Conceptemy : An issue in XML information retrieval

Vamsi Vutukuru[1]
Department of Computer Sciences
The University of Texas at Austin
Taylor Hall 2.124, Austin, TX – 78712
01 512 789 2571
vamsikv@cs.utexas.edu

Krupakar Pasupuleti
Department of Computer Sciences
The University of Texas at Austin
Taylor Hall 2.124, Austin, TX – 78712
01 512 478 7365
krupakar@cs.utexas.edu

Anuj Khare
Department of Electrical and Computer Engineering.
The University of Texas at Austin
ACES 2.106, Austin, TX – 78712
01 512 789 2571
anujkhare@hotmail.com

Amit Garg
Department of Computer Sciences
The University of Texas at Austin
Taylor Hall 2.124, Austin, TX – 78712
01 512 560 6970
amitji@cs.utexas.edu

Abstract

We identify the problem of *Conceptemy* that is caused by the artificial imposition of a hierarchal semantics over large XML datasets. Conceptemy is an original term coined to reflect the ambiguity caused by use of the same tag with the same semantic content in different hierarchical contexts. A mechanism has been designed to leverage the ambiguity caused by the problem to help the user refine searches over XML datasets. We take the search results of an XML search engine, cluster them and present them to the user in semantically distinct groups. As XML becomes increasingly widespread, we expect that *Conceptemy* will become increasingly important both in academic research and industry products.

Keywords

XML, Context-based Information Retrieval, Clustering, Conceptemy,

1. Introduction

The enormous quantities of information online would not constitute the Internet if there were no search engines to let users navigate to their topic of interest. In an XML world, information retrieval will continue to be the dominant application. Furthermore one would expect that with the additional semantic structuring provided by XML, the quality of searching could be improved. Several papers [1][2] address this problem. Companies like [4][5] specialize in software engines that perform this function.

In designing and implementing our own XML search engine, we identified a novel problem caused by attempting to ascribe a hierarchal semantics to a wide variety of concepts. Upon further investigation we successfully designed a clustering mechanism that takes advantage of the inherent ambiguity to help the user refine her search. Unfortunately we do not have access to datasets of sufficient complexity where

[1] In reverse alphabetical order

this problem is more likely to manifest itself; hence we designed our own dataset and performed experiments that illustrate the power of our approach.

2. Conceptemy

Conceptemy is an original term coined to reflect the ambiguity caused by use of the same tag (hence no synonymy[2]) with the same semantic content (thus no polysemy[3]) in different hierarchical contexts. Consider the example of two documents (Figure 1) , both containing the tag *name*. In both the documents *name* is the same word and has the same meaning, but refers to distinct entities. In one document it's the name of a firm, in second it's the name of an author. Thus the hierarchical path in which the tag occurs determines the context of the tag (see Figure 2).

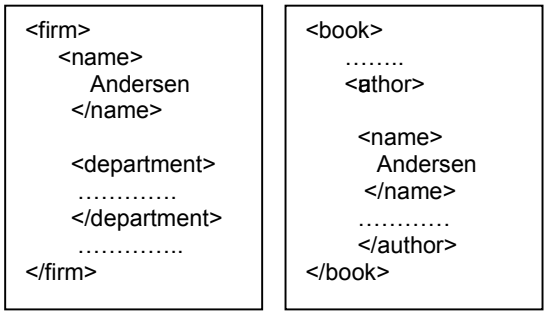

Figure 1

If the user performs a query with the tag 'name' and keyword 'andersen', then it would be more helpful if we can present the results in groups of semantically related documents – here those documents which refer to firms and those which refer to books.

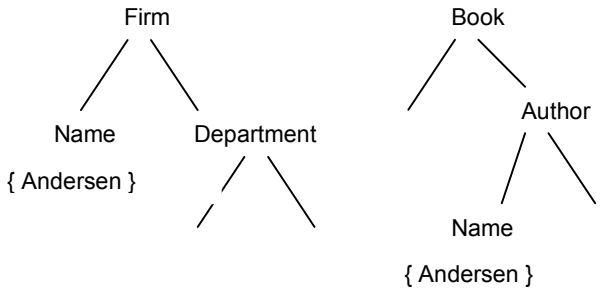

Figure 2

2.1 Concept Clustering

We describe a technique that addresses the problem of conceptemy and the lack of standardization in XML DTDs . These will also permit us to mine link collections for sets representing the same virtual community, as discussed previously.

Clustering of hierarchies. Given a tag in an XML document, its ancestors define its context . Together these define a concept hierarchy ie., an ordered list of ancestor tags represent the concept hierarchy of the tag, Hierarchy clusters are obtained by clustering these ordered lists. Such an approach serves well, for example, to separate different concepts yielded by a particular search online - thereby alleviating to some extent the problem of conceptemy. We can characterize the expected output of the method to return relatively few clusters (since they are all the result of one specific search) containing several documents.

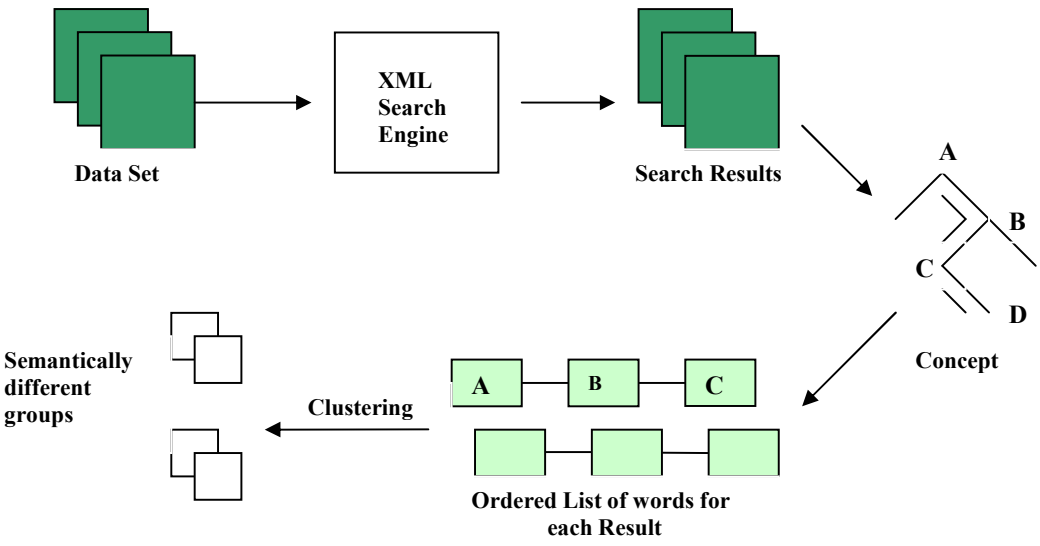

Figure 3

[2] Synonymy is the property of two or more distinct words having the same meaning
[3] Polysemy is the property of same word having multiple meanings

Figure 3 presents a graphical view of the concept clustering process. The 'concept clustering' module takes the search results as input. From each document, the hierarchical path in which the key tag occurs is extracted (this is done by our XML search engine) and stored in the form of an ordered list. Thus the problem of clustering search results has been reduced to clustering ordered lists. The main problem in clustering ordered lists is in defining a similarity measure between ordered lists. We have come up with a similarity measure between such ordered lists and perform clustering using graph partitioning techniques using this similarity measure.

2.1.1 Similarity Measure

Similarity among ordered lists can be viewed along the same lines as similarity in strings. Popular string similarity measures like the Levenshtein [6] distance, work well with strings but do not easily map to ordered lists. This is because they does not take into account the hierarchy or the order present in the lists. We define a similarity measure as:

$$Similarity(x,y) = \frac{1}{l_x + l_y}\left(\frac{w_{seq}}{r_{seq}}A_{xy} + \frac{w_{str}}{r_{str}}B_{xy} + \frac{w_{words}}{r_{words}}C_{xy}\right)$$

- A_{xy} = Length of Largest Common Subsequence(Seq)
- B_{xy} = Length of Largest Common substring (Str)
- C_{xy} = Number of Common words (Words)
- r_i = Average distance from the start node where the match occurs
- w_i = Weights attached to different parameters

r indicates the position in the hierarchy where match occurs so that sequences where a match occurs closer to the start node are semantically more related to each other.

The optimal values of weights would be determined by experiments. It is expected that $w_{Seq} > w_{Str} > w_{Words}$ since a higher weight age would be assigned to subsequence match when compared to either string or number of common words.

2.1.2 Clustering

We computed the similarity measures between each pair of lists. Then we used a standard graph-partitioning software METIS [5] to obtain the clusters.

3. Experimental Results

The main impediment to our experiment was to acquire a dataset which will have a variety of tag structures as to enable us to show conceptemy or concept clustering in action. But since XML data sources usually adhere to the same or to a set of similar DTDs, we couldn't get such an ideal dataset. We manually constructed a dataset which will have such a variety in tag structure. Our experimental dataset consisted of 50 documents with varied tag structure. Our experiments results shown in Table 1 and 2 show that our method in fact helps in getting a meaningful and useful clustering of search results.

Results

The running of the experiments involved giving a query in terms of the 'tag' and 'keyword' followed by analyzing the results obtained and updating weights(if necessary).We performed the following queries.

- Tag - 'name', Keyword - 'anderson'.

The resultant documents were 6 in number and were grouped into 3 clusters – Table 1. The documents were analyzed and the following conclusions were made. The documents in cluster 1 consisted of documents where the 'name' referred to the name of author, Clutser 2 consisted of documents where 'name referred to the name of a company and in cluster 3 it referred to the name of a book. Thus we were able to capture semantic information of the document by examining the hierarchical tag structure of the documents.

Table 1

Cluster 1	
<author> <name> Anderson morris </name> <book> ransom </book> </author>	<people> <author> <name> matt Anderson </name> <book> blood sport </book> </author> </people>
Cluster 2	
<author> <company> <name> Anderson consulting </name> </company> <book> Soccer Fans </book> </author>	<book> <author> <company> <name> Anderson consulting </name> </company> <hometown> California </hometown> </author> <year> 1978 </year> </book>
Cluster 3	
<author> <works> <book> <name> peter and Anderson </name> <contents> story of 2 friends </contents> </book> <paper> clustering data </paper> </works> </author>	<author> <book> <name> Anderson Giant </name> <year> 1999 </year> </book> </author>

- Tag - 'title', Keyword - 'professor'.

The resultant documents were 6 in number and were grouped into 3 clusters – Table 2. The documents were analyzed and the following conclusions were made. The documents in cluster 1 and 2 consisted of documents where the 'title ' referred to the title of a book while Cluster 3 consisted of documents where 'title' referred to the title of a person who was author in this case. Between clusters 1 and 2, we examined that although the 'title' occurred in the context of books, the books themselves occurred in different contexts. In cluster 1 the book formed a part of a conference while in cluster 2 it formed a part of a bookstore.

4. Conclusions

Our solution to the problem of Conceptemy offers several advantages. Foremost, the user gets results clustered in accordance with the semantics of his query. Documents, where the 'concept' occurred in similar semantic meaning, occur together and are thus easier to examine. In so doing, we reduce the search space for the user. By examining just a few documents from each cluster, the user can make out which cluster she is primarily interested in.

The basic idea behind 'concept clustering' is that we capture the semantics of the documents with respect to a given 'concept' by examining the hierarchical path of the document. Thus this scheme is actually language independent, it can deduce the context of the document with respect to the given 'concept' in any language, be it French, German or Chinese.

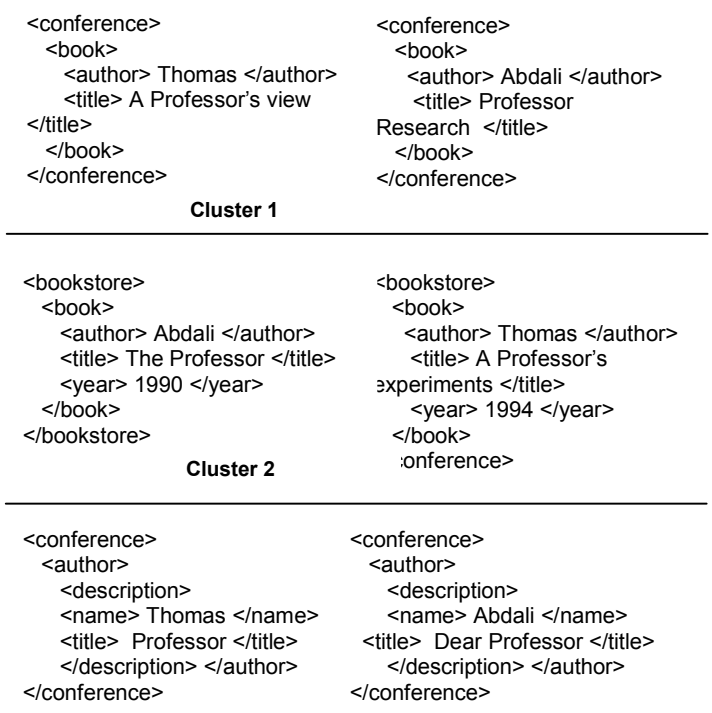

Table 2

5. References

1. D. Quass, J. Widom. R. Goldman, K. Haas, Q. Luo, J. McHugh, S. Nestorov, A. Rajaraman, H. Rivero, S. Abiteboul, J. Ullman, and J. Wiener. LORE: A Lightweight Object REpository for Semistructured Data. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, 1996
2. Xyleme: A dynamic warehouse for XML data of the Web. To appear in *IEEE Data Engineering Bulletin*. http://osage.inria.fr/verso/xyleme/short_paper.html
3. Daniel Egnor and Robert Lord. Structured Information Retrieval using XML. *Workshop On XML and Information Retrieval, ACM SIGIR 2000*
4. GoXML.com - XML Search Engine - Search and index XML documents http://www.goxml.com/
5. George Karypis, Rajat Aggarwal, Vipin Kumar, and Shashi Shekhar. Multilevel Hypergraph Partitioning: Applications in VLSI Domain. 34th *Design Automation Conference.*
6. Eric Ristad, Peter Yianilos. Learning string edit distance. In *Proc. 14th International Conference on Machine Learning*, 1997