B.Tech Project Report

A Multi-Lingual, Meaning Based Search Engine

Submitted in partial fulfillment of the requirements
for the degree of

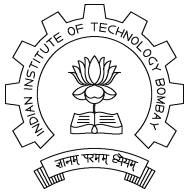
Bachelor of Technology

by

Sarvjeet Singh
99005029

under the guidance of

Dr. Pushpak Bhattacharyya



Department of Computer Science and Engineering
Indian Institute of Technology
Bombay

Acknowledgment

I would like to thank **Dr. Pushpak Bhattacharyya** for his invaluable support, encouragement and guidance, without which my BTP would have been an exercise in futility.

Abstract

In this report, a meaning based search engine that can be used as a multi-lingual platform for all sorts of search queries is presented. Universal Networking Language (UNL) is used as the underlying communicator. We try to surpass the language barrier at the World Wide Web (WWW) level. WWW is the largest repository of knowledge known and a language gap here is obviously a big drawback. Although, this hiatus is surmountable and the search engine is an early effort in this direction.

We discuss the need to develop a meaning based, inter-lingual search engine and its underlying principles. A brief discussion of UNL, the underlying intermediate language, is also given. We present the model, explaining its various modules and their implementations. Furthermore, the strengths of our search methodology, with respect to other techniques, are highlighted.

Contents

Chapter 1

Introduction

1.1 Introduction

Internet plays an important and at times vital role, in the day-to-day functioning of our life. The amount of information on the Internet has grown exponentially over the last few years. The vastness of knowledge available on the WWW is the cause of our ever-increasing vulnerability to "not the best" knowledge available. As is known, search engines are the largest contributors towards the mining of knowledge from the Internet. With the use of search engines, Internet users can quickly locate information they want from the vast amount of knowledge available on the Internet.

Unfortunately, most of the current search engines are monolingual. If multilinguality can be introduced in a search engine, it will completely revolutionize the information access scenario. Agro Explorer was established keeping the same goal in mind.

This project is being undertaken at Media Labs Asia, Maharashtra Hub. It envisions creating a system, which can provide the Indian Farmer access to the vast amount of agricultural information available on the Internet. The hurdles in the path of achieving this objective were the language barrier and the simplistic pattern matching approach of today's search engines.

Agro Explorer is a language independent search engine with multilingual information access facility. Instead of searching on plain text it does the search on the *meaning representation*, an Interlingua form called *UNL expressions* [1]. The system also employs Interlingua based approach of machine translation. The queries and the documents are translated into UNL. The translation is carried out in two steps. The first step *Analysis* deals with the conversion of the source language to an intermediate form (UNL), followed by the second step *Generation* which is conversion of the target language from the intermediate form. The Interlingua can be common for a number of languages, which is generally the case. The Analysis in the system is handled by EnConverter subsystem and the Generation is handled by the DeConverter subsystem. This method turns out to be economic when multiple languages are involved.

1.2 Motivation

As already discussed, the Internet is the largest repository of information and Search engines are the tools for accessing this plethora of information. Most of the information present on the Internet is available in English, the chief communication language worldwide. At present

count, the information content on the net is estimated to be a staggering 6 trillion bytes of data. 80 percent of this information is present in English. In a stark contrast to these statistics are figures, which show that only 40 percent of all Internet users have English as their medium of communication. This has forced many non-English speakers to spend time and resources over learning it. Despite this, the handicap still exists. So, these non-English speakers cannot harness the power of Internet and gain from it.

Most of the information being in English causes it to be effectively unavailable to the rural masses unqualified in English. The benefits of IT have not been derived by a large section of Indian population, mainly in rural areas. The reasons are given as lack of infrastructure, inadequate dissemination of information and so on. However, the problem of *language barrier* should be cited as one of the primary reasons.

Most of the advanced information for the agricultural domain should be in local languages. This should be available on the web for the farmers to read, assimilate and use. There is also the need for cross-language information transfer where climatic and agricultural conditions are similar (like Bengal, Bihar, Assam, and Orissa), thereby avoiding duplication of research and information hunting effort. The need for multilingual information processing is enormous for a country like India.

A sizable amount of information present on the Internet is in languages other than English, such as Spanish, French, Russian etc. This also creates a language barrier as the people who do not know these languages cannot access the information available in them. World Wide Web (WWW) is largest repository of knowledge known and a language gap here is obviously a big drawback.

With growing amount of information available on the WWW, it is becoming more and more difficult for the people to find the information they are looking for on the net, even with the help of search engines. It is because most of the current search engines do a pattern based search on the documents. They treat a web document and the query as nothing more than a bag of words. They do not try to find out the meaning of the query and pin-point the exact information the user is looking for. This leads to retrieval of lots of irrelevant pages as well (Problem of low relevance). And in some cases, many relevant pages are missed out (Problem of low recall). Thus, a user has to search through hundreds of pages to find out relevant information.

With the steadily growing power and reliability of Natural Language Processing, the UNL [1] can be a generous contributor in the realization of highly dependable search engines. In this report, a meaning based search engine is presented which can be used as a multi-lingual platform for all sorts of search queries. Because the search engine is multi-lingual, its reach is automatically expanded. The meaning based part of the search engine ensures that the user gets all and only the relevant pages in response to his query.

1.3 Related work

In this section a brief review of the work related to this project is presented. This allows us to put our model in perspective of the field.

1.3.1 Existing Search Engines

The most common search engine on the web, Google, is widely believed to be the best example of a traditional search engine, which is restricted to a single language (English in this case). It's a text based search engine that considers a query and documents as nothing more than a bag of meaningless patterns. Also, it returns a lot of useless and sometimes garbage information, which not only take up a lot of computing and bandwidth resources, but also is very cumbersome. It uses the famous PageRank algorithm [3], which ranks the documents according to the hyperlink structure, coupled with the local query specific score to give the final rank to a page [2].

1.3.2 Existing Meaning-Based Search Engines

Only a few meaning based search engines have been developed so far. Though the attempts made are highly laudable but their results are nowhere close to the mark and therefore they failed to be commercial successes. A few of such earlier attempts are mentioned below.

Search engines like oingo.com, excite.com and simpli.com also provide meaning based searching. Launched in October 1999, Oingo has already introduced three fully functional products: DirectSearch, DomainSense and AdSense. DirectSearch, a meaning-based search technology, uses the company's ontology to provide more precise and effective search results. DomainSense, Oingo's meaning-based domain name suggestion technology, currently increases domain name sales for leading registrars around the world. AdSense serves the most highly targeted advertisements on the Internet; effectively targeting advertisements based on search meanings rather than keywords. But none of these search engines is a true meaning based search engine as none of them considers the meaning emerging from the interconnection of words. Their results for many queries are sometimes even worst than Google, which, as we know, is not a meaning based search engine.

Dmitry Sergeevich Ermolaev, a Russian programmer, invented an *Intelligent Semantic, Clever, meaning-based Searching System for Text Information* which took into consideration the meaning of specific words or queries being searched. Unfortunately, this could not gain much industrial popularity.

Another step in the direction of meaning based searching was taken by a project of Chinese Academy of Sciences. This project, accomplished in 1998, worked on simplified Chinese and English. This project provided the flexibility of adding Traditional Chinese (Big5) and Traditional Chinese (EUC) in the future. Its established system consisted of two subsystems: Organization based subsystem and Web page based subsystem. Organization based subsystem was developed specially for users to find whether a certain organization in China had its own website and some detailed information about that organization. The web page based subsystem was developed for users for searching information in the web documents.

1.3.3 Existing Multilingual Search and Information Retrieval resources

A Multilingual Information Retrieval Tool Hierarchy (MIRTH) [5] for the World Wide Web gave a general model of multilingual information retrieval for Web searching. It coped with both English and Chinese information retrieval. MIRTH first created an index file that contained key information about different Web pages. MIRTH indexed both document titles and document

contents. Users could key in queries of search terms directly via a Web browser. Then, MIRTH started a search program that explored the pre-computed index files in real time and yielded search results accordingly.

MULINEX: Multilingual Web Search and Navigation Tool [6], developed by a consortium consisting of five European companies, supports English, French and German. It supports selective information access, navigation and browsing in a multilingual environment. The project emphasizes a user-friendly interface, which supports the user by presenting search results along with information about language, thematic category, automatically generated summaries, and allows the user to sort results by multiple criteria.

A comprehensive review of the work done in the field of multilingual information management can be found in a report titled *Multilingual Information Management: Current Levels and Future Abilities* [7]. A site containing links to two language multilingual search engines can be found at [8]. The European Commission has been working very hard to promote multilinguality, especially within EU in its V EU Framework Programme.

1.4 Organization of Report

We will then look at Universal Networking Language (UNL), which is a language independent way of representing text documents, in chapter 2. The Search engine model will be described in chapter 3. In chapter 4 we will study the Focused crawler in detail. HTML Parser is explained in chapter 5. Chapter 6 explains the page ranking algorithms which are used to order the search results retrieved by the search engine. Experimental results are presented in chapter 7. Chapter 8 gives the current status of implementation of the search engine. Conclusions and future work will be presented in chapter 9. The document used for experiments is attached in appendix A.

Chapter 2

Universal Networking Language

2.1 Introduction

The Universal Networking Language (UNL) project is a Project of Network-oriented Multilingual Communication initiated by the University of United Nations based in Tokyo. The UNL system employs an Interlingua approach to machine translation. The project has proposed a standard for encoding the meaning of Natural language expressions in the form of Semantic hyper-graphs.

Universal Networking Language or UNL has been developed to serve the purpose of an intermediate language in the interlingua approach adopted to overcome the language barrier. Universal Networking Language (UNL) [1] is an electronic language for computers to express and exchange every kind of information. Since the advent of computers, researchers around the world have worked towards developing a system that would overcome language barriers. While many different systems have been developed by various organizations, each has its special representation of a given language. This results in incompatibilities between systems. It is therefore impossible to break language barriers all over the world, even if all the results are combined in one system. Against this backdrop, the concept of UNL as a common language for all computer systems emerged.

UNL represents information sentence by sentence. This information is represented in the form of semantic hyper-graphs. The hyper-graphs have concepts as nodes with semantic attributes and arcs bearing semantic relations. The hypergraph can also be alternatively described as a set of directed binary relations, each between a two of the concepts present in the sentence.

The UNL expresses information by classifying objectivity and subjectivity. Objectivity is expressed using UW's and relations. Subjectivity is expressed using attributes attached to the UW's.

UNL can be said to comprise of the following elements:

1. UNL expressions
2. Binary Relations
3. Universal Words (UW)
4. Attributes

The UW's are the vocabulary of the UNL and the Relations and Attributes constitute the syntax of the UNL. We will now take these elements in succession and study their nature.

2.2 UNL expression

A UNL expression is a hypergraph, where a node may be simple or may recursively contain a hypergraph. Hypernodes may also be present in UNL expressions. However, the graphs and sub-graphs must contain a special node called the entry node. The UNL represents the meaning of a sentence through a hyper-graph having concepts as nodes and relations as arcs. Any component, such as a word or a sentence of a natural language can be represented with UNL expressions. A UNL expression therefore consists of a UW (in case of single word) or a set of binary relations (in case of a complete sentence).

2.2.1 Binary Relation

Binary relations are the building blocks for UNL documents. The semantic relations between the UWs form the arcs in the UNL hypergraph. The relations between the UWs have different labels according to the different roles they play. There are a total of 41 relation labels defined in the UNL specifications. For example,

- agt: defines the thing in focus which initiates an action.
- obj: defines the thing in focus which is directly affected by an event or state.
- plc: defines the place where an event occurs or a state if true.
- mod: a thing which restricts the focused thing.

Binary relations are constructed as follows:

<Binary Relation> ::= <Relation Label> | **":" <Compound UW-ID >** | **"("{ <UW1>** | **":" <Compound UW-ID1>} ","{<UW2>** | **":"<Compound UW-ID2>} ")"**

The elements used are defined in following manner

Relation Label	A relation
UW1 and UW2	Universal Words
UW-ID	two characters of '0' - '9' and 'A' - 'Z'
Compound UW-ID	two-digit decimal number (00 - 99)

When, a Compound UW-ID appears in the position of a UW, so-called the *scope-node*, it is used to refer to a Compound UW previously defined. Binary relations indicated by the Compound UW-ID define the contents of the scope. A scope-node always begin with ":" followed by the two digits of a Compound UW-ID. UW-IDs can be omitted from the UNL expressions. This is permissible when a UW is unique in a UNL expression. The UW-ID is used to indicate referential information, in cases where there are two or more occurrences of a UW-concept and they are not co-referent.

2.2.2 Universal Words

Universal Words (UWs) are character strings representing simple or compound concepts. They are the nodes of a UNL hypergraph. Concepts are universal and have a representation in all Natural languages. Universal Words are annotated with attributes to that provide information about how the concept is being used in a particular sentence. There are two classes of UW's

- Simple, unit concepts called *Universal Words*.
- Compound structures of binary relations grouped together and called *Compound UWs*. These are indicated with Compound UW-IDs

Syntax of UW

A Universal Word is made up of a character string, an English Language word, followed by a list of constraints and a list of lexical attributes.

The syntax for defining a UW is as follows:

<UW> ::= <Head Word> | **<Constraint List>** || **":"** **<UW-ID>** || **"."** **<Attribute List>**

where,

- **Head Word :** An English word interpreted as a label for a set of all the concepts that correspond to that word in English. The set comprises of all the concepts that may correspond to that in English. A Basic UW, an UW with no constraints, maps to this set. A Restricted UW, defined by its constraint list, denotes a subset of this set that is defined by its constraint List. Extra UW's denote new sets of concepts, which don't have a parallel in English. Thus, the Head Word serves to organize the concepts.
- **Constraints :** The Constraint List restricts the interpretation of a UW to a subset or to a specific concept included within the Basic UW, thus the term *Restricted UWs*.

  The Basic UW *drink*, with no Constraint List, includes the concepts of *putting liquids in the mouth*, *liquids that are put in the mouth*, *liquids with alcohol*, *absorb* and others.

  The Restricted UW *drink(icl>do,obj>liquid)* denotes the subset of these concepts that includes *putting liquids in the mouth*, which in turn corresponds to verbs such as *drink*, *gulp*, *chug* and *slurp* in English.

  The restrictions of Restricted UW's, their Constraint Lists, are Constraints. The Constraints that use the Relation Labels defined above can be interpreted as an abbreviated notation for full binary relations: drink(icl>do,obj>liquid) is the same as obj(drink(icl>do),liquid).
- **Attributes :** Provides information on how the concept is being used. Ex. @past, @plural.
- **UW-ID :** Used to indicate some referential information.

Types of UW's

Universal Words are of three types:

1. Basic UW's, which are bare head words. It denotes all the concepts that may correspond to that in English. They are used to structure the knowledge base and as a fallback method for establishing correspondences between different language words when more specific correspondences cannot be found. For example: go, take, house, state

2. Restricted UW's, which are head words with a constraint list. It denotes a subset of the concept that may correspond to that in English as defined by its Constraint List. Each Restricted UW represents a more specific concept and limits our attention to a particular sense of the word. For example: state(icl>situation), state(icl>nation), state(icl>government)

3. Extra UW's, denote concepts alien to the English Language. Foreign-language words are used as Head Words using English characters. The restrictions give some idea as to the nature of the concept. For example: ikebana(icl>do,obj>flowers) (art of flower arrangement), samba(icl>dance) (a kind of a dance), soufflé(icl>food,pof>egg) (a dish prepared from eggs)

Compound UWs

These are a set of binary relations that are grouped together to express a concept. A sentence itself is considered as a compound UW. Compound UWs are indicated by the means of Compound UW-ID's. For example:

Women who wear big hats in Movie theaters should be asked to leave.

Without Compound UWs, it would be impossible to build up complex ideas like "*women who wear big hats in move theaters*" and then relate them to other concepts. A Compound UW can be defined by placing a Compound UW-ID immediately after the relation label in all the binary relations that are to be grouped together. For instance, in the case of the above sentence

```
agt:01(wear(icl>do), woman(icl>person).@pl)
obj:01(wear(icl>do), hat(icl>thing))
aoj:01(big(aoj>thing), hat(icl>thing))
plc:01(wear(icl>do, theater(icl>place))
mod:01(theater(icl>place), movie(icl>entertainment))
agt:01(leave(icl>do).@entry, woman(icl>person).@pl)
```

After this group has been defined, the compound UW ID "01" can be used to cite the Compound UW. A Compound UW is sited using its Compound UW-ID as the UW. The ID is interpreted as the whole set of binary relations defined in it. It is a hypernode and should have an entry node of its own.

2.2.3 Attributes

Attributes of UWs are used to describe the subjectivity of sentences. They show what is said from the speaker's point of view: how the speaker views what is said. On the other hand, Relations and UW's describe the Objectivity of a sentence. The Attributes enrich the information content of the UNL by providing information like the speaker's view of emphasis, focus and topic; the time with respect to the speakers. The UNL group has a provided a very rich set of attribute which makes it possible to capture a great deal of real world situations into the UNL form. For example: In a sentence like - It was snowing yesterday - @past is used as attribute, because the speaker is talking about that has happened in the past

2.2.4 An Example

We give here an example of a UNL expression to provide an insight of the formulation.

Only a few farmers could use information technology in the early 1990's.

Core Sentence: Farmers use technology. Specific modifiers are used to enhance this sentence so that it assumes its given form.

Equivalent UNL Expression:

```
agt(use(icl>do).@ability-past, farmer(icl>person).@pl)
obj(use(icl>do), technology(icl>thing))
mod:01(farmer(icl>person), few(icl>number).@indef)
mod(:01, only)
mod(technology(icl>thing), information)
mod:02(1990(icl>time), early)
tim(use(icl>do), :02)
```

We can represent the above sentence in the graph form as shown in figure 2.1. Concepts are nodes and Relations are the arcs. The Root of the Graph is the Entry Node.

2.3 Enconverters and Deconverters

Enconverters and Deconverters provides the translation facility to the system. The process of converting a source natural language into UNL expressions is called enconversion, whereas, converting the UNL expressions into a target language is called deconversion.

The softwares used for this processes are are called Enconverters (ENCO) and Deconverters (DECO) respectively. These softwares are designed as a language independent parsers. They can work for any language by simply adapting a different set of the grammatical rules and Word Dictionary of a language.

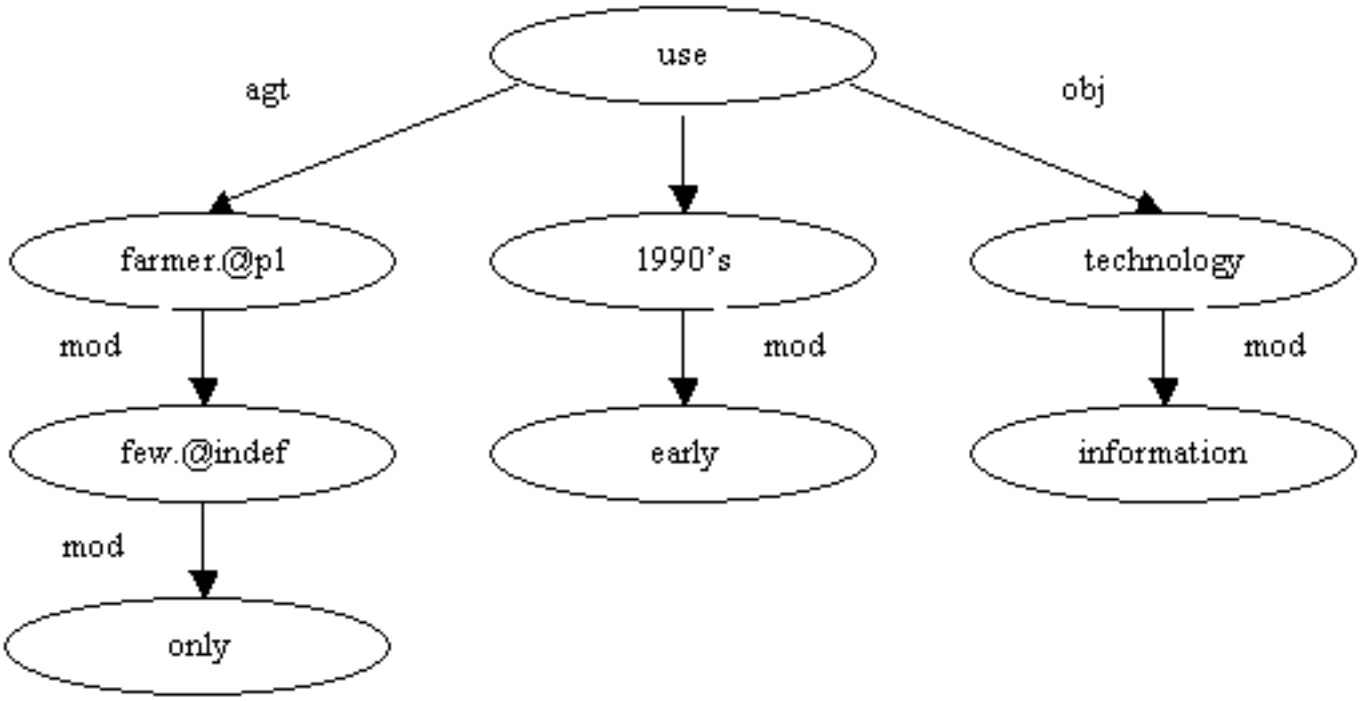


Figure 2.1: The UNL graph for the example

For the process of enconversion and deconversion, two files are used. The first file is a dictionary that lists correspondence between the Universal Words of UNL and the words of a native language. The second file lists grammatical rules. Each of these files is specific to a particular language and are developed according to the UNL Specifications and the UNL Knowledge Base.

Presently ENCO and DECO softwares for many languages including English, Hindi, Marathi, Spanish, Russian and Italian are under various stages of development.

2.3.1 Enconverter

The EnConverter is a language independent parser, which provides a framework for morphological, syntactic, and semantic analysis synchronously. It would be impossible to solve all the ambiguities even in a morphological analysis if the syntactic or semantic analysis is performed synchronously. Also, it would be impossible to solve every ambiguity in a syntactic analysis in the absence of semantic analysis.

The EnConverter scans the input string from left to right. When an input string is scanned, all matched morphemes with the same starting characters are retrieved from the dictionary and become the candidate morphemes. The rules are applied to these candidate morphemes according to the rule priority in order to build the syntactic tree and the semantic network for the sentence. The left character string is scanned from the beginning according to the applied rule; the process continues in the same manner. The output of the whole process is a semantic network expressed in the UNL format.

2.3.2 Deconverter

The DeConverter is a language independent generator providing a framework for syntactic and morphological generation as well as co-occurrence-based word selection for natural collocation. It can deconvert UNL expressions into a variety of native languages, using a number of linguistic data such as Word Dictionary, Grammatical Rules and Co-occurrence Dictionary of each language.

The DeConverter transforms the sentence represented by a UNL expression - i.e., a set of binary relations - into a directed hyper-graph structure called Node-net. The root node of a Node-net is called Entry Node and represents the main predicate of the sentence. The DeConverter applies generation rules to every node in the Node-net and generates the word list in the target language. In this process, the syntactic structure is determined by applying Syntactic Rules. Morphemes are similarly generated by applying Morphological Rules.

The DeConverter's generation ability is similar to that of a Turing machine. It is capable of generating all types of sentences, applicable to all languages. Co-occurrence Relations between words contribute to a better word selection. This means it is possible to generate more natural sentences by using Co-occurrence Relations.

Chapter 3

The Search Engine Model

3.1 Introduction

This chapter describes the overall structure, data and control flow of the meaning based multilingual search engine. Various modules that make up the search engine and brief description of their implementation is presented here.

3.2 Modules

All the modules of the search engine are shown in the figure 3.1.

3.2.1 Crawler Module

The World Wide Web, having over 350 million pages, continues to grow at a million pages per day. About 600 GB of text changes every month. This has posed serious problems of scale for the crawler and the search engine.

Undoubtedly, we need a general purpose crawler for a general search engine, but presently we are considering only the domain of agriculture as the search engine's domain. This is because the English to UNL Enconverter is not fully developed yet and will require some more research and time before it can convert all of the English sentences into UNL successfully. So, till that is done, we decided to focus on a narrow domain for our search engine.

Because of the above mentioned problem, we will need the corpus only for the agricultural domain. One way of doing this will be to use a general purpose crawler which will crawl the whole web and then select out the documents relevant to the agricultural domain from it. But a giant, general purpose web crawler is nor necessary not sufficient for this purpose. It will lead to a lot of time and space wastage. It is unreasonable to crawl and index 350 million pages just to distill pages related to agriculture from it. Also, the overhead will severely dampen the crawler's ability to refresh the pages related to agriculture domain.

A *focused crawler* will be much more relevant for our problem. Instead of crawling and indexing the whole web, a focused crawler only crawls the web pages that are likely to be most

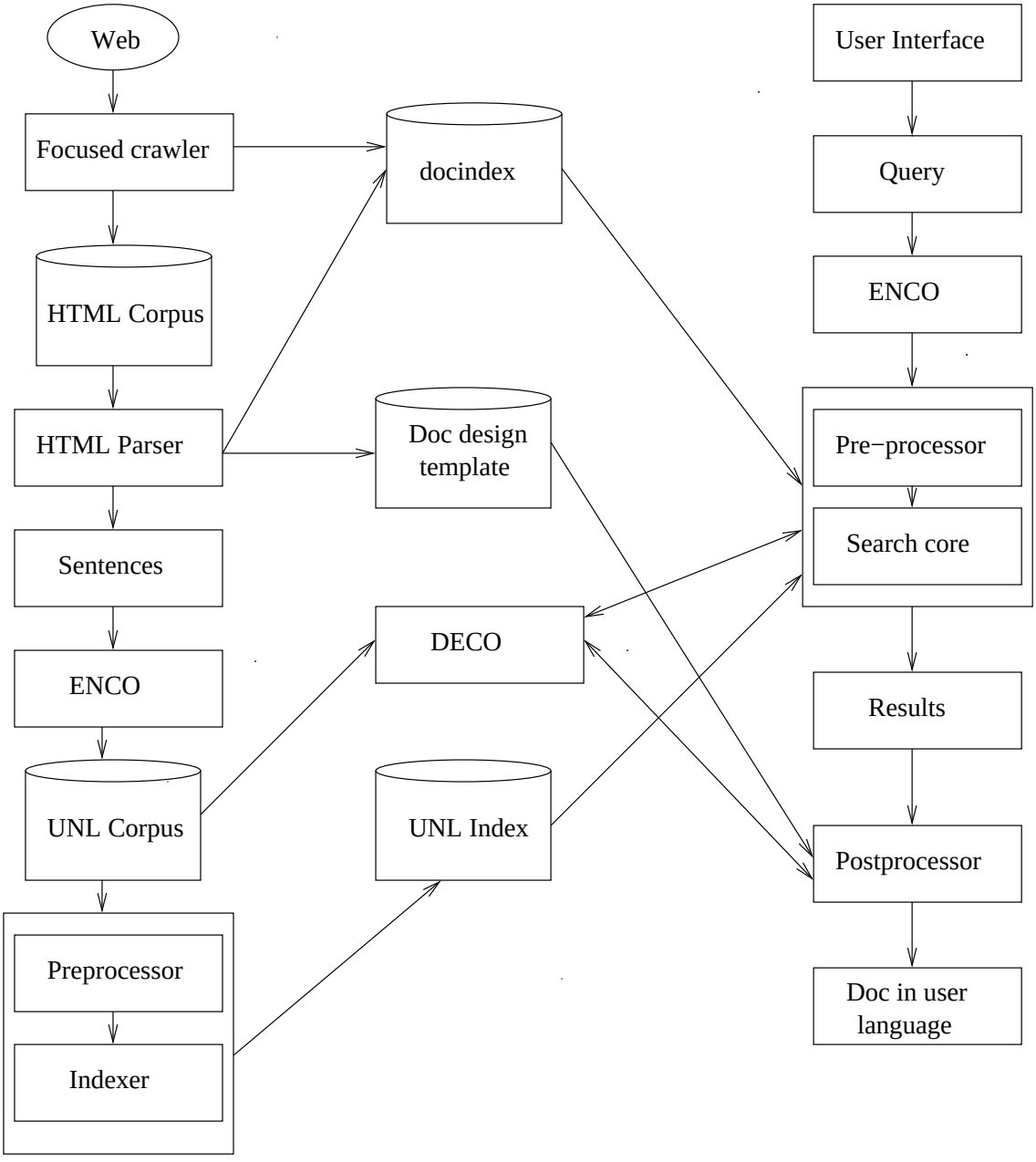


Figure 3.1: Block Diagram of the Search Engine

relevant for the domain at hand and avoids irrelevant pages of the web. This leads to significant savings in hardware and network resources and keep the crawl more up-to-date.

As shown in the figure 3.1. The crawler module will crawl the web and make a HTML corpus, which will be then fed into the HTML Parser. At this stage each document is assigned a unique id. The mapping from document-id to the document details is kept in a table called *docindex*. The crawler updates docindex with details of the documents crawled. docindex contains some important details about the documents like its docid, name/title, its web link, date of crawl, document language and number of lines within document. The last two fields are updated by the HTML Parser when it processes the document.

3.2.2 Enconverter module

Enconverter module, as the name suggests, converts the natural language documents into their equivalent UNL representation. As described earlier, *ENCO* is the software that is used for this purpose. This software is designed as a language independent parser. Because the search engine is multi-lingual, it will naturally consist of more than one set of dictionaries and rule bases, one for each language to UNL conversion.

This search engine is being developed on the Linux platform, but the ENCO software is available only for the windows platform. To overcome this difficultly, the Enconverter module is implemented on windows and a special Enconverter server is setup on a Windows 2000 machine. This server has ENCO software running on its back end.

Whenever the system (running on linux) needs to perform enconversion, it sends the sentence(s) to the enconverter server. Based on the language specified by the user in a special variable, the enconverter loads the corresponding dictionary in the ENCO software. The ENCO then converts the sentence to its UNL representation using the dictionary and the rule base. The enconverter server then replies back (to the requesting process) with the UNL equivalent of the query.

This system strips off the tags like [S] and the comments which are produced by the ENCO software and returns *only* the UNL expressions. In case, when there is only a single word in a sentence, the system replies with a "W" followed by a newline and then the UW corresponding to the word. This is done, so that the search core can distinguish between single word and set of binary relations and use different algorithms for matching accordingly.

3.2.3 Deconverter module

This module is similar to the Enconverter module except that it will convert the UNL representation back to any language (for which it has DECO software). This module is also implemented on the windows platform. It will be required to convert the documents which the search engine returns in the user's language.

Both the Enconverter and Deconverter modules are implemented as a PHP script running on apache web server on a Windows machine. A single script (query2.php) is used for both these servers. Based on what is requested (enconversion or deconversion) through a special parameter (*what*), the PHP script runs the appropriate software with the correct language rule base and dictionary (found out using another parameter -*lang*) and returns the result.

3.2.4 HTML Parser

This module parses the HTML documents crawled by the crawler in order to separate the formatting (HTML tags) from the sentences. The design of the document is stored in the document design template, which consists of only HTML tags with the placeholders for sentences. For example, for a HTML document:

```
<HTML>
<HEAD>
<TITLE>A sample document</TITLE>
</HEAD>
<BODY>
<H1>This is the heading.</H1>
<P>This is the document line 1. This is document line 2.</P>
</BODY>
</HTML>
```

the document design template is (using "-" as the sentence placeholder, in actual practice a *rare* character like ASCII(001) is used in place of "-")

```
<HTML>
<HEAD>
<TITLE>-</TITLE>
</HEAD>
<BODY>
<H1>-</H1>
<P>--</P>
</BODY>
</HTML>
```

and the sentences extracted are:

```
A sample document
This is the heading.
This is the document line 1.
This is document line 2.
```

Thus by just merging the sentences and the document design we can get back the original HTML document. Even if we translate the sentences into some other language, we can still

present the user with the same look and feel of the original web page by merging together the translated sentences and the document design template.

The HTML parser also determines the language of the document by using the language META tag or other heuristics and the number of sentences present in a document. These are then updated in the docindex. The sentences extracted by the HTML parser are then sent to the appropriate ENCO (based on the language of the document) for enconversion into UNL. Thus a UNL corpus is built, which consists of UNL representation of the documents.

3.2.5 Indexer Module

This module takes as input the UNL corpus and generates a UNL index, which is used by the search module to quickly find the relevant documents and calculate their relevance. This module itself consists of two sub-modules.

Preprocessor

This module preprocesses the UNL documents and converts them to a intermediate form which is then indexed by the indexer. Following steps are performed by this module:

1. All unnecessary spaces and tabs are removed.
2. All attributes of the Universal Words (UWs) are removed. Removing the attributes do not effect the search results and in some case may even improve the recall of the search engine.
3. Some of the UWs are not assigned any UW-ID by the Enconverter software. UW-IDs can be omitted from the UNL expressions when a UW is unique in a UNL expression. For consistency in indexing procedure, a unique dummy UW-ID is assigned to the UWs not having any UW-ID. This also simplifies the partial matching algorithm.
4. All the compound UW ids are replaced by the actual UNL expression of all the relations in the subgraph, sorted alphabetically and separated by an unique character ("~" in our case). All the normal UW ids present inside such a subgraph is removed before substituting it for the compound UW id.
5. The UW id of such a subgraph is kept as the compound UW id representing that subgraph in the original UNL expression.
6. All the compound UW ids are removed.

For example, if input UNL expression is:

```
[S]
aoj(contribute(icl>support>be(aoj>thing,obj>thing)):0S.@entry.@present.@pred, :01)
gol(contribute(icl>support>be(aoj>thing,obj>thing)):0S.@entry.@present.@pred,
    productivity(icl>quality>property):10)
man(contribute(icl>support>be(aoj>thing,obj>thing)):0S.@entry.@present.@pred,
```

```
    immeasurably(icl>how):13)
mod(productivity(icl>quality>property):10,    farm(icl>place):1J)
and:01(genetics(icl>natural science>science):0J.@entry,
    breeding(icl>activity>abstract thing)):06)
mod:01(breeding(icl>activity>abstract thing)):06,
    plant(icl>living thing>concrete thing):00)
[/S]
```

the resulting output after these steps will be:

```
[S]
aoj(contribute(icl>support>be(aoj>thing,obj>thing)):0S,
  and(genetics(icl>natural science>science),breeding(icl>activity>abstract thing)))
  ^mod(breeding(icl>activity>abstract thing)),
    plant(icl>living thing>concrete thing)):01)
gol(contribute(icl>support>be(aoj>thing,obj>thing)):0S,
  productivity(icl>quality>property):10)
man(contribute(icl>support>be(aoj>thing,obj>thing)):0S,
  immeasurably(icl>how):13)
mod(productivity(icl>quality>property):10,farm(icl>place):1J)
and(genetics(icl>natural science>science):0J,
  breeding(icl>activity>abstract thing)):06)
mod(breeding(icl>activity>abstract thing)):06,
  plant(icl>living thing>concrete thing):00)
[/S]
```

Indexer

This module separates the resulting UNL expressions got from the preprocessor into relation, first UW, second UW, first UWs id, second UWs id. It than updates the UNL index with these values, which is again a table in the mysql database.

If a sentence consists of only single Universal Word, it is also indexed into the index with the value of first UW as the UW given and all other fields as null.

3.2.6 Search Module

This is the nucleus of the search engine. It takes as input the UNL index generated by the interface module and a query (in UNL form) and returns a list of documents matched according to decreasing order of their relevance along with lines matched and the relevance (rank) of the

page. This module uses the same preprocessor used by the indexer module to preprocess the UNL query. This module does a *partial* search on the UNL documents. The relevance returned is calculated by combining the global page rank and the query specific page rank of the document as explained in chapter 6.

For sentences consisting of a single UW, the normal partial matching algorithm is not used. This case is handled separately in the search Module. As we can clearly see, in case of single UW, we can simply find the document sentences having those UWs, as the index (unlindex) contains all the UWs present in a sentence (UW1 and UW2). The $r_q(s)$ (refer chapter 6) for the document sentences found is taken as 1. Other (non-matching) sentences are given $r_q(s) = 0$.

The core of this module is written in C++. It just takes as input the query in UNL. A PHP script is used as a wrapper for this module. This script, after getting the query in users language, first converts it into UNL. Then it calls the C++ search program, which returns the results in a raw format. Only the relevant line numbers are returned, as the core is not aware of the language of interface from which it is being called. This PHP script parses the results and with the help of deconverter module translates the relevant lines into the users language and displays all the results in a HTML page generated on the fly.

3.2.7 Post Processor

In the output produced by the search module, user has links to view the document in different languages. This module is called when user wants to see a document in a particular language. The inputs to this module are the doc id and the language in which to show the document. This module, with the help of Deconverter module translates the UNL document into the language requested by the user. But this translation will be just a number of sentences. Now it takes the document design template, created by the HTML parser and merges the sentences with it to produce a complete HTML page.

This module first checks if the deconversion that is requested is already present in cache. If the document is present in cache (public_html/data directory), it uses the cached sentences. In case the deconversion is not available it calls the Deconversion server running on windows machine to deconvert the document into the user requested language and caches its output. Now it uses this sentence cache to generate the HTML page.

This module is implemented with the help of a PHP script (postp.php) and a C++ program (postp.C).

3.2.8 Interface Module

This module is responsible for interaction with the user and giving him back the results in a user friendly way. This module is implemented entirely in HTML and PHP. Interfaces for Hindi and English query are currently implemented.

A number of interfaces will have to be developed, one each for each different language supported by the search engine. An interface for a particular language, say X, displays the query page in language X. When the user enters the query in X language, this interface sends this query to the search module that returns all the relevant information needed by the interface to display the results. The interface will then take this output and prints the results according to decreasing order of relevance, a link to the original document, the original language of the document, the lines matched in the document (converted into language X) which provide as a

short summary of the document, the relevance ratio (rank) and the link which points to the document converted to language X.

IITK to DYTT-Yogesh Converter

Presently, in the Hindi interface the users input in taken in IITK format. But, to display the Hindi script on the HTML web page, we need to convert it into a viewable font. We are using the DVTT-Yogesh font for this purpose. The software for converting the plain text IITK format into DVTT-Yogesh font is available. Unfortunately, this software is also available for windows platform.

Therefore, it is necessary to send the text in IITK format to windows machine for conversion into DVTT-Yogesh font. We are using the same PHP script, which is being used for enconverter and deconverter modules, for this purpose also. When the variable *what* is set to "convert", this script runs the IITK to DVTT-Yogesh converter and sends the resulting text back to requesting process.

3.3 Organization of Data

The system organizes its data in various tables and directories as explained below:

1. **HTML Corpus:** As explained earlier, this corpus is built by the crawler as it crawls the web. This corpus is kept in a directory called **html**. This directory contains the HTML files in plain text format. The file names are the docid's assigned to the document (together with .html extension). The mapping from docid to documents is kept in docindex table.

2. **Document Index :** This index is kept in a MYSQL table named unldocindex. The fields of this table are:

   - *docid* : The unique id assigned to document. (primary key)
   - *orilink* : The link from which the document was crawled.
   - *language* : Language of the original document.
   - *numlines* : Number of sentences in the document.

   In future, more fields (e.g. date/time crawled etc.) can be added to this table as the need arise.

3. **Document design template :** This is the *design* of the document with the placeholders for sentences. These designs are kept in the **public_html/design** directory. The filenames are the docid's of the corresponding documents and the extension is *des*.

4. **UNL Corpus :** The UNL expressions of the documents is kept in **public_html/data** directory. The filenames is docid's of the corresponding documents and the extension is *unl*.

5. **Sentences :** The sentences extracted by the HTML Parser are kept in **public_html/data** directory with filename as the docid of corresponding document and extension as *.Language*, where language is the source natural language of the sentences. For example the extension will be *.English* for English sentences.

   This directory is also used as cache directory by the post-processor with the same naming convention as described above. So, if the original document is in English and user wants to see the document in English, the English are not generated afresh, but the same original sentences extracted by the HTML Parser are used.

   Currently, the files in cache are not deleted. In future, we may want to delete old cached sentences file when the size of this cache directory exceeds a limit.

6. **UNL Index :** The UNL documents are indexed in this index. It is kept in a mysql table named *unlindex*. The fields of this table are:

   - *REL* : Relation name.
   - *UW1* : First parameter of the relation.
   - *UW2* : Second parameter of the relation.
   - *UW1ID* : ID associated with UW1.
   - *UW2ID* : ID associated with UW2.
   - *docid* : Unique ID assigned to the document.
   - *sent* : Sentence number.

3.4 Control Flow

The modules described above and their inter-connections are shown in figure 3.1. As UNL is the underlying representation used, it makes the search engine both meaning based and multilingual. All documents are stored in their UNL form, which is nothing but the *meaning* of the documents. This eliminates the language barrier. When the user enters a query, the query is also converted to its meaning (UNL). Then a meaning based search is performed on the UNL documents and results are ordered according to their ranks. The user can give a query in various languages and the results will be displayed in the language in which the user gave the query. The documents returned will also be translated to the user's language.

The user interface is the place where the user submits his query. The query is in turn converted into a UNL expression by the EnConverter and then given to the search core. The Search Core then searches the query using the unlindex generated by the indexer module. The results returned by the search core, which also contains the relevant line numbers, are parsed by a PHP script (a different PHP script is used for each language). The PHP script feeds the relevant line numbers to the DeConverter or reads them from the cache if they are available, and then displays the deconverted sentences in the user's language so that the user gets an idea about the document.

When the use clicks on a link on the results page, the post processor is called to generate the document's HTML page in users language. The postprocessor calls the deconverter to convert the document sentences to user language and then merges it with doc design template to finally generate the document in users language.

Chapter 4

Focused Crawler

4.1 Introduction

The World Wide Web, having over 350 million pages, continues to grow at a million pages per day. About 600 GB of text changes every month. This has posed serious problems of scale for the crawler and the search engine. The present day crawlers, such as Alta Vista's Scooter and Google's Google Bot, uses a massive amount of resources to crawl and index the huge web. In spite of these efforts, these bots are able to cover only 30 to 40 percent of the web and the refreshes takes one or two months on an average. This problem is partly due to their efforts to try to cater to every possible query that might be made on the web.

Compared to web, human brain has grown only linearly from 400 to 1400 cubic centimeters in the last 3.5 million years. Clearly, this *information explosion* is causing information overload for a web user. Most of the web users are usually concerned with information belonging to a specific domain. Whenever they give a query, they are mostly interested in results from a specific domain. A giant, all-purpose crawl is neither necessary nor sufficient for this purpose. It is also unreasonable to have to first crawl and index 350 million pages in order to distill pages of a small domain! Much of this index would never be used, but, burdened by the responsibility of maintaining this huge index, the crawler would not be able to preferentially and frequently refresh and further explore relevant regions of the web.

A *focused crawler* will be much more relevant in this case. Instead of crawling and indexing the whole web, a focused crawler only crawls the web pages that are likely to be most relevant for the domain at hand and avoids irrelevant pages of the web. This leads to significant savings in hardware and network resources and keep the crawl more up-to-date.

4.2 Algorithm

For making an automatic focused crawler we need to somehow present to it the domain of crawling. There can be many ways to do this, but the most widely used technique is to give it some positive example of pages that belong to the domain. We can give the focused crawler a set of pages that represents a domain.

We initially present the crawler with Yahoo categories, which describe a domain. The crawler can then crawl the categories given as well as any subcategories present in them and extract all the web links to the pages inside these categories. These pages are then taken as

the model for the domain of crawling. These set of pages are then presented to a document classifier [9] which forms of a model for the domain. Based on this model the classifier can give relevance score to new documents, which is a measure of their relevance to the domain presented.

After the classifier is trained, we can start the main focused crawler. Our aim is to crawl as many relevant pages possible while avoiding the non-relevant pages. To achieve this, we note that pages that are relevant to the domain of crawling have greater probability of having out-links to pages that are also relevant. At any given time, all the links, which are not yet crawled, are kept in a table along with their priority of crawling. This priority is measure of our belief that the link will point to a page belonging to domain of crawling. The crawling priority of a link is taken as the relevance of its parent page (i.e the page from which the link was extracted). The crawler picks up the link of highest priority and crawls it. Then it presents this page to the classifier that gives it a relevance score. If this score is below a threshold the page is not included in the corpus. Otherwise, the link present in this page are extracted and added to the links table. There priorities are taken as the relevance of the current page. In this fashion, the crawl continues, until all links are exhausted or the crawler is stopped (in case sufficient corpus is already built).

4.3 Implementation

A primitive focused crawler, inspired by the focused crawler described in [4], was implemented in this project. This crawler was written in Unix shell script. The crawler takes as input a file containing yahoo directory's categories, which we believe represents a domain of crawling. It then extracts all the out-links from the page using the *lynx* package. Out of these out-links, the out-links to other yahoo categories are identified, which can be easily done as the links of yahoo categories are of the form http://dir.yahoo.com/XXX. From these, only those links that are sub categories of the present categories are marked for further crawling. Rest of the categories are saved in special file called *suggestions*. Rest of the links, which are links to the web pages representing the domain are saved in another file called *links*.

After links from all yahoo categories, and their sub categories are extracted, the web page links present in the *links* file are crawled using *wget* utility. Some of the links (images, other html document) in the crawled pages may be relative. We need to convert these relative links into absolute links because we will be showing the user a cached version of page in which the relative links will not work. To convert these links, we can either add a base URL tag on the top of the crawled page or prefix the relative links with the base url of the page. After the crawl is over we will get a set of html pages, which can be then presented to the classifier for building a model, which will be used by the second stage crawler to start a focused crawl as described above.

This crawler was started by giving http://dir.yahoo.com/Science/Agriculture as the initial category. At the end of the crawl it gave 319 yahoo categories as suggestions and crawled 1798 web pages.

4.4 Current Status and Future Work

Only a part of focused crawled was completed in the current project. The second part of the crawler, which will crawl the web based on the suggestions from the classifier, is still to be implemented.

Chapter 5

HTML Parser

5.1 Introduction

Parsing is the process of structuring a linear representation in accordance with a given Grammar . The *linear representation* may be a sentence, a computer program, a knitting pattern, a sequence of geological strata, a piece of music, actions in ritual behavior, in short any linear sequence in which the preceding elements in some way restrict the next element. For some of the examples the grammar is well known, for some it is an object of research and for some our notion of a grammar is only just beginning to take shape. Here we have to identify the structure of the HTML document so as to separate the information and the style of the document.

The objective of the HTML parser is to recognize the structure of the HTML document so as to break it into HTML tags and sentences. The HTML tags define the look and style of the document. The sentences contain the information, which form the knowledge base for the search engine.

5.2 Impetus

The Focused Crawler is going to crawl the web and collect web pages from a specific domain, in this case the agricultural domain. Later these web pages are stored in a database. Textual Information from these HTML formatted documents is to be extracted which is then passed to enconverter to make the UNL corpus.

The HTML Parser stores design of the document in the document design template, which consists of only HTML tags with the placeholders for sentences. The sentences are stored in a separate file, with each sentence separated by a newline character.

A HTML document is composed of tags besides textual information. These tags decide how the document looks. If we separate the sentences from the tags, we can have these sentences available for search engine, also if the document is required to be translated into another language, it can be and later the tags can be reattached to the translated sentences. This would preserve the style of the document. Thus the user shall see the retrieved document in his native language with exactly the same style as that of the original non-translated document.

5.3 Algorithm

The HTML code for a web page consists of various tags and attributes as well as textual information. The tags are identified by the simple structure <*ABC*>. So anything inside a tag is irrelevant as far as HTML parser is concerned. However lot of processing is needed for the textual part.

The only thing relevant between the <HEAD> and the </HEAD> tag is the title of the document which can be easily extracted by extracting the string between the <TITLE> and </TITLE> tag. Everything except the title can be dumped into doc design template as the HEAD part of HTML code does not contain any sentences.

However the textual information can be in many forms like paragraphs, sentences, points and tables. The current version of the HTML Parser assumes that textual information inside web pages appear as simple sentences terminated by standard sentence delimiters like ".", "?" and "!".

The parser dumps all the HTML tags and other style information in doc design template as long as it does not encounter some textual part. As soon as it encounters a sentence, it begins to write the sentence in the sentence file. Now the main problem is how to find the end of sentence. We note that the end of sentence can be marked either by a sentence delimiter or by a sentence ending tags like
, <P>, </P> etc. After finding out the end of sentence by the above heuristics, it writes the sentence in sentence file followed by a newline and a special character is written to the doc design template in place of this sentence to mark its position.

5.4 Problems faced and proposed solutions

- The dot coming after the abbreviations like Dr., Mr., Mrs. should not be treated as a sentence terminating character. In other words the program should distinguish between sentence terminating full stop and the dot after an abbreviation. This issue can be resolved by keeping a list of abbreviations. Whenever a dot is encountered and word just preceding it is one of the abbreviations then the dot is not considered as a sentence terminator.

- If any HTTP address is present in the document then the question mark or the dots appearing in them should not be treated as the sentence terminating characters. We can identify the HTTP addresses by the "http://" part, or by "www." part, or by finding out if there are two or more single words separated by a dot without any spaces between them. Combination of one or more of these heuristics will help us to identify a HTTP address in the document.

- There are special characters, which occur in a HTML text like "`<`", "`&#22;`" and "` `". These are Numerical Entity codes. Whenever we encounter such characters inside the textual information part of a HTML document, the symbol is read and referred to a list to find out the character it is representing. Once it is known the Numerical Entity code is replaced by the corresponding character in the text buffer and the normal processing is carried on.

- Also certain HTML text needs to be handled separately e.g. *This is a <B>big</B>*

string. In this case we cannot preserve the <B> and </B> tags with the scheme mentioned. To overcome this problem, we can simply drop these kind of tags. Note that this will result in loss of style of the HTML document. Another approach can be to expand the current simple *put a character to mark a sentence* scheme and add some more information in the doc design template to find out where exactly such kinds of tags are to be put when the HTML document is being generated from the translated sentences and the doc design template.

5.5 Future Vision

In future as more and more web documents will be analyzed, we will be finding more complexities, which can be solved by devising solutions and adding suitable code for the same in the original code. The simple fact is that the HTML documents subscribe to various standards like the W3C, Microsoft Internet Explorer Extensions, Netscape Navigator Extensions etc. Besides we have different versions of HTML like HTML2.0, HTML3.2 which again serve as independent standards. So it is highly likely to find some aberrations in the HTML Parser outputs. The only solution to the problem lies in continually testing the parser against various web pages and adding new rules so that it is able to handle more and more HTML pages.

Chapter 6

Relevance and Ranking

6.1 Introduction

A web search engine not only returns a set of pages in response to the user's query, but it also has the job of arranging the pages in decreasing order of their *relevance*. The relevance of a web page is the measure of the web page's importance with respect to a search query. The importance of web page is inherently a subjective matter, which depends on reader's interests, knowledge and attitudes. In spite of this there is much that can be said objectively about the relative importance of web pages that are retrieved in response to a search query.

This kind of ranking of web pages is important in many respects. It saves time and energy of the search engine user, because the user will find the pages that are most likely to be relevant on the top of the search results. The definition and implementation of the relevance directly affects implementation details of many other aspects of search engine such as indexing and data structures used for representing the processed corpus. Finally, it affects the Precision and Recall of the search engine, the two most important measures used for judging the results of a search engine.

The relevance of a page to a given query is function of the global page rank and query specific page rank.

6.2 Global Page Rank

The global page rank measures the relative importance of the web pages. It does not take into account the user's query to calculate the rank of the page, instead the hyperlink structure of the web is considered to calculate the page rank.

Even without the knowledge of the actual contents of a web page, we can predict a lot about the relative importance of web pages but looking at the overall link structure of the web. Generally, highly linked pages are more important than pages with few links. This simple citation counting is not sufficient for finding the importance of a page. For example, if a web page is linked by Yahoo home page it will be more important than a page which has ten links, but from obscure places.

6.2.1 PageRank

PageRank [3] captures the above notion of the importance of a web page. PageRank, along with a query specific page rank, is currently used by Google to rank its results [2]. Intuitively, a page has high PageRank if the sum of the PageRank of its backlinks is high. This covers both the case when a page has many backlinks and when a page has a few highly ranked backlinks. PageRank is defined as follows:

We assume page A has pages $T_1 \ldots T_n$ which point to it (i.e., are citations). The parameter d is a damping factor which can be set between 0 and 1. d is usually set to 0.85. Also $C(A)$ is defined as the number of links going out of page A. The PageRank of a page A is given as follows:

$$PR(A) = (1-d) + d\{\frac{PR(T_1)}{C(T_1)} + \ldots + \frac{PR(T_n)}{C(T_n)}\}$$

Note that the PageRanks form a probability distribution over web pages, so the sum of all web page's PageRanks will be one. PageRank or $PR(A)$ can be calculated using a simple iterative algorithm, and corresponds to the principal eigenvector of the normalized link matrix of the web.

6.2.2 Random Surfer Model

PageRank can be thought of as a model of user behavior. We assume there is a *random surfer* who is given a web page at random and keeps clicking on links, never hitting "back" but eventually gets bored and starts on another random page. The probability that the random surfer visits a page is its PageRank. And, the d damping factor is the probability at each page the random surfer will get bored and request another random page. This model is known as the *Random Surfer Model*.

6.3 Query Specific Page Rank

The Global page rank is independent of the search query. Obviously, the relevance of the web page will also depend on the search query given. For example, in a text based search engine, if the query X Y (X and Y are words) is given, a page containing both the terms X and Y will be more relevant than a page having only X or only Y, assuming the PageRank is same for both pages.

As described in chapter 3 both the query and the document is converted into UNL before a meaning based search is done. For each sentence in the document, we will have one UNL graph. Thus, essentially, we will have a collection of UNL graph (document) and a given UNL graph (query) which need to match with the document. If the query graph is subgraph of any sentence graph in the document, then we can say that the sentence is relevant for the query and the document should be retrieved in the results. Intuitively, we need to do a subgraph matching between the query graph and the document sentence graphs.

From experience and common sense, the attributes of Universal words (UW) do not effect to the query matching. So at the preprocessing stage, all the attributes of the UWs are stripped off.

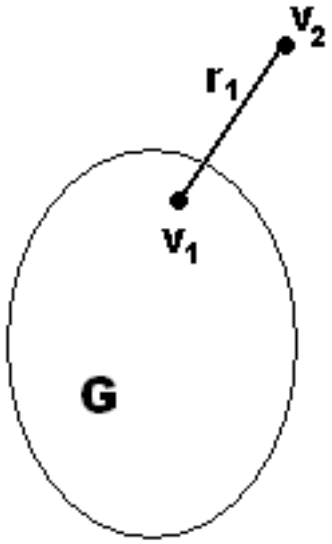


Figure 6.1: Query graph

6.3.1 Complete Matching

From the above discussion, the first and easiest algorithm for finding the relevant documents will be as follows. After the query graph is found, a subgraph checking is done on every sentence graph in the document. If the query graph is subgraph of a sentence, that sentence is considered *relevant* to the query. A document having more proportion of relevant sentences will be more relevant to a query. Mathematically, this can be expressed as:

$$R_q(d) = \frac{\sum_{s \in S_d} r_q(s)}{|S_d|}$$

where, $R_q(d)$ is relevance of document d to the query q. S_d is set of sentences in the document d. $r(s)$ is relevance of sentence s to the query q. As mentioned above $r(s) = 1$ if the query is subgraph of the sentence graph, 0 otherwise.

To find out the results for a query q, we need to look at each and every sentence of the all the documents! Clearly this is very time consuming and impractical as number of documents on the Internet is huge. Thus we need *indexing* to reduce the time consumed in finding the results for a given query.

The most obvious and relevant indexing scheme will be to index the whole corpus of UNL documents on the *edges* of the UNL graph. An edge of the UNL graph is a triplet

$$r(U_1, U_2)$$

where r is the relation-label and U_1 and U_2 are Universal Words.

For each edge present in the corpus, we will store the (document number, sentence number) pair(s) in the index. Now it will be easy to find what all (d,s) pair has all the edges of the query by taking intersection of sets of (d,s) for each edge of the query.

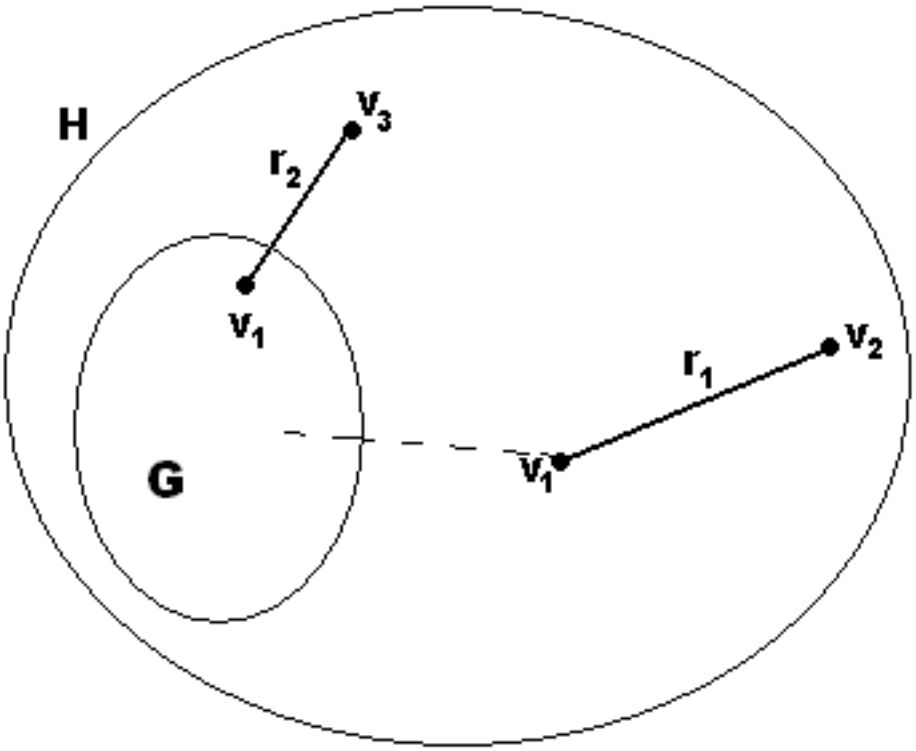


Figure 6.2: Sentence graph

This approach has a serious problem associated with it. It may sometimes return an document as relevant to a query, even when none of the sentences in the document has the query as its subgraph. To understand this more clearly, consider a UNL graph G. The vertices of this graph will be the UWs. Let vertex $v_1 \in V(G)$. Consider a query, $Q = G \bigcup r_1(v_1, v_2)$ as shown in figure 6.1. Now consider a document have a sentence graph H, with $G \subset H$ and it has $r_2(v_1, v_3)$ instead of $r_1(v_1, v_2)$ present in the query (figure 6.2). Also, H has another edge $r_1(v_1, v_2)$ inside it. But this v_1 is not same as the v_1 present in G. Clearly, if we apply the above algorithm, this document will also be returned even when it does not contain the query subgraph.

To overcome this problem, some processing will be needed after finding the relevant sentences to make sure that the edges found inside the sentence have the same connections as they have in the query. Whether two UWs (vertices) are two different occurrences of the same concept is indicated by their UW-IDs. In order to establish connectively of two edges $r_1(u_1, u_2)$ and $r_2(u_2, u_3)$, we only need to make sure that UW-ID of u_2 in first relation is same as that of u_2 in the second relation.

6.3.2 Partial Matching

Another drawback in the complete matching approach is that its one-or-none matching. As already mentioned a sentence will be either relevant to a query or it won't be. There is no concept of *partial* or *approximate* matching. Undoubtedly, complete matching approach will

lead to high precision but low recall. To overcome this problem, we can introduce a partial matching scheme which has lower precision but higher recall.

Two different occurrences of same UW in two different edges is said to be *linked* if the UW-IDs of these two occurrences is same. In the document graph, we say that a link between two occurrences of same UWs, is a *correct link* if there is link corresponding to this link, between the UWs in the query graph also. A correct link is defined only for the common edges in the document and the query.

As we did in complete matching, we will index the documents on their edges. The relevance of a document $R_q(d)$ is same as given in the complete matching scheme. But for partial matching the $r_q(s)$ is now defined as:

$$r_q(s) = \alpha\frac{n}{N} + (1-\alpha)\frac{l}{L}$$

where, n is number of relation edges (of the query) found in the sentence. N is total number of relation edges in the query. l is number of correct links in the sentence and L is the total number of links between all UWs in the query. α is a empirical constant. Here is linear relation is assumed, but the relation can as well be exponential, with the value of $r_q(s)$ increasing exponentially with linear increase in percentage match of the query.

In both of these approaches we have assumed that all the sentences have equal importance. But search engines like Google, give more weight to terms if they are present in the title or in bigger font. We can do the same by giving more weight those $r_q(s)$ which corresponds to sentences having bigger font or title sentence(s).

The partial matching approach treats a edge as the smallest unit of *meaning*, which may not be desirable in some cases. As discussed in chapter 7 this may not be desirable in some cases. We can remedy this problem by adding an extra term in $r_q(s)$ which depends on how many UW's of the query are present in the document.

6.3.3 Partial Matching Algorithm

As already mentioned, all the documents are indexed on the triplet - Relation, UW1 and UW2. As explained in chapter 3, the indexer module adds following entries for each relation edge in the document in the unlindex table - document id, sentence number, UW1, UW2, UWID1, UWID2, where UWID1 and UWID2 are ids associated with the first and the second Universal words of the relation.

For calculating $r_q(s)$ we need to know values of n, N, l and L. N and L are independent of s and can be calculated by simply considering the query q. As explained earlier, for a query q, N is the total number of relation edges in query. L, which is the total number of links in the query can be easily calculated if we observe the following. For a query graph $G(E,V)$, L is given by

$$L = \sum_{v \in V}(degree(v) - 1)$$

The algorithm given in figure 6.3 can be used to calculate L and N.

$UWID1_q$ and $UWID2_q$ are the IDs of first and second parameters of the current relation in the query. uidset is a set containing UWID's from the query.

Now for each (document,sentence) pair, we need to know the values of n and l to find out $R_q(d)$, the relevance of a document. All the relation edges present in the documents, matching

```
Procedure CalculateLN
Input:  UNL expression for the query
Output: L
Begin
    N = Number of relation edges in the Input.
    Initialize uidset and L = 0
    for each relation edge in the query
        if UWID1_q ∈ uidset // We have got a link
            L++
        else uidset = uidset ⋃ {UWID1_q}
        if UWID2_q ∈ uidset // We have got a link
            L++
        else uidset = uidset ⋃ {UWID2_q}
    return N and L
End
```

Figure 6.3: Algorithm for finding N and L

the relation edges of the query can be found out by giving a simple SQL query as we have indexed on relation and the universal words. By ordering the results by document id and sentence number, we will get all the matching relations of a (document,sentence) pair together. We can run the algorithm given in figure 6.4 on this input to find out n and l.

$UWID1_q$ and $UWID2_q$ are the IDs of first and second parameters of the current relation edge in the query. uidtable is a map from UWIDs of the query to set of UWIDs present in the document. uidtable($UWID1_q$) returns the set associated with $UWID1_q$. This algorithm assumes that for each relation edge in the query, we have a unique matching relation edge in a (document,sentence) pair.

Procedure RELEVANCE
Input: All the matching relation edges of a query for a (document,sentence) pair
Output: $r_q(s)$
Begin
n = Number of relation edges in the Input.
Initialize uidtable and $l = 0$
for each relation edge in the query
find the corresponding (matching) edge in the input
if the edge is found
if $UWID1_q \in$ uidtable // We have got a link
if $UWID1$ of the current relation $\in$ uidtable($UWID1_q$) // Its a correct link
$l++$
else uidtable($UWID1_q$) = uidtable($UWID1_q$) $\bigcup$ $UWID1$ of the current relation
else uidtable($UWID1_q$) = { $UWID1$ of the current relation }

if $UWID2_q \in$ uidtable // We have got a link
if $UWID2$ of the current relation $\in$ uidtable($UWID2_q$) // Its a correct link
$l++$
else uidtable($UWID2_q$) = uidtable($UWID2_q$) $\bigcup$ $UWID2$ of the current relation
else uidtable($UWID2_q$) = { $UWID2$ of the current relation }
$r_q(s) = \alpha\frac{n}{N} + (1-\alpha)\frac{l}{L}$
End

Figure 6.4: Algorithm for finding relevance of a sentence

Chapter 7

Experimental Results

This chapter provides insight into the working of search engine by considering some experimental queries and analyzing their results. These queries were made through the English interface of the search engine. We have indexed only one document in the search engine for these example queries. The document is attached in Appendix A for reference.

The results are presented as the line numbers of the sentences matched along with the sentence in English language. The reader is encouraged to check out the UNL expressions of the lines matched to gain insight into the matching algorithm.

7.1 Some Representative Example Queries

- **Query :** agriculture

  Enconverted Query :

  `agricultural(mod<thing):00.@entry`

  Results :

  1. *Line 5*: However, India would have been in an even better position now both in terms of agricultural output and economic development had our planners given the required importance to its development it deserved in the early years since independence.
  2. *Line 6*: Even today, the farmers in India are able to obtain only 15 per cent of their requirements of agricultural credit from banks.

  Comments : Both these lines have the UW of the query present in their UNL expressions.

- **Query :** public investment

  Enconverted Query :

  `mod(investment(icl>assets):07.@entry,public(icl>general(aoj>thing)):00)`

  Results :

1. *Line 12*: Though the overall growth of Indian economy has depended much upon the performance of agriculture, over the years, not much public investment has been made on its development.
2. *Line 13*: There is a steady deceleration in public investment in gross capital formation in agriculture.
3. *Line 14*: In 1980-81, the public investment as a percentage of gross capital formation in agriculture was 38.7 percent which fell to 16.2% in 1996-97.

- **Query :** moneylenders exploit farmers

  Enconverted Query :

```
obj(exploit(agt>thing,obj>thing):0D.@entry.@present,
    farmer(icl>occupation):0L.@pl)
agt(exploit(agt>thing,obj>thing):0D.@entry.@present,
    moneylender(icl>occupation):00.@pl)
```

  Results :

  1. *Line 10*: The farmers are still being exploited by moneylenders who provide finance at exorbitant interest rates and there are cartels of traders who pay very little for their produce even in the well recognized mandies in the country.

- **Query :** farmers exploit moneylenders

  Enconverted Query :

```
obj(exploit(agt>thing,obj>thing):08.@entry.@present,
    moneylender(icl>occupation):0G.@pl)
agt(exploit(agt>thing,obj>thing):08.@entry.@present,
    farmer(icl>occupation):00.@pl)
```

  Results : No results.

  Comments : There were no lines present in the document mentioning anything about farmers exploiting moneylenders. The last two queries clearly highlight the power of meaning based search. A simple pattern based search engine would have returned the same results for both these queries.

- **Query :** green revolution

  Enconverted Query :

```
mod(revolution(icl>event):06.@entry,green(icl>colour):00)
```

Results : No results.

Comments : Line number 2 of the document contains above phrase
It has gone through a green revolution, a white revolution, a yellow revolution and a blue revolution.
and the corresponding UNL expression is:

```
agt(go through(agt>thing,obj>thing).@entry.@present.@complete,
    it(icl>thing))
obj(go through(agt>thing,obj>thing).@entry.@present.@complete, :01)
and:01(revolution(icl>event):03,revolution(icl>event):02)
and:01(revolution(icl>event):04,revolution(icl>event):03)
and:01(revolution(icl>event):05.@entry,revolution(icl>event):04)
mod:01(revolution(icl>event):02.@indef,green(icl>color))
mod:01(revolution(icl>event):03.@indef,white(icl>color))
mod:01(revolution(icl>event):04.@indef,yellow(icl>color))
mod:01(revolution(icl>event):05.@indef,blue(icl>color))
[/S]
```

We can see from the sixth UNL expression that UW used for green in the document and the one generated by ENCO is different. The spelling of color in the icl tag does not match. This problem of non-standard UWs is a common in many queries posed to the search engine.

- **Query :** government agencies

Enconverted Query :

```
mod(agency(icl>organization>group):0B.@entry.@pl,
    government(icl>governmental organization):00)
```

Results : No results.

Comments : Line number 8 of the document contains above phrase

and the corresponding UNL expression is:

```
[S:8]
and(:02.@entry,:01)
aoj:01(able(aoj>thing).@entry,farmer(icl>occupation).@def.@pl)
mod:01(percent(icl>ratio),farmer(icl>occupation).@def.@pl)
qua:01(percent(icl>ratio),:03)
fmt:03(23.@entry,30)
```

```
mod:01(:03,only(mod<thing))
tim:01(able(aoj>thing).@entry,present(icl>time))
pur:01(able(aoj>thing).@entry,derive(agt>thing,obj>thing):05)
obj:01(derive(agt>thing,obj>thing):05, benefit(icl>profit):06.@pl)
mod:01(benefit(icl>profit):06.@pl,any(mod<thing))
mod:01(benefit(icl>profit):06.@pl,
    extension service(icl>activity):07.@pl)
obj:01(provide(icl>give(agt>thing,gol>thing,obj>thing)):08.@complete,
    extension service(icl>activity):07.@pl.@topic)
agt:01(provide(icl>give(agt>thing,gol>thing,obj>thing)):08.@complete,
    agency(icl>organization).@pl)
mod:01(agency(icl>organization).@pl,
    government(icl>governmental organization))
mod:01(agency(icl>organization).@pl,various(mod<thing))
obj:02(lose(agt>thing,obj>thing).@entry.@present.@complete,
    percent(icl>ratio).@topic)
qua:02(percent(icl>ratio).@topic,20)
man:02(20,about(icl>how))
mod:02(percent(icl>ratio),crop(icl>food).@def)
dur:02(lose(agt>thing,obj>thing).@entry.@present.@complete,
    year(icl>time))
mod:02(year(icl>time),every(mod<thing))
rsn:02(lose(agt>thing,obj>thing).@entry.@present.@complete,:04)
and:04(spillage(icl>result),mishandle(agt>thing,obj>thing).@progress)
and:04(flood(icl>phenomenon).@pl,spillage(icl>result))
and:04(drought(icl>phenomenon).@pl,flood(icl>phenomenon).@pl)
and:04(pest(icl>animal).@pl,drought(icl>phenomenon).@pl)
and:04(disease(icl>state).@entry.@pl,pest(icl>animal).@pl)
[/S]
```

Different UWs are being used for *agency* in the enconverted query and the document. This is again problem of standardization of UWs.

- **Query :** progress in agriculture

  Enconverted Query :

  ```
  scn(progress(icl>movement):00.@entry.@pl,agriculture(icl>activity):0C)
  ```

  Results : No results.

  Comments : Line number 1 of the document contains above phrase
  India has made lot of progress in agriculture since independence in terms of growth in output, yields and area under many crops.
  and the corresponding UNL expression is:

```
[S:1]
agt(make(agt>thing,obj>thing).@entry.@present.@complete,
    India(iof>country))
obj(make(agt>thing,obj>thing).@entry.@present.@complete,
    progress(icl>movement))
qua(progress(icl>movement),lot(icl>quantity))
scn(make(agt>thing,obj>thing).@entry.@present.@complete,
    agriculture(icl>activity))
tmf(make(agt>thing,obj>thing).@entry.@present.@complete,
    independence(icl>value))
man(make(agt>thing,obj>thing).@entry.@present.@complete,:01)
obj:01(in terms of(icl>how(obj>thing)).@entry, growth(icl>phenomenon))
mod:01(growth(icl>phenomenon),:02)
and:02(yield(icl>gain).@pl,output(icl>product))
and:02(area(icl>place).@entry,yield(icl>gain).@pl)
plc:02(area(icl>place),under(icl>place))
bas:02(under(icl>place),crop(icl>food).@pl)
qua:02(crop(icl>food).@pl,many(aoj>thing))
[/S]
```

As we can see from the second and forth UNL expressions the relation between progress and agriculture is not a direct one in the agriculture. As we have explained earlier, we are considering a relation as the smallest unit of meaning in the subgraph matching. Due to this, the above line was not matched by the search engine. This can be corrected either by taking a UW as smallest unit of meaning or doing some additional graph analysis at the graph matching stage to identify such kind of indirect relationships. In the former case, we have to modify the current partial matching algorithm slightly in order to take into account matching of UW's.

Chapter 8

Current status of the implementation

8.1 Indexer module

Status: Completed
Comments: This module was successfully completed and thoroughly tested.

8.2 Search module

Status: Completed
Comments: This module was successfully completed and thoroughly tested.

8.3 Post Processor

Status: Completed
Comments: This module was successfully completed and thoroughly tested.

8.4 Interface module

Status: Completed
Comments: Query input and result interfaces for English and Hindi is completed. Marathi interface for query input and result is also complete but it was not integrated with the search engine, as currently the Marathi ENCO software is not available.

8.5 IITK to DVTT-Yogesh Converter

Status: Completed
Comments: This module was successfully completed and thoroughly tested.

8.6 Enconverter module

Status: Completed

Comments: Presently the Hindi and English ENCO softwares are integrated with this module. More ENCO softwares can be easily added by modifying the query2 PHP script.

8.7 Deconverter module

Status: Completed
Comments: No DECO softwares were integrated with this module as presently no satisfactorily working DECO system was available. Hence all deconversion was done *manually* and the deconverted sentence file was kept as cache.

8.8 HTML Parser

Status: Partially Completed
Comments: A simple HTML parser was implemented in this project. It needs to be improved to handle more complex HTML pages.

8.9 Crawler Module

Status: Partially Completed
Comments: A primitive focused crawler was implemented in this project. Given a set of yahoo categories representing a domain, this crawler crawls the categories as well as any sub categories present in them recursively and extracts the links of pages which represent the domain. It also suggests addition categories that may be relevant to the domain presented. Then it crawls the pages to build a model for the classifier. The rainbow classifier is not currently integrated with the system.

Chapter 9

Conclusions and Future work

9.1 Strengths of the search engine

1. This search engine eliminates the language barrier. Language barrier is the biggest obstacle in taking knowledge to the masses. Generally all the information, especially on the web, is in English or other major world languages. The majority of the population of the world doesn't understand these languages. To make this information available to all, the information has to be made language independent. There are two methods for doing this:

   - The first method is to convert every language into all other possible languages. But this is not feasible. If we consider that there are only 10 languages in the world, we will have ${}^{10}P_2 = 90$ translators.
   - The second method is to consider an intermediate language. Universal Networking Language (UNL) is one such language. All the documents are converted into this intermediate language and a document can be converted back into any other language, from this intermediate representation. This makes the method extremely compact. For 10 languages, we need only $10 \times 2 = 20$ translators.

   The second method is used in this project. Initially, we are considering only English language.

2. Our search engine is a Meaning Based Search Engine. UNL, the intermediate language, uses Meaning Representation, i.e. it not only stores a word but also its meaning and attributes. For example, a word like "drink" will have different meanings in different sentences. It might mean "putting liquids in the mouth", or "liquids that are put in the mouth", or "liquids with alcohol", or "absorb" etc. But a UNL representation "drink(icl>do,obj>liquid)" denotes the subset of these concepts, "putting liquids into the mouth" which in turn corresponds to "drink", "gulp", "chug" and "slurp" in English. Attributes of a word provide information about how these concepts are being used in a particular sentence. As both the documents and queries are in their UNL representations, they are unambiguous and only documents that exactly match the query are retrieved.

   The results are much more accurate than any other techniques currently used. There are a few Meaning Based search engines on the web like www.oingo.com and www.simpli.com but their results are vague. Universally recognized as the best search engine, www.google.com,

is not a meaning based search engine. It uses text-matching techniques. Its results are highly relevant but it returns a lot of extraneous pages that are not related to the searched query. For example, for a query "Prime Minister of India", its initial results are accurate but the documents at the end have only partial matches, i.e. they might only have "Prime" or "Minister" or "India" or some combination of these words.

In Google, we can specify that only exact matches be retrieved. Hence, only pages that have "Prime Minister of India" as a phrase will be retrieved. But pages that have "Indian Prime Minister" will not be retrieved. In our search engine, both these cases will be matched.

9.2 Conclusion

In this report, a different approach to the problem of meaning based search engine and multi-linguality is presented. We believe that making searches on the web meaning based is becoming the need of the hour. Also, the importance of multi-linguality should at least be at par with the other aspects of a search engine. For the search results to be realistic and meaningful, they must encompass the typical user's requirements and specifications.

The model in this report is an amalgamation of two independent features. We integrated the user's language requirement with the relative importance of knowledge the user seeks. This has been possible by using the UNL as an intermediary language. UNL representation is language independent and captures the relationship between the words and their attributes. Hence multi-lingual and meaning based properties can be incorporated together rather than using separate language translators in search engines.

The bottleneck created by the exponentially expanding content of the web, more so in dominant languages, has been drawing attention to the area of sense disambiguation and linguistic issues, due to which the demand of producing more rigorous solutions, and hence more complex ones, has been constantly rising. Our model is a unique and early attempt to address this issue.

9.3 Future Work

Future work will touch upon the following areas:

- Testing the scalability of the search engine on a bigger corpus.
- Currently, the ENCO and DECO softwares fails for many sentences. They have to be improved before this search engine can be used on a bigger scale.
- The HTML parser developed is quite primitive and may fail on bigger, complex and as often is the case, erroneous HTML pages. We need to add more rules to it to make it able to handle complex HTML documents.
- Currently, the global page rank (relevance) is not implemented. It can be incorporated into the relevance returned to improve the ranking of the search engine.
- Improving the current search algorithm by adding reasoning and inferencing capabilities to it.

Appendix A

Document used in experiments

The following document was used for the experimental queries that was given to the search engine and were analyzed in chapter 7.

```
[S:1]
; India has made lot of progress in agriculture since independence in terms of
; growth in output, yields and area under many crops.
agt(make(agt>thing,obj>thing).@entry.@present.@complete,India(iof>country))
obj(make(agt>thing,obj>thing).@entry.@present.@complete,progress(icl>movement))
qua(progress(icl>movement),lot(icl>quantity))
scn(make(agt>thing,obj>thing).@entry.@present.@complete,agriculture(icl>activity))
tmf(make(agt>thing,obj>thing).@entry.@present.@complete,independence(icl>value))
man(make(agt>thing,obj>thing).@entry.@present.@complete,:01)
obj:01(in terms of(icl>how(obj>thing)).@entry, growth(icl>phenomenon))
mod:01(growth(icl>phenomenon),:02)
and:02(yield(icl>gain).@pl,output(icl>product))
and:02(area(icl>place).@entry,yield(icl>gain).@pl)
plc:02(area(icl>place),under(icl>place))
bas:02(under(icl>place),crop(icl>food).@pl)
qua:02(crop(icl>food).@pl,many(aoj>thing))
[/S]
[S:2]
; It has gone through a green revolution, a white revolution, a yellow revolution
; and a blue revolution.
agt(go through(agt>thing,obj>thing).@entry.@present.@complete,it(icl>thing))
obj(go through(agt>thing,obj>thing).@entry.@present.@complete, :01)
and:01(revolution(icl>event):03,revolution(icl>event):02)
and:01(revolution(icl>event):04,revolution(icl>event):03)
and:01(revolution(icl>event):05.@entry,revolution(icl>event):04)
mod:01(revolution(icl>event):02.@indef,green(icl>color))
mod:01(revolution(icl>event):03.@indef,white(icl>color))
mod:01(revolution(icl>event):04.@indef,yellow(icl>color))
mod:01(revolution(icl>event):05.@indef,blue(icl>color))
[/S]
```

```
[S:3]
; Today, India is the largest producer of milk, fruits, cashewnuts, coconuts and
; tea in the world, the second largest producer of wheat, vegetables, sugar and
; fish and the third largest producer of tobacco and rice.
tim(:01,today(icl>day).@entry)
aoj:01(producer(icl>thing):02.@entry.@def,India(iof>country))
and:01(:05,:04)
and:01(:06,:05)
mod:04(producer(icl>thing):02.@entry.@def,:07)
man:07(large(icl>big(aoj>thing)):7a.@entry,most(icl>how):19)
mod:04(producer(icl>thing):02.@entry.@def,:08)
and:08(fruit(pof>plant).@pl,milk(icl>beverage))
and:08(cashewnut(icl>fruit).@pl,fruit(pof>plant).@pl)
and:08(coconut(icl>fruit).@pl,cashewnut(icl>fruit).@pl)
and:08(tea(icl>beverage).@entry,coconut(icl>fruit).@pl)
plc:04(producer(icl>thing):02.@entry.@def,world(icl>region).@def)
mod:05(producer(icl>thing):02.@entry.@def,:10)
man:10(large(icl>big(aoj>thing)):15.@entry,most(icl>how):18)
mod:05(:10,2.@ordinal)
mod:05(producer(icl>thing):02.@entry.@def,:11)
and:11(vegetable(icl>food).@pl,wheat(icl>cereal))
and:11(sugar(icl>seasoning),vegetable(icl>food).@pl)
and:11(fish(icl>animal).@entry,sugar(icl>seasoning))
mod:06(producer(icl>thing):02.@entry.@def,:13)
man:13(large(icl>big(aoj>thing)):16.@entry,most(icl>how):17)
mod:06(:13,3.@ordinal)
mod:06(producer(icl>thing):02.@entry.@def,:14)
and:14(rice(icl>grain).@entry,tobacco(icl>product))
[/S]
[S:4]
; The per capita availability of foodgrains has risen in the country from 350 gm
; in 1951 to about 500 gm per day now, of milk from less than 125 gm to 210 gm
; per day and of eggs from 5 to 30 per annum despite the increase in population
; from 35 crores to 95 crores.
plc(rise(gol>thing,obj>thing):04.@entry.@present.@complete,
    country(icl>region).@def)
coo(rise(gol>thing,obj>thing):04.@entry.@present.@complete,
    increase(icl>phenomenon).@def.@contrast)
mod(increase(icl>phenomenon).@def.@contrast,population(icl>person))
src(increase(icl>phenomenon).@def.@contrast,:17)
gol(increase(icl>phenomenon).@def.@contrast,:18)
qua:17(crore(iof>10 000 000):19.@entry.@pl,35)
qua:18(crore(iof>10 000 000):20.@entry.@pl,95)
and(:02,:01)
and(:03,:02)
```

```
per(availability(icl>accessibility):06.@def,capita(icl>person))
obj:01(rise(gol>thing,obj>thing):04.@entry.@present.@complete,
    availability(icl>accessibility):06.@def)
mod:01(availability(icl>accessibility):06.@entry.@def,foodgrain(icl>cereal).@pl)
src:01(rise(gol>thing,obj>thing):04.@entry.@present.@complete,:07)
gol:01(rise(gol>thing,obj>thing):04.@entry.@present.@complete,:08)
qua:07(gm(icl>weight unit):09.@entry,350)
per:01(:07,day(icl>period):11)
dur:01(:07,1951)
qua:08(gm(icl>weight unit):10.@entry,500)
man:08(500,about(icl>how))
per:01(:08,day(icl>period):11)
tim:01(:08,now(icl>time))
obj:02(rise(gol>thing,obj>thing):04.@entry.@present.@complete,
    availability(icl>accessibility):06)
mod:02(availability(icl>accessibility):06.@def,milk(icl>beverage))
src:02(rise(gol>thing,obj>thing):04.@entry.@present.@complete,:12)
gol:02(rise(gol>thing,obj>thing):04.@entry.@present.@complete,:13)
qua:12(gm(icl>weight unit):14.@entry,125)
bas:12(less(aoj>thing),125)
per:02(:12,day(icl>period):16)
qua:13(gm(icl>weight unit):15.@entry,210)
per:02(:13,day(icl>period):16)
obj:03(rise(gol>thing,obj>thing):04.@entry.@present.@complete,
    availability(icl>accessibility):06)
mod:03(availability(icl>accessibility):06.@def,egg(icl>food).@pl)
src:03(rise(gol>thing,obj>thing):04.@entry.@present.@complete,3)
gol:03(rise(gol>thing,obj>thing):04.@entry.@present.@complete,30)
per:03(3,annum(icl>year))
per:03(30,annum(icl>year))
[/S]
[S:5]
; However, India would have been in an even better position now both in terms of
; agricultural output and economic development had our planners given the
; required importance to its development it deserved in the early years since
; independence.
tim(:01.@entry.@contrast,now(icl>time))
man(:01.@entry.@contrast,:02)
scn:01(India(iof>country):08.@entry.@probable.@complete,position(icl>state).@indef)
mod:01(position(icl>state).@indef,:09)
man:09(good(aoj>thing),more(icl>how))
man:01(:09,even(icl>how))
obj:02(in terms of(icl>how(obj>thing)).@entry, :03)
and:03(development(icl>action):07.@entry,output(icl>product))
mod:03(output(icl>product),agricultural(mod<thing))
```

```
mod:03(development(icl>action):07,economic(mod<thing))
con(:01.@entry.@contrast,:04)
agt:04(give(agt>thing,gol>thing,obj>thing).@entry.@past.@complete,
    planner(icl>person):06.@pl)
mod:04(planner(icl>person):06.@pl,we)
obj:04(give(agt>thing,gol>thing,obj>thing).@entry.@past.@complete,
    importance(icl>value).@def)
mod:04(importance(icl>value).@def,required(mod<thing))
gol:04(give(agt>thing,gol>thing,obj>thing).@entry.@past.@complete,
    development(icl>action):07)
aoj:04(deserve(aoj>thing,obj>thing):05.@past,development(icl>action):07)
mod:04(development(icl>action):07,India(iof>country):08)
obj:04(deserve(aoj>thing,obj>thing):05.@past,importance(icl>value).@def)
dur:04(deserve(aoj>thing,obj>thing):05.@past,year(icl>time).@def.@pl)
mod:04(year(icl>time).@def.@pl,early(mod<thing))
tmf:04(year(icl>time).@def.@pl,independence(icl>value))
[/S]
[S:6]
; Even today, the farmers in India are able to obtain only 15 per cent of their
; requirements of agricultural credit from banks.
tim(:01,today(icl>day).@entry)
man(today(icl>day).@entry,even(icl>how))
aoj:01(able(aoj>thing).@entry, farmer(icl>occupation).@def.@pl)
plc:01(farmer(icl>occupation).@def.@pl,India(iof>country))
pur:01(able(aoj>thing).@entry,obtain(icl>get(agt>volitional thing,obj>thing)))
obj:01(obtain(icl>get(agt>volitional thing,obj>thing)),percent(icl>ratio))
qua:01(percent(icl>ratio),15)
mod:01(15,only(mod<thing))
mod:01(percent(icl>ratio),requirement(icl>necessity).@pl)
mod:01(requirement(icl>necessity).@pl,credit(icl>money))
mod:01(requirement(icl>necessity).@pl,they(icl>persons))
mod:01(credit(icl>money),agricultural(mod<thing))
src:01(obtain(icl>get(agt>volitional thing,obj>thing)),bank(icl>institution).@pl)
[/S]
[S:7]
; The various state seed corporations are able to produce only 1 0 per cent of
; the seeds required by our farmers.
aoj(able(aoj>thing).@entry,corporation(icl>company).@def.@pl)
mod(corporation(icl>company).@def.@pl,seed(pof>plant))
mod(corporation(icl>company).@def.@pl,state(icl>government))
mod(corporation(icl>company).@def.@pl,various(mod<thing))
pur(able(aoj>thing).@entry,produce(agt>thing,obj>thing))
obj(produce(agt>thing,obj>thing),percent(icl>ratio))
qua(percent(icl>ratio),10)
mod(10,only(mod<thing))
```

```
mod(percent(icl>ratio),seed(pof>plant).@def.@pl)
obj(require(icl>necessitate(agt>thing,gol>place)).@complete,
    seed(pof>plant).@def.@pl.@topic)
agt(require(icl>necessitate(agt>thing,gol>place)).@complete,
    farmer(icl>occupation).@pl)
mod(farmer(icl>occupation).@pl,we)
[/S]
[S:8]
; At present only 23-30 per cent of the farmers are able to derive any benefits
; of extension services provided by various government agencies and every year
; about 20 per cent of the crop is lost due to mishandling, spillage, floods,
; droughts and pests and diseases.
and(:02.@entry,:01)
aoj:01(able(aoj>thing).@entry,farmer(icl>occupation).@def.@pl)
mod:01(percent(icl>ratio),farmer(icl>occupation).@def.@pl)
qua:01(percent(icl>ratio),:03)
fmt:03(23.@entry,30)
mod:01(:03,only(mod<thing))
tim:01(able(aoj>thing).@entry,present(icl>time))
pur:01(able(aoj>thing).@entry,derive(agt>thing,obj>thing):05)
obj:01(derive(agt>thing,obj>thing):05, benefit(icl>profit):06.@pl)
mod:01(benefit(icl>profit):06.@pl,any(mod<thing))
mod:01(benefit(icl>profit):06.@pl,extension service(icl>activity):07.@pl)
obj:01(provide(icl>give(agt>thing,gol>thing,obj>thing)):08.@complete,
    extension service(icl>activity):07.@pl.@topic)
agt:01(provide(icl>give(agt>thing,gol>thing,obj>thing)):08.@complete,
    agency(icl>organization).@pl)
mod:01(agency(icl>organization).@pl,government(icl>governmental organization))
mod:01(agency(icl>organization).@pl,various(mod<thing))
obj:02(lose(agt>thing,obj>thing).@entry.@present.@complete,
    percent(icl>ratio).@topic)
qua:02(percent(icl>ratio).@topic,20)
man:02(20,about(icl>how))
mod:02(percent(icl>ratio),crop(icl>food).@def)
dur:02(lose(agt>thing,obj>thing).@entry.@present.@complete,year(icl>time))
mod:02(year(icl>time),every(mod<thing))
rsn:02(lose(agt>thing,obj>thing).@entry.@present.@complete,:04)
and:04(spillage(icl>result),mishandle(agt>thing,obj>thing).@progress)
and:04(flood(icl>phenomenon).@pl,spillage(icl>result))
and:04(drought(icl>phenomenon).@pl,flood(icl>phenomenon).@pl)
and:04(pest(icl>animal).@pl,drought(icl>phenomenon).@pl)
and:04(disease(icl>state).@entry.@pl,pest(icl>animal).@pl)
[/S]
[S:9]
; In fruits and vegetables the loss is around 30 per cent.
```

```
aoj(percent(icl>ratio).@entry,loss(icl>event).@def)
qua(percent(icl>ratio).@entry,30)
man(30,around(icl>how))
mod(loss(icl>event).@def,:01)
and:01(vegetable(icl>food).@entry.@pl,fruit(pof>plant).@pl)
[/S]
[S:10]
; The farmers are still being exploited by moneylenders who provide finance at
; exorbitant interest rates and there are cartels of traders who pay very little
; for their produce even in the well recognized mandies in the country.
and(:02.@entry,:01)
obj:01(exploit(agt>thing,obj>thing).@entry.@present.@progress.@complete,
    farmer(icl>occupation).@def.@pl.@topic)
agt:01(exploit(agt>thing,obj>thing).@entry.@present.@progress.@complete,
    moneylender(icl>occupation).@pl)
tim:01(exploit(agt>thing,obj>thing).@entry.@present.@progress.@complete,
    still(icl>how))
agt:01(provide(icl>give(agt>thing,gol>thing,obj>thing)).@present,
    moneylender(icl>occupation).@pl)
obj:01(provide(icl>give(agt>thing,gol>thing,obj>thing)).@present,
    finance(icl>economy))
cob:01(provide(icl>give(agt>thing,gol>thing,obj>thing)).@present,:04)
mod:04(rate(icl>charge).@entry.@pl,interest(icl>profit))
mod:01(:04,exorbitant(mod<thing))
aoj:02(exist(aoj>thing).@entry.@present,cartel(icl>syndicate).@pl)
mod:02(cartel(icl>syndicate).@pl,trader(icl>occupation).@pl)
agt:02(pay(agt>thing,obj>thing,pur>thing).@present,cartel(icl>syndicate).@pl)
obj:02(pay(agt>thing,obj>thing,pur>thing).@present,little(aoj>thing))
man:02(little(aoj>thing),very(icl>how))
gol:02(pay(agt>thing,obj>thing,pur>thing).@present,produce(icl>result))
mod:02(produce(icl>result),they(icl>persons))
plc:02(pay(agt>thing,obj>thing,pur>thing).@present,:03)
man:02(:03,even(icl>how))
mod:03(mandi(icl>market).@entry.@def.@pl,recognized(mod<thing))
man:03(recognized(mod<thing),well(icl>how))
plc:03(mandi(icl>market).@entry.@def.@pl,country(icl>region).@def)
[/S]
[S:11]
; Gross Capital Formation in Agriculture
scn(formation(icl>arrangement).@entry,agriculture(icl>activity))
mod(formation(icl>arrangement),:01)
mod:01(capital(icl>assets),gross(mod<thing))
[/S]
[S:12]
; Though the overall growth of Indian economy has depended much upon the
```

```
; performance of agriculture, over the years, not much public investment
; has been made on its development.
dur(:01.@entry,year(icl>time).@def.@pl)
and:01(:03.@entry,:02)
aoj:02(depend upon(aoj>thing,obj>thing).@entry.@present.@complete.@although,
    growth(icl>phenomenon).@def)
mod:02(growth(icl>phenomenon).@def,economy(icl>abstract thing))
mod:02(economy(icl>abstract thing),Indian(aoj>thing))
mod:02(growth(icl>phenomenon).@def,overall(mod<thing))
obj:02(depend upon(aoj>thing,obj>thing).@entry.@present.@complete.@although,
    performance(icl>operation))
man:02(depend upon(aoj>thing,obj>thing).@entry.@present.@complete.@although,
    much(icl>how):04)
mod:02(performance(icl>operation).@def,agriculture(icl>activity))
obj:03(make(agt>thing,obj>thing).@entry.@present.@complete,
    investment(icl>assets).@topic)
qua:03(investment(icl>assets).@topic,much(mod<thing):05.@not)
mod:03(investment(icl>assets).@topic,public(icl>general(aoj>thing)))
gol:03(make(agt>thing,obj>thing).@entry.@present.@complete,development(icl>action))
mod:03(development(icl>action),agriculture(icl>activity))
[/S]
[S:13]
; There is a steady deceleration in public investment in gross capital formation
; in agriculture.
aoj(exist(aoj>thing).@entry.@present,deceleration(icl>reduction).@indef)
mod(deceleration(icl>reduction).@indef,steady(icl>stable))
mod(deceleration(icl>reduction).@indef,investment(icl>assets))
mod(investment(icl>assets),public(icl>general(aoj>thing)))
scn(investment(icl>assets),formation(icl>arrangement))
scn(formation(icl>arrangement),agriculture(icl>activity))
mod(formation(icl>arrangement),:01)
mod:01(capital(icl>assets).@entry,gross(mod<thing))
[/S]
[S:14]
; In 1980-81, the public investment as a percentage of gross capital formation in
    agriculture was 38.7 percent which fell to 16.2% in 1996-97.
tim(:02,:01.@entry)
fmt:01(1980.@entry,1981)
aoj:02(percent(icl>ratio):06.@entry.@past,:05)
aoj:05(percentage(icl>ratio).@indef.@entry,investment(icl>assets).@def)
mod:05(investment(icl>assets).@def,public(icl>general(aoj>thing)))
qua:02(percent(icl>ratio):06.@entry.@past,38.7)
obj:03(fall(obj>thing,gol>thing).@entry.@past,:05)
gol:03(fall(obj>thing,gol>thing).@entry.@past,:07)
qua:07(%,16.2)
```

```
dur(:03,:04)
fmt:04(1996.@entry,1997)
mod:05(percentage(icl>ratio).@indef.@entry,formation(icl>arrangement))
scn:05(formation(icl>arrangement),agriculture(icl>activity))
mod:05(formation(icl>arrangement),:06)
mod:06(capital(icl>assets).@entry,gross(mod<thing))
[/S]
[S:15]
; During this period the share of private investment, however, rose from 61.3%
; to 83.8%.
obj(rise(gol>thing,obj>thing).@entry.@contrast,share(icl>part).@def)
mod(share(icl>part).@def,investment(icl>assets))
mod(investment(icl>assets),private(mod<thing))
dur(rise(gol>thing,obj>thing).@entry.@contrast,period(icl>time))
mod(period(icl>time),this(mod<thing))
src(rise(gol>thing,obj>thing).@entry.@contrast,:01)
qua:01(%:02.@entry,61.3)
gol(rise(gol>thing,obj>thing).@entry.@contrast,:03)
qua:03(%:04.@entry,83.8)
[/S]
[S:16]
; This rise is attributed mostly to better terms of trade offered by the
; government to agriculture vis-a-vis industry.
obj(attribute to(agt>thing,obj>thing,gol>thing).@entry.@present.@complete,
    rise(icl>phenomenon))
mod(rise(icl>phenomenon),this(mod<thing))
man(attribute to(agt>thing,obj>thing,gol>thing).@entry.@present.@complete,
    mostly(icl>how))
gol(attribute to(agt>thing,obj>thing,gol>thing),term(icl>condition).@pl)
mod(term(icl>condition).@pl,good(aoj>thing))
man(good(aoj>thing),more(icl>how))
mod(term(icl>condition).@pl,trade(icl>activity))
obj(offer(icl>give(agt>thing,gol>thing,obj>thing)).@complete,
    term(icl>condition).@pl.@topic)
agt(offer(icl>give(agt>thing,gol>thing,obj>thing)).@complete,
    government(icl>governmental organization).@def)
gol(offer(icl>give(agt>thing,gol>thing,obj>thing)).@complete,:01)
and:01(industry(icl>activity).@entry.@contrast,agriculture(icl>activity))
[/S]
[S:17]
; The fall in public sector investment is attributed to increase in current
; expenditure to meet higher subsidies on food, fertilizers, electricity,
; irrigation, credit and other farm inputs rather than creating assets.
obj(attribute to(agt>thing,obj>thing,gol>thing).@entry.@present.@complete,
    fall(icl>phenomenon).@def)
```

```
mod(fall(icl>phenomenon),investment(icl>assets))
scn(investment(icl>assets),:03)
mod:03(sector(icl>group).@entry,public(icl>general(aoj>thing)))
gol(attribute to(agt>thing,obj>thing,gol>thing).@entry.@present.@complete,
    increase(icl>phenomenon))
mod(increase(icl>phenomenon),expenditure(icl>expense))
mod(expenditure(icl>expense),current(aoj>thing))
pur(increase(icl>phenomenon),:01)
and:01(create(agt>thing,obj>thing).@entry.@contrast.@progress,
    meet(agt>volitional thing,obj>thing))
obj:01(meet(agt>volitional thing,obj>thing).@entry,subsidy(icl>gift).@pl)
mod:01(subsidy(icl>gift).@pl,high(aoj>thing))
man:01(high(aoj>thing),more(icl>how))
pur:01(subsidy(icl>gift).@pl,:02)
and:02(fertilizer(icl>functional thing).@pl,food(icl>functional thing))
and:02(electricity(icl>phenomenon),fertilizer(icl>functional thing).@pl)
and:02(irrigation(icl>activity),electricity(icl>phenomenon))
and:02(credit(icl>money),irrigation(icl>activity))
and:02(input(icl>functional thing).@entry.@pl,credit(icl>money))
mod:02(input(icl>functional thing).@pl,farm(mod<thing))
mod:02(input(icl>functional thing).@pl,other(mod<thing))
obj:01(create(agt>thing,obj>thing).@contrast.@progress,asset(icl>property).@pl)
[/S]
[S:18]
; In recent years public expenditure on maintenance of existing irrigation
; projects has gone up and investment on new projects has fallen.
and(fall(obj>thing,gol>thing).@entry.@present.@complete,
    go up(icl>rise(gol>thing,obj>thing,src>thing)).@present.@complete)
tim(go up(icl>rise(gol>thing,obj>thing,src>thing)).@present.@complete,:01)
mod:01(year(icl>time).@entry,recent(mod<thing))
mod:03(expenditure(icl>expense):04.@entry,public(icl>general(aoj>thing)))
pur:03(expenditure(icl>expense):04.@entry,maintenance(icl>activity):05)
mod:03(maintenance(icl>activity):05,project(icl>plan):02.@pl)
mod:03(project(icl>plan):02.@pl,irrigation(icl>activity))
aoj:03(exist(aoj>thing).@progress,project(icl>plan):02.@pl)
obj(go up(icl>rise(gol>thing,obj>thing,src>thing)).@present.@complete,:03)
obj(fall(obj>thing,gol>thing).@entry.@present.@complete,investment(icl>assets))
mod(project(icl>plan):06.@pl,new(aoj>thing))
scn(investment(icl>assets),project(icl>plan):06.@pl)
[/S]
[S:19]
; Many of the projects started in early 1980's are still awaiting completion
; because of cost overruns and non-availabilty of necessary timely funds.
tim(await(icl>expect(agt>thing,obj>thing)).@entry.@present.@progress,
    still(icl>how))
```

```
agt(await(icl>expect(agt>thing,obj>thing)).@entry.@present.@progress,:01)
obj(await(icl>expect(agt>thing,obj>thing)).@entry.@present.@progress,
    completion(icl>action))
obj:01(start(icl>begin(agt>thing,obj>thing)).@entry.@complete,many(icl>thing))
mod:01(many(icl>thing),project(icl>plan).@def.@pl)
dur:01(start(icl>begin(agt>thing,obj>thing)).@entry.@complete,1980.@pl)
mod:01(1980.@pl,early(mod<thing))
rsn(await(icl>expect(agt>thing,obj>thing)).@entry.@present.@progress,:02)
and:02(availability(icl>accessibility).@entry.@not,overrun(icl>exceed).@pl)
mod:02(availability(icl>accessibility).@entry.@not,fund(icl>assets).@pl)
mod:02(fund(icl>assets).@pl,necessary(aoj>thing))
mod:02(fund(icl>assets).@pl,timely(mod<thing))
mod:02(overrun(icl>exceed).@pl,cost(icl>expenditure))
[/S]
[S:20]
; In the 8th plan it was proposed to raise the investment in agriculture to 18.7%
; of the total plan outlay, but the actual investment turned out to be about
; 10-11% only.
dur(:01,plan(icl>idea):12.@entry.@def)
mod(plan(icl>idea):12.@entry.@def,8.@ordinal)
and:01(:03.@entry.@contrast,:02)
obj:02(propose(agt>thing,obj>thing).@entry.@past.@complete,:04.@topic)
obj:04(raise(icl>increase(agt>thing,gol>thing,obj>thing)).@entry,
    investment(icl>assets):08.@def)
scn:04(raise(icl>increase(agt>thing,gol>thing,obj>thing)):08.@def,
    agriculture(icl>activity))
gol:04(raise(icl>increase(agt>thing,gol>thing,obj>thing)).@entry,:09)
qua:09(%,18.7)
mod:04(09,:05.@def)
mod:05(outlay(icl>expense).@entry,plan(icl>idea):13)
mod:04(:05.@def,total(mod<thing))
obj:03(turn out(obj>thing,gol>thing).@entry.@past,investment(icl>assets):10.@def)
mod:03(investment(icl>assets):10.@def,actual(mod<thing))
gol:03(turn out(obj>thing,gol>thing).@entry.@past,:06)
aoj:06(:11.@entry,investment(icl>assets):10.@def)
qua:11(%,:07)
fmt:07(10,11)
man:06(:07,about(icl>how))
mod:06(:07,only(mod<thing))
[/S]
[S:21]
; Growth in Foodgrain Production
mod(growth(icl>phenomenon).@entry,production(icl>action))
mod(production(icl>action),foodgrain(icl>cereal))
[/S]
```

```
[S:22]
; The foodgrain production in the 1950s rose in India because of expansion in area.
obj(rise(gol>thing,obj>thing).@entry.@past,production(icl>action).@def)
mod(production(icl>action).@def,foodgrain(icl>cereal))
dur(rise(gol>thing,obj>thing).@entry.@past,1950.@pl)
plc(rise(gol>thing,obj>thing).@entry.@past,India(iof>country))
rsn(rise(gol>thing,obj>thing).@entry.@past,expansion(icl>activity))
mod(expansion(icl>activity),area(icl>place))
[/S]
[S:23]
; In the 1960s, the growth rate was very poor which resulted in large scale
; imports.
dur(:01,1960.@entry.@pl)
aoj:01(poor(aoj>thing).@past.@entry,rate(icl>magnitude).@def)
mod:01(rate(icl>magnitude).@def,growth(icl>phenomenon))
man:01(poor(aoj>thing).@past.@entry,very(icl>how))
agt:01(result in(icl>cause(agt>thing,obj>thing)).@past,rate(icl>magnitude).@def)
obj:01(result in(icl>cause(agt>thing,obj>thing)).@past,import(icl>output).@pl)
mod:01(import(icl>output).@pl,large-scale(mod<thing))
[/S]
[S:24]
; The development, production and use of better seeds increased the productivity
; of wheat in 1970s and rice in 1980s.
agt(increase(agt>thing,obj>thing).@entry.@past,:02)
mod:02(:01.@entry.@def,seed(pof>plant).@pl)
and:01(use(icl>action).@entry,production(icl>action))
and:01(production(icl>action),development(icl>action))
mod:02(seed(pof>plant).@pl,good(aoj>thing))
man:02(good(aoj>thing),more(icl>how))
obj(increase(agt>thing,obj>thing).@entry.@past,
    productivity(icl>fruitfulness).@def)
mod(productivity(icl>fruitfulness).@def,:03)
and:03(rice(icl>grain).@entry,wheat(icl>cereal))
dur(rice(icl>grain),1980.@pl)
dur(wheat(icl>cereal),1970.@pl)
[/S]
[S:25]
; The eighties was also a period of green revolution which enabled India to become
; self-sufficient in foodgrain production and even a marginal exporter.
man(:01,also(icl>how).@entry)
aoj:01(period(icl>time).@entry.@def,eighty(icl>time).@def.@pl)
mod:01(period(icl>time).@entry.@def,:02)
mod:02(revolution(icl>event).@entry,green(icl>color))
agt:01(enable(agt>thing,obj>thing).@past,:02)
obj:01(enable(agt>thing,obj>thing).@past,India(iof>country))
```

```
pur:01(enable(agt>thing,obj>thing).@past,:03)
obj:03(become(gol>thing,obj>thing).@entry,India(iof>country))
gol:03(become(gol>thing,obj>thing).@entry,:04)
and:04(exporter(icl>occupation).@entry.@indef,self-sufficient(aoj>thing))
scn:04(self-sufficient(aoj>thing),production(icl>action))
mod:04(production(icl>action),foodgrain(icl>cereal))
mod:04(exporter(icl>occupation).@entry.@indef,marginal(mod<thing))
[/S]
[S:26]
; In the Nineties, however, the annual growth rate in production has fallen to
; 1.66 per cent from 3.54 per cent recorded in the eighties.
dur(:01.@contrast,ninety(icl>time).@entry.@def.@pl)
obj:01(fall(obj>thing,gol>thing).@entry.@present.@complete,
    rate(icl>magnitude).@def)
mod(rate(icl>magnitude),annual(aoj>thing))
mod(rate(icl>magnitude),growth(icl>phenomenon))
mod(growth(icl>phenomenon),production(icl>action))
gol(fall(obj>thing,gol>thing).@entry.@present.@complete,:02)
qua:02(percent(icl>ratio):04,1.66)
src(fall(obj>thing,gol>thing).@entry.@present.@complete,:03)
qua:03(percent(icl>ratio):05,3.54)
obj(record(icl>document(agt>thing,obj>thing)).@complete,percent(icl>ratio):05)
dur(record(icl>document(agt>thing,obj>thing)).@complete,eighty(icl>time).@def.@pl)
[/S]
[S:27]
; This is a matter of serious concern for the country as this growth rate is just
; matching the annual growth rate of population and 40 percent of the population
; in the country is still living below the poverty line.
aoj:01(matter(icl>abstract thing).@entry.@indef,this(icl>thing))
mod:01(matter(icl>abstract thing).@entry.@indef,concern(icl>feelings))
mod:01(concern(icl>feelings),serious(aoj>thing))
aoj:01(matter(icl>abstract thing).@entry,country(icl>region):07.@def)
rsn(:01.@entry,:02)
and:02(:04.@entry,:03)
aoj:03(match(icl>correspond(aoj>thing,obj>thing)).@entry.@present.@progress,
    rate(icl>magnitude):05)
man:03(match(icl>correspond(aoj>thing,obj>thing)).@entry.@present.@progress,
    just(icl>how))
mod:03(rate(icl>magnitude):05,this(mod<thing))
mod:03(rate(icl>magnitude):05,growth(icl>phenomenon):5A)
obj:03(match(icl>correspond(aoj>thing,obj>thing)).@entry.@present.@progress,
    rate(icl>magnitude):06)
mod:03(rate(icl>magnitude):06,growth(icl>phenomenon):6A)
mod:03(growth(icl>phenomenon):6A,population(icl>person):09)
mod:03(growth(icl>phenomenon):6A,annual(aoj>thing))
```

```
agt:04(live(agt>person).@entry.@present.@progress,percent(icl>ratio))
qua:04(percent(icl>ratio),40)
mod:04(percent(icl>ratio),population(icl>person):10.@def)
plc:04(population(icl>person):10.@def,country(icl>region):08.@def)
man:04(live(agt>person).@entry.@present.@progress,still(icl>how))
plc:04(live(agt>person).@entry.@present.@progress,below(icl>place))
bas:04(below(icl>place),line(icl>shape).@def)
mod:04(line(icl>shape).@def,poverty(icl>state))
[/S]
[S:28]
; The production of foodgrains in the country fell to 192.4 million tonnes in
; 1997-98 from 199.4 million tonnes in 1996-97.
mod:01(production(icl>action).@entry.@def,foodgrain(icl>cereal).@pl)
plc(:01,country(icl>region).@def)
obj(fall(obj>thing,gol>thing).@entry.@past,:01)
gol(fall(obj>thing,gol>thing).@entry.@past,:02)
qua:02(tonne(icl>metric ton):12.@entry.@pl,:03)
qua:03(million:11,192.4)
dur(:02,:04)
fmt:04(1997:08.@entry,1998)
src(fall(obj>thing,gol>thing).@entry.@past,:05)
qua:05(tonne(icl>metric ton):13.@entry.@pl,:06)
qua:06(million:10,199.4)
dur(:05,:07)
fmt:07(1996.@entry,1997:09)
[/S]
[S:29]
; The production of wheat dropped by 3.5 million tonnes and coarse cereals by 3
; million tonnes.
obj(drop(obj>thing).@entry.@past,production(icl>action).@def)
and(:03,:02)
mod:03(production(icl>action).@entry.@def,cereal(icl>food))
mod:02(production(icl>action).@entry.@def,wheat(icl>cereal))
mod:03(cereal(icl>food).@pl,coarse(icl>rough(aoj>thing)))
man(drop(obj>thing).@entry.@past,by(icl>how(obj>thing)))
obj(by(icl>how(obj>thing)),tonne(icl>metric ton).@pl)
qua:02(tonne(icl>metric ton).@pl,:04)
qua:04(million:07.@entry,3.5)
qua:03(tonne(icl>metric ton).@pl,:05)
qua:05(million:08.@entry,3)
[/S]
[S:30]
; The kharif 1998-99 crop was also no better.
aoj(good(aoj>thing).@entry.@not.@past,crop(icl>food).@def)
man(good(aoj>thing).@entry.@not.@past,more(icl>how))
```

```
man(good(aoj>thing).@entry.@not.@past,also(icl>how))
mod(crop(icl>food).@def,kharif(iof>crop))
dur(crop(icl>food).@def,:01)
fmt:01(1998.@entry,1999)
[/S]
[S:31]
; However, the 1998-99 Rabi crop is expected to be good with a 5.2 million tonnes
; higher foodgrain output expected from it.
obj(expect(agt>thing,obj>thing):01.@entry.@present.@complete.@contrast,
    crop(icl>food).@def.@topic)
mod(crop(icl>food).@def.@topic,Rabi(iof>crop))
dur(crop(icl>food).@def.@topic,:02)
fmt:02(1998.@entry,1999)
gol(expect(agt>thing,obj>thing):01.@entry.@present.@complete.@contrast,
    good(aoj>thing))
aoj(good(aoj>thing),crop(icl>food).@def.@topic)
aoj(have(aoj>thing,obj>thing),crop(icl>food).@def.@topic)
obj(have(aoj>thing,obj>thing),output(icl>product).@indef)
mod(output(icl>product),foodgrain(icl>cereal))
mod(output(icl>product),high(aoj>thing))
man(high(aoj>thing),more(icl>how))
man(high(aoj>thing),:04)
qua:04(tonne(icl>metric ton).@entry.@pl,:03)
qua:03(million.@entry,5.2)
obj(expect(agt>thing,obj>thing):05.@complete,output(icl>product).@topic)
frm(expect(agt>thing,obj>thing):05.@complete,crop(icl>food).@def.@topic)
[/S]
[S:32]
; The crop is estimated at 96.5 million tonnes.
obj(estimate(icl>calculate(aoj>thing,obj>thing)).@entry.@present.@complete,
    crop(icl>food).@def.@topic)
gol(estimate(icl>calculate(aoj>thing,obj>thing)).@entry.@present.@complete,
    tonne(icl>metric ton).@pl)
qua(tonne(icl>metric ton).@pl,:01)
qua:01(million.@entry,96.5)
[/S]
[S:33]
; Agricultural ministry has projected a 1998-99 ouput of foodgrains at more than
; 200 million tonnes.
agt(project(icl>calculate(agt>thing,obj>thing)).@entry.@present.@complete,
    Agricultural ministry(iof>organization))
obj(project(icl>calculate(agt>thing,obj>thing)).@entry.@present.@complete,
    output(icl>product).@indef)
mod(output(icl>product).@indef,foodgrain(icl>cereal).@pl)
dur(output(icl>product).@indef,:01)
```

```
fmt:01(1998.@entry,1999)
gol(project(icl>calculate(agt>thing,obj>thing)).@entry.@present.@complete,
 tonne(icl>metric ton).@pl)
qua(tonne(icl>metric ton).@pl,:02)
qua:02(million.@entry,:03)
bas:03(more(icl>how).@entry,200)
[/S]
[S:34]
; The union government had plans to cover about 2 million farmers in the
; country with a total credit of Rs. 168 crores in 1999 with'kisan cards'.
aoj(have(aoj>thing,obj>thing).@entry.@past,
    union government(icl>organization).@def)
obj(have(aoj>thing,obj>thing).@entry.@past,plan(icl>idea).@pl)
pur(plan(icl>idea).@pl,cover(agt>thing,obj>thing))
obj(cover(agt>thing,obj>thing),farmer(icl>occupation).@pl)
qua(farmer(icl>occupation).@pl,:01)
qua:01(million.@entry,2)
man(:01,about(icl>how))
plc(cover(agt>thing,obj>thing),country(icl>region).@def)
met(cover(agt>thing,obj>thing),credit(icl>money).@indef)
mod(credit(icl>money).@indef,total(mod<thing))
mod(credit(icl>money),rupee(icl>currency).@pl)
qua(rupee(icl>currency).@pl,:02)
qua:02(crore(iof>10 000 000).@entry.@pl,168)
dur(credit(icl>money),1999)
ins(cover(agt>thing,obj>thing),kisan card(icl>tool).@pl)
[/S]
[S:35]
; It has fixed interest rates on credit given to farmers through these cards at
; 13.26 per cent including tax on interest on the first Rs. 2 lakhs credit and,
; at 6.02 percent on amounts above Rs. 2.00 lakhs.
agt(fix(icl>decide(agt>thing,obj>thing)).@entry.@present.@complete,it)
obj(fix(icl>decide(agt>thing,obj>thing)).@entry.@present.@complete,:01)
mod:01(rate(icl>charge).@entry.@pl,interest(icl>profit):06)
aoj(on(aoj>thing,obj>thing):20,:01)
obj(on(aoj>thing,obj>thing):20,credit(icl>money):09)
obj(give(agt>thing,gol>thing,obj>thing).@complete,credit(icl>money):09.@topic)
gol(give(agt>thing,gol>thing,obj>thing).@complete,farmer(icl>occupation).@pl)
man(give(agt>thing,gol>thing,obj>thing).@complete,through(icl>how(obj>thing)))
obj(through(icl>how(obj>thing)),card(icl>tool).@pl)
mod(card(icl>tool).@pl,this.@pl)
gol(fix(icl>decide(agt>thing,obj>thing)).@entry.@present.@complete,
    percent(icl>ratio))
and(:05.@entry,:04)
qua:04(percent(icl>ratio).@entry,13.26)
```

```
qua:05(percent(icl>ratio).@entry,6.02)
aoj(on(aoj>thing,obj>thing):21,:05)
obj(on(aoj>thing,obj>thing):21,amount(icl>magnitude).@pl)
aoj(above(aoj>thing,obj>thing),amount(icl>magnitude).@pl)
obj(above(aoj>thing,obj>thing),rupee(icl>currency):10.@pl)
qua(rupee(icl>currency):10.@pl,:14)
qua:14(lakh(iof>100 000):15.@entry.@pl,2:16)
aoj(include(aoj>thing,obj>thing).@progress,:04)
obj(include(aoj>thing,obj>thing).@progress,tax(icl>charge))
aoj(on(aoj>thing,obj>thing):22,tax(icl>charge))
obj(on(aoj>thing,obj>thing):22,interest(icl>profit):07)
aoj(on(aoj>thing,obj>thing):23,interest(icl>profit):07)
obj(on(aoj>thing,obj>thing):23,credit(icl>money):08)
mod(credit(icl>money):08.@def,rupee(icl>currency):11.@pl)
qua(rupee(icl>currency):11.@pl,:03)
qua:03(lakh(iof>100 000):12.@entry.@pl,2:13)
mod(:03,1.@ordinal.@def)
[/S]
```

Bibliography

[1] UNL Center, UNDL Foundation. *The Universal Networking Language Specifications, Version 3*, 2001.

[2] Sergey Brin and Lawrence Page, *T*he anatomy of a large-scale hypertextual Web search engine. Computer Networks and ISDN Systems 30(1-7), 1998.

[3] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd, *T*he pagerank citation ranking: Bringing order to the web. Technical report, Stanford, Santa Barbara, CA 93106, January 1998. http://www-db.stanford.edu/ backrub/pageranksub.ps.

[4] Soumen Chakrabarti, Martin van den Berg and Byron Dom, *F*ocused crawling: a new approach to topic-specific Web resource discovery. In The proceedings of 8th International World Wide Web Conference, 1999.

[5] *M*IRTH – Chinese/English Search Engine: A Multilingual Information Retrieval Tool Hierarchy For World Wide Web "Virtual Corpus" and Training Resource in Computing and Linguistics and Literature. Thesis by Xioda Zhang, Leeds, UK, 1996.

[6] Erbach G., Neumann G., and Uszkoreit H. *M*ULINEX: Multilingual Indexing, Navigation and Editing Extensions for the World Wide Web. published in Hull, D. and Oard, D. eds. Cross-Language Text and Speech Retrieval - Papers from the 1997 AAAI Spring Symposium, AAAI Press, Stanford.

[7] Linguistica Computazionale, Volume XIV-XV, *M*ultilingual Information Management: Current Levels and Future Abilities, Insituti Editoriali e Poligrafici Internazionali, Pisa, Italy, 2001.

[8] Link to 2 language multilingual search engines, http://www.themillenniumsearch.com/

[9] McCallum, Andrew Kachites. *B*ow: A toolkit for statistical language modeling, text retrieval, classification and clustering. http://www.cs.cmu.edu/ mccallum/bow. 1996.