Personal Information Retrieval Visualization (PIRV): Clustering and Visualization of Web Document Search Results

Xiangyang Xu and Ernst L. Leiss
Department of Computer Science
University of Houston
Houston, TX 77207
{x228917@yahoo.com, coscel@cs.uh.edu}

Abstract

Conventional web search engines often return long lists of ranked documents as their output. This text-like data presentation for web search results has many limitations. Since only a part of the list of documents can be shown at a time, users cannot get a complete picture of the returned documents. Therefore, users do not know if these documents contain a document they are interested in, after reading the first few items of the list of documents. Due to the imprecise nature of current Web search engines and the explosive increase in the number of documents available, users are forced to spend a significant amount of time going through the list of the results or abandon the current search result.

In this project, we design and implement a system called PIRV (Personal Information Retrieval Visualization), which dynamically groups the search results into clusters and presents these clusters in 2-dimensional graphics. After receiving a query from a user, PIRV sends it to the search engine, receives the returned documents, clusters these documents according to similarity values between individual documents, transforms the data into a graphical representation, and then displays these graphics to the user. With this visual display, a user may use visual perception to evaluation these clusters and to make an intuitive judgment about the relevance of these documents without having to read a significant portion of each document. Furthermore, a user's search history is saved in the user's computer upon logging out; this can be used to assist in future searches. The saved search history file is automatically retrieved by PIRV upon login. A user can also view previous search results when doing multiple query searches.

Keywords: Internet search, clustering of results, visualization.

1. Introduction

Although much diverse information is available when using major search engines, users often suffer from too many results ([1, 2, 3]). Search engines often return excessively many documents or Web pages [4]. These results are then presented as an ordered list of documents with title and/or a brief text summary or description. Because the sheer size of information in the Internet and the imprecision of Web search engines, the number of returned documents through search engines can be huge. Furthermore, only a small part of a long ordered list of documents can be shown at a time.

Filtering the long ranked lists of returned document manually by the user may be time-consuming [1]. Since most of the documents are likely irrelevant, users must filter out many irrelevant documents before finding out what they really want. Users have to spend time going through long lists of documents, ignoring most documents after looking over the first entries in a list, or make additional queries, or switch to another search engine. This process is time-consuming and cumbersome; Web users may get frustrated with this information retrieval when the right web document cannot be found easily.

The records of users' query search histories are not maintained in most Web search services [5]. Users are likely to make multiple queries in order to find out what they want. Multiple queries may give users a set of returned documents based on refined queries or different queries. Users can get some ideas from the documents returned from multiple queries. Often a user may reformulate a query based on the results of multiple queries to improve the quality of search. However, most research engines do not provide the capability to store the query search history. Users may get lost or confused when the multiple query approach is used.

We designed and implemented an interface called "Personal Information Retrieval Visualization" (PIRV). PIRV was designed for clustering the retrieval results returned from a popular search engine (Yahoo [6]) and displaying the clusters visually to the user. Both graphical display and textual display are used in order to give users more information. The information visualization interface is our main focus, with textual display as a supplement. The

Web search results are clustered into groups through a clustering engine using the agglomerative hierarchical clustering algorithm. Users can perceive the clusters straightforwardly. PIRV also provides capabilities for users to manipulate the clusters dynamically. By examining the clusters, users save time and effort since they need not examine the data item by item. Additionally, facilities for storing previous queries are provided.

2. **Web Document Clustering**

In the text of each document extracted from HTML pages, there are many symbols such as numbers, punctuations, and signs, as well as short-words with fewer than four characters, such as a, the, of, is, are, etc. These non-word symbols and short-words are not used in the similarity calculation; they are removed from the string of text representing each document. Only the remaining words are kept and saved.

We group web retrieval documents into clusters according to key word relevance ([7], [8]). It is assumed that documents with the same topic are similar in their descriptions; these will be put into one cluster. Thus, the returned documents by search engines can be clustered into different groups according to their similarities. Our method to cluster the documents is to compare each key word in one document to the key words in every other document. The more matched key words there are between the two documents, the more similar the two documents are considered. The value of similarity between two documents is calculated according to the frequency of matched key words.

Measurement of content similarity between two documents is defined as the score of similarity. We have chosen to use Sorenson's similarity [9]. Sorensen's similarity between two documents X and Y is calculated according to the formula:

$$s = 2a / (2a + b + c)$$

where s = Sorensen's similarity
a = number of common key words in X and Y
b = total number of key words in X - a
c = total number of key words in Y – a.

Sorensen's similarity considers the balance between the total number of key words and the number of common keywords. It also considers the situations of too few common key words. The more common key words there are, the greater Sorensen's similarity is. If two documents have identical keywords, the Sorensen similarity is 1. For documents X and Y, there are two Sorensen similarity values, i.e., X compared to Y and Y compared to X. Since these two values are the same, all Sorensen's similarity values are saved in a triangular table for later clustering usage.

Hierarchical clustering groups documents into a hierarchical tree of clusters [10]. Agglomerative hierarchical clustering (AHC) [10] is the most popular type of clustering procedure and is commonly used for document clustering. Studies indicate that AHC produces clusters with high quality [11]. In this project, we used the AHC algorithm to group the documents into clusters according to their Sorensen similarity triangular table or Sorensen matrix [10]. Initially, every document is regarded as a cluster. There are three steps to find a cluster. First of all, we search the table and find the two clusters that have the closest similarity values. Then, these two clusters are combined into a new one. The average of the two similarities is the new combined similarity score. The third step is to recalculate the triangle table, i.e., the similarities between this new cluster and others are recalculated. We repeat this process until a given number of clusters is left. A user can select this number of clusters when making a query search.

In this project, we improved the AHC algorithm to reduce its time complexity. Previously, the computing time for the AHC algorithm had been reported to be $O(n^4)$ [12]. In our project, the similarity values are built into a triangular table. If there are n documents initially, the initial matrix requires n(n-1) / 2 calculations. However, after two documents are merged into a cluster, recalculating the triangle matrix requires on average $O(n)$ time (for one merge) provided the data structure for the matrix is properly organized. If these documents are organized in a Max heap structure, finding of the document with the maximum value can be done in time $O(1)$. Thus, the overall complexity is as following:

$$\frac{n(n+1)}{2} + \sum_{i=k}^{n-1} O(i) = O(n^2)$$

3. System Design and Architecture

The PIRV system architecture is based on a client-server model as shown in Figure 1. The system can be divided into three parts: 1) the far end components that include the query search subsystem, query search history subsystem, and user account subsystem, 2) the communication component, here the Apache Tomcat 4.0.4 server, that transports data between the client and the server, and 3) the client-side display components that display the results and allow users to interact with the system.

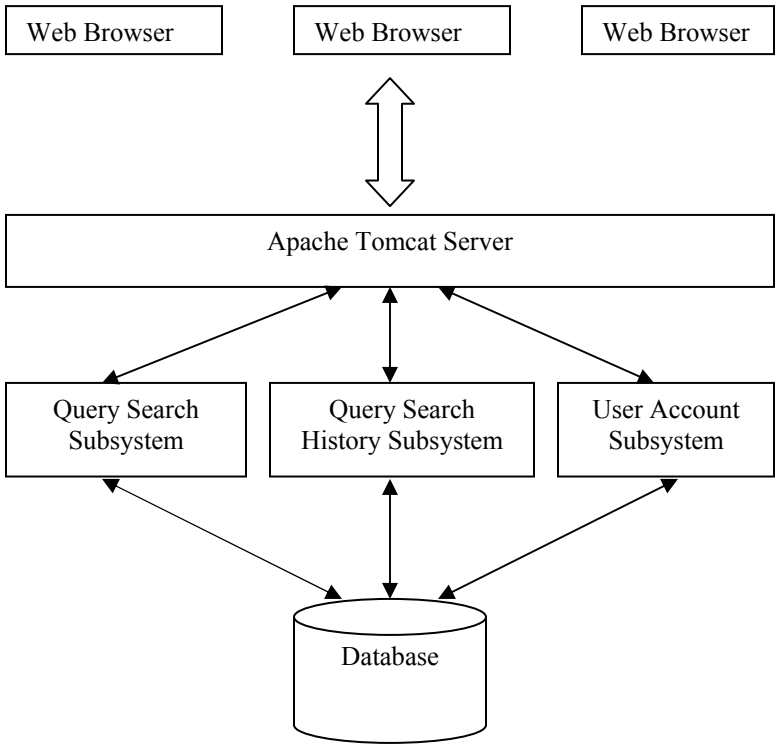

Figure 1 System Architecture of Personal Information Retrieval Visualization (PIRV)

In this project, an interface called "Personal Information Retrieval Visualization" (PIRV) was designed and implemented. PIRV was designed for clustering the retrieval results returned from a popular search engine (Yahoo [6]) and displaying the clusters visually to the user. It is a search engine using JSP and Applets for the GUI layer and using Java for retrieving and document clustering for the Apache Servlet ([13], [14]).

The query search subsystem is the major part of the entire PIRV system. It receives a user-entered query, sends the query to the search engine, collects documents from the search engine, produces clusters from these documents, and displays a graphical presentation of the cluster to the client. We have implemented the query search subsystem in four modules: 1) document retrieval module; 2) result processing module; 3) document cluster module; 4) data presentation interface.

The document retrieval module of the query search subsystem is initiated when a user submits a query via a browser. After receiving the user-entered query, the Java Servlet will transform the query into the URL-encoded format and send the query to the underlying search engine (Yahoo [6]). We use the Java URLconnection class to connect to the search engine. Then, the returned HTML page from the search engine will be received through the same connection channel. The returned HTML pages will be parsed.

The target of the result-processing module is the text of each document retrieved. Non-character symbols and other information must be removed. There are many "stop" words such as a, is, the, are, to, at, on, etc., which should be removed from the text since these "stop" words contribute little to the content of the description of each document. The text is first converted into lower case.

In order to make the document clustering more meaningful to users, the PIRV system gives the users the capability to specify how many clusters to create and how many results to retrieve from the search engine for this query. When a user makes a query, he can choose a number of clusters, from 3 to 15. The user can choose the number of documents, from 10 to 5000 from a Jlist, or all of the results from the search engine. He can also enter a number to specify the number of results.

The visualization of clusters is implemented on a class that extends the canvas. One of three buttons such as "Bar", "Pie", and "Dot" gives the signal and initializes the redraw process on the same canvas. Instances of three inner classes, chartCluster, pieCluster, and dotCluster, are used to represent the individual components such as bars in the bar graph view, pie slices in the

pie view, and circles in the dot view. These components will draw themselves on the canvas according to the values of the parameters that they have received. The components defined by the instances of these three inner classes all have a function to report their identities when users click on them so that the applet has to show the cluster information in the display panel.

The PIRV system has a query search history subsystem in order to allow users to see the results of previous queries. The search history may be important to some users who will do multiple queries and refined queries. Current query search results are temporarily saved in a server-side database so that the user can view them. When the user logs out, the query search results are sent to and saved in the user's computer upon request. The data in the server-side database are deleted. Next time when the user logs in, the file stored in his computer is automatically read and sent to the PIRV system. The data are extracted and installed in the server-side database.

The database was developed to hold account information and query search data of users. The tables are designed to hold query search information such as queries, clusters, and documents. Considering that writing to or reading from any text file is time-consuming, no text file was used in this project. Using the driver of the JDBC/ODBC Bridge is one of the common methods for a Java Servlet to support database connectivity [13]. We need both the ODBC driver and the JDBC/ODBC Bridge to establish a connection to the relational database. To access the database efficiently, we have built a database connection class, which has a constructor to make a connection to the database and has many functions to manipulate the contents of tables.

4. Paradigms in Visualization of Web Document Search Results

Information visualization investigates methods to support the exploration of large volumes of abstract data using graphical representations so that users may use their visual perception to evaluate and analyze the data ([15], [16], [17], [18]). This involves ways to transform the data into graphical expression. The data presentation interface in this project is to represent clusters in vivid 2-d graphic formats so that users may use their visual perception to evaluate and analyze the query search results. We have created three types of data presentation interface to

visualize the clusters, namely the bar graph view, the pie view, and the dot view. The three vitalization interfaces are implemented in separated canvases, which sit on three panels inside the Java applet. There are three command buttons for each panel. If a query search is successful, the three view buttons become active. The user can choose which view he likes by clicking on the corresponding command button.

The bar graph view arranges the clusters in the form of charts (see Figure 2), which is located on the left side of the applet. Each bar represents a cluster. The height of each bar represents the amount of documents of this cluster. The bars have different colors allowing users to differentiate the bars easily. The user clicks a bar to select the corresponding cluster. The scrolled list of documents of this selected cluster will show up in the display panel on the right side of the applet. Each document displays its document sequence number, a hyperlinked title, and a description. If the user wants to go to the page he is interested in, he clicks on the title and the selected page will show up in a separate window.

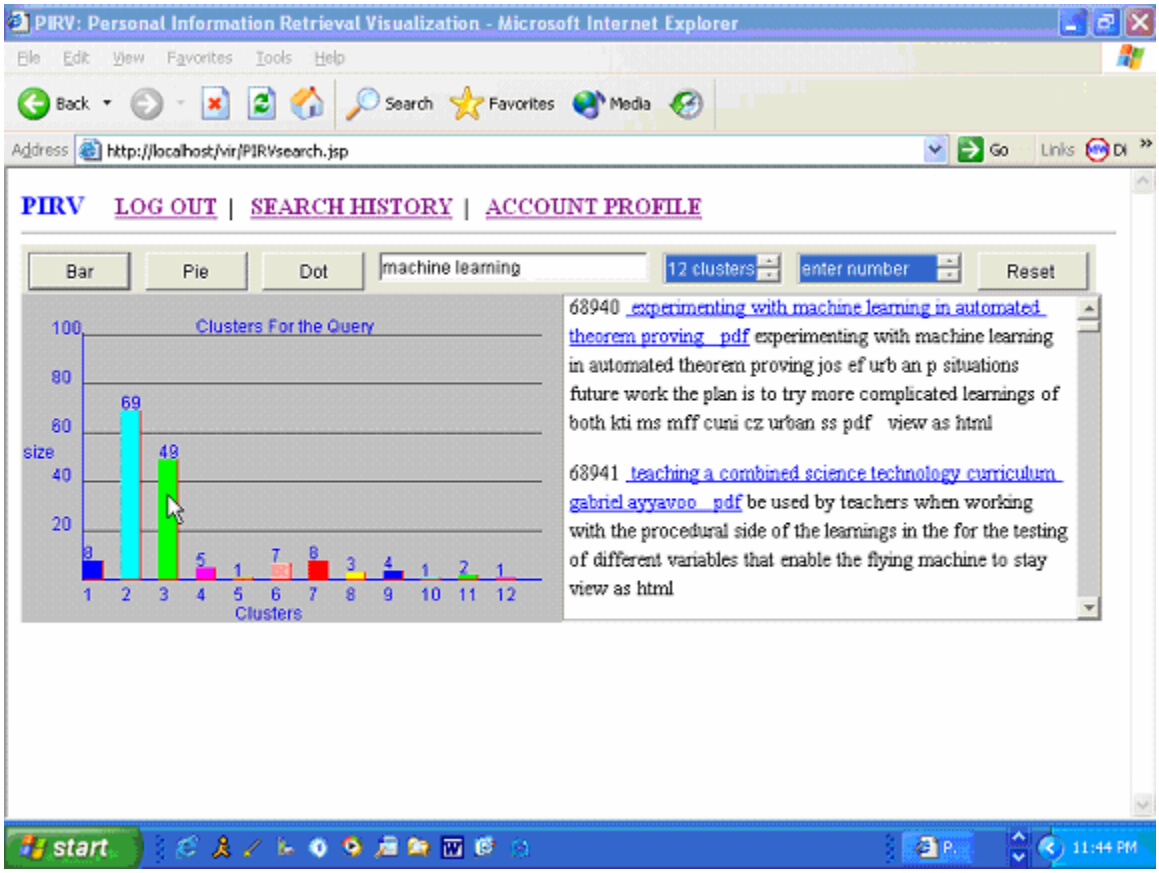

Figure 2 The Bar Graph View of the Data Presentation Interface

The pie view is to arrange the clusters in the form of a pie graph (see Figure 3). If the user clicks the pie button on the top of the applet, the pie view panel appears in the same place as the bar graph view. The pie is divided into different slices. Each cluster is represented by a pie slice that has a different color from its neighbors. The relative size of each piece in the pie represents the number of documents in this cluster. Similar to the bar graph view, the slices of the pie have links to their corresponding scrollable document lists that are shown in the display panel if selected. The user may easily select and view an individual cluster by clicking the corresponding slice in the pie graph.

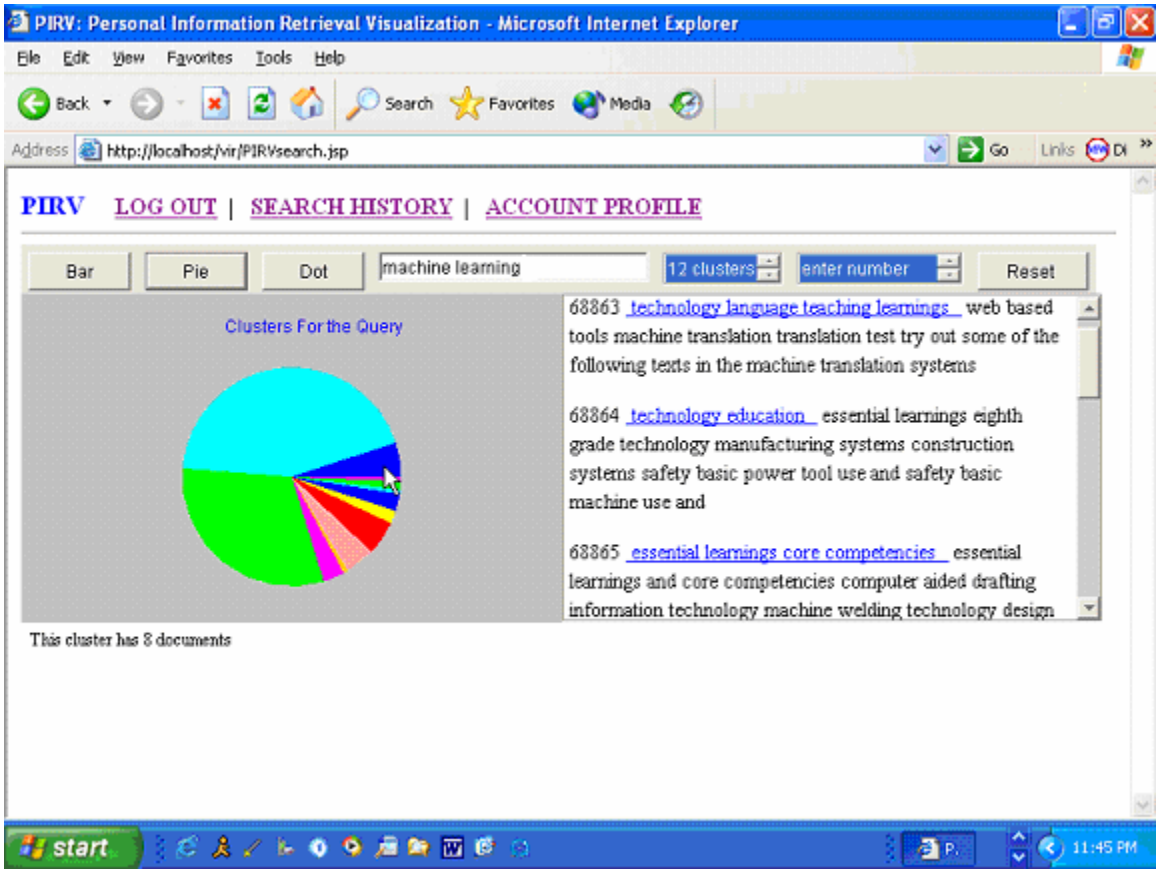

Figure 3 The Pie View of the Data Presentation Interface

The dot view is a concentric circular graphic with the dots located on it (see Figure 4). The graph has three concentric circles that give users a distance perception to the center. Dots represent the clusters. These dots have differing sizes, colors, and distances from the center.

The size of a dot represents the number of documents in this cluster. The colors of the dot allow the user to differentiate the clusters easily. The distance from the center is dependent upon the relevance to the query. The closer to the center, the more relevance to the query the dot has. Also, each dot is linked to the cluster information that is shown on the right side of the screen. The user may click a dot to select the corresponding individual cluster.

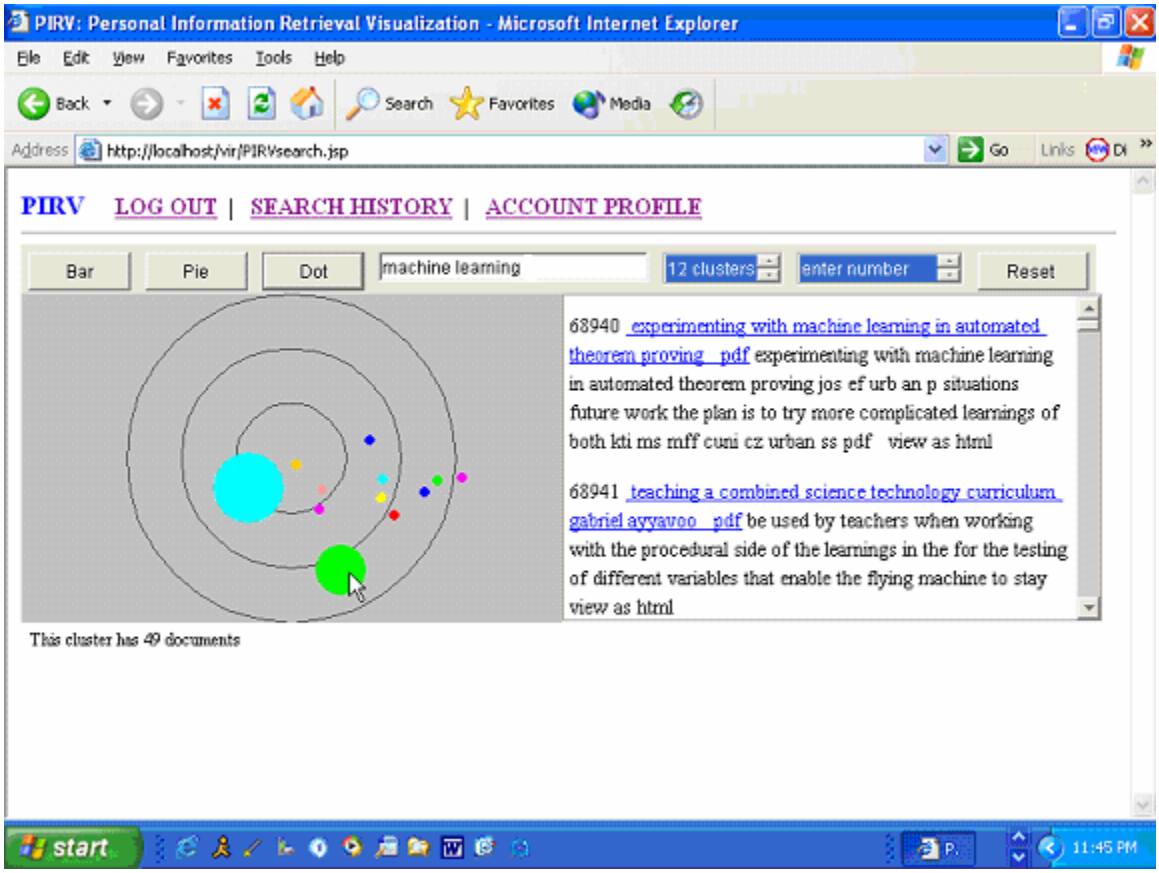

Figure 4 The Dot View of the Data Presentation Interface

5. Conclusions

In this paper we presented a Personal Information Retrieval Visualization (PIRV): a query search service tool with the capabilities of clustering and visualizing Web document search results. It provides the following features:

1) The system is an online system so that users can login into the system from anywhere in the world using a web browser.

2) Users can utilize this software to do query searches. The retrieval results are clustered and the clusters are displayed in the front-end so that users can view and browse these results.
3) The software provides three visualization panels that are serviced by the same data manager module. Users can visualize the retrieval results in three different ways, namely the bar graph view, the pie view, and the dot view.
4) The software provides a search history function. Users can save results between sessions. Subsequently, they can view and manipulate their saved search results.
5) The software design is independent of the search engine. PIRV can be used based on any search engine or meta-search engine with minimal coding changes.

The implementation of this system should have benefits for users by alleviating the difficulty of displaying many search results from Web search engines. Because our visual interface also provides the clusters' relevancy to a query, it allows users to make an intuitive judgment about the relevancy of documents. This tool may allow users to weed out quickly irrelevant clusters and concentrate on one or more relevant clusters. In this way, PIRV should be particularly useful when a large amount of results are retrieved from Web search engines.

We have identified a number of areas in which the design and implementation of the system can be enhanced. A study of evaluation of the PIRV display is needed to answer the question how much the clustering and visualization of this tool help users to find relevant documents more efficiently. More information about each cluster may be provided in order to give user a general idea about this cluster. This may need more computation in the server. The visual interface itself can be further improved by allowing users to zoom in on the clusters. In this way, a user can concentrate on a subset of documents and see more graphical detail.

Bibliography

[1] Eun-II Cho and Sung Hyon Myeng. Visualization of Retrieval Results Using DART. In Proc. of RIAO. Paris, April, 2000. http://133.23.229.11/~ysuzuki/Proceedingsall/RIAO2000/Friday/118BO6.pdf

[2] Oren Zamir and Oren Etzioni Grouper. A Dynamic Cluster Interface to Web Search Results. In Proc. of the Eighth International World Wide Web Conference (WWW8), 1999. http://www.cs.washington.edu/research/projects/WebWare1/etzioni/www/papers/www8.pdf

[3] A. Abdollahzadeh Barfourosh, H.R. Motahary Nezhad, M. L. Anderson and D. Perlis. *Information Retrieval on the World Wide Web and Active Logic: A Survey and Problem Definition*, 2002. http://www.cs.umd.edu/Library/TRs/CS-TR-4291/CS-TR-4291.pdf

[4] Offer Drori. *Improving Display of Search Results in Information Retrieval Systems – Users' Study*. http://shum.huji.ac.il/~offerd/papers/drori072001.pdf

[5] Susan Havre, Elizabeth Hetzler, Ken Perrine, Elizabeth Jurrus, and Nancy Miller. *Interactive Visualization of Multiple Query Result*s. http://www.pnl.gov/infoviz/sparklerinfovis01.pdf

[6] Yahoo http://www.yahoo.com

[7] M. D. Dunlop. Development and Evaluation of Clustering Techniques for Finding People. Proceedings of the Third International Conference on Practical Aspects of Knowledge Management, Basel, Switzerland, 30-31 October 2000. http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-34

[8] M. A. Hearst and J. O. Pedersen. Reexamining the Cluster Hypothesis. In Proceedings of SIGIR '96, pp.76—84, 1996. http://www.scils.rutgers.edu/~muresan/Docs/sigirHearst1996.pdf

[9] Coles, S.L., R.C. DeFelice, and D. Minton. Marine Species Survey of Johnston Atoll, Central Pacific Ocean. Report to U.S. Fish & Wildlife Service, Honolulu. Bishop Museum Technical Report 19, 2001. http://hbs.bishopmuseum.org/pdf/johnstonreport.pdf

[10] Michael Steinbach, George Karypis, and Vipin Kumar. A Comparison of Document Clustering Techniques. Department of Computer Science and Engineering, University of Minnesota, Technical Report #00-034, 2000. http://rakaposhi.eas.asu.edu/cse494/notes/clustering-doccluster.pdf

[11] Douglass R. Cutting, David R. Karger, Jan O. Pedersen, and John W. Tukey. Scatter/Gather: A Cluster-based Approach to Browsing Large Document Collections, SIGIR '92, pp.318 – 329, 1992. http://www.scils.rutgers.edu/~muresan/Docs/sigirCutting1992.pdf

[12] Oren Zamir and Oren Etzioni. Web Document Clustering: a Feasibility Demonstration. Proceedings of the 19th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'98), pp.46-54, 1998.

[13] Marty Hall and Larry Brown. *Core Web Programming*. New Jersey, Prentice Hall, 2001.

[14] The Java Servlet Technology Pages http://java.sun.com/products/servlet/index.html.

[15] Stuart K. Card, Jock D. Mackinlay, and Ben Shneiderman. *Readings in Information Visualization*. Morgan Kaufman, 1999.

[16] Robert Spence. *Information Visualization*. Addison-Wesley, 2001.

[17] David A. Carr. Guidelines for Designing Information Visualization Applications. Proceedings of the 1999 Ericsson Conference on Usability Engineering. http://www.sm.luth.se/~david/papers/VizGuidelines.pdf

[18] Matthew Carey, Frank Kriwaczek, and Stefan M. Ruger. *A Visualization Interface for Document Searching and Browsing*. http://www.doc.ic.ac.uk/~frk/frank/inforet.pdf