Abstract

The World Wide Web contains a huge amount of unclassified data and its continuous growth has made it a complex domain for information retrieval. Current web information retrieval (IR) systems (i.e., Search engines) very often overload the user with irrelevant search results. This has forced the user to perform a certain level of analysis on the results returned. Web IR systems are currently one of the most researched areas in the computer industry. So far there have been many attempts to incorporate Soft Computing techniques such as Fuzzy Logic, Neural Networks, Genetic Algorithms, etc. This paper focuses on how Fuzzy Logic can be introduced to IR systems. The current applications of Fuzzy techniques are analyzed and a concept called “Macro-clustering” is introduced as a solution for optimizing results of generalized search queries.

Keywords: Data Mining, Fuzzy Systems, Fuzzy Clustering, Information retrieval, Fuzzy Searching, Proxy Design Pattern

2. Problems in current web IR systems

There are many difficulties encountered during the retrieval of information on the web[1]. The key problems of current web IR systems are:

- Imprecision, and Uncertainty
- Lack of deduction capabilities
- Inability to take soft decisions
- No page ranking with respect to user queries
- No personalization
- Dynamism, Scale, and Heterogeneity

Imprecision, and Uncertainty: The aim of an IR system is to estimate the relevance of documents to users’ information needs, expressed by means of queries. This is a hard and complex task which most of the existing IR systems find difficult to handle due to the inherent imprecision and uncertainty related to user queries. Most of the existing IR systems offer a very simple modeling of retrieval, which privileges the efficiency at the expense of accuracy. Query processing in search engines, which are an important part of IR systems, is simple blind keyword matching. This does not take into account the context and relevance of queries with respect to documents

Lack of deduction capabilities: The current search engines have no deductive capability. For example, none of them gives a satisfactory response to a query like: How many computer science graduates were produced by South Asian universities in 2003?

Inability to take Soft Decisions: Current query processing techniques follow the principle of hard rejection while determining the relevance of a retrieved document with respect to a query. This is not correct
since relevance, itself, is a “gradual” property of the documents [2], not a crisp one.

Page Ranking with respect to user queries: Page ranks are important since human beings find it difficult to scan through the entire list of documents returned by the search engine in response to his/her query. Rather, one sifts through only the first few pages, say less than 20, to get the desired documents. Therefore, it is desirable, for convenience, to get the pages ranked with respect to “relevance” to user queries. However, there is no definite formula which truly reflects such relevance in top-ranked documents. The scheme for determining page ranks should incorporate 1) weights given to various parameters of the hit like location, proximity, and frequency; 2) weight given to reputation of a source, i.e., a link from yahoo.com should carry a much higher weight than a link from any other not so popular site; and 3) ranks relative to the user.

Personalization: It is necessary that IR systems tailor the retrieved document set as per users’ history or nature. Though some of the existing systems do so for a few limited problem domains, no definite general methodology is available.

Dynamism, Scale, and Heterogeneity: IR systems find difficulty in dealing with the problem of dynamism, scaling, and heterogeneity of web documents. Because of the time-varying nature of web data many of the documents returned by the search engines are outdated, irrelevant, and unavailable in the future, and, hence the user has to try his queries across different indexes several times before getting a satisfactory response. Regarding the scaling problem, Etzioni and Zamir[3] has studied the effect of data size on precision of the results obtained by the search engine. Current IR systems are not able to index all the documents present on the web and this leads to the problem of “low recall.” The heterogeneity nature of web documents demands a separate mining method for each type of data. (e.g. Image retrieval systems[4])

3. Fuzzification strategies in IR

The following section discusses various attempts made in order to incorporate a fuzzy logic to information retrieval techniques.

One of the earliest fuzzy based IR system, RUBRIC, was developed by R.Tong, V. Askman and J. Cunningham[5]. It was capable of retrieving based on the relevance of the document to the semantics of the users query. RUBRIC used rules to link words to concepts, which are also connected to semantically related concepts by rules. Each rule has a “relevance value” to indicate the strength of the association between its antecedent/words and consequent concepts. For example, if both “killing” and “politician” occur in a document, it suggests the document is somewhat related to assassination. This can be expressed as a rule with relevance value 0.5. During the information retrieval RUBRIC views the user’s query as a goal. Through goal back driven chaining, it determines the degree the document is relevant to the query. Fig 1. shows an example inference tree for finding documents related to the concept terrorism. The leaves that are found in a specific document are assigned a relevance value of 1. Otherwise, the term is given a relevance value of 0. These relevance values are propagated upwards in the inference tree using rules. Several rules have “auxiliary antecedents”, whose appearance in the document modifies (usually strengthens) the relevance value of the rule consequent. For example, a violent action is quite (0.8 degree) relevant to a terrorist event. However a violent action occurring together with assassination is completely (1.0) relevant to a terrorist event, as shown in the figure.

![Fig. 1](image-url)

The relevance of a inferred concept is calculated by RUBRIC based on the following equation:

\[ v[\text{consequent}]=a \times v[\text{primary}] + (b-a) \times v[\text{auxiliary}] \]

where \(a\) and \(b\) are relevance values associated with the primary antecedent and the auxiliary antecedent respectively, \(v[\text{consequent}]\) denotes the relevance of the current document to the concept \(c\). Relevance values are concept inferred from multiple rules and combined using fuzzy disjunction.

Using a measure of recall (i.e., the ration of number of relevant documents in the database) and a
measure of precision (i.e., the ratio of the number of relevant documents retrieved to the total number of documents retrieved). RUBRIC’s performance in retrieving information has been shown to be superior to a comparable approach using nonfuzzy rules. Even though RUBRIC’s approach is effective for retrieving documents related to a small set of predefined concepts, it is difficult to scale up the approach for general information retrieval systems. An alternative approach is to automatically generate a relevance measure between keywords by analyzing all documents in the database of an information retrieval system. S. Miyamato proposed an approach for generating fuzzy relevance measure (called fuzzy pseudothesaurus) based on the assumption that if two terms occur together frequently in documents, then they are relevant[6]. These IR systems have become models for web based IR systems.

Retrieving desired information from the Web is a tiresome process that every web user goes through every day. The main reason is the poor classification of the Web information. Different search engines use different techniques for this purpose and as a result their results significantly different. (e.g. the popular search engine “Google” uses link analysis to obtain its result). Thus the web user has to use several search engines in order to fulfill a particular information need.

There are plenty of Web search engines which utilize special robots in search for new Web pages, and when a page is found it is put in the ‘right’ classification category depending on the classification method the Web search engine is using. In a technique called “Metadata classification” embeds the parameters required for classification in the web object itself. This means that the task of classification is partially transferred to those who create and maintain Web elements. As an advanced solution for Web classification M. Marchiori [7] proposed a fuzzification method. He says that existing Web metadata sets do have attributes assigned to objects, but they either have them or do not have them. Instead, he argues that attributes should be fuzzified, i.e. each attribute should be associated with a “fuzzy measure of its relevance for the Web object, namely a number ranging from 0 to 1". This means that if an attribute is assigned value 0 it is not pertinent to the correspondent Web object. If the value is 0.4 relevance of the attribute to the Web object is 40%. Since classification by itself is an approximation, better of worse, fuzzification method allows flexibility within a predefined classification system providing more detailed ranking and allowing the basic set of concepts to be relatively small.

Similarly, Yager has described a framework for formulating linguistic and hierarchical queries[8]. It describes an IR language which enables users to specify the interrelationships between desired attributes of documents sought using linguistic quantifiers. Examples of linguistic quantifiers include “most,” “at least,” “about half.” Let \( Q \) be a linguistic expression corresponding to a quantifier such as "most" then it is represented as a fuzzy subset \( Q \) over \( I = [0,1] \) in which, for any proportion \( r \), belonging to \( I, Q(r) \) indicates the degree to which \( r \) satisfies the concept indicated by the quantifier \( Q \). Koczky and Gedeon [9] deal with the problem of automatic indexing and retrieval of documents where it cannot be guaranteed that the user queries include the actual words that occur in the documents that should be retrieved. Fuzzy tolerance and similarity relations are presented, and the notion of “hierarchical co-occurrence" is defined that allows the introduction of two or more hierarchical categories of words in the documents.

The usage of fuzzy logic on web IR has been demonstrated by Molinari and G. Pasi [10], when they developed a principled approach for assigning weights to different components, which are specified by tags in an HTML document. The rationale is that a word in the title carries much more weight than the same word appearing in other portions of the document. Therefore, it is possible to sort tags based on their degree of importance. For instance a sorted list maybe Title, Header1, Header2, emphasis, delimiters, etc. Based on the order in the list, fuzzy weight can be calculated for each tag.

\[
W_i = \frac{(n - i + 1)}{\sum_{i=1}^{n} i}
\]

where the \( n \) is the total number of tags in the sorted list. The total relevance measure of a document, denoted \( F \), to a query \( q \) is a weighted sum of relevance measure for each tag.

\[
F(q) = \sum_{i=1}^{n} W_{t_i} \times F_{v_{t_i}}(q)
\]

where \( F_{v_{t_i}}(q) \) denotes the degree that the content of tag \( t_i \) is relevant to the query \( q \). The aggregation is just the summation of the \( F_{v_{t_i}} \times W_i \) for all the \( i \) values.

4. Back-propagation to utilize relevance

So far we have seen how to calculate relevance of a Web object to the search query providing the information that is contained within the Web object. Since Web is a dynamic media inter-wound with hyperlinks it is a common fact that one Web object points to some other Web object or even to more of them. This brings us to the problem of calculating relevance of an object that is pointed to by some other
Suppose a certain Web object O has the associated metadatum A: v, indicating that the attribute A has fuzzy value v. If there is another Web object O’ with an hyperlink to O, then we can “back-propagate” this metadata information from O to O’. The intuition is that the information contained in O (classified as A: v) is also reachable from O’, since we can just activate the hyperlink.

However, the situation is not like being already in O, since in order to reach the information we have to activate the hyperlink and then wait for O to be fetched.

So, the relevance of O’ with respect to the attribute A is not the same as O’(v), but is in a sense faded, since the information in O is only potentially reachable from O’, but not directly contained therein. The solution to this problem is to fade the value v of the attribute multiplying it by a “fading factor” f (with 0 < f < 1). So, in the above example O’ could be classified as A: v· f. The same reasoning is then applied recursively. So, if we have another Web object O” with an hyperlink to O’, we can back-propagate the obtained metadatum A: v· f exactly in the same way, obtaining that O” has the corresponding metadatum A: v· f· f.

Experiments on a randomly chosen region of Web objects showed that the usage of back-propagation method can significantly improve effectiveness of the classification. They also showed that the critical mass of Web metadata classification usefulness is achieved when at least 16% of the Web use metadata classification, in contrast with 50% without incorporation of the back-propagation method. Furthermore, in order to achieve excellence level metadata need 53% of the Web to be classified, in contrast with 80% without described method.

Most of all, the method of back-propagation, which presuppose the fuzzification method, acts on top of any classification, and does not require any form of semantic analysis. Therefore, it is completely language independent which is very important when the number of non-English Web pages is constantly increasing.

5. Clustering of search results

Clustering techniques, in general, are used when there is no class of data to be predicted but rather when the instances of data are to be divided into natural groups. These clusters presumably reflect some mechanism at work in the domain from which instances are drawn, a mechanism that causes some instance to bear a stronger resemblance to one another than they do to remaining instances [16]. In the context of web information retrieval, clustering is used for automatically discovering groups of similar documents in a set of documents and a group of documents formed in the process is called a ‘cluster’.

Clustering algorithms such as K-means, Buckshot, Fractionation, Suffix Tree Clustering are being used in existing web IR systems. Many of these document clustering algorithms rely on off-line clustering of the entire document collection, but the Web search engines’ collections are too large and fluid to allow off-line clustering. Therefore clustering has to be applied to the much smaller set of documents returned in response to a query. Because the search engines service millions of queries per day, free of charge, the CPU cycles and memory dedicated to each individual query are severely curtailed. Clustering usually has to be performed on a separate machine, which receives search engine results as input, creates clusters and presents them to the user.

Clustering is important in several factors of information retrieval. In traditional information retrieval, one important means of speedup is to cluster data and to represent only a representative of each cluster in the database. [11]. When the source of information is the Internet, clustering the results allows more useful information to be presented on the first page of the results, allowing the user to determine which cluster is relevant.

The following search results illustrates the use of clustering in web IR. Here the keyword ‘salsa’ has been searched on the search engine WebCrawler[3].

Search results for the query: “Salsa”
Documents: 246, Cluster: 15
Clustering was introduced to the web as a method of limiting the number of documents that a user is shown. An early experiment in Web document clustering has shown that allowing relevant documents to appear in multiple clusters is advantageous [12]. Fuzzy clustering [13] is a well known generalization of clustering where each element can have non-zero membership in multiple clusters. Cluster exemplars are then computed taking into consideration the relative membership of each member of the cluster. Given the complexity of the results of most internet searches, a fuzzy clustering is likely to better represent the data than a crisp clustering. In addition, the ability to represent and use the degree of membership in the cluster when determining cluster exemplars for display and relevance ranking will help to mediate the effect of cluster outliers that could prevent the user from seeing documents in a cluster that would otherwise be relevant. Google, which seems to be the most effective search engine to date, currently supports simple hostname-based clustering.

Etzioni[3] has listed the key requirements of web document clustering as measure of relevance, browsable summaries, ability to handle overlapping data, snippet tolerance, speed and incremental characteristics. In [14], fuzzy c-medoids (FCNdd) and fuzzy c Trimmed medoids (FCTMdd) are used for clustering of web documents and snippets (outliers). In [15], a fuzzy clustering technique for web log data mining is de-scribed. Here, an algorithm called competitive agglomeration of relational data (CARD) for clustering user sessions is described, which considers the structure of the site and the URLs for computing the similarity between two user sessions. This approach requires the definition and computation of dissimilarity/similarity between all session pairs, forming a similarity or fuzzy relation matrix, prior to clustering. Since the data in a web session involves access method (GET/ POST), URL, transmission protocol (HTTP/FTP), etc., which are all nonnumeric, correlation between two user sessions and, hence, their clustering, is best handled using fuzzy set approach.

Although clustering improves the readability of search results, for general queries only a few large clusters are returned.

6. Macro-clustering

Web IR systems perform queries by considering web documents on an individual basis due to the architecture of the web. However, information is introduced to the web as websites having multiple related documents. This grouping formed by the creator of a web site is usually not taken into consideration by any IR retrieval technique. An objective of this paper is to demonstrate the use of a clustering method performed at the web site level.

As mentioned in the previous section existing clustering methods can improve the retrieval process by grouping results in a more meaningful format. On the other hand, for more generalized queries where the number of results in a cluster increases it is efficient to use a second level of clustering – known which the author names as “Macro-clustering”. This method takes in to consideration the possible classification of documents at a website level as shown in Fig. 3. A query submitted to a search engine can be of 2 forms.

A General Query is submitted by the user expecting a body of information. The number of occurrences of such information on the web is high and as a result a greater number of results are retrieved. On the other hand, Specific queries are submitted with the need for a specific piece of information. They produce in a fewer number of search results.

<table>
<thead>
<tr>
<th>Cluster no.</th>
<th>Size</th>
<th>Shared phrases and sample document titles</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>8</td>
<td>Puerto Rico; Latin Music</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1. Salsa Music in Austin</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2. LatinGate Home Page</td>
</tr>
<tr>
<td>2</td>
<td>20</td>
<td>Follow Ups post; York Salsa Dancers</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1. Origin and Development of Salsa?</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2. Re: New York Salsa Dancers are the best because...</td>
</tr>
<tr>
<td>3</td>
<td>40</td>
<td>music; entertainment; latin; artists</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1. Latin Midi Files Exchange</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2. Salsa Music On The Web. con Sabor!</td>
</tr>
<tr>
<td>4</td>
<td>79</td>
<td>hot; food; chiles; sauces; condiments; companies</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1. Religious Experience Salsa</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2. Arizona Southwestern Cuisine and Gifts</td>
</tr>
<tr>
<td>5</td>
<td>41</td>
<td>pepper; onion; tomatoes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1. Salsa Mixes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2. Salsa Q &amp; A</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Table 1
Clusters formed in the process of a general query usually contains fewer clusters and a large amount of documents within a cluster. Such clusters tend to be less useful from the perspective of the user. For such clusters, by applying “Macro-clustering” we reduce the number of results in a number of 2nd level clusters (Macro-clusters) that are more manageable to the user.

The number of results retrieved by a query is proportional to the specificity of the query string. This can also be given as,

\[ n_r \propto \frac{1}{n_k} \]

where \( n_r \) is the number of results and \( n_k \) is the number of key words in the query term. The most popular search engine to date, Google, accepts an upper bound of 10 key words.

For a more generalized query (e.g. \( n_k < 4 \)) the impact of clustering is reduced and the number of results within a cluster increases. Also, for such queries (e.g. query string such as “Artificial Intelligence” or “Neural Networks”) it is difficult point out a specific document in a website that is related to the subject. For such situations, this second level of clustering will make the results of the query more useful. However, Macro-clustering may not be required for more specific queries (e.g. \( n_k > 3 \)) as the results retrieved are more likely to be distinct documents in a website. As a result the decision to use this technique could be determined by the IR system by analyzing the query string.

Consider the following query strings.

- \( q_1 \) - “Fuzzy Logic”
- \( q_2 \) - “Applications of extension principle in fuzzy logic”

The above queries can be classified as generalized and specific query strings, respectively. By implementing a Macro-clustering the results retrieved for \( q_1 \) should be mainly website based and, similarly document based for \( q_2 \).

An experiment was carried out to distinguish the relationship between the number of query terms and the results retrieved. It was performed by executing search phrases \( q_1 \) and \( q_2 \), on several top ranking search engines. The results obtained are given in the following table.

<table>
<thead>
<tr>
<th>Search Engine</th>
<th>No. of hits for ( q_1 )</th>
<th>No. of hits for ( q_2 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Google</td>
<td>599,000</td>
<td>11,500</td>
</tr>
<tr>
<td>2. Alltheweb</td>
<td>294,095</td>
<td>649</td>
</tr>
<tr>
<td>3. Altavista</td>
<td>123,121</td>
<td>1,984</td>
</tr>
<tr>
<td>4. Wisenut</td>
<td>132,060</td>
<td>2,070</td>
</tr>
</tbody>
</table>

Table 2

These results are agreeable with the concept discussed above. Considering the given query phrases, a general query produces results in the order of \( 10^6 \) where as a specific query produces results in the order of \( 10^3 \).

Relevance ranking can be incorporate into this method by using a fuzzification technique described in section 2 of this paper. By using a fuzzification technique we can assign a fuzzy value for the closeness of a document to a particular query phrase and subsequently using a clustering algorithm, document clusters can be formulated. The fuzzy relevance values of the documents can used to index documents within a cluster.

With Macro-clustering we can perform a sorting of results within the cluster by calculating the aggregate value of fuzzy terms for each website. This can found for the \( j^{th} \) website referenced in the cluster using,

\[ F_{j_{aggregate}} = \sum_{i=1}^{n} F_{ij} \]

where \( F_{j_{aggregate}} \) is the total of fuzzy values for a website and \( F_{ij} \) is the fuzzy value of the \( i^{th} \) document in the collection of \( n \) documents in the \( j^{th} \) website. In each cluster the website having the most number of relevant documents will appear at the beginning of the list.

The results of Macro-clustering will be a list of websites and upon selecting a website in the list, the user will be shown all of documents that match the query that belongs to the website. Here, the user will also be given the chance of visiting the home page of a website from where he could browse using the navigation system available in the site itself.
A method of classifying a website is required in implementing this method. The best method to achieve this is by performing link analysis within the search domain. Here, we can extract the domain information to identify documents coming from the same website. This information can be used as the basis for Macro-clustering. Here the user classification at the website level is deduced by the system by analyzing the URLs. An assumption is made here, that all web documents in a website have similar URL patterns. Another method that could be used for identifying a document’s parent website is to use identification that can be specified by the web document creator as meta data. For example, we can use the following notation to label documents in a website.

\[ \text{<META NAME= "website" CONTENT="www.fuzzylogic.com"}> \]

However, this method leaves a certain level of responsibility on the hands of the web document creator, and therefore, not very effective.

In the implementation Macro-clustering search engines would have to first determine the type of search performed based on the number of keywords. The following pseudocode shows a high level algorithm that can be used to process a query.

```
Accept user query phrase
Determine fuzzy values of documents in search domain
Determine query category
If ‘specific-query’
  Cluster documents based on Concepts
Else If ‘general-query’
  Cluster documents based on Concepts
  For each Cluster
    Perform link analysis on documents
    Form Macro-clusters based on domain
    Calculate Fuzzy totals within Macro-cluster
    Order Macro-clusters based on Fuzzy totals
  End For
End If
Display Results
```

In the following example Macro clustering is applied to a very small sub set of the world wide web(3 web sites). Here, we assume that a search performed using the term “fuzzy logic” retrieves the results given in table 3.

<table>
<thead>
<tr>
<th>Index</th>
<th>Document URL</th>
<th>Rel.</th>
<th>FRF</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td><a href="http://www.cs.berkeley.edu/fz/internet.htm">www.cs.berkeley.edu/fz/internet.htm</a></td>
<td>15</td>
<td>1.0</td>
</tr>
<tr>
<td>2</td>
<td><a href="http://www.pdn.ac.lk/fl/index.htm">www.pdn.ac.lk/fl/index.htm</a></td>
<td>14</td>
<td>0.9</td>
</tr>
<tr>
<td>3</td>
<td><a href="http://www.pdn.ac.lk/fl/fuz.htm">www.pdn.ac.lk/fl/fuz.htm</a></td>
<td>13</td>
<td>0.8</td>
</tr>
<tr>
<td>4</td>
<td><a href="http://www.cs.berkeley.edu/fz/papers.htm">www.cs.berkeley.edu/fz/papers.htm</a></td>
<td>12</td>
<td>0.8</td>
</tr>
<tr>
<td>5</td>
<td><a href="http://www.pdn.ac.lk/fl/defuz.htm">www.pdn.ac.lk/fl/defuz.htm</a></td>
<td>11</td>
<td>0.7</td>
</tr>
<tr>
<td>6</td>
<td><a href="http://www.fuzzylogic.com/index.htm">www.fuzzylogic.com/index.htm</a></td>
<td>10</td>
<td>0.6</td>
</tr>
<tr>
<td>7</td>
<td><a href="http://www.fuzzylogic.com/tech/index.htm">www.fuzzylogic.com/tech/index.htm</a></td>
<td>9</td>
<td>0.6</td>
</tr>
<tr>
<td>8</td>
<td><a href="http://www.cs.berkeley.edu/fz/zadeh.htm">www.cs.berkeley.edu/fz/zadeh.htm</a></td>
<td>8</td>
<td>0.5</td>
</tr>
<tr>
<td>9</td>
<td><a href="http://www.fuzzylogic.com/tech/papers.htm">www.fuzzylogic.com/tech/papers.htm</a></td>
<td>7</td>
<td>0.4</td>
</tr>
<tr>
<td>10</td>
<td><a href="http://www.pdn.ac.lk/fl/research.htm">www.pdn.ac.lk/fl/research.htm</a></td>
<td>6</td>
<td>0.4</td>
</tr>
<tr>
<td>11</td>
<td><a href="http://www.fuzzylogic.com/tech/functions.htm">www.fuzzylogic.com/tech/functions.htm</a></td>
<td>5</td>
<td>0.3</td>
</tr>
<tr>
<td>12</td>
<td><a href="http://www.pdn.ac.lk/fl/new.htm">www.pdn.ac.lk/fl/new.htm</a></td>
<td>4</td>
<td>0.2</td>
</tr>
<tr>
<td>13</td>
<td><a href="http://www.cs.berkeley.edu/basic.htm">www.cs.berkeley.edu/basic.htm</a></td>
<td>3</td>
<td>0.2</td>
</tr>
<tr>
<td>14</td>
<td><a href="http://www.fuzzylogic.com/extension.htm">www.fuzzylogic.com/extension.htm</a></td>
<td>2</td>
<td>0.1</td>
</tr>
<tr>
<td>15</td>
<td><a href="http://www.fuzzylogic.com/neurofuzzy.htm">www.fuzzylogic.com/neurofuzzy.htm</a></td>
<td>1</td>
<td>0.0</td>
</tr>
</tbody>
</table>

Rel. – Relevance value   FRF – Fuzzy Relevance Factor

Table 3

For the sake of simplicity, we limit the area of the search to 3 pre-specified websites. The F.R. Factor is determined in relation to the order of the search results. (i.e. most relevant result appears first). Then we fuzzify the set of results with reference to the first result, which we consider to have complete membership in the fuzzy set \( R = \{ \text{documents that match the search query} \} \). The related function for the fuzzy set is given in fig.4.

![Fig 4 - Membership function of Fuzzy Set R](image)

The results in the Table 3 are aggregated based on link names(web sites). The F.R.F. total for each domain is
also calculated, and the resulting table (Table -4) is sorted according to the F.R.F. total. Subsequently we obtain a list of websites related to the search term in order of relevance.

<table>
<thead>
<tr>
<th>MC Index</th>
<th>Domain(Website)</th>
<th>FRF Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td><a href="http://www.pdn.ac.lk/fl/">www.pdn.ac.lk/fl/</a></td>
<td>3.18</td>
</tr>
<tr>
<td>2</td>
<td><a href="http://www.cs.berkeley.edu/">www.cs.berkeley.edu/</a></td>
<td>2.53</td>
</tr>
<tr>
<td>3</td>
<td><a href="http://www.fuzzylogic.com/">www.fuzzylogic.com/</a></td>
<td>2.24</td>
</tr>
</tbody>
</table>

Table 3
The development of XML technologies for Internet applications has received attention in the industry as it’s use give more meaning to the structure of a web document. XML is becoming a new standard of data representation and exchange, and more documents are expected to be available on the web. In XML, the structures and possibly the meaning of data are explicitly indicated by element tags. Therefore, incorporating XML will no doubt increase the accuracy of fuzzy content based information retrieval.

The concept ‘Macro-clustering’ was presented as a solution to improve search queries with a few search terms. The need for this arises because a majority of queries performed on search engines belong to this category.

7. Conclusion

The path to an ‘perfect’ web IR systems is still far ahead and the model for such as system cannot be fixed as the World Wide Web changes at a rapid rate. Researchers have identified the use of soft computing methods( Fuzzy Logic, Neural Networks, Genetic Algorithms) as tools in making this goal a reality. This paper discussed the part played by Fuzzy Logic. Finally, the concept of “Macro-clustering” was introduced as a solution for optimizing results of generalized search queries. This concept contains many aspects on which further research can be carried out.

References