu.edu/unlsys/unl/unls30.doc
2. Bharati, A., Chaitanya, V., Kulkarni, A. P. and Sangal R., Language Access : An Information Based Approach, International Conference on Knowledge Based Computer Systems, Mumbai, 2000.
3. Sinha, R.M.K., Machine Translation: The Indian Context, in International Conference on Applications of Information Technology in South Asian Languages, AKSHARA'94, New Delhi, 1994.
4. Rao, D., Mohanraj, K., Hegde, J., Mehta, V. and Mahadane, P, A Practical Framework for Syntactic Transfer of Compound-Complex Sentences for English-Hindi Machine Translation, International Conference on Knowledge Based Computer Systems, Mumbai, 2000.
5. Clark, P., Thompson, Holmback, H., Duncan, L., Exploiting a Thesaurus-Based Semantic Net for Knowledge-Based Search, 12th Conf on Innovative Applications of AI (AAAI/IAAI'2000), 2000.
6. Senniappan, K. and Bhattacharyya, P., Automatic Generation of Hyperlinks using Semantic Information, International Conference on Information Technology (CIT 2000), Bhubaneswar, 2000.

7. Green, S., Building Hypertext links by computing Semantic Similarity, IEEE Transactions on Knowledge and Data Engineering, Vol. 11-5, 1999.

8. Earley, J., An Efficient Context-free Parsing Algorithm, Communications of the ACM, 13(2), 1970.

9. EnConverter Specification Version 2.1, UNU/IAS/UNL Centre, Tokyo 150-8304, Japan, 2000.

10. Hutchins, W. J. and Somers, H. L., An Introduction to Machine Translation, Academic Press, 1992.

11. DeConverter Specification Version 2.1, UNU/IAS/UNL Centre, Tokyo 150-8304, Japan, 2000.

12. Shachi Dave and P. Bhattacharyya, Knowledge Extraction from Hindi Texts, Journal of Institution of Electronic and Telecommunication Engineers, 18(4), July, 2001.