Using Semantic Information to Improve Case Retrieval in Case-Based Reasoning Systems

J. Akshay Iyer and Pushpak Bhattacharyya

Indian Institute of Technology Bombay
{akshay,pb}@cse.iitb.ac.in

Abstract. Conventional Case-Based Reasoning (CBR) systems rely on word knowledge to index and search cases from its memory. On being presented with a problem, the Case-Based Reasoning system tries to retrieve a relevant case based on the words that appear in the problem sentence without considering their respective senses. Drawbacks of such systems become more evident in cases where the input is in the form of a sentence in a natural language. Ignoring semantic information in this case may not result in retrieval of desired case or may result in retrieval of an undesired case. In this paper we present a method that tries to improve the precision of retrieval by also taking into account semantic information available to us about the words in the problem sentence. Towards this goal, Universal Networking Language (UNL) is made use of, which provides a semantic representation of natural language text to capture sentence structure. Lexical resource like WordNet is used for finding semantic similarity between two concepts. Using an existing commercial Case-Based Reasoning system as basis for comparison, we demonstrate that considering such semantic information helps in improving case retrieval.

1 Introduction

Case-Based Reasoning (CBR) Systems are one of the most widely used systems in the field of problem solving and planning. A number of such systems are developed and reported [5, 8, 11]. Typically, a number of cases are stored in memory and upon being presented with a problem, a set of relevant cases is retrieved and presented as a solution to the problem [7]. One of the fundamental issues in such systems concerns this retrieval process. Information from the input problem is extracted out and this information is used to index (or search) in the memory to locate the desired case. In systems where a problem is input in Natural Language form, the issue becomes more profound. Traditionally, a number of statistical methods are used for extracting information from the input problem and using it in turn for identifying cases that are relevant to the problem. However, since such methods do not employ any natural language understanding, they fail in situations when mere knowledge about words is not sufficient.

In this paper we propose a method by which we could use information, both semantic and syntactic, from natural language text to compare and retrieve relevant cases. The rest of the paper is organized as follows. Section 2 discusses the shortcomings of traditional methods and sets out the motivation for using sentence structure and semantic knowledge in case-searches. In order to capture sentence structure we propose to use Universal Networking Language (UNL) [4], whose introduction and generation process are given in section 3 and 4 respectively. In section 5 we present our algorithm to measure sentence similarity that simultaneously takes into account the structural similarity as well as the similarity of concepts involved. This is done using UNL and WordNet [6]. The results obtained from our reference Case-Based Reasoning System and the results obtained through our method are compared and described in section 6. We conclude the paper in section 7 with a note that using semantic information helps in improving the case retrieval in Case-Based Reasoning Systems.

2 Role of Semantic Knowledge in Case Searches

In order to understand and appreciate the role and importance of semantic knowledge and sentence structure in case-retrieval process, we need to understand the working of systems that do not use this information and rely only on the word knowledge. *CHEF* [5] is a CBR system developed at Yale University by Hammond. The input to the system is a set of goals to be satisfied by a single integrated case. The cases in this system describe recipes for various dishes. *CHEF* stores its cases in memory indexed by various features like dish-type, ingredients etc. The input to the system is the type of dish that is to be prepared and a list of desired ingredients which are used as keywords to search for relevant cases in its index. Since the system uses only these features as inputs, the system is limited to the jargon of culinary only. The system offers no flexibility in that the user is expected to follow a set representation for input. Any input that falls outside the representation will not be able to produce desired results. Also, a user is not allowed the freedom to annotate his input with any remarks or comments that may be useful while preparing a plan for the dish.

CONSULT [11], developed by *Tata Consultancy Services*, is a more generic Case-Based Reasoning System. Each case in *CONSULT* pertains to a single problem and contains questions that are posed to the user for an interactive diagnosis of the problem [14]. Based on the inputs given by the user, a relevant case is retrieved and output to the user. Here, every case contains a *Title* field that describes the problem whose solution is contained therein. The user enters a problem in natural language text and a case is retrieved from the memory to perform further diagnosis. The problem of searching for a case that contains a problem similar to the one input by the user hence gets reduced to finding cases whose *Title* is the most similar to the problem input by the user. A set of questions are posed to the user, the answers to which are compared to the ones listed out under the relevant cases that are retrieved. A question-answer pair typically behaves as an attribute value pair. The answers provided by the user to the questions posed are compared to these values and a match is found. Our efforts have been directed toward devising an approach that will make the initial case retrieval based on the similarity between a problem statement and the case *Titles* more fruitful.

Let us look into an example that illustrates this. We consider a case-based system that is modeled on the *CONSULT* system. Let us consider three cases, whose respective *Title* fields contain the following

– My computer in office is not running
– Cannot run MS Office on my computer
– My machine is not working

Though the first two sentences are talking about two different problems, it is difficult to know this difference until we consider the meanings of the words present. The two sentences share most of their words and hence would seem very similar to each other to a system that follows conventional methods like stemming, gramming [9], etc. It might present both the cases as being relevant to a single problem. On the other hand, the first and third cases, though seemingly different at the word level, are highly related to each other. A system ignoring the meaning of words will not be able to capture this similarity.

We also need to appreciate the importance of sentence structure in sentence similarity measure. In our method, sentence structure similarity measure denotes whether similar concepts are playing similar roles in the sentences being compared.

A sentence is represented using an interlingua called Universal Networking Language (UNL) [4]. Information in every sentence is captured at three levels: the concepts that are involved, the role they play in the sentence and attributes that describe their properties. The role of concepts in the sentence with respect to each other is represented using UNL relations and it is these relations that we consider to capture sentence structure. In the next section, we present a brief introduction to UNL and how it extracts and represents information out of a natural language text.

3 Universal Networking Language

Information contained in natural language text sentences needs to be captured effectively and exhaustively to be useful for understanding and processing. Universal Networking Language (UNL), proposed by United Nations University [4], represents natural language in the form of a semantic network where the concepts form the nodes of the graph and the relations among these concepts form the links among them (see Figure 1). UNL represents information sentence by sentence. This hypergraph is also represented as a set of directed binary relations, each between two concepts present in the sentence. Concepts are represented as character strings called Universal Words (UW).

The knowledge within a document is represented in three dimensions:

– **Universal Words (UW)**: describe concepts that are present in a document. Since concepts are universal and are expected to be independent of any one language, the Universal Words also are language independent. The Universal Words are accompanied by restrictions that describe the sense of the word, given by the UW, in a given context. For example, *drink* can describe either *putting liquids in the mouth*, *liquids that are put in the mouth*, *liquids with alcohol* or *absorb* etc. But a concept with a restriction like *drink(icl$>$liquor)* describes the sense of *drink* as

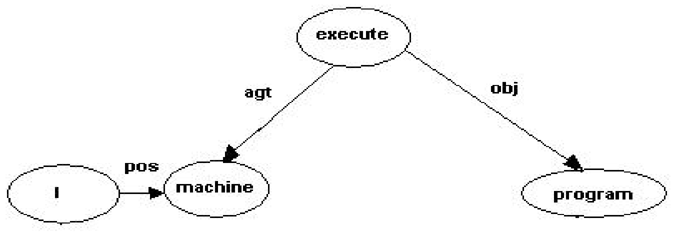

Fig. 1. UNL Graph for ***"MY MACHINE IS EXECUTING PROGRAMS"***

a noun standing for a type of *liquor*. The *icl* represents an *IS-A* kind of relationship between the concept and what follows *icl*. Therefore, each Universal Words represents a unique sense.

– **UNL Relations**: describe the relations between the concepts involved in the sentence and the roles (e.g. *subject* or *object* in case of nouns) that they play in conveying the meaning of a sentence. UNL uses a standard set of 41 relations to capture this knowledge. Each relation describes a kind of role that concepts can play towards the overall meaning of a sentence. Let us illustrate this with the help of an example- *My printer is not working*- the UNL for which is given below.

  agt(work(icl>function).@not.@present.@progress
  .@entry, printer(icl>machine))
  pos(printer(icl>machine), I(icl>person))

  The relations in UNL are binary defined as *rel(UW1, UW2)*. Here, not only the facts that *printer* is a kind of machine and *work* is a kind of action, but also, the relation between these two concepts, *printer* being an **agent** (*agt*) of the action of **not working**, are presented. Also, *I* is shown to be related to *printer* as its **possessor** (*pos*).

– **UW attributes**: capture and represent properties of concepts like tense of a verb, and speaker's perspective, attitude etc. In the above example, the use of *.@present* and *.@progress* with the UW for *work* describe the action of **working** as happening in the present time. Speaker's attitude like affirmation, contradiction, exclamation are also represented using these attributes.

  To illustrate this, consider the previous example. The presence of an attribute *.@not* indicates that the process of **working** is not happening. *.@entry* is a special attribute that indicates the main predicate of a sentence. This attribute is attached to the node from where generation of target natural language begins. In the following sections, we briefly describe the process of generation of UNL from a source natural language text, also known as *EnConversion.*

4 Generation of UNL from Source Natural Language

The process of conversion of a natural language text into its equivalent UNL representation is called *EnConversion* and the machine that performs this is called *EnCon-*

verter [12]. EnConverter (EnCo) is a language independent parser that performs morphological, syntactic and semantic analysis synchronously [3].

EnCo analyzes the source text sentence by sentence. It makes use of a knowledge rich lexicon of concepts and an exhaustive rule base for analysis. The EnConverter's function can be compared to that of a multi-headed Turing machine. The input sentence is converted into a node-list representation where a token (word or blank) forms a node. The EnCo works on this node-list through its windows (or heads), namely Analysis Windows (AW) and Condition Windows (CW). There are two Analysis Windows, but there can be any number of Condition Windows. EnCo checks the nodes under its two AWs, and the nodes that appear under its CWs. Based on the attributes of these nodes, it performs actions as described in the rule-base [1]. The various actions performed could be deletion, exchange, composition, forming a relation etc. Since there are only two Analysis Windows and at a time, operations are performed only on the nodes that appear under them, all UNL relations that are generated are binary relations. For example, consider a sentence, *Machine is executing*. The initial node-list that would be generated is shown.

/>>/Machine is executing/>>/

Here, >> and << indicate sentence head and sentence tail markers respectively. The EnCo picks up relevant entries and attributes for these words from a UW dictionary. In this case, the concept for **machine** will have attributes *N, INANI etc.*, whereas the attributes for **execute** will be *VRB, VOA-ACT etc.* A sample rule in rulebase is given below.

>(SHEAD){N::agt}{VRB:::}P10;

The above rule states that if there is a *noun* under the Left *AW* preceded by a sentence head indicator *SHEAD* (checked by a *CW*) and followed by a *verb* under the Right *AW*, then the *noun* is related to the *verb* as the **agent** of the action indicated by the relation *agt* in the rule.

In addition to forming relations between two nodes, EnConverter can add or delete attributes from a node. For example, in a sentence *Program will run*, the word *will* does not appear in the final UNL representation but EnCo adds an attribute *.@future* to the predicate of the sentence that is *run*. It is the UNL relations that we consider in our system when comparing two sentences for structural similarity.

5 Measuring Sentence Similarity

As mentioned previously, similarity between two sentences is measured on two counts: how similar are the concepts involved in the two sentences and how similar roles do the concepts play in the sentence? Since relations describe the roles that concepts play in the meaning of the sentence, similar structure sentences will have similar relations in their respective UNL representations. For example, *My machine is running* and *The printer is not working*, though different in meaning, have a very similar structure by virtue of the role *machine* and *printer* play (*i.e. agt*) in the meaning of their respective sentences. We now present the algorithm used in the CONSULT system followed by our algorithm to measure the similarity among sentences.

5.1. Sentence Similarity in CONSULT

Consult uses k-Nearest Neighbor (kNN) algorithm to determine the similarity between cases. The kNN computes a weighted Euclidean distance between two cases. Each case is represented as a set of attribute-value pairs. One such attribute of a case is the case *Title* and its value is a string that describes the problem to which the case is related. A case-similarity search involves matching of the case *Titles* too. We now describe briefly the string matching that is undertaken in CONSULT [11].

Let the input string be
Sytem hanged while doing a btch program.

Let the case Title of a case be
Software crashes when I run batch process.

It may be noted that the two strings share not a single word and that the words *System* and *batch* are misspelled in the input string. Sentence similarity in CONSULT proceeds in the following steps:

– **Stemming**: In this step, the input string is taken and all the words, that are derived or inflected, are reduced to their root forms. Hence, the input string now looks like

Sytem ***hang*** *while* ***do*** *a btch program.*

The words that were changed are indicated in boldface.

– **Synonym Rewriting and Auto Correction**: In this step, common spelling errors are rectified and variants of a word that mean the same (i.e. synonyms) are reduced to a standard form. In our example, the word *Sytem* is auto-corrected to *System* and *btch* to *batch*. Also, the words *System* and *do* are reduced to their standard forms of *Software* and *run* respectively. At this stage, our input string becomes
Software crash *while* ***run*** *a batch program*

– **Stripping of Noise Words**: Noise words (also called stop-words), that do not add anything significant to the overall meaning of the sentence, are excluded from the sentence before matching. In current example, the words "while" and "I" in the input string are dropped. Thus, our input sentence now looks like
Software crash run batch program

– **Gramming**: After the first three steps, it may be noted that the two strings seem very similar to each other. Each string is now broken down into an unordered set of strings of fixed length or grams. This sequence of grams is then used for comparison. Similarity is computed as a function of the cardinality of the intersection at the gram level.

Thus, as is evident, CONSULT does not even attempt to utilize word meanings or the role that they play in the given problem sentence. However, this approach is prone to failure in the example three sentences mentioned in Section 2.

5.2. Comparing Concept Similarity

Taking UNL representation for one *Case Title* at a time, we compare each of the concepts occurring in it's UNL representation with the concepts that appear in the UNL representation for the problem sentence. The similarity score
is computed using the method proposed by Resnik [10] where similarity of two concepts is determined by the information that they share indicated by the most specific concept that subsumes them both in a concept hierarchy. Resnik used WordNet [6] (see the appendix) for this hierarchy of concepts. For every concept, its likelihood of occurring in the document is calculated by counting the number of instances of itself and the concepts subsumed by it in the document. Therefore, the more general a concept, the more number of occurrences it will have. Probability (or likelihood) of occurrence of a concept is given as

$$P(c) = Nc / N \tag{1}$$

where *Nc* is the number of times a concept *C* occurs in the document and *N* is the total number of words in the document. Using the Information Content Theory [13], the *Information Values* associated with each concept *C* is negative log of the likelihood of occurrence of the concept.

$$IC(c) = -\ln(P(c)) \tag{2}$$

We too used WordNet to arrange concepts in a hierarchy and assign them *Information Content Values* in the manner proposed by Resnik. However, in Resnik's method, the sense of the concepts being matched is not known. Therefore, a similarity score is measured for all senses of the two concepts and the maximum among them is chosen. While this may work in most of the cases, it is not always very effective. Use of UNL Universal Words helps us restrict our attention to only one sense of a concept and therefore produces the most useful similarity score.

If there are N_1 and N_2 nodes (or words) in the two sentences S_1 and S_2 respectively, then the concept similarity measure is calculated as

$$\frac{\sum_{n_1 \in S_1} \sum_{n_2 \in S_2} SimScore(n_1, n_2)}{(N_1 * N_2)} \tag{3}$$

The sum of all the similarity scores over all pairs of concepts, that are matched for two sentences, is taken and averaged over the number of comparisons made. This is done to ensure the number of occurrence of a concept does not affect, influence or mislead the final similarity score.

5.3. Comparing Sentence Structure Similarity

Given all the cases in our Case-Based Reasoning system, we begin by obtaining the UNL representations for the *Titles* in each case. We consider the graph representations of UNL for our system. However, we represent a labeled edge in a UNL graph as sequence of two links; one from the initial *concept node* to a *relation node* that is labeled after the relation and another link from the latter to the destination *concept node* of the original link as shown in Figure 2. The structural similarity of two sentences is obtained by calculating the total number of subgraphs that their respective UNL graphs share. The method of calculation of common subgraphs is based on the method proposed by [2]. The smallest unit of subgraph in this context is either an edge with exactly one *relation node* or a *concept node*. The common subgraph calculation for a pair of graphs being compared is performed by calculating, for all pairs of nodes taken from the two graphs, the sum of number of common subgraphs rooted at the given nodes.

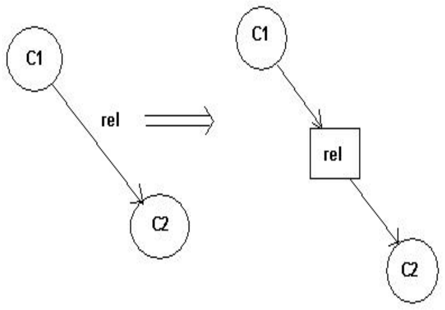

Fig. 2. Modification of Links in UNL Graphs

The recursive formula for common-subgraph calculation is given in [2].

$$C(n_1,n_2)=\prod\nolimits_{(x,y)\in sim(n_1,n_2)}(C(x,y)+2)-1 \tag{4}$$

where *sim(n₁,n₂)* is the set of common descendants of n_1 and n_2.

$$sim(n_1,n_2)=\left\{(x,y)\mid \begin{array}{c} x\in children(n_1)\\ y\in children(n_2)\\ label(x)\approx label(y)\end{array}\right\} \tag{5}$$

The condition *label(x) ≈ label(y)* in the above definition denotes similarity in the words' underlying concepts as computed by our concept matching algorithm.

Using the above definitions, the total number of common subgraphs between two graphs G_1 and G_2 is

$$C(G_1,G_2)=\sum\nolimits_{n_1\in G_1,n_2\in G_2,label(n_1)\approx label(n_2)}C(n_1,n_2) \tag{6}$$

The structural similarity between two sentences (or graphs) is now computed as

$$C(G_1,G_2)/N(G_1)*N(G_2) \tag{7}$$

where $N(G_1)$ and $N(G_2)$ are the total number of subgraphs in G_1 and G_2 respectively.

In the end, the concept and structural similarity scores obtained using the two methods are combined together to give us a cumulative similarity score.

6 Results

6.1. Experimental Setup

Our experiments based on the ideas and algorithms presented in section 5 were carried out on a case base of 120 cases. The cases dealt with problems faced by users in varied domains like *printer-related problems*, *DOS/Windows-related problems*, *Internet-related problems* and *DB2-related problems*. A set of 10 queries from each domain was taken and input to our system. The results obtained thus were compared with the ones that were obtained from CONSULT. We illustrate the performance our system with the help of a few examples.

Since our system uses both concept similarity and structural similarity, there are certain advantages and disadvantages with this approach. For example, let us consider the first query in the *DOS* category that was input to the system, *Windows creates a lot of temporary files*. This query did not have any corresponding relevant case in the Case-Base. The CONSULT and our system, however, returned the same case as the seemingly most relevant one, *I am not able to create a file in MS-DOS*. Obviously, the similarity here was established only by means of term matching. In addition to cases that matched due to presence of *files*, our system could also return results that talked about *programs*, *mails* etc. since they too are documents and are similar to *files*. This way, we could also include those cases that could have been relevant to this query but could have been ignored. As an illustration, an input query *My printer is not printing in Windows 98* also returns *Printer not writing to LPT1*. Note that our system could find a similarity between the concepts of *print* and *write* in the given context.

UNL Attributes also play an important role in concept matching. A system that relies only on terms and their grams will not be able to distinguish between sentences that are differentiated by the presence of a *not*. As another illustration, an input query *Unable to rename a file in DOS* returns *I am unable to create a file in DOS* due to its high structural similarity with respect to the relations that are shared by *unable*, *file* and *DOS* with the main verb of the each sentence, as well as concept similarities.

Sometimes, precision of case-retrieval may suffer due to concept similarities. This is illustrated in the following example.

– The input sentence was: My HP printer is not working

Table 1. Case Matching Results – Sentence 1

Sentence	Score (Our Method)	Score (CONSULT)
My mouse is not working in MS-DOS	0.53	–
My printer does not print the whole page	0.43	50
The printer is not writing to LPT1	0.40	35
I cannot read the print on my page when it is printed	–	30

As mentioned previously, this is an instance where concept similarity generates undesired results. A *mouse* is considered similar to *printer* since they both are devices and the two sentences' structural similarity is very high resulting in a close match.

– The input sentence was: I am unable to download pictures from the Internet

Table 2. Case Matching Results – Sentence 2

Sentence	Score (Our Method)	Score (CONSULT)
I am not able to download exe files from the Internet	0.38	60
I am not able to hear music from the Internet	0.31	30
I am unable to view the Internet connection icon on my desktop	0.35	30
Internet Explorer quits opening web pages while surfing	0.33	30

Note the high structural similarity of the first three cases with the query sentence. Both structural similarity and concept similarity give a high score to the first case in this example.

– The input sentence was: I am unable to surf the web

Table 3. Case Matching Results – Sentence 3

Sentence	Score (Our Method)	Score (CONSULT)
I am unable to browse the Internet	0.86	–
I am unable to print entire screen image	0.51	–
I am unable to install my Lexmark printer	0.42	9.09

The above example illustrates the power of our system. Given that *surf* is a common word used to describe the activity of *browsing* the Internet and *web* being synonymous with *Internet* or a computer network, we are able to identify the similarities between these concepts here and produce a match. CONSULT, however, was not able to come up with a match in this case since it could not find any common terms. The other two cases that are retrieved are similar to the original query by way of structural similarity with very little concept similarity.

– The input sentence was: I cannot create tables in DB2

Table 4. Case Matching Results – Sentence 4

Sentence	Score (Our Method)	Score (CONSULT)
We cannot create DB2 database for our project	0.62	50
I am not able to create index on table	0.56	50
I am not able to create table in database	0.53	50
I cannot use database	–	50

The first case here matches the query due to its strong similarity in its structure with respect to *create* and also the presence of *DB2*. The second and third too are quite similar to the query. CONSULT also provides *I cannot use database* as a retrieved case which our system does not pick.

We ran our system on a case-base of 120 cases. The precision for our system as well as that of CONSULT was calculated. We define *precision* as the number of relevant cases among those that are retrieved by the system. Overall, our system provided a higher precision than CONSULT in all domains.

7 Conclusions

In this paper, we presented a **novel method that uses semantic information to improve relevant case retrieval in Case-Based Reasoning systems**. As is observed, conventional methods of sentence similarity measures that do not take word meanings into account, fail miserably in scenarios where two different words in two sentences may be talking about the same concept. Such systems are also prone to failure in scenarios where presence of same words in two sentences convey different meanings altogether. This was highlighted by some example sentences given in section 2. These problems are duly and effectively handled by our system because it not only considers words in a sentence, but also, their correct senses. The capabilities of this system are taken another step forward by taking into account the similarity in sentence structure. In short, use of additional semantic information, obtained through UNL in our case, helps us to better evaluate similarity of sentences.

Appendix: WordNet

WordNet is a lexical resource that organizes words and concepts based on their similarity in meaning [6]. It divides the lexicon into five categories; noun, verbs, adjectives, adverbs and functional words, each of which follows a different semantic organization. It organizes concepts in terms of word meanings. A word meaning is represented by a set of all the word forms that can be used to express it called a *Synonym Set* or *Synset*. These synsets designate meaning to a word. The organization of WordNet describes a number of semantic relations between concepts represented as pointers between these *Synsets*. Some of the semantic relations found in WordNet are:

– **Synonymy**: defines a relation between concepts that mean the same. By synonymy, we mean that usage of one Synset can be replaced by the other without changing the meaning of the concept.
– **Antonymy**: is relation that is formed between word forms and not word meanings. This is because, an opposite of ***x*** is not always *not-**x***.
– **Hypernymy**: is a semantic relation that, along with *hyponymy*, defines a *IS-A* hierarchy between two concepts. This relation is transitive and asymmetrical and generates a hierarchical semantic structure. This is what is used by Resnik, and subsequently by us, to generate *Information Content Values* for concepts that occur in our Case-Base.

References

1. Shah C., Parikh J., Soni T.,"Conversion of English Langage Texts to Universal Networking Language", *B.E. Dissertation,Dharamsinh Desai Institute of Technology, Nadiad,* 2000.
2. Collins M., Duffy N., "Parsing with a Single Neuron: Convolution Kernels for Natural Language Problems", *Technical Report, University of California and Santa Cruz,* 2001.
3. UNL Centre/UNDL Foundation, "EnConverter Specification Version 3.3", April 2002.

4. Uchida H., Zhu M., Della S.T., "UNL: A gift for a millenium", *The United Nations University*, 2000.
5. Hammon K., "Explaining and Repairing plans", *Journal of Artificial Intelligence Research,* 1990.
6. Miller G. A.,Beckwith R.,Fellbaum C.,Gross D.,Miller K. J., "Introduction to WordNet: An Online Lexical Database", *Technical Report, Princeton University,* 1993.
7. Mitchell T., "Machine Learning", *McGraw Hill Companies Inc,* 1983.
8. Bhattacharyya P.,Choudhury S.R.,Gupta S.S., "Man power Planning with Case Based Reasoning: The Selector", *International Conference on Knowledge-based Computer Systems (KBCS),* December 1998.
9. Porter M., "An algorithm for suffix stripping", *Readings in Information Retrieval, San Fransisco, US,* 1997, 313-316.
10. Resnik P., " Semantic similarity in a taxonomy: An infomation-based measure and its application to problems of ambiguity in natural language", *Journal of Artificial Intelligence Research,* 1999.
11. Chakraborti S.,Balaraman V.,Vattam S.S., "Using CBR inexact search for intelligent data retrieval", *International Conference on Knowledge-based Computer Systems (KBCS),* 2000.
12. Pairkh J., Dave S., Bhattacharyya P., " Interlingua based english hindi machine translation and language divergence", *Journal of Machine Translation,* September 2002.
13. Shannon C., "A Mathematical Theory of Communication", *The Bell System Technical Journal,* July, October 1948, 379-423, 623-656.
14. Tata Research, Development and Design Centre, "CBDM – A Case-Based Development Methodology", *Technical Report*, 2000.