English to UNL (Interlingua) Enconversion

Manoj Jain and Om P. Damani
Department of Computer Science and Engineering
Indian Institute of Technology Bombay, India
manojk@it.iitb.ac.in, damani@cse.iitb.ac.in

Abstract

We describe a system for converting English sentences into expressions of an interlingua called Universal Networking Language (UNL). UNL represents knowledge in form of semantic network, where nodes represent concepts and links represent semantic roles between concepts. UNL nodes also contain semantic attributes like number, tense, aspect, mood, negation etc. Our system uses a lexicalized probabilistic parser to get the typed dependency tree and the phrase structure tree for a given English sentence. The system then converts dependency relations into UNL relations and attributes based on the POS tags of the words involved in the relation, and their semantic attributes obtained from the Princeton Wordnet. UNL hypernodes called scopes are generated by considering the relative positions of the words in the phrase structure tree. Correct handling of UNL scopes is a distinctive aspect of our work. We are not aware of any other enconversion system that attempts generating scopes, which are essential for the eventual deconversion of the UNL into some other natural language. We measure the accuracy of our system by computing the BLEU score on the Hindi sentences generated from the UNL. On 60 sentences taken from a real life agricultural corpus, we achieve a BLEU score of .26 compared to a BLEU score of .33 for the manually generated UNLs, showing the promise of our approach.

1. Introduction

In interlingua based machine translation, a source language sentence, is transformed into a language independent interlingual representation. A target language sentences is then generated out of the interlingual representation. Given N languages, this method requires N enconversion and N Deconversion modules compared to N^2 modules needed in the normal analysis, transfer, and generation approach [4].

Universal Networking Language [1] is a relatively new interlingua which was proposed in mid 90s and was undergoing revisions till 2005. The process of converting a source language (natural language) expression into the UNL expression is referred to as "enconversion". The process of converting UNL expressions into a target language representation is called "deconversion".

2. UNL Structure

UNL is composed of three main elements: Universal Words (UWs), relations, and attributes. UWs are inter-linked with other UWs to form a UNL expression corresponding to a natural language sentence. These links, called relations, specify the role of each word in a sentence. UWs can also be annotated with *attributes* like *number*, *tense, etc.*, which provide further information about how the concept is being used in the specific sentence. Of special significance is the @entry attribute, typically attached to the main predicate. Consider the English sentence below in Example 1, and its UNL expression. A visual representation of this UNL expression is given in Figure 1.

Example 1: John worked specially for the social fund.
[UNL]
agt(work(agt>human).@past.@entry, John(iof>person))
man(work(agt>human).@past.@entry, specially)
pur(work(agt>human).@past.@entry, fund(icl>money))
mod(fund(icl>money), social(aoj>thing))
[/UNL]

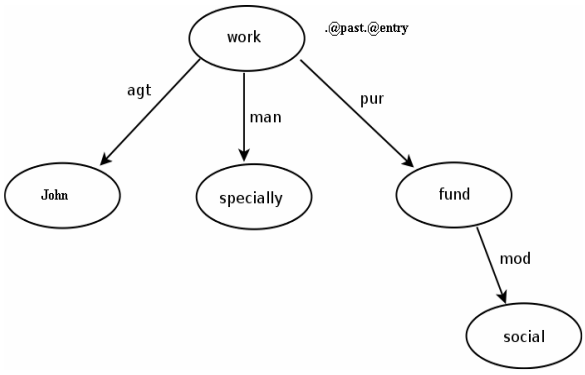

Figure 1: UNL graph for Example 1

Here *agt* (agent), *man* (manner of the action), *pur* (purpose) and *mod* (modifier) are the UNL relations. *work*(*agt*>*human*), *specially, fund*(icl>*money*) etc. are

the Universal Words. These UWs have restrictions mentioned in parentheses for the purpose of denoting a unique sense. Here *icl* stands for inclusion and *iof* stands for instance of.

2.1. UNL Scopes

UNL represents coherent sentence parts (like clauses and phrases) through Compound UWs, also called scope nodes. These scope nodes are like graphs within graphs. These sub graphs have their own environment and the @entry node. UNL graph for the sentence in Example 2 is given in the Figure 2.

Example 2: Funding for the first stage will be provided by government administrations and corporate sponsors.

The phrase "*government administrations and corporate sponsors*" is considered as being within a scope. The scope is given a compound UW ID:03 to denote a separate environment of knowledge representation. The information for number, tense, aspect, mood, negation, etc., are represented using UNL attributes while gender and language specific morphological attributes like- vowel ending of nouns, adjectives, verbs, etc., are stored in the UNL-Target language dictionary.

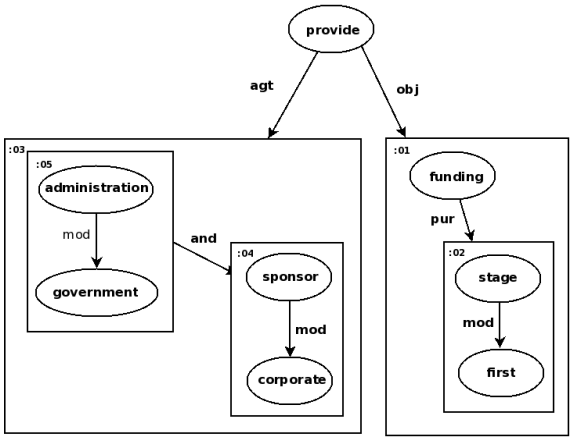

Figure 2: UNL graph for Example 2

2.2. Our System

In this paper we present our work on Enconverter for English. Our system uses a lexicalized probabilistic parser to get the typed dependency tree and the phrase structure tree for a given English sentence. The system then converts dependency relations into UNL relations and attributes based on the POS tags of the words involved in the relation, and their semantic attributes. UNL hypernodes called scopes are generated by considering the relative positions of the words in the phrase structure tree. Correct handling of UNL scopes is a distinctive aspect of our work. We are not aware of any other enconversion system that attempts generating scopes, which are essential for the eventual deconversion of the UNL into some natural language sentence.

The motivation for our work comes from [2] which used the concept of Semantically Relatable Set (SRS). We recognized that the information obtained from SRS can be simply obtained by using a dependency parser. Unlike SRS, dependency parsing is an active area of research, and hence we can benefit from the efforts of other researchers in the field. Since several dependency parsers are publicly available, we decided to opt for the dependency parsing route. Still, our rule formats are very much inspired by the rule formats in [2].

3. Related Work

The UNDL Foundation provides a Universal Parser[2] that takes an annotated natural sentence as input and generates a UNL graph as output. The annotations required are the UNL attributes and relations. Hence, in effect, it takes a linearized UNL graph and delinearizes it. Thus, this parser merely reduces the problem of generating UNL graph to that of generating linearized graphs.

Other than Semantically Relatable Sequence (SRS) based approach presented in [2] and [8], hardly any public information exist on Enconversion. A semantically relatable sequence (SRS) of a sentence is a group of words in the sentence, not necessarily consecutive, that appear in the semantic graph of a sentence as linked nodes. For example consider the sentence, "*The professors made comments on the paper.*" Some of the SRSes for this sentence are (made, comments), (professors, made), (comments, on, paper). In this approach English text is first converted into SRS, and then SRS is converted to UNL.

4. Architecture of our English to UNL Enconverter

The architecture of our system is shown in Figure 3. English to UNL enconversion process consist of six phases. Out of these phases parsing is done by the Stanford Parser [5]. Stanford Parser gives two types of parse trees: phrase structure tree and dependency tree. These parse trees are converted into UNL expressions by using rule bases. Before describing each of the six phases in detail we will first describe the Stanford Parser.

4.1. Parsing

In Stanford Parser, both semantic (lexical dependency) and syntactic (PCFG: probabilistic context free grammar) structures are scored with

separate models. It produces two types of parse trees: phrase structure tree and dependency tree. A typed dependency parse represents dependencies between individual words in a sentence with grammatical relations as labels, such as *subject* or *object*. Stanford Parser generates typed dependency parse tree from the phrase structure parses. An example is given next. Consider the sentence in Example 3, its output phrase structure tree (bracketed form), and the typed dependency parse tree. First word in each grammatical relation is the head and the second word is dependent. Each word is given a unique number.

Example 3: This will reduce the spread of germs and contagious diseases.

Phrase Structure Parse

```
(ROOT
    (S
     (NP (DT this))
     (VP (MD will)
      (VP (VB reduce)
       (NP
        (NP (DT the) (NN spread))
        (PP (IN of)
         (NP
          (NP (NNS germ))
          (CC and)
          (NP (JJ contagious)
           (NNS disease)))))))
    (. .)))
```

Dependency Parse

```
nsubj(reduce-3, this-1)
aux(reduce-3, will-2)
det(spread-5, the-4)
dobj(reduce-3, spread-5)
prep_of(spread-5, germ-7)
amod(disease-10, contagious-9)
conj_and(germ-7, disease-10)
```

Forty-eight grammatical relations used in Stanford Parser are arranged in a hierarchical manner, rooted with the most generic relation as *dep* (dependent). When the relation between a head and its dependent can be identified more precisely, relations further down in the hierarchy can be used. For example *dep* (dependent) relation can be specialized to *aux* (auxiliary), *conj* (conjunct), or *mod* (modifier).

4.2. Preprocessing multi-word prepositions

For multi-word prepositions like "according to", Stanford Parser does not give correct dependency parse as output. So we first identify these multi-word prepositions by looking in the list of multi-word prepositions obtained from [2], then clubbing them and giving a tag IN (Preposition or subordinating conjunction). Input sentence "*Fertilizers should be given according to the soil examination*." will be preprocessed and input to the parser will be "*Fertilizers should be given according-to/IN the soil examination.*" This is allowed because Stanford Parser can take partially POS-Tagged input.

4.3. Parse Tree Post Processing

While multi-word prepositions needed preprocessing because parser could not handle it correctly, certain post processing is also needed even with a correct parse tree because of multi-word nouns, phrasal verbs etc. In this phase some modification takes place on dependency parse of the sentence. Some of these modifications are as follows:

4.3.1. Multi-Word Nouns: Stanford Parser itself recognizes multi-word nouns and produces *nn* grammatical relation for them. Since in UNL, proper nouns form a single UW, we club parts of proper nouns together by looking at the POS tag of the word in *nn* relations. If both are NNP (Proper noun), then they will be clubbed together to give a single word. As shown in the Example 4, in the dependency parse, there is a grammatical relation "nn(Singh-2, Udai-1)". And both the word *Singh* and *Udai* are tagged as proper noun (NNP) in the phrase structure parse. So they will be clubbed together to get a single word '*Udai Singh*'. For common noun multi-word, a lookup is performed in wordnet. If the multi-word is present in the wordnet then we club them.

Example 4: Udai Singh and his family had wisely moved to the safety of the nearby hills.

Parts of Dependency Parse

```
nn(Singh-2, Udai-1)
nsubj(move-8, Singh-2)
```

Modified Dependency Parse

```
nsubj(move-8, Udai Singh-2)
```

4.3.2. Phrasal Verbs: Parser produces *prt* grammatical relation for phrasal verbs. So we club them together to give a single word. As shown in the Example 5 below, there is a grammatical relation *prt* between *pick* and *up*. So we club them to get a phrasal verb '*pick up*'.

Example 5: He picked up the book cheerfully.

Dependency Parse

```
nsubj(pick-2, he-1)
prt(pick-2, up-3)
```

Modified Dependency Parse

```
nsubj(pick up-2, he-1)
```

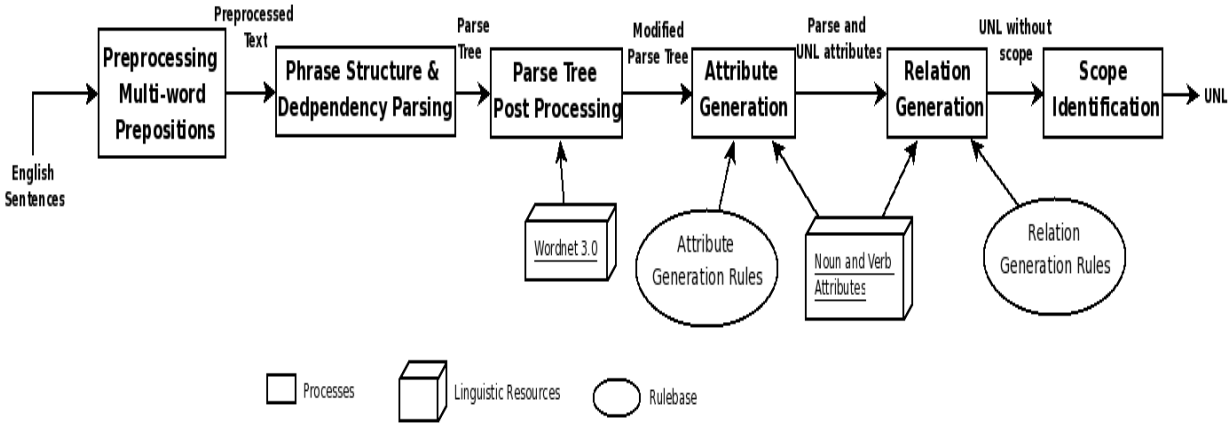

Figure 3 English->UNL Encoverter Architecture

Aux String	Type	UNL Attributes
MD will VB	Simple	.@future
VBZ have VBN be VBN	Simple	.@present.@complete.@passive
VBZ have VBN be VBN	Interrogative	.@interrogative.@present.@complete.@passive

Table 1 Syntax of Auxiliary Conversion Rules

Grammatical Relation	Head Word Attributes	Dependent Word Attributes	Head Word	Dependent Word	UNL Relation	UW1	UW2	UW1 Attributes	UW2 Attributes
prep_in	-	:PLACE	-	-	plc	1	2	-	-
prep_in	VB	:ABS	-	-	scn	1	2	-	-
prep_in	VB	:TIME	-	-	tim	1	2	-	-
conj_but	-	-	-	-	and	1	2	@contrast	-
xcomp	VB:Intransitive	-	-	-	pur	1	2	-	-
nsubj	VB:UnErgBe	-	-	-	aoj	1	2	-	-
nsubj	VB:UnErgDo	-	-	-	agt	1	2	-	-

Table 2 Syntax of Relation Generation Rules (UnErgBe: Unergative Be type Verb, UnErgDo: Unergative Do type Verb, ABS: Abstract)

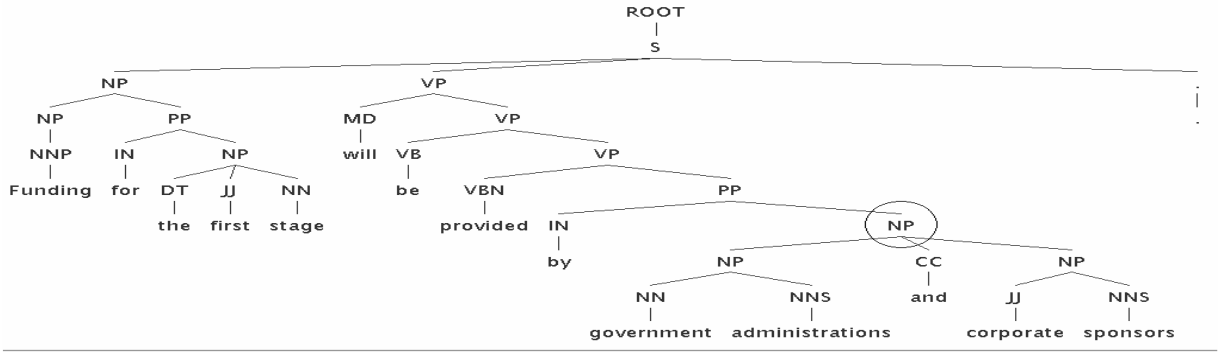

Figure 4: Phrase structure tree for Example 1

4.3.3. Relative Clauses: When there is a relative clause in a sentence and two clause are attached with relative pronoun (that, which, what etc.) or wh-word (when, where) then the dependency parse of that sentence contains a relation *rcmod* (relative clause modifier) between the heads of the two clauses. The parser also produces either *nsubj*, *dobj,* or *advmod* between the head of the second clause and the relative pronoun or wh-word as shown in the Example 6. The dependency parse is modified in a way so that pronouns or wh-words are replaced with their antecedents.

There are three cases as shown below. In all cases *rcmod* relation will be deleted.

1. If second relation is *nsubj* or *dobj*, then relative pronoun in *nsubj* or *dobj* is replaced with the head of the *rcmod* relation. As shown in the Example 6, the modified dependency parse contains only *nsubj* relation with dependent *quality*.
2. If second relation is *advmod* and word attaching two clause is *when* then *advmod* dependency relation will be changed to *tmod* (temporal modifier) and also dependent of the *advmod* will be changed to head of the *rcmod*.
3. If second relation is *advmod* and word attaching two clause is *where* then *advmod* dependency relation will be changed to *plcmod* (place modifier) and also dependent of the *advmod* will be changed to head of the {*rcmod*}.

Example 6: This knowledge implies reflection about the common ground between all individuals as well as the qualities that differentiate them.

Dependency Parse
nsubj(differentiate-18, that-17)
rcmod(quality-16, differentiate-18)

Modified Dependency Parse
nsubj(differentiate-18, quality-16)

4.4. Attribute Generation

In this phase two types of attributes are generated: morphological attributes and attributes from auxiliary verbs.

4.4.1 Morphological attribute: Morphological attribute @pl (plural word) is generated based on the POS tag of the word. If it is NNPS (Proper noun, plural) or NNS (Common Noun, plural), then @pl should be attached with the word. As shown in the Example 3 *germs* and *diseases* are tagged as NNS, hence @pl should be attached with both the words.

4.4.2. Attributes from auxiliary verbs: Parser generates two types of dependency relations for auxiliary verbs, *aux* (auxiliary) and *auxpass* (passive auxiliary). In Example 3, presence of "aux(reduce-3, will-2)" in dependency tree shows that the sentence contains a auxiliary verb *will* for the main verb *reduce*. Auxiliary verbs can be used for generating attributes describing speaker's view on aspects of event (@progress, @complete etc.), attributes describing time with respect to the speaker (@present, @past etc.) and attributes describing speaker's attitudes (@imperative, @interrogative). For finding the exact attribute all auxiliaries and their POS (part of speech) tag is used with the main verb's POS tag (if any).

1. Rules for generating the attributes are given in Table 1. Here "Aux String" represents the string to be matched. Let us look at the rule in the second row which says there should be two auxiliaries, *have* and *be* with POS tag VBZ (Verb, 3rd person singular present) and VBN (Verb, past participle) respectively and main verb should have POS tag VBG (Verb, gerund or present participle). Type represents type of the sentence: simple, interrogative, or imperative. And "UNL attributes" shows UNL attributes generated for the word.

In Example 3, modal *will* is followed by a verb, and hence the sentence will match "MD will VB" as given in rule 1 in Table 1. Hence as per the rule, @future will be attached to the word *reduce*.

Interrogative and Imperative sentences: For interrogative sentences, parser produces a clause level tag SBARQ (Direct question introduced by a wh-word or wh-phrase) or SQ (Inverted yes/no question, or main clause of a wh-question, following the wh-phrase in SBARQ) in the phrase structure tree. By looking at these tags interrogative sentences can be identified as shown in the Example 7 below.

Example 7: How do I sell this product?

Phrase Structure Parse

```
(ROOT
  (SBARQ
    (WHADVP (WRB how))
    (SQ (VBP do)
      (NP (PRP I))
      (VP (VB sell)
        (NP (DT this) (NN product))))
    (. ?)))
```

In case of imperative sentences (Example 8), the first word of the sentence will have the tag VB (Verb,

Imperative). All other sentences will be simple sentences.

Example 8: Use pure seeds to prevent the disease.

4.5. Relation Generation

In this stage every grammatical relation is converted into UNL relations and attributes. Rules for conversion use semantic attributes of the words[1], POS tag, and word itself. Syntax of the rules is given in Table 2.

For e.g. if the relation is 'prep_in' and the dependent word has attribute 'PLACE' then the relation should be converted into *plc* relation. As shown in the Example 9 below, word *area* has 'PLACE' attribute, hence it should be converted into *plc* relation.

Example 9: Do not let the she-goats to feed in the disease-infected area.
prep_in(feed-7, area-11) → plc(feed,area)

4.5.1 Residual attribute generation: While most of the attributes are generated in the attribute generation phase, some attributes are generated in this phase. As shown in the fourth rule of the Table 2 for *conj_but* relation @contrast attribute is also attached to UW1.

4.6. Scope Identification

As discussed in Section 2.1 scope is a mechanism used in the UNL format to express compound concepts in a sentence as well as coordinating concepts. Clauses can be considered as compound concepts and these are usually marked with a scope.

For identification of scope, UNL relations are divided into two types of relations:

Cumulative relations: Cumulative relations include *and*, *or* and *mod*. Let us say node n3 is the first common parent of node n1 and n2 in a phrase structure tree. If the node n1 of the UNL graph has a cumulative relation r with node n2, which then other relations on n1 are processed in this way:

1. All relations which are not r and fall below node n3 should have been processed earlier in recursive way.
2. All relation r which falls below n3 are grouped with r.
3. All other relations should be processed later in recursive way.

Other relations: All other relations fall in this category. When there is a relation from node A to node B of this type, and node B also have some outgoing relations (1 or more), then all descendent nodes of B and B itself are grouped together to give a scope.

We next explain our scope identification algorithm with the help of an example. For Example 10, the dependency parse is given below and the phrase structure tree is given in Figure 4.

Example 10: Funding for the first stage will be provided by government administrations and corporate sponsors.

Dependency Parse
nsubjpass(provide-8, Funding-1)
det(stage-5, the-3)
amod(stage-5, first-4)
prep_for(Funding-1, stage-5)
aux(provide-8, will-6)
auxpass(provide-8, be-7)
nn(administration-11, government-10)
agent(provide-8, administration-11)
amod(sponsor-14, corporate-13)
conj_and(administration-11, sponsor-14)

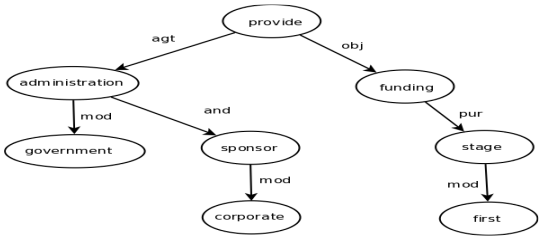

Figure 5: UNL graph without scope for Example 10

The output UNL graph after the relation generation stage is shown in Figure 5. Now we have to identify scopes in this UNL graph. Look at the *and* relation between *administration* and *sponsor*. According to our algorithm, there should be a scope containing the *and* relation and its arguments. But before this we have to look for the other relation attached with the arguments of the *and* relation. These are *government* and *corporate*. In the parse tree we have to look if these nodes fall below the first common parent of *administration* and *sponsor*. In the parse tree first common parent of *administration* and *sponsor* is the circled NP and both *government* and *corporate* fall below the common parent NP. Hence both should be processed first and it will result in scope :04 and :05. Now scope :03 will combine scope :04 and :05 with the *and* relation. The UNL graph after scope generation stage is shown in Figure 6.

If there was some other node not falling below the common parent NP then that would have been processed after scope :03. For example, consider Example 11 and phrase tree showed in Figure 7. Here there is an *and* relation between *restaurant* and *cinema*

[1] We are getting semantic attributes for noun using Princeton Wordnet [10] and the list of ergative verbs from Wikitionary [11].

and in the parse tree first common parent of *restaurant* and *cinema* is circled NP. But *Maharastra* does not fall below the first common parent NP. Hence it should be processed after the *and* relation is taken into a scope :2. The corresponding UNL graphs are shown in Figures 8 and 9.

Example 11: This has been started at restaurants and cinemas of Maharashtra .

Dependency Parse

nsubjpass(start-4, this-1)
aux(start-4, have-2)
auxpass(start-4, be-3)
prep_at(start-4, restaurant-6)
conj_and(restaurant-6, cinema-8)
prep_of(restaurant-6, Maharashtra-10)

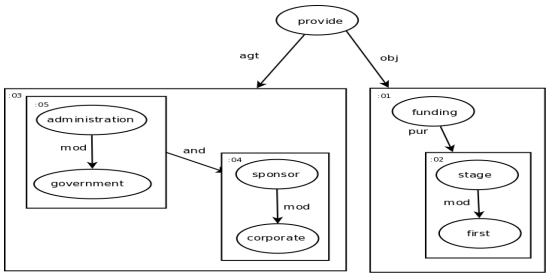

Figure 6: UNL graph (with scope) for Example 10

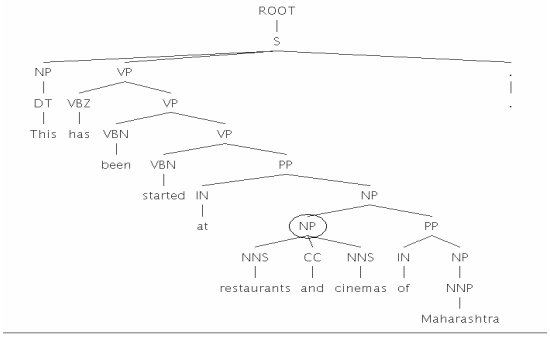

Figure 7: Phrase structure parse tree for Example 11

5. A Complete Example

We will take Example 3 and show step-by-step details of UNL generation. Because there are no multi-word prepositions, so no preprocessing is done. After parsing, the output parse were given earlier with the example itself. No parse modification will take place because there are no *nn*, *prt* or *rcmod* grammatical relations. Attribute generation will result in @pl for *germ* and @pl for *disease*. In relation generation phase *nsubj* will converted into *agt* relation, *det* will give @def attribute for *spread*, *dobj* will change into *obj* relation, *prep_of* will change into *mod* relation, *amod* will also change into *mod* relation, and *conj_and* will change into *and* relation. The final UNL expression after scope generation is given below. UNL graph for this is given in Figure 11.

[UNL]
obj(reduce.@future.@entry, :01)
agt(reduce.@future.@entry, this)
mod:01(spread.@def, :02)
and:02(germ.@entry.@pl, :03)
mod:03(disease.@pl, contagious)
[/UNL]

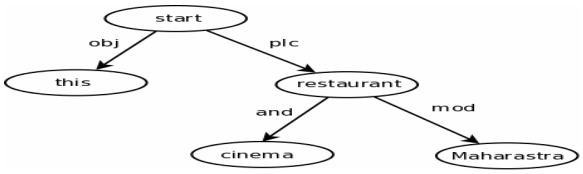

Figure 8: UNL graph without scope for Example 12

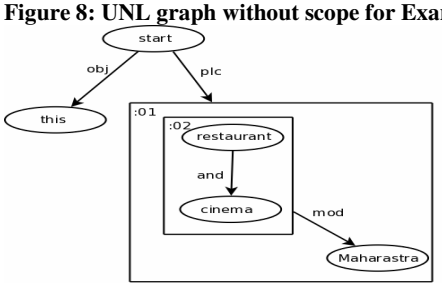

Figure 9: UNL graph with scope for Example 12

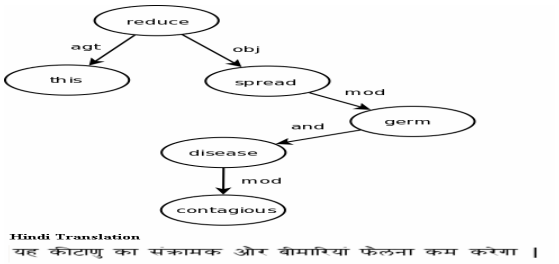

Figure 10: UNL graph and Hindi translation (without scope) for Example 3

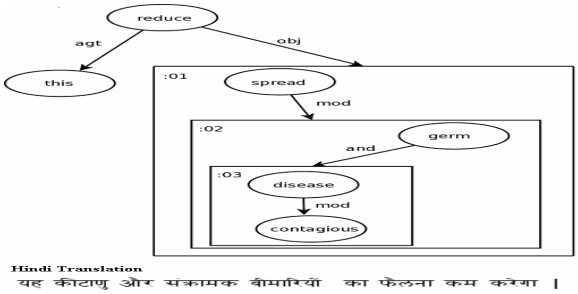

Figure 11: UNL graph and Hindi translation (with scope) for Example 3

5.1. Impact of generating scopes: To look at the impact of generating scopes in translation, we show Example 3 and its Hindi translation obtained by using Hindi Deconverter [6]. UNL graph without scope and with scope for the Example 3 is given in Figure 10 and Figure 11 respectively. As shown, the output translation has really improved after generating scopes.

6. Evaluation

Since there are not many published enconverters, hence there is no standard evaluation methodology. In [2], precision and recall metrics are defined for the enconversion task. The quality of enconversion is measured by the number of entries that match between a gold standard UNL and the one generated by the system. An entry is said to be matched, if the UNL expression finds the UW1, UW2 and the relation correctly. For example, entries 1 and 2 below match but entries 1 and 3 do not match.

1 agt(reduce.@future.@entry, this)
2 agt(reduce.@future.@entry, this)
3 aoj(reduce.@future.@entry, this)

However, we are uncomfortable with this definition for two reasons. First, it completely ignores the issue of scope. Second, the purpose of UNL generation is to facilitate machine translation. It is not clear how the precision and recall measures correlate with the final translation quality. We measure the accuracy of our system by computing the BLEU score on the Hindi sentences generated from the UNL.

On 60 sentences taken from a real life agricultural corpus, we achieve a BLEU score of .26 compared to a BLEU score of .33 for the manually generated UNLs, thus showing the promise of our approach. Since we do not have access to the SRS based system, we cannot compare our results with theirs.

6.2 Error Analysis

The main reason for UNL generation failure is the incorrect parsing in case of attachment ambiguity and stranded prepositions.

Another reason is the absence of semantic information for a large number of verbs. We do not have access to the kind of information discussed in [9]. For verbs we are only using the information whether verb is ergative or not. If some more properties of verbs are used then accuracy of the system can improve significantly. For example, dependency parse of sentence "*He will appear in court.*" has a dependency "nsubj(appear-3, he-1)". Here *nsubj* can be converted into *agt* or *aoj* relation based on the type of verb. If it is "*unergative do type verb"* then output UNL relation will be *agt*, if it is *"unergative be type verb"* then output relation will be *aoj*. Correct UNL relation can be generated, using this information. From [11] we have found a list of 200 ergative verbs but there are thousands of other verbs in English. We plan to prepare a detailed analysis of verb properties from [7].

7. Conclusions

A new approach for enconverting English sentences into UNL is proposed, designed, and implemented. On deconverting the generated UNLs, we achieve a BLEU score of .26 compared to a BLEU score of .33 for the manually generated UNLs, showing the promise of our approach. The system still needs many improvements like handling parser errors, improving rule base, compiling verb properties etc.

Acknowledgements

We would like to give our sincere thanks to the SRS based encoverter team. This sytem remains our main source of inspiration. We also want to specially thank Amit Sangodkar for all his help with Hindi Deconverter.

8. References

[1] Universal Networking Digital Language Foundation. http://www.undl.org/

[2] R. Mohanty, A. Dutta and P. Bhattacharyya, Semantically Relatable Sets: Building Blocks for Repesenting Semantics, MT Summit, 2005.

[3] Universal Parser, UNDL Foundation. http://www.undl.org/unlsys/uparser/UP.htm

[4] Hutchins W., Somers H. (1992). An Introduction to Machine Translation, Academic Press, New York

[5] D. Klein and C. Manning. "Fast Exact Inference with a Factored Model for Natural Language Parsing." NIPS 2002, 2003,

[6] S. Singh, M. Dalal, V. Vachhani, P. Bhattacharyya, O. Damani. Hindi Generation from Interlingua (UNL), Machine Translation Summit XI, 2007.

[7] Beth Levin. English Verb Classes and Alternations: A Preliminary Investigation. U. Chicago Press, 1993.

[8] S. Dave, J. Parikh and P. Bhattacharyya, Interlingua Based English Hindi Machine Translation and Language Divergence, Journal of Machine Translation (JMT), Volume 17, September, 2002.

[9] R. Mohanty and P. Bhattacharyya, Lexical Resources for Semantics Extraction, LREC 2008.

[10] Princeton Wordnet. http:// wordnet.princeton.edu/

[11] Wikitionary, http://en.wiktionary.org/wiki/Category:English_ergative_verbs