Dynamicity vs. Effectiveness: A User Study of a Clustering Algorithm for Scatter/Gather

Weimao Ke, Cassidy R. Sugimoto, and Javed Mostafa
Laboratory of Applied Informatics Research
School of Information and Library Science
University of North Carolina at Chapel Hill
{wke, csugimoto, jm}@unc.edu

ABSTRACT

We proposed and implemented a novel clustering algorithm called LAIR2, which has linear worst-case time complexity and constant running time average for on-the-fly Scatter/Gather browsing [4]. Our previous experiments showed that when running on a single processor, the LAIR2 on-line clustering algorithm was several hundred times faster than the parallel Buckshot algorithm running on multiple processors [11]. This paper reports on a study that examined the effectiveness of the LAIR2 algorithm in terms of clustering quality and its impact on retrieval performance. We conducted a user study on 24 subjects to evaluate on-the-fly LAIR2 clustering in Scatter/Gather search tasks by comparing its performance to the Buckshot algorithm, a classic method for Scatter/Gather browsing [4]. Results showed significant differences in terms of subjective perceptions of clustering quality. Subjects perceived that the LAIR2 algorithm produced significantly better quality clusters than the Buckshot method did. Subjects felt that it took less effort to complete the tasks with the LAIR2 system, which was more effective in helping them in the tasks. Interesting patterns also emerged from the subjects' comments in the final open-ended questionnaire. We discuss the implications and future research.

Categories and Subject Descriptors

H.3.3 [**Information storage and retrieval**]: Information Search and Retrieval—*Clustering*; H.3.3 [**Information storage and retrieval**]: Information Search and Retrieval—*Search process*

General Terms

Algorithms, Measurement, Performance, Experimentation

Keywords

Information retrieval, Scatter/Gather, clustering, user study, visualization, effectiveness, efficiency

1. INTRODUCTION

Effective and efficient browsing methods for large text collections have been widely examined. Among existing methods, Scatter/Gather browsing is well known for its ease of use and effectiveness in situations where it is difficult to precisely specify a query [4, 9]. It combines search and interactive navigation by gathering and reclustering user-selected clusters.

The Scatter/Gather browsing method was first proposed by Cutting et al. (1992) [4]. In each iteration of this browsing method, the system scatters the dataset into a small number of clusters/groups, and presents short summaries of them to the user. The user can select one or more of the groups for future study. The selected groups are then gathered together and clustered again using the same clustering algorithm. With each successive iteration the groups become smaller and more focused. Iterations in this method can help users refine their queries and find the desired information from a large data collection.

Researchers also applied Scatter/Gather to browsing retrieved documents from query-based searches. It was found that clustering was a useful tool for the user to explore the inherent structure of a document subset when a similarity-based ranking did not work properly [8]. Relevant documents tended to appear in the same cluster(s) that could be easily identified by the users [9, 17].

Since the Scatter/Gather method requires on-the-fly clustering on a large data corpus, fast clustering algorithms are essential. Clustering efficiency is often more important than accuracy because it is the real-time interaction with the user that potentiates the value of Scatter/Gather [8].

Two linear time clustering algorithms, namely the Buckshot and the Fractionation, were implemented for the original Scatter/Gather method [4]. Both algorithms have $O(kn)$ time complexity, where k is the number of desired clusters and n the total number of documents. As compared to the Buckshot, the Fractionation algorithm is a little slower but with higher accuracy. Although better than a quadratic time complexity, $O(kn)$ is not fast enough for large document collections. A parallel version of the Buckshot algorithm, which achieved a $O(n \log n)$ time complexity, was further proposed and evaluated [10].

Research also investigated a method that used a precomputed hierarchy of meta-documents for further expansion of selected items and reclustering of the subset [5]. Only dealing with a subset of M meta-documents in each iteration, the algorithm achieved constant interaction-time for Scatter/Gather browsing. However, the reclustering process is

not efficient enough for real time interaction because M cannot be too small ($M >> k$, the number of clusters needed). On the other hand, by summarizing descendant documents, meta-documents might be too large to be reclustered efficiently, or too small to be accurately representative.

A limited number of user studies have been performed on Scatter/Gather interfaces. Research analyzed how often the subjects chose the clusters with relevant documents after issuing the first search and reclustering the results by means of the Scatter/Gather function. It was found that users predominately chose the cluster with the largest number of relevant documents [9]. It was also shown that Scatter/Gather induced a more coherent view of the text collection than query-based search and supported exploratory learning in the search processes [17].

2. PROPOSED CLUSTERING METHOD

Notwithstanding the usefulness of the Scatter/Gather method, real-world Scatter/Gather systems are rarely found. The existing algorithms for efficient clustering have not sufficiently enabled on-the-fly clustering for responsive Scatter/Gather services, particularly when dealing with large data collections and many concurrent users. This research focused on the design and evaluation of a new algorithm for this purpose.

In Section 2.1, we elaborate on the proposed LAIR2 algorithm, which takes advantage of a precomputed hierarchy but, different from [5], does not rely on meta-documents for reclustering. In Section 2.2, we discuss findings from previous research regarding the efficiency of LAIR2 [11]. Then we move onto the focus of this study and discuss the value and potential effectiveness of the LAIR2 algorithm in Section 2.3 and 2.4, followed by research questions in Section 3.

2.1 LAIR2 Algorithm

There exist a large number of data clustering algorithms, which can be classified into two main categories, namely, hierarchical algorithms and partitional algorithms [6]. A hierarchical clustering algorithm generates a cluster hierarchy, which is called a dendrogram. A dendrogram is a tree that records the process of clustering. Similar items are connected by links whose level in the tree is determined by the similarity between the two items. The hierarchical algorithms can be further divided into agglomerative and divisive methods. The major difference between the two is that an agglomerative approach works in a bottom-up manner and a divisive approach top-down.

A partitional clustering algorithm obtains a single layer of data partition rather than a cluster hierarchy. In other words, it is flat as compared to the hierarchical methods. One major advantage of the partitional clustering algorithms is the efficiency. The widely used K-means method and its variances belong to this category.

We present a novel on-line clustering algorithm called LAIR2, which can greatly improve the response time of Scatter/Gather browsing sessions. The algorithm is composed of two phases. In the off-line phase, a cluster hierarchy is generated using traditional hierarchical clustering algorithms. Later in the on-line phase, drawing on the previously generated hierarchy, the on-line LAIR2 algorithm is used to cluster user selected data items in almost constant time.

In the first phase of the LAIR2 clustering algorithm, an arbitrary agglomerative (or divisive) hierarchical algorithm

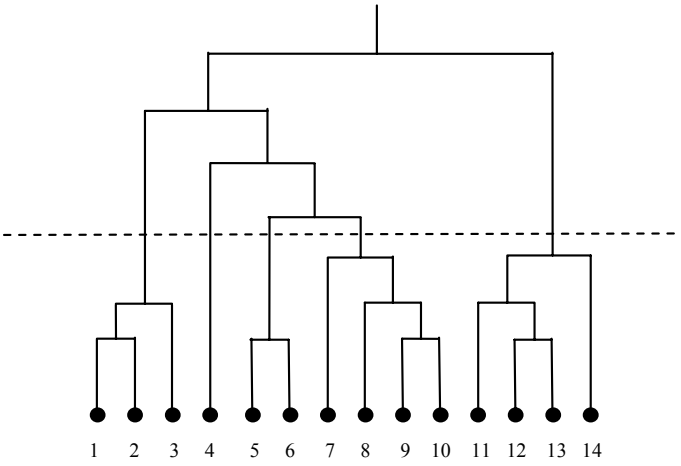


Figure 1: A dendrogram shows how the clusters are agglomerated hierarchically. By cutting the tree at different heights, different number of clusters can be generated. For example, the dashed line in this figure generates five clusters from the dendrogram.

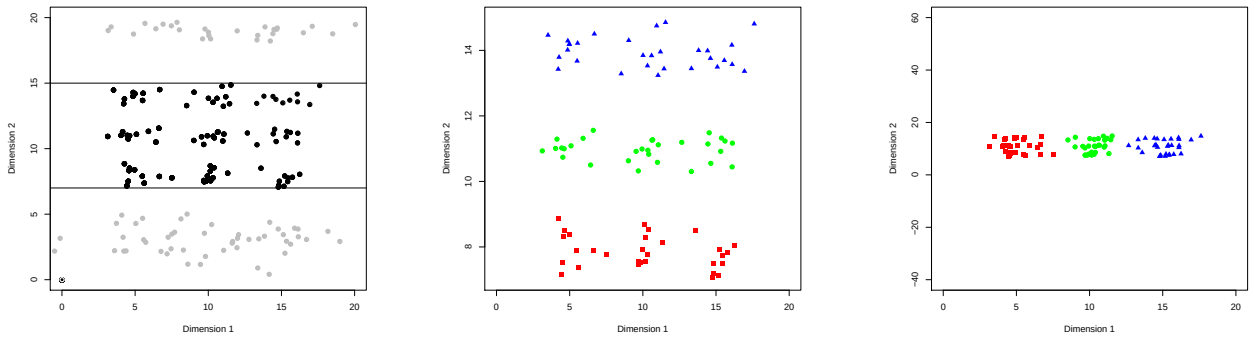
can be used to construct a dendrogram of clusters. The result is represented by a sequence of the agglomerated pairs of data points. Table 1 shows a possible agglomeration sequence of the dendrogram visualized in Figure 1.

	Cluster 1	Cluster 2	Similarity	New Cluster No.
1	1	2	0.94	n+1
2	12	13	0.89	n+2
3	5	6	0.70	n+3
4	9	10	0.69	n+4
...	...	...	...	...
n-1	2*n-5	2*n-2	0.21	2*n-1

Table 1: A possible agglomeration sequence of a dendrogram

In each iteration of the on-line clustering, we decide upon the initial k centroids by making use of the agglomeration sequence constructed in the first phase. Suppose the desired number of the clusters is k and the number of the clusters in the user selected subset is k', and obviously there exists $k' < k$. Now the problem is transformed to finding k centroids of the data points which are previously clustered into k' groups. Instead of calculating the k centroids from scratch, we make use of the previous knowledge, the agglomeration sequence table.

Since we already have k' centroids in the current working data collection (subset), we just need to find more centroids to make the total number of centroids equal to k. To achieve this, we split the current k' clusters according to the precomputed dendrogram in a top-down manner. We scan the dendrogram from the top (or the sequence table from the bottom), skipping those cluster pairs that have at least one cluster out of the current working data collection. After the first cluster pair whose data points are all in the working collection is found, we split it by removing this entry and adding its two sub-clusters to the proper positions in the table. This process is repeated until $k - k'$ clusters have been split, which means k centroids have been identified for



(a) Initial space and user selection (b) Weigh Y dimension more important (c) Weigh Y dimension less important

Figure 2: Local Feature Reweighing and *Dynamic* Clustering

the current data collection. The updated agglomeration sequence table is kept for later use in the next Scatter/Gather iteration.

2.2 Efficiency

Given the size of the agglomeration sequence table $n-1$, the worst case time of the split process is $n+k-k'-2$, which is $O(n)$. In most cases, the split process will stop in $c(k-k')$ steps, where c is a constant related to k and k'. Therefore, this algorithm has a constant time complexity.

Our previous research has demonstrated the high efficiency of the proposed LAIR2 algorithm [11]. On a data collection containing tens of thousands documents, the on-line LAIR2 clustering algorithm runs several hundred times faster than the parallel Buckshot algorithm [4, 10]. When the size of data collection increases to hundreds of thousands, the clustering time of our algorithm is still satisfactory. For instance, even with an unusually large number of targeted clusters $k = 256$, the average response time of the LAIR2 algorithm (based on a single processor) is 0.4 seconds on a dataset of $256,000$ documents. Please refer to [11] for detailed results on the efficiency of the LAIR2 algorithm and how it supported a real-time system for on-line Scatter/Gather browsing.

2.3 Novelty and Flexibility

The LAIR2 method looks similar to another algorithm that used precomputed hierarchical meta-documents [5]. They both use a precomputed hierarchy for re-scattering and achieve constant interaction-time. One might question the value of the new algorithm here because of the similarities. Nonetheless, there are several essential differences. Firstly, although both are of constant interaction-time, our approach simply traverses the hierarchy and expand the selected clusters without *local* reclustering. This further improves the on-line interaction efficiency, essential to a system that provides responsive services.

Secondly, our approach remains flexible for a user to select any clusters in each iteration. Although some researchers observed that approaches of this kind were too restrictive and could serve only one cluster at each presentation [5], this is not necessarily the case, as described in Section 2.1. Whereas the method in [5] focused on coarse-grained patterns of local subsets by reclustering meta-documents, our approach maintains a global view of the local ones and reasonably skips local reclustering.

Previous research has supported the usefulness of traversing a hierarchy without local reclustering. A cluster hierarchy was used in research to build an interactive browser for a hypertext collection and was shown to be sufficiently comprehensive and flexible to support a variety of user searches [3]. Whereas LAIR2 provides efficient on-line clustering, the off-line pre-computation can use incremental hierarchical clustering methods such as [18] to better serve text collections that have frequent updates (e.g., news).

2.4 Dynamicity and Effectiveness

Notwithstanding the fact that the LAIR2 algorithm is able to dynamically generate various combinations of clusters based on users' choices, one may argue that this type of algorithm is still "static" in the sense that the clusters existed before user selection. A real *dynamic* reclustering process based on the *local* subset selected by the user will arguably produce better results. In the text below, we will illustrate that this *dynamicity*[1], by using *new information* that emerges from user selection, is potentially desirable. However, very few existing clustering methods take advantage of user selections to achieve better *dynamicity* and thus effectiveness [20]. Hence , we will argue that the LAIR2 algorithm based on a static precomputed structure is essentially not different from and not inferior to classic methods in terms of clustering *effectiveness*.

In order to simplify the discussion and to visualize the clustering process, assume we have a set of documents, each of which can be represented with two dimensions (or features). Thus, the documents can be plotted on a 2D space, shown in Figure 2 (a).

If, for instance, the user chooses documents in the middle (darker dots in Figure 2 (a)), this selection provides new information and there are different heuristics we can use for further clustering. One may reason that, because the user chooses something based on dimension Y (Y values in the

[1]By *dynamicity*, we mean a clustering method's capability of understanding the user's information need and generating new clusters for the need. The key of a dynamic method is to take advantage of the new and dynamic information based on what is selected vs. what is not, which is only available during user system interactions.

middle), it is potentially important to the user and we should continue to let the user further utilize this dimension for successive clustering. In this case, we should weigh dimension Y more significantly for the local reclustering. Visually, this is to zoom in on dimension Y and do the clustering, which will produce clusters shown in Figure 2 (b).

Another reasoning is that if the user has already differentiated the clusters based on Y, then the dimension is no more discriminative and the user is ready to see the documents from other perspectives. Thus, we shall weigh less the significance of dimension Y. By minimizing Y, it will produce another subset of clusters, shown in Figure 2 (c).

Based on the illustration above, reweighing the local dimensional (feature) space appears to be necessary in order to have the benefits of local re-clustering and *dynamicity*. For example, one may treat the local subset as a new document collection and use it to recompute DF (or Document Frequency) values. Another possibility is to identify the commonalities and differences between the chosen clusters and the unselected ones and use that information to recompute the subspace structure.

Although utilization of *local* (or *subspace*) structure for clustering was discussed in the literature, very few researchers looked into the issue of *dynamic*, or query specific, clustering based on user selections/queries [14, 20]. Research found such dynamic clustering outperformed static clustering under some experimental conditions. However, further research is needed to validate the findings in a broader context [20].

When used for Scatter/Gather browsing, existing methods are not assumed to apply *local* reclustering for potentially better *effectiveness*. Therefore, we argue that the LAIR2 algorithm is not inferior to existing methods in terms of clustering *effectiveness*. In other words, without closer scrutiny, it should not be taken for granted that the LAIR2 algorithm, based on a precomputed hierarchy, is less dynamic and sensitive to user selections.

3. RESEARCH QUESTIONS

Without further evidence, the effectiveness of the proposed LAIR2 algorithm remains arguable. The major objective of the Scatter/Gather clustering is to better serve users' information needs through interactive navigation. The question becomes whether the LAIR2 algorithm can satisfy the users better in their search tasks, as compared to some classic clustering algorithm for Scatter/Gather.

Furthermore, by shifting the computational burden to the off-line phase, one can fine tune the off-line process to produce high quality hierarchical clusters for on-line Scatter/Gather. Seen in this light, the LAIR2 is potentially better than classic algorithms that have to find trade-off between efficiency and effectiveness on the fly. If this is true, the proposed LAIR2 algorithm will have benefits of both efficiency and effectiveness, thus enabling research on the Scatter/Gather browsing to move forward from experimental prototyping to real-world application. To sum, we have three levels of research questions below.

1. User-centric quality: Do users perceive clusters produced by LAIR2 as better than those produced by a classic clustering algorithm for Scatter/Gather? (We will elaborate on the classic Buckshot algorithm for comparison in Section 4.2.1.)

2. Task-oriented usability/utility: Is the LAIR2 algorithm more effective in helping Scatter/Gather users find relevant information?

3. Overall user satisfaction: Are users more satisfied with the Scatter/Gather system supported by the LAIR2 algorithm?

We are additionally interested in assessing the users' interactions with and perceptions of the Scatter/Gather visualization interface. We are also interested in the type of tasks and contexts in which the Scatter/Gather interface is useful, in comparison with more traditional search tools.

4. TWO SYSTEMS FOR COMPARISON

This study evaluated a Scatter/Gather information retrieval interface, by comparing the difference between two different algorithms. Two systems, one based on the Buckshot clustering [4] and the other the proposed LAIR2 algorithm, were implemented. The two shared common features in the overall information flow (Section 4.1.1), document representation (Section 4.1.2), feature selection (Section 4.1.3), similarity measure (Section 4.1.4), and the user interface (or the Scatter/Gather browser, Section 4.1.5). They differed from each other in the clustering algorithms that supported the on-the-fly gather-and-scattering. Section 4.1 elaborates on the commonalities while Section 4.2 discusses the algorithmic differences.

4.1 Common Features of the Two Systems

4.1.1 Information Flow

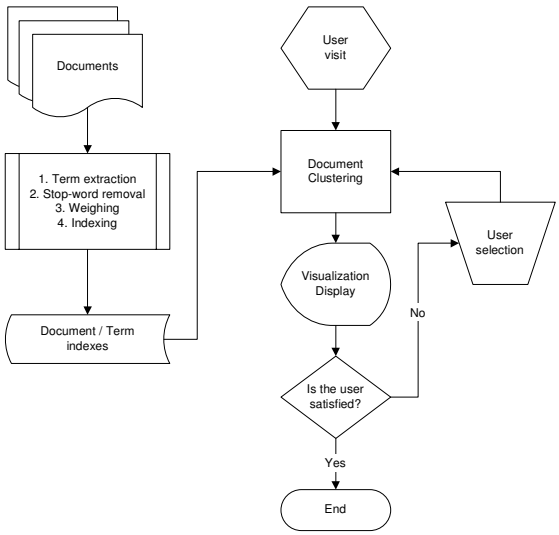


Figure 3: System information flow

Figure 3 shows the information flow of both Scatter/Gather systems. It began with preprocessing a text collection: term extraction, stop-word removal, term weighing, and indexing. We used the Lemur toolkit for language modeling and information retrieval to process and index the documents [16], which was then used by the on-line browser for clustering and document retrieval. When a user began a session, the system clustered the whole collection and presented the results. The user selected favored clusters and *gathered & scattered* them to produce new clusters. This process was

continued to refine the search results until the user was satisfied.

4.1.2 Document Representation

For document representation, we used the well known Vector-Space Model [2] to construct document-term vectors. Feature selection (see Section 4.1.3) produced a thesaurus for a document collection. This thesaurus was then used to represent each document using the TF*IDF (Term Frequency * Inverse Document Frequency) weighing scheme. A document was then converted to a numerical vector where the i^{th} item was computed by:

$$W_i = T_i \cdot log(\frac{N}{n_i}) \quad (1)$$

where T_i is the frequency of the i^{th} term of the thesaurus in the new document, N is the total number of documents in the representative document set, and n_i is the number of documents in the representative document set containing the i^{th} term of the thesaurus. TF*IDF is a well-known and effective technique for term weighing [2].

4.1.3 Feature Selection

In order to further improve clustering efficiency for both systems, feature selection (or dimensionality reduction) was used to reduce the size of the feature space without sacrificing clustering quality. Research compared several feature selection methods and showed that the document frequency thresholding (DF) method delivered competitive effectiveness for categorization tasks until more than 90% of the features were removed [21]. A similar pattern was also found for text clustering: DF maintained good clustering quality until more than 96% of the terms were removed [15].

DF thresholding is a simple technique and scales well to large data collections [21]. Its complexity is linear to the number of the training documents. Given this efficiency, and the effectiveness discussed above, DF thresholding was used for both Scatter/Gather systems to remove both frequent and rare terms.

4.1.4 Similarity Measure

To cluster documents, we used a similarity measure based on document-term vectors. Document-term vectors were produced by the TF*IDF representation scheme described above while each cluster was represented using the centroid of documents assigned to the cluster. Then a similarity measure, called Cosine Similarity Coefficient [2], was used to compute the cosine of an angle between two vectors (document-document, cluster-cluster, or document-cluster). Given two non-null vectors $X = [x_1, .., x_t]^T$ and $Y = [y_1, .., y_t]^T$, their cosine similarity was computed by:

$$\frac{\sum_{i=1}^{t} x_i \cdot y_i}{\sqrt{(\sum_{i=1}^{t} x_i^2) \cdot (\sum_{i=1}^{t} y_i^2)}} \quad (2)$$

A clustering method then used the similarity values to determine what to split or what to merge during each clustering iteration.

4.1.5 User Interface

We implemented a Scatter-Gather browser based on the proposed LAIR2 algorithm. Using information visualization techniques, this browser would help users refine their search and narrow down search results interactively and visually. Another system enabled by the Buckshot clustering algorithm [4] was implemented as well (see Section 4.2.1). The two interfaces looked identical (Figure 4). Features of the interface were carefully designed in terms of what representations fitted Scatter/Gather, what information could be efficiently conveyed through visualization, and how all this could be integrated to support Scatter/Gather interactions [12, 7]. As shown in Figure 4, the primary elements of the user interface include (more details in [11]):

1. Cluster: visualized by color-coded circles, is a group of documents in the database which are similar/related to each other.

2. Cluster size: determined by the number of documents that belong to a cluster. Note that it is based on a log function of the number in order to visualize very large and very small clusters.

3. Cluster color: determined based on the homogeneity of the given cluster–the warmer the color, the higher the level of homogeneity of that cluster.

4. Cluster position: determined by the similarity of two given clusters–the closer the clusters, the higher their similarity.

5. Gather & Scatter button: function for iterating through the database after selecting the desired cluster/s. If the button is clicked without selecting any of the clusters, an error message is displayed asking the user to select at least one cluster.

6. Back button: used to return to the previous cluster selection at any time in the Scatter/Gather process.

7. Reset button: used to reset the Scatter/Gather browser to its default initial state, i.e., top level clusters.

8. Slider: used to increase or decrease the number of clusters to be displayed on the screen after each iteration. Moving the slider to the left decreases the number of clusters desired and vice versa.

Using the above mentioned functionalities, the system operates in the following way. The initial index page of the Scatter/Gather browser shows, by default, seven clusters/nodes displaying seven main topics of the the text collection. These clusters are arranged near or away from each other, based on the similarity of the associated documents. Moving the cursor over a specific cluster displays more information about it in the middle window.

The list of articles related to the shown clusters is displayed in the bottom window. The initial list shows the first ten related documents with brief descriptions and links to detailed information. Links to additional document lists appear at the bottom of the current list. Clicking on a title will display the document in the bottom-right frame, where the user can read the article and determine if it is relevant.

For searching on the desired topic, the user selects one or more clusters by clicking on the cluster/s. A blue border appears around the selected cluster/s, identifying it as chosen for further iteration. To deselect the cluster, the user clicks again on the same cluster and the blue border disappears. To produce the iteration, the user presses the Gather & Scatter button, located on the top left side of the window. This produces a new display of clusters, showing information related to the selected cluster/s.

In both the article list and article display frames, the user is presented with a set of rating buttons, i.e., the icons showing one star to three stars. The user can rate how relevant

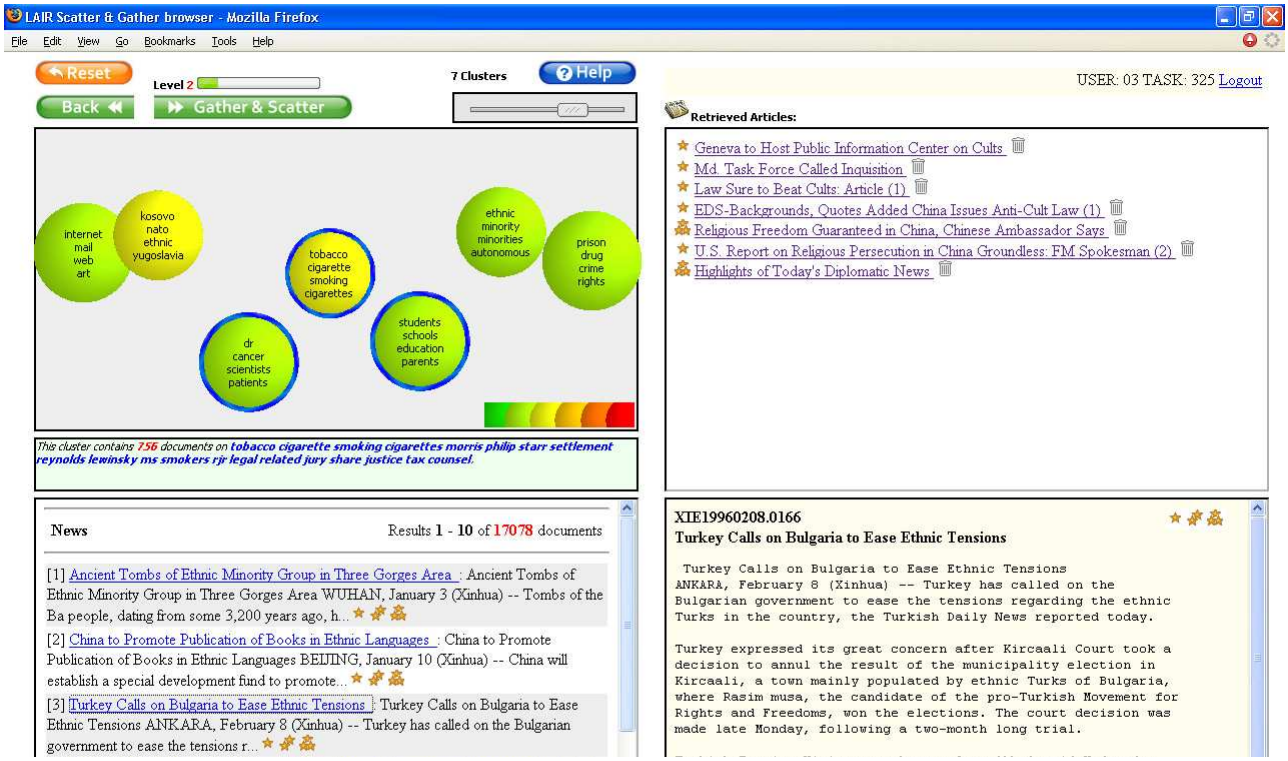


Figure 4: Scatter/Gather browser

the article is by clicking one of the "star" buttons (one-star denotes somewhat relevant and three-stars highly relevant). Once selected, the article will appear in the retrieved articles frame on the upper right side of the display. A "delete" button (with a trash can icon) is provided in case the user later decides that an article is irrelevant. The user can continue to search and select articles in this way. At any time, the user can click on the *Reset* button to return to the default initial page (without losing any of the retrieved/ranked documents).

4.2 Differences between the Two Systems

In this section, we will discuss the algorithmic differences between the two systems for comparison. Section 4.2.1 elaborates on the processes involved in the Buckshot clustering, a replication of [4]. Section 4.2.2 introduces the specific algorithm we used for the off-line phase of the LAIR2 method to pre-compute a hierarchical structure–the on-line LAIR2 clustering has been closely discussed in Section 2. In Section 4.2.3, we present strategies used to make the two systems comparable in terms of efficiency in order not to bias the subjects' evaluations on effectiveness.

4.2.1 Classic System: Buckshot

The initial paper on the Scatter/Gather method proposed a three-phase algorithm for on-line clustering [4]: 1) Find k centers; 2) Assign each document in the collection to a center; and 3) Refine the partition so constructed. Our implementation of the classic system followed this method exactly.

Phase I: Buckshot for Finding Initial Centers

In the first phase, we used the Buckshot method, which randomly sampled $\sqrt{kn}$ documents, where k is the number of desired clusters and n the number of documents related to the user-chosen clusters. Then it applied a clustering subroutine to the sampled set and returned the centers of clusters found. This method maintained comparable quality and was much faster than the Fractionation method also proposed in the paper, desirable for on-the-fly Scatter/Gather [4]. For the clustering subroutine within the $\sqrt{kn}$ documents, we used the classic K-means algorithm with refinements. Algorithm 1 shows how basic K-means clustering works.

Algorithm 1 Basic K-means Algorithm

1: Let $\widehat{k}$ be the number of clusters desired.
2: select $\widehat{k}$ points as the initial centroids.
3: **repeat**
4: assign all points to the closest centroid.
5: recompute the centroid of each cluster.
6: **until** the centroids don't change.

Phase II: Nearest Center Assignment

When the k centers from the $\sqrt{kn}$ documents were presented, we simply assigned each document in the entire chosen set to a center that maximized the pairwise similarity. The representation of each center used the average vector values of all documents that had been assigned to it. The cost of the assigning procedure was kn.

Phase III: Partition Refinement

Now the cluster centers' quality might not be sufficiently good because it was only based on the $\sqrt{kn}$ samples. We needed to refine them further. The first strategy was to re-

peat the Nearest Center Assignment until the centers did not move further or a finite number of times was reached. The refinement improved the clustering very quickly within the first few iterations but might worsen the partition if continued excessively [4]. In our implementation, the refinement ran for five iterations.

The second refinement strategy was to split the most heterogeneous clusters while the third merged the most similar clusters. The implementation repeated five times for both split and merge. It maintained k clusters after all the steps. Please refer to [4] for more details on the method.

4.2.2 New System: LAIR2

As per discussed, the LAIR2 clustering had two phases: an off-line phase for producing a cluster hierarchy and an on-line phase for clustering using the existing hierarchy. The off-line phase used Bisecting K-means, i.e., Algorithm 2, to produce a clustering hierarchy (binary tree). See section 2.1 for details on the LAIR2 clustering algorithm, particularly the on-line clustering.

Research has shown the superior effectiveness and efficiency of the Bisecting K-mean method on several benchmark datasets [19]. The method, Algorithm 2, uses the K-means with $\widehat{k} = 2$ in a recursive hierarchical manner. It also repeats the K-means to refine the clustering quality in each divisive iteration.

Algorithm 2 Bisecting K-means Algorithm

1: Let k be the number of clusters desired
2: Let all documents belong to one cluster initially
3: **repeat**
4: pick the largest cluster to split
5: **repeat**
6: split the cluster into two using the basic K-means algorithm. (Bisecting step, $\widehat{k} = 2$)
7: **until** the predefined m times, e.g. $m = 5$
8: take the split that produces the clustering with the highest overall similarity.
9: **until** the desired number of clusters k is reached.

4.2.3 Strategies to "Blur" Efficiency Difference

Our previous research found that the LAIR2 algorithm was much faster than the parallelized Buckshot algorithm running on multiple processors [10, 11]. In this research, the focus is on the clustering effectiveness of the LAIR2 algorithm, i.e. the quality of the produced clusters and how well they helped the users find relevant documents. However, the presence of efficiency difference might have biased the users' perception of effectiveness. The following strategies were used to blur the efficiency difference.

The efficiency difference was supposedly huge particularly when the top level clusters were to be produced, which involved the entire collection of documents. For Buckshot clustering on the top levels, we used a caching mechanism to reuse the results. Before the user study, the researchers created automatic scripts to run through all possible combinations of top-three-level user selections and cached the clustering results.

Note that in order to limit the number of all possible selections, we purposely fixed the number of desired clusters in each Scatter/Gather session to seven. In the actual study, the Buckshot system worked comparably efficient using the cache and the majority of subjects (19/24) were not aware of a system difference when asked during the exit debriefing.

5. EXPERIMENTAL DESIGN

The purpose of this study was to determine if the LAIR2 algorithm could produce better clusters than the classic Buckshot algorithm and to determine if users were more satisfied with the results from the new algorithm. We argued that, besides the huge efficiency gain, the proposed LAIR2 algorithm is at least equally effective as compared to the classic Buckshot algorithm. That is, we hypothesized that $LAIR2 \geq Buckshot$ in terms of the three levels of clustering effectiveness as stated in the research questions (Section 3).

5.1 Methodology

The methodology employed for this study was a within-subject study of 24 undergraduate students. Each subject performed two of the four tasks (see Section 5.3) on the LAIR2 system while the other two on the Buckshot method. Combinations of task order and system order were created using a Latin-square rotation and randomly assigned to the subjects, which enabled three complete cycles of system-task rotations. Through the system-task rotation, we made sure that, for each task, half of the subjects completed with the LAIR2 system and half with Buckshot. In addition, to avoid potential learning effects, half of the subjects searched on the LAIR2 system first while the other Buckshot first. Subjects were unaware of which algorithm they were using and additionally unaware of the fact that they were comparing two systems.

5.2 Subjects

Twenty four subjects were recruited by means of a mass email sent to all undergraduate students at a single institution and offered $15 in exchange for one and a half hours of their time. Once they arrived at the study and before conducting any search tasks, the subjects were asked to complete a demographic questionnaire which identified their age, sex, year in school, major area of study, computer experience, web searching experience, and the frequency with which they read news articles.

The average age of the subjects in this study was 20 years of age, with 50% of the subjects within the first two years of their undergraduate career and the other 50% in the their third and fourth years of study. Seventeen (71%) of the subjects were female and seven (29%) of the subjects were male. The subjects represented a variety of majors (17 different areas of study), with the majority of subjects (4) studying journalism. The majority of the subjects indicated that they were experienced with computers (91.6% indicated very or fairly experienced) and web searching (87.5% indicated daily searching; 12.5% indicated weekly searching). They also indicated high levels of frequency with reading news articles (62.5% reporting reading news articles daily; 33.3% weekly).

5.3 Document Collection and Tasks

The collection used for this study was a modified version of the TREC 2005 High Accuracy Retrieval from Documents (HARD) track–a corpus of news documents from 1996-2000 [1]. From the original corpus of documents, we removed all those documents which had no relevance assessments. The remaining subset was comprised of 33,660 documents

containing 6,479 unique relevant documents with 6,561 relevance records for the 50 search topics/tasks (some documents were judged relevant to more than one topic). In order to index this corpus, we removed 136,226 unique terms: including stopwords (742 terms), frequent terms (document frequency $\geq$ 3,000), rare terms (document frequency $\leq$ 30), and all terms which began with a number. This left 20,167 unique terms for indexing/document representation.

Four tasks (search topics) were chosen from the TREC 2005 HARD track. These tasks were chosen for their diversity, their accessibility to a large audience, and because they each had at least 100 relevant documents within the collection (a total of 818 relevant records in the 33,660 documents). The selected topics were: Ireland Peace talks (topic 404), cult lifestyles (topic 325), teenage pregnancy (topic 658), and abuses of e-mail (topic 344).

5.4 Procedure

The researcher and subjects met one-on-one in a research lab located in a university building. Subjects were offered consent forms immediately upon arrival. After agreeing to participate and filling out a demographic questionnaire, they were shown a 10 minute video tutorial (using TREC topic 383: drugs for mental illness). Following the tutorial, subjects were issued one of the four assigned topics and were allowed up to 15 minutes to search on each task. After each task, they were asked to evaluate the system using a post-task questionnaire, shown in Figure 5.

Q1: How did the clustering affect your ability to complete the task? (1 more difficult ... 7 made it easier)
Q2: How satisfied were you with the results you received? (1 not at all satisfied ... 7 very satisfied)
Q3: How satisfied were you with the new clusters/topics after each time you clicked the Gather & Scatter button? (1 not at all satisfied ... 7 very satisfied)
Q4: How much effort did it take to complete this task? (7 a lot of effort ... 1 very little effort)
Q5: How appropriate were the documents that the system provided for the clusters you selected? (1 not at all appropriate ... 7 very appropriate)
Q6: How confident were you of your ability to make the system work to accomplish the assigned task? (1 not at all confident ... 7 very confident)
Q7: How effective did you feel the system was in helping you complete this task? (1 not at all effective ... 7 very effective)
Q8: How familiar were you with the topic? (1 not at all familiar ... 7 very familiar)

Figure 5: Post-Task Questions

Following completion of the four tasks and four post-task questionnaires, subjects were asked to fill out a final open-ended questionnaire, which was used to assess their overall satisfaction with the Scatter/Gather interface.

6. RESULTS

6.1 Post-Task Questionnaire

The post-task questions examined the subjects' satisfaction and perception of clustering effectiveness during the search tasks. Table 2 shows the mean values and difference of the two systems in terms of each question. T-tests of the differences and probability values are shown in the last two columns.

Note that Q8 was about a subject's familiarity with each search topic. We controlled for topic familiarity and did not expect any difference in Q8 so as not to bias the evaluations on clustering effectiveness. We found significant differences at the 0.05 level in Q3 (satisfaction with the produced clusters), Q4 (effort), and Q7 (system effectiveness). The LAIR2 system is significantly better than the Buckshot system in terms of these three questions. Note that for Q4 (effort), a lower value means better system effectiveness that demanded less effort. Although the mean values of LAIR2 are all higher than Buckshot in terms of the other questions, no significant differences were found (Q2 and Q5 were significant at the 0.1 level). A comparison of the mean values (with error bars indicating standard errors $\pm\sigma$ of the differences) are plotted in Figure 6. We will interpret and discuss these results further in Section 7.

PQ	Buckshot	LAIR2	Difference	t	Pr	
Q1	4.33	4.81	0.48	1.53	0.130	
Q2	4.21	4.83	0.63	1.83	0.071	.
Q3	4.13	4.77	0.64	2.22	0.029	*
Q4	4.17	3.52	-0.65	-2.18	0.032	*
Q5	4.56	5.06	0.50	1.76	0.082	.
Q6	4.65	4.92	0.27	0.90	0.368	
Q7	4.38	5.00	0.63	2.16	0.033	*
Q8	3.90	3.75	-0.15	-0.44	0.665	
Signif. codes: '*' p<.05, '.' p<0.1						

Table 2: Results: Post-Task Questionnaire

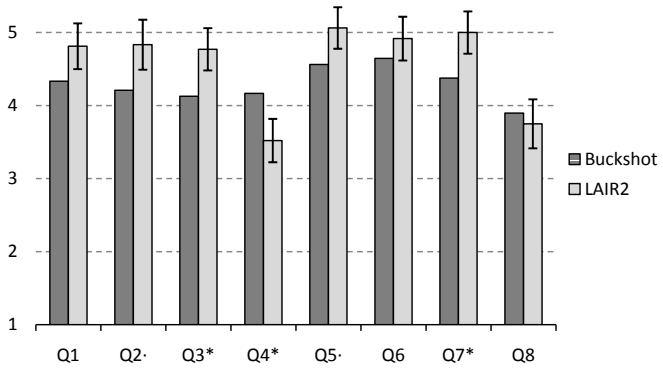


Figure 6: Comparison for each post-task question (Signif. codes: '*' p<.05, '.' p<0.1)

6.2 Retrieval Effectiveness

In each task, we recorded the subject's retrieved documents and compared them to the relevance base. The result in Table 3 shows the mean values and system differences in terms of precision, recall, and F_1 measures [13]. Although the mean values of the LAIR2 system were all higher (better) than the Buckshot system, there was no significant difference.

6.3 Final Questionnaire

The final questionnaire was used to collect general information about the subjects' experiences with the interface, regardless of the underlying system differences. Four questions were asked in the final questionnaire:

1. What did you like about the system?

Measure	Buckshot	LAIR2	Diff	t	Pr
Precision	0.603	0.651	0.047	0.695	0.489
Recall	0.061	0.082	0.012	1.373	0.173
F_1	0.172	0.223	0.021	1.367	0.175
Signif. codes: '*' p<.05, '.' p<0.1					

Table 3: Results: Retrieval Effectiveness

2. How could this system be improved?

3. How did you like the clustering element of the system? Did it improve or impede your searching?

4. Have you ever used a searching tool that used visualizations, such as the one you just used? If so, please either name or describe the tool which you used.

In response to question one, the subjects indicated that they favored the ease of use, system design, and narrowing functionality of the interface. Many subjects remarked that the system had a short learning curve. They also appreciated the ability to see all functions (searching, saving, clustering) on one screen and not have to navigate between multiple screens. Finally, they enjoyed the narrowing functionality of the system, noting they liked the ability to quickly eliminate large amounts of unrelated material.

In terms of improvements, a large number of subjects indicated that they would have liked a hybrid approach-in which they could use keyword searching to narrow down the initial clusters and then proceed with the *Gather & Scatter* function after the initial search. Some felt that the initial groupings were vague and other clusters repetitive or indistinguishable. They requested more clusters within each step with a greater degree of *specificity*.

When asked whether the system improved or impeded their searching, the subjects presented mixed responses, in which they indicated that topic *specificity* and topic *familiarity* dictated to a large degree whether the system hindered or helped their searching. Most respondents agreed that it was easier to search *broad* topics within the system, while *specific* topics were more difficult. They also felt that in the cases where they knew a lot about the topic, the system slowed them down. The comments from the subjects imply that the system was best suited for *exploratory* searching and learning [4, 17], rather than fact finding.

In the last question, subjects were asked whether or not they had used visualization search systems. All subjects replied that they had never used a search system similar to this one.

7. DISCUSSION

In the user study, we controlled the order of the two systems to be used and the four tasks to be performed by the subjects–they had equal opportunities to appear first/next in the user tasks. From the results in Table 2, we also find no significant difference in terms of topic familiarity (post-task Q8). That is, the tasks performed on the two systems were equally familiar to the subjects. This ensured that the evaluations were not biased by potential learning effects nor by the tasks/topics.

The significant difference of post-task Q3 indicates that subjects were more satisfied with clusters produced by the LAIR2 system after they clicked the Gather & Scatter button. It implies that LAIR2 produced better quality clusters.

At the task-oriented usability/utility level, significant differences were also found. The subjects indicated that it took less effort to complete the tasks using the LAIR2 system (post-task Q4) and they felt that the LAIR2 system was more effective in helping them complete the tasks (post-task Q5).

In terms of objective effectiveness measures, there was no significant difference between the two systems (see precision, recall, and F_1 in Table 3). At the system level, the subjects felt equally confident to make the systems work to complete the assigned tasks. It appears that, although clusters produced by the LAIR2 algorithm were of better quality and subjects perceived them as more helpful in the searches, they did not significantly improve the retrieval results.

Many subjects indicated in the final questionnaire that when they Gathered & Scattered to a very focused subset, the clusters became less differentiable and the system(s) impeded the searches. It also impeded the searches when the subjects were familiar with the topics to be searched. In these cases, they indicated that a traditional query-based search was desirable, or at least a hybrid approach in which the initial searching was done via keyword, and the refining done by means of Scatter/Gather [17]. This reinforces previous studies of hybrid approaches [9] and encourages more investigation in this area. It also suggests future research investigate the potential usefulness of selection-biased (or query specific) automatic cluster summarization [20].

Many of the comments from the subjects indicated that the perceived usefulness of such a system would be in exploratory searching, or searching where it is difficult to precisely specify a query, reinforcing previous literature on Scatter/Gather browsing [4, 9, 17]. Several subjects quit searching when they felt they found sufficient documents–the collection of relevant documents is larger than they expected. This supports previous findings that Scatter/Gather is effective in exploratory search and involves time [17]. It also explains why the retrieval effectiveness results in terms of recall were low with both systems (Table 3).

The fact that clusters became less distinguishable within a focused subset reiterates the importance of *local reclustering* for potential *dynamicity*. Document/cluster representation using *global* (or collection-wide) properties did not produce sufficiently discriminative features for the local subset (or subspace). Again, we argue that in order to produce more dynamic and differentiable clusters in a subspace, methods that take advantage of user selection information to reweigh local features are desirable. Future research on this will be worthwhile.

8. CONCLUSION

The results of this study provide evidence on the competency of the LAIR2 algorithm. The user evaluations supported that, as compared to the Buckshot algorithm (a classic algorithm for Scatter/Gather), the proposed LAIR2 algorithm produced significantly better quality clusters from the subjects' perspective. Subjects also indicated that it took less effort to complete the tasks on the LAIR2 system and they felt the LAIR2 system was more effective in helping them in the tasks.

In terms of overall retrieval effectiveness, i.e., the objective measures of precision and recall, we found no significant differences even though the mean values of LAIR2 were all better and some of them were significant at the 0.1 level. In other words, the LAIR2 algorithm was at least equally ef-

fective, if not better, in facilitating the retrieval of the news articles.

Provided the fact that the on-line LAIR2 method is far more efficient than the classic algorithms, its improved clustering quality and comparable retrieval effectiveness, as revealed in the experiments, will make this work a significant contribution to the whole body of research on Scatter/Gather. This will enable real-world systems to provide responsive Scatter/Gather services to on-line concurrent users.

As the comments of the subjects suggested, we need to investigate what made clusters within a focused subset less differentiable and how a system can improve cluster/document representation by utilizing the new information based on user selections. Continued research on *local reclustering* and the impact of *dynamicity* on clustering *effectiveness* will be worthwhile.

Acknowledgments

We appreciate valuable comments and help from Diane Kelly, Kiduk Yang, and Jane Greenberg. We also thank Alex Berry and Sujit Gadkari for their collaboration on the Scatter/Gather browser implementation. This work was partially supported by the NSF Grant 0333623.

9. REFERENCES

[1] J. Allan. Hard track overview in TREC 2005: High accuracy retrieval from documents. In *TREC '05: Proceedings of the Text REtrieval Conference*, 2005.

[2] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison Wesley Longman publishing, 2004.

[3] D. B. Crouch, C. J. Crouch, and G. Andreas. The use of cluster hierarchies in hypertext information retrieval. In *HYPERTEXT '89: Proceedings of the second annual ACM conference on Hypertext*, pages 225–237, New York, NY, USA, 1989. ACM Press.

[4] D. R. Cutting, D. Karger, J. O. Pedersen, and J. W. Tukey. Scatter/Gather: A cluster-based approach to browsing large document collections. In *The 15th Annual ACM SIGIR*, pages 318–329, 1992.

[5] D. R. Cutting, D. R. Karger, and J. O. Pedersen. Constant interaction-time Scatter/Gather browsing of very large document collections. In *SIGIR '93: Proceedings of the 16th annual international ACM SIGIR conference on research and development in information retrieval*, pages 126–134, New York, NY, USA, 1993. ACM Press.

[6] J. Han, M. Kamber, and A. L. H. Tung. *Spatial Clustering methods in data mining: a survey*. New York, 2001.

[7] M. A. Hearst. *Modern Information Retrieval*, chapter 10. Addison-Wesley Longman Publishing, 2004.

[8] M. A. Hearst, D. R. Karger, and J. O. Pedersen. Scatter/Gather as a tool for the navigation of retrieval results. In *Working Notes AAAI Fall Symp. AI Applications in Knowledge Navigation*, 1995.

[9] M. A. Hearst and J. O. Pedersen. Reexamining the cluster hypothesis: Scatter/Gather on retrieval results. In *SIGIR '96: Proceedings of the 19th annual international ACM SIGIR conference on research and development in information retrieval*, pages 76–84, New York, NY, USA, 1996. ACM Press.

[10] E. C. Jensen, S. M. Beitzel, A. J. Pilotto, N. Goharian, and O. Frieder. Parallelizing the Buckshot algorithm for efficient document clustering. In *CIKM '02: Proceedings of the eleventh international conference on Information and knowledge management*, pages 684–686, New York, NY, USA, 2002. ACM Press.

[11] W. Ke, J. Mostafa, and Y. Liu. Toward responsive visualization services for Scatter/Gather browsing. In *ASIS&T '08: Proceedings of the annual meeting of the American Society for Information Science and Technology 2008*, 2008.

[12] A. J. Kleiboemer, M. B. Lazear, and J. O. Pedersen. Tailoring a retrieval system for naive users. In *Proceedings of the Fifth Annual Symposium on Document Analysis and Information Retrieval (SDAIR)*, Las Vegas, NV, 1996.

[13] D. D. Lewis. Evaluating and optimizing autonomous text classification systems. In *Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 246–254, 1995.

[14] T. Li, S. Ma, and M. Ogihara. Document clustering via adaptive subspace iteration. In *SIGIR '04: Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 218–225, New York, NY, USA, 2004. ACM.

[15] T. Liu, S. Liu, Z. Cheng, and W.-Y. Ma. An evaluation on feature selection for text clustering. In *Proceedings of the Twentieth International Conference on Machine Learning (ICML-2003)*, Washington DC, 2003.

[16] P. Ogilvie and J. P. Callan. Experiments using the lemur toolkit. In *Text REtrieval Conference (TREC)*, 2001.

[17] P. Pirolli, P. Schank, M. Hearst, and C. Diehl. Scatter/Gather browsing communicates the topic structure of a very large text collection. In *CHI '96: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 213–220, New York, NY, USA, 1996. ACM.

[18] N. Sahoo, J. Callan, R. Krishnan, G. Duncan, and R. Padman. Incremental hierarchical clustering of text documents. In *CIKM '06: Proceedings of the 15th ACM international conference on Information and knowledge management*, pages 357–366, New York, NY, USA, 2006. ACM.

[19] M. Steinbach, G. Karypis, and V. Kumar. A comparison of document clustering techniques. In *KDD Workshop on Text Mining*, 2000.

[20] A. Tombros, R. Villa, and C. J. V. Rijsbergen. The effectiveness of query-specific hierarchic clustering in information retrieval. *Inf. Process. Manage.*, 38(4):559–582, 2002.

[21] Y. Yang and J. Pedersen. A comparative study on feature selection in text categorization. In *Proc. of the 14th International Conference on Machine Learning ICML97*, pages 412–420, 1997.