Q C S:
An Information Retrieval System for Improving Efficiency in Scientific Literature Searches

Final Report for Version 1.0

Daniel M. Dunlavy

May 16, 2003

Acknowledgements

I would like thank Dianne O'Leary, my thesis advisor at the University of Maryland and co-advisor for the QCS project, for all of her invaluable help throughout the entire development of the QCS system. I would also like to thank John Conroy, an applied mathematician at the Center for Computing Sciences and co-advisor for the QCS project, for giving me the idea to develop the QCS system and listening to all my ideas of how I wanted to change everything originally proposed.

I would also like to thank David Levermore and Bill Dorland, the instructors and project leaders of the Advanced Scientific Computation course at the University of Maryland, College Park, for which the QCS system was originally developed. Their questions about and ideas for the implementation of the QCS system proved very helpful. Input from my fellow students in the course, Jim Cooley, Darran Furnival, and John Harlim has also helped contribute to the development of the QCS system.

Finally, I would like to thank my wife Nancy and daughter Maisy for their patience and support throughout every minute of the development of the QCS system. This project is dedicated to them as they are so dedicated to helping me succeed.

Contents

List of Tables

Abstract

Conducting scientific research most often involves a search through existing literature in order to avoid repeating research efforts, review methods already developed for solving a problem, gain a better understanding of a problem, etc. Typically, this search is performed using the Internet, which is a convenient portal to various databases of books, journal articles, technical reports, preprints, etc.

The Query, Cluster, Summarize (QCS) information retrieval system is presented in an attempt to improve efficiency in these literature searches. Given a query, QCS retrieve documents relevant to the query, separates the retrieved documents into topic clusters, and creates a single summary for each of topic clusters. Latent Semantic Indexing is used retrieval, generalized spherical k-means (gmeans) is used for the document clustering, and a hidden Markov model coupled with a pivoted QR decomposition is used to create a single extract summary for each topic cluster.

Algorithm and implementation details of the current version of the QCS system, QCS v1.0, are presented, and a description of the user interface to the system is discussed.

Examples of the use of QCS v1.0 are presented using data from the Document Understanding Conferences, a conference series dedicated to furthering progress in the area of automatic summarization.

Chapter 1

Introduction

This report details the development of and completion of the initial version of the Query, Cluster, Summarize (QCS) information retrieval (IR) system, QCS v1.0. In the first section, background material and motivation of the problem that the QCS system attempts to solve, along with a brief description of the originally proposed QCS system, is presented. (For a more detailed account of the proposal, an interested reader is directed to the original proposal [7].) Chapter 2 gives the details of the algorithms implemented in QCS v1.0. Implementation issues are discussed in Chapter 3, highlighting work completed to date on the QCS system, a user interface to the system, and accompanying software tools created or utilized in facilitating the development of the QCS system. Validation of the algorithms used in the QCS system is presented in Chapter 4. Finally, possible future directions and open issues are discussed in Chapter 5.

As with many software projects of this scale, some of the ideas originally proposed for the QCS system in [7] have been phased out for various reasons. The specific changes and the rationale behind these changes are discussed throughout this report in the sections concerning components of the QCS system where major changes were made.

1.1 Background

Conducting scientific research most often involves a search through existing literature in order to avoid repeating research efforts, review methods already developed for solving a problem, gain a better understanding of a problem, etc. Typically, this search is performed using the Internet, which is a convenient portal to various databases of books, journal articles, technical reports, preprints, etc.

In using this approach for IR, a researcher has the advantage of being able to search through great amounts of reference material. However, along with this great access comes the challenge of efficiently retrieving and processing only relevant material. When using an IR engine to search through electronic resources, simple queries can return too many documents or documents not relevant to the intended search criteria.

1.2 Motivation

As a motivating example, consider a physicist researching which methods have been used to solve problems in the area of plasma physics. As an initial attempt at finding scientific papers on methods developed to solve problems in this area, the physicist searches the World Wide Web using Google (www.google.com), a search engine for sifting through an index of more 3 billion HTML-hyperlinked documents available. Entering the query, *methods "plasma physics"*, into the Google search engine returns more than 32,800 documents (as of the date of this report). This is simply too many documents to read and many of the documents returned contain redundant information, so the physicist reads the first dozen or so and then turns to an electronic resource of papers in the area of physics, the arXiv (xxx.lanl.gov) preprint server. The same query applied to both abstracts and the full text of the papers available at the arXiv server returns 60 papers. Although this is a much more manageable number of papers (or just abstracts) to read, the arXiv server is a small collection of author-submitted preprints and most likely does not contain all of the papers written on methods for solving problems in the area of plasma physics. This may act as a good starting point, but by no means should it suffice as an exhaustive search.

Using these two examples as motivation, it would be useful to have an IR system that could perform the following tasks:

- retrieve documents relevant to a query,
- separate the retrieved documents into clusters by topic, and
- create a single summary document for each topic cluster.

Ideally, such a system would facilitate more efficient literature searches. By categorizing and summarizing the set of documents that best match a researcher's query terms, a large amount of information can be presented which is reduced in size and organized in a categorical hierarchy. If scalable, it would be able to handle large amounts of information, similar to Google, while producing a more compact, accessible representation of the relevant documents. Also, it would address the shortcomings of arXiv while retaining its beneficial qualities – it would be able to search more extensive collections and yet produce a manageable number of results by summarizing and reducing redundant information.

1.3 The Proposed QCS System

The QCS system was proposed in [7] to solve the IR tasks presented in the previous section. QCS was designed to index a set of documents, retrieve documents relevant to a query, cluster the subset of retrieved documents, and produce a single summary for each of the clusters.

The indexing of a document set allows for efficient retrieval of documents in that numeric representations of the documents are stored for retrieval rather than the actual ASCII character bytes. The specific model for the data is called a *vector space model* and is described in Section 2.1.

The querying component of QCS takes a query, i.e., a set of words for which to search a document set, and returns a subset of documents relevant to the query. The amount of

relevance of the returned documents to the query has been parameterized in QCS to allow an individual user to control this factor.

The clustering component of QCS separates the retrieved documents into a set of clusters by topic. That is, a disjoint set of partitions of the retrieved documents is created which reflects the categorical similarities and differences between the documents. If two documents roughly cover the same material, they are placed into the same cluster. The clustering algorithm in QCS allows for an adaptive number of clusters, but a user has the ability to specify that a specific number of clusters should be returned if so desired.

Finally, the summarization component of QCS summarizes each of the documents and produces a single multi-document summary for each of the clusters.

The interface to the system described above is platform-independent and consists of dynamic HTML pages created by a Java servlet. The separation of the interface from the computational components allows development of the latter components to be optimized for a specific hardware platform without tying the entire system to that platform. This in turn allows the use of highly-tuned numeric libraries that would not be possible in fully platform-independent code. All of the implementation details are given in Chapter 3.

1.4 Data and Examples

The examples presented throughout this report consist of simple self-contained examples as well as examples of using the QCS system with a real set of documents. In the former case, all information about each example is introduced in the section preceding it. In the latter case, the examples are results of using the QCS system for IR on a set of documents used to test automatic summarization systems. Starting in 2000, a conference series, the Document Understanding Conferences (DUC), has been run by the National Institute of Standards and Technology (NIST) to "further progress in summarization and enable researchers to participate in large-scale experiments." Each year two sets of documents are released to participants of the conferences and are intended to be used to train and test summarization algorithms.

Source	*Year(s)*	*Number*
Associate Press	1989–1990	355
San Jose Mercury News	1991	92
Los Angeles Times	1989–1990	39
Foreign Broadcast Information Service	1996	39
Wall Street Journal	1987–1992	23
Financial Times	1991–1994	19
Total Number of Documents		567
Total Size of Document Set		2.6 Mb

Table 1.1: Sources of documents used in QCS examples in this report

The document set used for the examples in this report is the testing set from DUC 2002.

The documents in this set are ASCII-formatted SGML[1] files from several news agencies and newswire services. Statistics about the documents are provided in Table 1.1. This is a rather small test set, but will suffice to illustrate the use of the QCS system.

The topics of the documents in this collection are limited, making this a good set for testing the clustering and summarization algorithms used in the QCS system. There are an average of 10 documents related to particular topics or subjects under the following categories:

1. a single natural disaster event within a seven day coverage
2. any single event within a seven day coverage
3. multiple distinct events of a single type
4. biographical information about a single person

Examples topics and/or subjects for each of these areas are given in Table 1.2. Although there are only four major areas from which the topics are drawn, it is evident from the example topics that there is enough diversity within this document set to allow for many quite different queries of the data.

Area	*Topic*
1	Hurricane Gilbert pounds the Caribbean, October 1988 California earthquake, 17 October 1989 Floods in China June 1994
2	Iraq invades Kuwait 1 Aug 1990 Schoolyard shootings in Stockton, California, Jan. 1989 Tiananmen Square Revolt June 1989
3	Ship sinkings Olympic gold medal events Grievances and strikes of miners around the world
4	Leonard Bernstein Margaret Thatcher Andrei D. Sakharov

Table 1.2: Example topics of documents used in QCS examples in this report

[1]Standard Generalized Markup Language

Chapter 2

The QCS System

The QCS (Query, Cluster, Summarize) system attempts to improve efficiency in IR by combining query-based IR, categorical clustering, and multi-document summarization into a single IR tool. The model of the data used in the QCS system is a vector space model and is described in Section 2.1. The steps for processing the source documents to be used include detecting and marking the sentence boundaries, indexing the words of each document, and categorizing the sentences within a document. These processes are described in Section 2.2.

The specific computational methods proposed to carry out the tasks of querying a set of documents and returning relevant documents, clustering a set of documents by topic, and creating a summary of multiple documents are presented in Table 2.1, with details given in Sections 2.3, 2.4, and 2.5, respectively.

Task	*Method*	*Reference*
Querying	Latent Semantic Indexing (LSI)	[4]
Clustering	Generalized Spherical k-Means	[6]
Summarization	Hidden Markov Model + QR Algorithm	[16]

Table 2.1: Computational methods used in QCS v1.0.

2.1 The Vector Space Model

A set of documents can be represented using an $m \times n$ term-document matrix, A, where m is the number of terms and n is the number of documents in the set. Although a term can be defined in several ways, terms in the QCS project represent the words (white space delimited) in a document with the exception of pre-designated *stop words* (e.g. "a", "the", "in", etc.). See Appendix A for the list of stop words used in QCS v1.0.

The value of an entry of the matrix A is a product of three scaling terms:

$$a_{ij} = \tau_{ij} \cdot \gamma_i \cdot \delta_j \qquad (i = 1, ..., m; j = 1, ..., n) \tag{2.1}$$

where τ_{ij}, γ_i, and δ_j, are the *local weight*, *global weight*, and *normalization factor*, respectively. These parameters are chosen so that the value a_{ij} best represents the importance (or weight)

of term i in document j for a particular document set. The jth column of A, a_j, is commonly referred to as the *feature vector* of document j.

Example 2.1. As a simple example of a term-document matrix, consider the set of four one-line documents presented in Table 2.2.

Document	*Contents*
1	Hurricanes are described herein.
2	Particular hurricanes cause floods.
3	People probably like neither floods nor earthquakes.
4	Earthquakes are the better of the two.

Table 2.2: An example document set of one-line documents.

After removing the stop words, the remaining words are *hurricanes*, *earthquakes*, and *floods*. The *term frequency* of each term, or the number of times that each of these words appears in each document, is presented in Table 2.3.

	d_1	d_2	d_3	d_4
hurricanes	1	1	0	0
earthquakes	0	0	1	1
floods	0	1	1	0

Table 2.3: The unscaled term-document matrix for the example one-line document set.

Column d_j in the table represents document j in Table 2.2. This is an example of a term-document matrix with no scaling. □

The various scaling schemes for a term-document matrix used in QCS v1.0 are presented in Table 2.4 (Kolda and O'Leary present a similar table of schemes in [11] along with the original references for each scheme). The values f_{ij} and f_i are the number of times term i appears in document j and the number of times term i appears in the entire document collection, respectively.

The local *binary* weighting is used when it is important whether or not a term appears in a document (as in the case with a document set with very little overlap in terms across the document set), whereas the *log* weighting would be used to damp the effects of large differences in term frequencies within a single document.

The purpose of using a global weighting scheme is to reduce the weight of terms that occur frequently within a document or across several documents while giving a greater weight to terms that occur infrequently. An interested reader can follow the reference links in [11] for the theoretical development of the global weighting schemes.

Finally, the normalization factor is used to remove any bias based on document size by scaling each document feature vector (columns of the term-document matrix A) so that each has unit length in the Euclidean norm.

Scaling scheme triplet will be represented by a three letter code using the symbols in Table 2.4. This reflects references to the various schemes seen in the literature, with a single exception: *tfx* is sometimes referred to as *tf.idf*. This is typically the scheme that researches

	Local Weights (τ_{ij})	
t	Term Frequency	f_{ij}
b	Binary	$\chi(f_{ij}) = \begin{cases} 0 & f_{ij} = 0 \\ 1 & f_{ij} > 0 \end{cases}$
l	Log	$\log(f_{ij} + 1)$
	Global Weights (γ_{ij})	
x	None	1
n	Normalized	$\left(\sum_i f_{ij}^2\right)^{-1/2}$
f	Inverse Document Frequency (IDF)	$\log\left(\frac{n}{\sum_j \chi(f_{ij})}\right)$
F	IDF Squared(IDF2)	$\log\left(\frac{n}{\sum_j (\chi(f_{ij}))^2}\right)$
e	Entropy	$1 - \sum_j \frac{p_{ij}\log(p_{ij})}{\log n}$ with $p_{ij} = f_{ij}/f_i$
	Normalization (δ_{ij})	
x	None	1
n	Normalized	$\left(\sum_i (\tau_{ij}\gamma_i)^2\right)^{-1/2}$

Table 2.4: Scaling factors for a term-document matrix

use to derive performance baselines for new methods and is the scheme used in the examples of QCS v1.0 presented in this paper.

The representation of documents as described above (barring specific choices for what constitutes a term and scaling factors) is a *vector space model* of the document set and is the most widely used model for representing documents in IR.

2.2 Preprocessing the Documents

The preprocessing of the document set for use with QCS consists of converting the documents to a standardized format, detecting and marking the sentence boundaries, and categorizing sentences for use in the summarization algorithms.

All documents are converted into SGML-encoded documents. This consists of placing start and end tags around each part of the text, where each tag represents the type of information found between the start and end tags. For example, the tags <DOC> and </DOC> are placed at the beginning and end of the document to specify where the document begins and ends in the data. Currently, the SGML tagsets incorporated into QCS consists of those used by the news agencies listed in Table 1.1.

Tagging of the sentence boundaries is done primarily using tools created by the Edinburgh Language Technology Group (`http://www.ltg.ed.ac.uk/`). Specifically, various components of that group's LT TTT (v1.0) parsing system are used. These tools were chosen due to their flexibility in handling both SGML and ASCII text documents, as well as their capability in handling most of the preprocessing tasks required by the QCS system.

The main tool used for detecting and tagging the sentence boundaries is LT POS [13], a tool in the LT TTT suite. LT POS is a probabilistic part-of-speech tagger and sentence splitter based on a combination of hidden Markov and maximum entropy models. The default models, trained on the Brown corpus [10], are used in QCS v1.0.

DTD	Filename	SGML Tag	*stype*
Associated Press	ap.dtd	<TEXT>	1
		<HEAD>	0
San Jose Mercury News	sjmn.dtd	<TEXT>	1
		<LEADPARA>	1
		<CAPTION>	0
		<DESCRIPT>	0
		<HEADLINE>	0
		<MEMO>	0
Los Angeles Times	latimes.dtd	<TEXT>	1
		<HEADLINE>	0
		<SUBJECT>	0
		<GRAPHIC>	0
Foreign Broadcast Information Service	fbis.dtd	<TEXT>	1
		<TI>	0
		<H1>, . . . , <H8>	0
Wall Street Journal	wsj.dtd	<TEXT>	1
		<LP>	1
		<HL>	0
Financial Times	ft.dtd	<TEXT>	1
		<HEADLINE>	0

Table 2.5: Mapping of SGML tags to *stype* values

An important part of preprocessing the data for use in the summarization tool of QCS is determining the value of the content of each sentence base on the role of that sentence in the document. One of the benefits of using SGML-encoded documents is that one can specify such roles using SGML tags.

Using the SGML document type definition (DTD) for a document allows one to determine the set of all possible SGML tags that exist in documents of that type. Using these tag sets, it is possible to distinguish which sentences 1) are candidates for extract summaries, 2) contain key terms or phrases that would aid in creating a summary, and 3) contain no useful information for the task of summarization. To this end a new attribute was created for the SGML tag denoting a sentence boundary, <s>, in order to denote each of these three types of sentences. This new attribute is the *stype* of a sentence, and its possible values are 1, 0, and -1, corresponding to the three types of sentences described above. Table 2.5 presents the values of *stype* used for sentences embedded into the SGML tags encountered in the several types of documents currently used for testing the QCS system. Tags not shown are assigned $stype = -1$.

Choosing to embed information into the document itself instead of creating a processing module in the summarization algorithm allows for the flexibility of using the information throughout the various stages of the QCS system. Furthermore, it will allow for the expansion of the types of sentence classification without affecting the implementation of the summarization tool.

2.3 Querying Documents

2.3.1 Goals

In developing a tool for query-based IR, there were two goals for QCS: 1) to retrieve documents that are most relevant to a user-specified query, and 2) to order the results based on a measure of relevance. In classic IR systems, lexical (exact word) matching is used to match documents to a query. However, there are many drawbacks to this approach. Several are listed below.

- *Pseudonyms:* A person may be referred to by several names, and when one of those names is part of a query, an IR system should be able to retrieve documents containing any of those names. For example, Mark Twain and Samuel Clemens refer to the same individual and query about him should return documents containing either name.
- *Synonyms:* If two words essentially have the same meaning and one is part of a query, an IR system should return documents containing either word. From our motivating example in Section 1.2, the query should retrieve documents about "methods," but also "techniques" and "algorithms", since these words play the same role in that context.
- *Stemmed words:* If one of the query words is "methods", an IR system should be able to return documents containing "method" and "methodology" as well.

2.3.2 Details

The method for uncovering the association of terms and documents in QCS v1.0 is Latent Semantic Indexing (LSI). The LSI algorithm attempts to reveal implied relationships and remove term ambiguity, while preserving the most characteristic features of each document.

LSI attempts to accomplish this using the singular value decomposition (SVD) of the term-document matrix, A:

$$A = U\Sigma V^T = \sum_{i=1}^{r} \sigma_i u_i v_i^T \tag{2.2}$$

where U and V are matrices, each with columns that form orthonormal sets ($u_1, \ldots, u_n$ and $v_1, \ldots, v_n$, respectively), and Σ is a diagonal matrix with monotonically decreasing positive values ($\sigma_1, \ldots, \sigma_r$) along the diagonal. As is typical, it is assumed that $m \geq n$, i.e., that there are more terms than documents, and that $\text{rank}(A) = n$, where n is the number of unique documents in the document set. We can make this assumption for the rank, since we will only represent a single copy of each document in the term-document matrix, despite the possibility that there may be more than one copy of a document in a document set.

The truncated SVD can be used to approximate A using a rank-k matrix:

$$A \approx A_k = U_k \Sigma_k V_k^T = \sum_{i=1}^{k} \sigma_i u_i v_i^T \tag{2.3}$$

which is the *best* rank-k approximation to A in the sense that

$$\|A - A_k\| \leq \|A - B_k\| \qquad \forall B_k \text{ s.t. } \operatorname{rank}(B_k) = k \tag{2.4}$$

where $\|\cdot\|$ is any unitarily invariant norm.

The choice of k depends on the size and nature of the document set, as well as the spread of terms throughout the individual documents and the entire document set, and is the number of *derived factors* used to represent each of the documents. The columns of U_k represent the derived term vectors, and the columns of V_k represent the derived document vectors. By using the truncated SVD, we have a means of capturing the underlying semantic structure of the document set while reducing the noise and variability. Furthermore, terms occurring in similar documents (documents containing many of the same words) will be be close together in the k-dimensional factor space even though they may never appear in the same document. It is not uncommon to use values of $k \approx 100$ for document sets with more than 1000 documents using greater than 5000 terms as features.

Example 2.2. (Taken from [1].) Consider the words *car*, *automobile*, and *elephant*. The terms *car* and *automobile* are synonymous, with *elephant* being an unrelated term. If none of these terms occur in the same document, then in a search for *car*, documents containing the term *automobile* have the same likelihood of being returned as documents containing the term *elephant*. However, the terms *car* and *automobile* will be close to each other in the k-dimensional factor space under the very likely assumption that they co-occur in documents with the same terms (e.g., *motor*, *engine*, *vehicle*, *model*, etc.). Therefore, using the truncated SVD, LSI increases the likelihood that documents containing the word automobile should be returned for queries containing the term *car*. □

A query can be represented in exactly the same manner as documents in the vector space model, i.e., using a query vector, $q \in \mathbb{R}^m$. A query is typically much more sparse than document vectors (contains far fewer terms than an average document) and does not necessarily use the same scaling scheme.

Once the query vector has been scaled using one of the schemes from Section 2.1, we can project the vector q onto the k-dimensional term space using U_k (from the truncated SVD), scale it with the k derived factors using Σ_k (the singular values), and measure how close the query is to each document in the k-dimensional document vector space. A vector of scores, s, can be computed by computing inner products of the projected and scaled query vector with the projected document vectors (columns of V_k^T):

$$\tilde{q} = \Sigma_k (U_k^T q) \tag{2.5}$$

$$s = \tilde{q}^T V_k^T = q^T U_k \Sigma_k V_k^T = q^T A_k \tag{2.6}$$

Typically, the query vector and the columns of A have been normalized (using the Euclidean norm), and hence, the scores in s turn out to be cosine similarity scores. That it, they

represent the cosine of the angle between the projected and scaled query and each document in the k-dimensional document vector space (a space spanned by the columns of V_k^T).

Given a fixed (but usually user-specified) matching tolerance, *tol*, document j is considered a match to the query, q, if

$$s_j = q^T (a_k)_j > tol \tag{2.7}$$

where $(a_k)_j$ is the jth column of A_k.

Example 2.3. Using the term-document matrix, A, from Example 2.1, the rank-2 and rank-1 approximations are given in Table 2.6 as A_2 and A_1, respectively. Note that the columns of A have been normalized to remove any bias due to document length.

$$\underset{A}{\begin{pmatrix} 1.00 & 0.71 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.71 & 1.00 \\ 0.00 & 0.71 & 0.71 & 0.00 \end{pmatrix}} \quad \underset{A_2}{\begin{pmatrix} 0.83 & 0.82 & 0.12 & -0.17 \\ -0.17 & 0.12 & 0.82 & 0.83 \\ 0.33 & 0.47 & 0.47 & 0.33 \end{pmatrix}} \quad \underset{A_1}{\begin{pmatrix} 0.33 & 0.47 & 0.47 & 0.33 \\ 0.33 & 0.47 & 0.47 & 0.33 \\ 0.33 & 0.47 & 0.47 & 0.33 \end{pmatrix}}$$

Table 2.6: Example term-document matrix and low-rank approximations

Now, if we were searching for documents containing the word "hurricanes", the query would be

$$q = (1, 0, 0)^T \ ,$$

and the computed cosine similarity scores would be

$$\begin{aligned} q^T A &= \begin{pmatrix} 1.00 & 0.71 & 0.00 & 0.00 \end{pmatrix} \ , \\ q^T A_2 &= \begin{pmatrix} 0.83 & 0.83 & 0.12 & -0.17 \end{pmatrix} \ , \\ q^T A_1 &= \begin{pmatrix} 0.33 & 0.47 & 0.47 & 0.33 \end{pmatrix} \ . \end{aligned}$$

Using the original matrix A is equivalent to performing exact matching of the query terms to the documents. With this approach, there is no way to determine the relevance of documents that do not contain any of the query terms, such as documents 3 and 4.

The rank-2 approximation gives a different set of relevance scores – documents 1 and 2 still are the most relevant, but now document 3 also has some relevance to the query. The reason for this is that document 3 contains the word "floods" which occurs in document 2 along with the query word "hurricanes". The negative score of document 4 is mapped back to 0, as are all negative scores when using these scores as a measure of the relevance.

The rank-1 approximation does not contain enough information to adequately represent the underlying semantic structure. Documents 1 and 2 are no longer the most relevant documents, as they should be; document 3, which may or may not be relevant to the query, is scored as one of the highest; and document 4, which has no apparent relevance to the query is given equal weight as a document with one of the query terms in it (document 1). Clearly, this is not a good approximation for computing similarity scores. □

Although it is very simple, Example 2.3 shows that if the rank of the approximation is too high LSI too closely resembles exact matching, and if it is too low LSI blurs the terms too much to produce good results. For each new document set, the most appropriate low-rank approximation may take some time to discover.

2.3.3 Example

The DUC 2002 document set was preprocessed and indexed, giving 7767 unique terms across the 567 documents. Since there are several documents that contain information about natural disasters, the query "hurricane earthquake" was run through the querying tool of QCS. The top 10 scoring documents are presented in Table 2.7 with the LSI score and the subject line of the document. Appendix B presents the top scoring documents with a positive score. These subject lines are the first sentence from each document that has *stype* = 0 (see the description of *stype* in Section 2.2 for more details of which tags contain "subject lines".)

Score	*Subject Line*
.90	Hurricane Latest in String of Disasters to Hit Historic City
.85	Hurricane Forecasters Carry On Amid Chaos
.85	Forecasting Aided By Supercomputers, But Still An Uncertain Science
.84	Killer Storm Hits South Carolina Coast
.83	Scientists: Warming Trends Could Mean Fiercer Hurricanes
.82	City Sends Money To Charleston In Repayment Of 211-year-old Debt
.82	150,000 Take Off As Hugo Takes Aim At Ga., Carolina
.82	Loss Of Life Low Because People Were Prepared
.81	Hurricane Gilbert Heading for Jamaica With 100 MPH Winds
.80	Gilbert: Third Force 5 Hurricane This Century

Table 2.7: Query results with query, "hurricane earthquake", on DUC 2002 documents

Clearly, LSI has found several documents about hurricanes. However, some of the subject lines are quite ambiguous and would be difficult to categorize the documents on the basis of these alone. If a query tool using LSI were performed with no other processing of the data, and the subject lines were returned with pointers to the original documents (as is typically the case with query tools), a user would still have many articles to read and no idea whether or not the high-ranking documents contained redundant information.

This leads the the next tool in the QCS system – clustering.

2.4 Clustering Documents

2.4.1 Goals

In developing a clustering tool to partition a set of documents matching a query, there were several goals for QCS: 1) to cluster the documents so that the documents in each cluster are related by topic, 2) to allow for a variable number of clusters, and 3) to use information gathered and/or produced by the query tool of QCS.

2.4.2 Details

The method for clustering the documents used in QCS v1.0 is called generalized spherical k-means, (gmeans) [6]. A presentation of the problem of clustering is presented here, followed by details of the gmeans algorithm.

Consider a set of N documents returned from a query tool, where $N \leq n$, the number of documents in the entire collection queried. We would like to find a partition of those N documents such that the documents in each partition all pertain to a single topic (in the sense that the terms are highly correlated across the documents within a given partition). Furthermore, we would like each partition to correspond to a *different* topic.

In general, solving this problem could require N partitions (or disjoint sets). However, the documents that we are considering are assumed to all have a very similar underlying semantic structure in some k-dimensional factor space, and so it is likely that we may indeed require far fewer than N partitions to cluster the documents by topic. In order to discern if there is any topic similarity amongst the N documents, we return to the original m-dimensional vector space, where m is the total number of terms used to create the term-document matrix A. Thus, we return to the original matrix A to extract the columns corresponding to the N documents of interest.

A clustering of these N documents $d_1, \ldots, d_N$ is a partitioning into k disjoint subsets, $\pi_1, \ldots, \pi_k$. That is,

$$\bigcup_{j=1}^{k} \pi_j = \{d_1, \ldots, d_k\} \qquad \pi_j \cap \pi_l = \emptyset, \quad j \neq l \tag{2.8}$$

Based on cosine similarity (assuming that the extracted columns of A have been normalized with respect to the Euclidean norm), the *coherence* of the cluster π_j can be defined as

$$\sum_{d_i \in \pi_j} d_i^T c_j, \tag{2.9}$$

where c_j is the normalized centroid of cluster π_j:

$$c_j = \frac{\sum_{d_i \in \pi_j} d_i}{\| \sum_{d_i \in \pi_j} d_i \|} \tag{2.10}$$

Aggregating over all of the clusters, we can define the following combined coherence function:

$$\mathfrak{C}\left(\{\pi_j\}_{j=1}^{k}\right) = \sum_{j=1}^{k} \sum_{d_i \in \pi_j} d_i^T c_j \tag{2.11}$$

Now we would like to maximize this function to give us a partitioning with the optimal coherence. That is, since the inner product gives a score relating how close two unit vectors are in the m-dimensional term space, we are trying to partition the documents such that the normalized centroids of all the clusters are a minimal distance (in a average sense) away

from each document in the corresponding cluster. This is one of the classical approaches to k-means clustering. (There is an alternative approach of minimizing the radii of the clusters. The two problems can be shown to be equivalent.)

The gmeans algorithm is an iterative method for maximizing the combined coherence function in (2.11). The algorithm is described succinctly in [5] and detailed in [6].

One advantage of using the gmeans algorithm is that it employs an efficient strategy for computing the feature vector similarities (distances), which is the computational bottleneck of classical k-means algorithms. At each iteration, upper bounds for the dot products in (2.9) and the change in the centroid are stored for each cluster. For example, at iteration t, the stored values are

$$\max_{d_i \in \pi_j} d_i^T c_l^{(t-1)} \qquad l = 1, 2, \ldots, k \tag{2.12}$$

$$\|c_l^{(t)} - c_l^{(t-1)}\| \qquad l = 1, 2, \ldots, k \tag{2.13}$$

Since the vectors have all been normalized to unit length, we have the following relation for each document vector d in cluster l:

$$|d^T c_l^{(t)} - d^T c_l^{(t-1)}| \leq \|d\| \, \|c_l^{(t)} - c_l^{(t-1)}\| \leq \|c_l^{(t)} - c_l^{(t-1)}\| \tag{2.14}$$

which implies

$$d^T c_l^{(t-1)} - \|c_l^{(t)} - c_l^{(t-1)}\| \leq d^T c_l^{(t)} \leq d^T c_l^{(t-1)} + \|c_l^{(t)} - c_l^{(t-1)}\| \, , \tag{2.15}$$

which gives a similarity upper bound for $d^T c_l^{(t)}$ that can be used during iteration t. This means for all documents in cluster j, if the dot product $d^T c_j^{(t)}$, $j \neq l$, is greater than the similarity upper bound for cluster l, the dot product $d^T c_l^{(t)}$ does not have to be computed explicitly. After the first few iterations of most k-means algorithms, the clusters do not change dramatically. This means that the potential savings are great using similarity estimation. An example of the actual savings when using gmeans on a large document set can be found in [5], Figure 1.4.

The gmeans algorithm also allows for adaptive values of k, the number of clusters. An upper bound on the number of clusters must be specified so that $k \leq N$, otherwise gmeans will try to put fewer and fewer document vectors in more and more clusters. Typically, choosing k such that $k \ll N$ gives the best results. In QCS v1.0, the default value for the upper limit on the number of clusters is 10% of N.

2.4.3 Example

The results of the query from the example in Section 2.3.3 were run through the clustering tool of QCS. As the clustering can be a computational bottleneck with respect to the other components of QCS, choosing an initial partitioning based on the query results is a good idea. This can be thought of as an approximation to the desired clustering. If it is chosen well, the clusters will not change much, and the performance of the clustering tool will not dominate the computational time during a run through QCS.

The choice for the initial clustering, or seeding, in QCS v1.0 is 5 clusters, cluster i containing documents with query scores of $.2(i-1)+.01$ through $.2i$. That is, the scores of the documents in the five clusters are in the ranges .01–.20, .21–.40, . . . , .81–1.00, respectively. Preliminary results show that this is a good approximation and keeps the computational time of clustering comparable to the computational times of the other components of the QCS system.

Since the clustering algorithm allows for an adaptive number of clusters, an upper limit must be placed on the number of clusters allowed. Otherwise, the number of clusters could equal the number of documents, with each cluster containing only one document. That would certainly maximize the coherence function but would not be very helpful in achieving the goals of the QCS system.

The results of the clustering with an upper limit of clusters set to 10 are presented in Table 2.8. The clusters are sorted by mean query score.

Cluster	*Documents*	*Mean Score*
1	17	.74
2	17	.72
3	11	.44
4	5	.38
5	11	.36
6	3	.27
7	9	.21
8	4	.13
9	3	.04
10	2	.03

Table 2.8: Clustering results using query scores for initial seeding

The results show that for this example a majority of the documents are in clusters with the highest query scores. This is representative of many of the preliminary tests of the clustering tool in QCS v1.0.

To see examples of the content of the documents in these clusters, Table 2.9 presents the top 3 scoring documents in each of the top 3 clusters. As with the query example in Section 2.3.3, the query score and subject lines are shown. All of the results of the clustering for this example are presented in Appendix C.

2.5 Summarizing Documents

2.5.1 Goals

In developing a summarization tool that produces both single-document and multi-document summaries, there were several goals for QCS: 1) to create single-document summaries of all of the documents within each cluster, 2) to produce a multi-document summary for each of the clusters using the single-document summaries, and 3) to remove redundant information

Cluster	*Score*	*Subject Line*
1	.83	Scientists: Warming Trends Could Mean Fiercer Hurricanes
	.81	Hurricane Gilbert Heading For Jamaica With 100 Mph Winds
	.80	Gilbert: Third Force 5 Hurricane This Century
2	.90	Hurricane Latest In String Of Disasters To Hit Historic City
	.85	Hurricane Forecasters Carry On Amid Chaos
	.85	Forecasting Aided By Supercomputers, But Still An Uncertain Science
3	.51	A Special Session, With Speed
	.48	The Bay Area Quake; Pressure Points
	.47	The World Series; Oakland Athletics Vs. San Francisco Giants

Table 2.9: Sample of clustering results from the top 3 scoring clusters

from the final multi-document summaries. These multi-document summaries are presented to the user as the output of the QCS system.

2.5.2 Details

The method of producing the single-document summaries uses a hidden Markov model (HMM) to compute the probabilities that sentences are good summary sentences, and the method for producing multi-document summaries and removing redundancy is the pivoted QR algorithm. The summarization tool in QCS v1.0 is based on the the work by Conroy and O'Leary [3], and most closely resembles their system submitted to DUC 2003 for evaluation [8].

For each of the clusters produced using the clustering tool, the QCS system creates one multi-document extract summary. An extract summary is a set of sentences extracted from a set of documents which represents a summary of all of the documents in that set. Typically, the number of sentences in an extract summary is far fewer than the number of sentences in the original set of documents.

Creating a multi-document extract summary consists of computing the probability that each of the sentences is a summary sentence for the document in which it occurs. This probability is computed for each sentence in each of the N documents, and then the sentences are ordered in decreasing order of these probabilities (i.e., from the most likely to least likely candidates for summary sentences). A 9-state HMM, built to extract four lead sentences and supporting sentences, is used to compute these probabilities. For an introduction to the theory behind the use of HMM's, an interested reader id referred to [14].

The HMM uses features based upon terms as specified in Section 2.2. The features used for the HMM in QCS v1.0 use "signature" and "subject" terms:

- the number of signature terms, n_{sig}, in the sentence—value is $o_2(i) = \log(n_{sig} + 1)$;
- the number of subject terms, n_{subj}, in the sentence—value is $o_1(i) = \log(n_{subj} + 1)$;
- the position of the sentence in the document—built into the state-structure of the HMM.

The signature terms are the terms that are more likely to occur in the document (or document set) than in the corpus at large. To identify these terms, we use the log-likelihood statistic suggested by Dunning [9] and first used in summarization by Lin and Hovy [12]. The statistic is equivalent to a mutual information statistic and is based on a 2-by-2 contingency table of counts for each term.

The subject terms are a special subset of the signature terms. These are terms that occur in sentences with $stype = 0$, e.g., headline and subject heading sentences.

The features are normalized component-wise to have mean zero and variance one. In addition, the features for sentences with *stype* 0 and -1 are coerced to be -1, which forces these sentences to have an extremely low probability of being selected as summary sentences.

Once the probabilities that the sentences are summary sentences have been computed, a term-sentence matrix is formed in a similar fashion to the term-document matrix. The entries of this matrix are the term frequencies scaled so that the Euclidean norms of the columns (sentences) are equal to the corresponding probabilities computed above. For each cluster, the number of columns (sentences) to be included in the term-sentence matrix is determined by the length of the multi-document summaries. For a summary containing w words, enough sentences are chosen so that the total number of words in all the sentences for a cluster is at least $2w$.

The columns of the term-sentence are scaled so that their Euclidean norm is equal to the probability that the corresponding sentence is a summary sentence, i.e. equal to the HMM output score for the corresponding sentence. In order to remove redundant sentences, i.e. those sentences which basically contain the same information, a pivoted QR algorithm is used on the scaled term-sentence matrix.

The pivoted QR algorithm is a modification of the classical Gram-Schmidt orthogonalization algorithm for matrices. In the classical algorithm, the columns of a matrix are successively normalized to unit length and subtracted from the columns to the right in the matrix. When all of the columns have gone through this process, the columns of the new matrix represent an orthonormal basis for the column space of the original matrix. In the pivoted QR algorithm, instead of moving through the columns from left to right, the next column that goes through the normalization and subtraction steps is the one with the largest norm.

With respect to summarization, the initial effect of doing this is that the first column (sentence) chosen is the one with the greatest norm (highest probability of being a summary sentence), which we would like to do. Next, the components of that column are subtracted from the remaining columns (after the column is scaled to unit length). The effect of doing this is that the information contained in that first sentence is now removed from the remaining columns, thus removing any redundant information contained in the other sentences. This process is repeated using the remaining columns, starting each new cycle with the column with the largest norm after the successive subtractions have been performed. The exact number of sentences extracted depends on the size of the summary a user requests at the time of entering a query, the the process terminating once the number of words in the sentences already chosen as pivot columns is greater than or equal to w. Note that this choice for termination produces slightly larger summaries than requested.

More details on using the pivoted QR algorithm to extract non-redundant summary sentences can be found in [3].

2.5.3 Example

The results of the clustering from the example in Section 2.4.3 were run through the summarization tool of QCS.

From Table 2.9, we can guess that that cluster 1 contains documents about Hurricane Gilbert using just the information presented in the subject lines. However, it is not so easy to guess the topics for clusters 2 and 3, as there is greater variability in the contents of the subject lines.

The multi-document summaries produced for the top 3 scoring clusters are presented in Table 2.10. We can see that cluster 1 does indeed document Hurricane Gilbert, cluster 2 is about catastrophe insurance, and cluster 3 is about an earthquake in California.

The flow of the summaries in representative of the output of QCS v1.0 for the queries tested. They do not read like human-generated summaries, but the hope is that they will suffice to inform a user of the content of the documents contained in each cluster. Some of the sentences are misplaced as well, most likely due to an issue with the clustering component of QCS. For example, sentence 2 of the summary for cluster 3 refers to Iran and not California.

The examples in Sections 2.3.3, 2.4.3, and 2.5.3 present the results of the output from the entire QCS system. They are representative of the capabilities of QCS v1.0.

Cluster	*Mean Score*	*Multi-Document Summary*
1	.74	Gilbert reached Jamaica after skirting southern Puerto Rico, Haiti and the Dominican Republic. Pereira, who spoke by telephone from Mexico City, said heavy rain was falling over the peninsula and communications with Cancun and Cozumel were out. The hurricane center said Gilbert was the most intense storm on record in terms of barometric pressure. Jamaican Prime Minister Edward Seaga said late Tuesday that at least six people were killed, and an estimated 60,000 were left homeless in "the worst natural disaster in the modern history of Jamaica".
2	.72	Such increased demand for reinsurance, along with the losses the reinsurers will bear from these two disasters, are likely to spur increases in reinsurance prices that will later be translated into an overall price rise. For example, insurers may seek to limit their future exposure to catastrophes by increasing the amount of reinsurance they buy. Nationwide Insurance Co., a mutual company based in Columbus, Ohio, said Hugo "is the single largest claims disaster" it has seen in its 63-year history. Hugo could have a marginal impact on third-quarter income at Travelers Corp., Aetna Life & Casualty Insurance Co. and Chubb Corp., according to industry analysts.
3	.44	The 7.7-magnitude quake was the largest ever recorded in that area, where two major plates of the earth's crust meet, Needhams said. The shock wave traveled through the mountainous section of coastal Iran where most of the buildings are built on a flood plain of loosely deposited soil that shifts in an earthquake and allows structures to collapse, he said. The nearest metropolitan area to Tuesday's earthquake, San Jose, has seen nearly a dozen earthquakes of 5 magnitude or greater in the last 10 years, but several of them have been on the Calaveras fault, which generally runs just east of San Jose, more than 10 miles east of San Andreas at this point .

Table 2.10: Multi-document summaries for the top 3 scoring clusters

Chapter 3

Implementation

QCS v1.0 is a collection of software tools developed and tested on the SunOS 5.8 (Solaris 8) and Linux (kernel v2.4) Unix operating systems. Most of the tools are provided free of charge under the GNU General Public License (GPL) or some other licensing mechanism allowing free use for research purposes. The remaining tools have only recently been created and will be available under the GNU GPL in the near future.

QCS v1.0 has been developed as a client-server application. The implementation details of the server side are presented below and include all of the components presented to this point. The client and the interface between the client and server are presented in Section 3.5.

3.1 Preprocessing

As mentioned in Section 2.2, the preprocessing tools primarily consist of components from the LT TTT v1.0 [1] parsing software. All other tools used for preprocessing are Perl scripts.

The insertion of the *stype* attribute into the SGML sentence tags is done using a Perl module called HTML-Parser v3.27 [2] developed by Gisle Aas and Michael A. Chase. The module allows for very flexible SGML parsing using very few lines of code.

3.2 Querying Documents

3.2.1 Indexing

The indexing of the terms and documents is done using the General Text Parser (GTP)[3], developed by Michael Berry and his associates at the University of Tennessee, Knoxville. GTP also produces the SVD of the term-document matrix for use in the LSI querying tool (see below).

GTP was chosen for use in QCS v1.0 since it was a full implementation of LSI for query-based retrieval. GTP is written in C++ and is incorporated into QCS v1.0 as a static object library. Minor changes to the code were necessary to provide a consistent interface to the

[1] Available via the Edinburgh Language Technology Group – http://www.ltg.ed.ac.uk

[2] Available via the Comprehensive Perl Archive Network (CPAN) – http://www.cpan.org

[3] Available via http://www.cs.utk.edu/ lsi/soft.html

data represented in the vector space model. Specifically, the code dealing with the storage of the term-document matrix needed to be adjusted to match that used in the clustering algorithm.

Currently, the indexing is done "offline", in that it is performed one time for a static document set or during system downtime for a dynamic set. The reason for this is that the parsing and indexing outweighs the computational costs of the other components of the QCS system. Adding this computation to the system while users are accessing the system negatively affects the performance of the system.

For large dynamic sets, GTP has several choices as to how to incorporate new documents. The SVD can either be updated or recomputed, or the new documents can simply be incorporated into the current low-rank approximation of the term-document matrix. Certainly, the latter method is the least costly computationally. However, if the new documents differ in content too much from the current document set, the low-dimensional factor space may not adequately represent the underlying semantic structure of the new documents.

3.2.2 Parallel Indexing

A parallel version of GTP, PGTP, has also been incorporated into QCS v1.0. PGTP is written in C++ and uses the Message Passing Interface (MPI) library specification for implementing the code in parallel. PGTP as part of QCS v1.0 has been compiled and tested on both the SunOS and Linux operating systems using the MPICH[4] implementation of MPI developed at Argonne National Laboratory.

The part of GTP that can be efficiently performed in parallel is the computation of the SVD of the term-document matrix. The details of the parallel implementation of the SVD in PGTP can be found in [2].

Timing results for PGTP using 14 Sun Ultra10 workstations are presented in Figure 3.1. The specific times presented in the figure are the real, or wall clock, time and the user, or computational, time required to compute the SVD in parallel for the term-document matrix produced from the DUC 2002 document set. From the figure, we can see that there is definitely a speedup when using more than one processor. Specifically, the best speedup factor in real time is more than 6 (4 processors), and for user time it is more than 25 (4 processors). However, we can see that there is essentially no difference in user time when using more than 3–4 processors for the DUC 2002 data set. Moreover, the increase in real time for more than 4 processors shows that the parallel startup and communication costs start to outweigh the computational cost when using only a few processors.

Although the DUC 2002 document set is considered to be a small test set for information retrieval (567 documents and 7767 terms), results presented in [2] reflect similar behavior for PGTP run on several larger test sets (more than 130,000 documents and 270,000 terms).

3.2.3 Matching Documents to a Query

The implementation of the querying tool in QCS v1.0 is called *gtpquery* and was developed as part of the GTP system. It is implemented in C++ and was to developed to work with

[4]Available via Argonne National Laboratory – http://www-unix.mcs.anl.gov/mpi/mpich

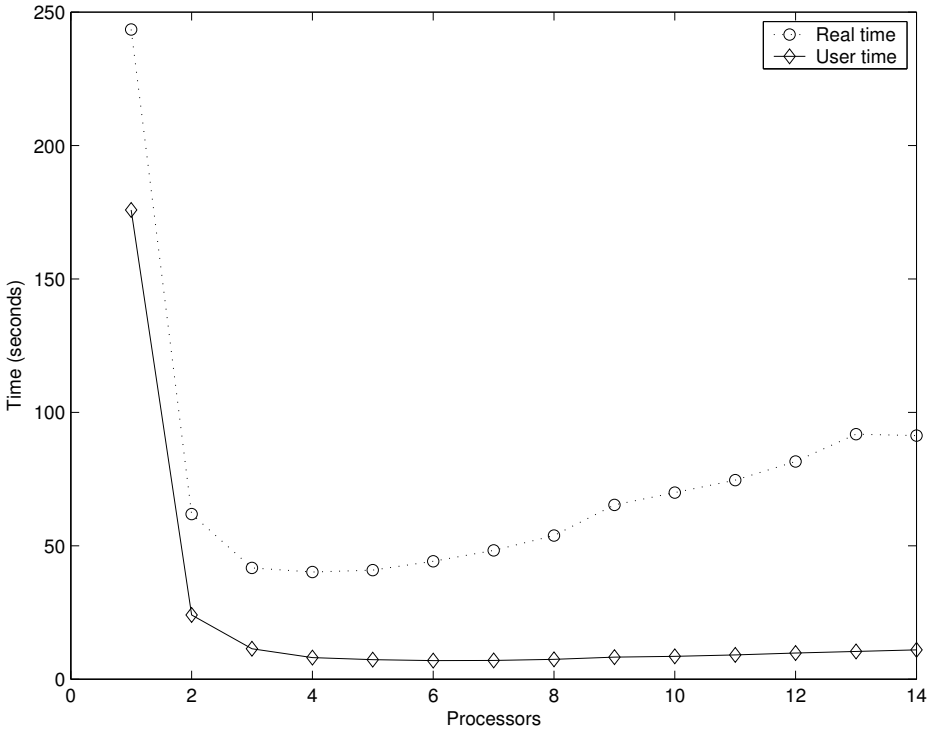

Figure 3.1: Timing results for PGTP indexing DUC 2002 on 14 Sun Ultra10 workstations

GTP, but not PGTP. An attempt was made to alter the original gtpquery code for use with PGTP so that the query could be processed in parallel as well, but the resulting tool was very inefficient compared to the serial version. Hence, the serial version of gtpquery is used in QCS v1.0.

Gtpquery parses the query in the same manner as a document is parsed during the indexing of a document set, normalizes the resulting vector, and calculates the cosine similarity scores using a low-rank approximation of the term-document matrix. A very helpful feature implemented in gtpquery is the ability to use different low-rank approximations without having to recompute the SVD. Since the components of the SVD are stored instead of the reassembled low-rank approximation, a user is able to choose the rank of the approximation to be used for each query up to the number of singular values computed during the SVD computation by GTP. If all of the singular values are stored, the user has the option of performing queries ranging from exact matches (using all of the singular values) to extremely conceptual matches (using just a few singular values). This is the approach used in QCS v1.0.

The specific documents matching a query can be chosen by either specifying the number of documents to be retrieved or a cutoff of the query score. In QCS v1.0, 100 documents are returned so as to have a large enough subset of the documents to guarantee good clustering and summarization output. The potential downside to this is that, depending on the specific query, many of the retrieved documents may have very low query scores. Since the main topics of the documents in the DUC 2002 document set are well known, however, this did

not pose a problem during testing of the QCS system with this document set.

3.3 Clustering Documents

The implementation of gmeans[5] used in QCS v1.0 was developed by Yuqiang Guan at the University of Texas, Austin. It is implemented in C++ and is incorporated into QCS v1.0 as a static object library. Only slight modifications to the original code were necessary so that the interface to the data in the vector space model matched both the query and summarization tools.

As presented in Section 2.4.3, the number of clusters ranges from 5 to 10 in QCS v1.0. Eventually, this is one of the options that will be specified by the user.

Gmeans includes several distance measures, only one of which has been tested extensively in QCS v1.0. These distance measures are spherical k-means (used in QCS v1.0), Euclidean distance, Kullback-Leibler divergence, and diametric distance. More testing on the use of these distance measures will help determine their usefulness in producing good clusters for use in summarization.

3.4 Summarizing Documents

A prototype of the HMM and pivoted QR algorithm was developed by John Conroy of the Center for Computing Sciences and Dianne O'Leary of the University of Maryland, College Park. This code was developed using Matlab, and thus needed to be converted to code that could be incorporated into QCS, i.e., compiled. Matlab provides a tool called the Matlab Compiler to perform such a conversion. The Matlab code was converted to C instead of C++, due to performance issues associated with the C++ code. The C code is incorporated into QCS v1.0 as a static object library. This C code was never modified directly throughout the development of QCS v1.0, as much of the code produced by the Matlab Compiler is difficult to follow in the opinion of this author. Instead, the Matlab code was altered and new C code was produced for any changes that needed to be made.

Of all implementations of the algorithms used in QCS v1.0, the summarization tool is certainly the fledgling code. The current implementation has been evolving over the past couple of years to reflect improvements in the summarization techniques used. Specifically, the state space of the HMM, the parameters of the HMM, and the features used to compute the HMM scores are constantly evolving (and possibly will be in the next several iterations of development of the QCS system).

3.5 The QCS Client

The "client" in QCS v1.0 consists of dynamically created HTML pages accessible through any HTML-aware browser. Using this approach seems to make the QCS system as portable as possible from the perspective of its users.

[5] Available at http://www.cs.utexas.edu/users/yguan/datamining/gmeans.html

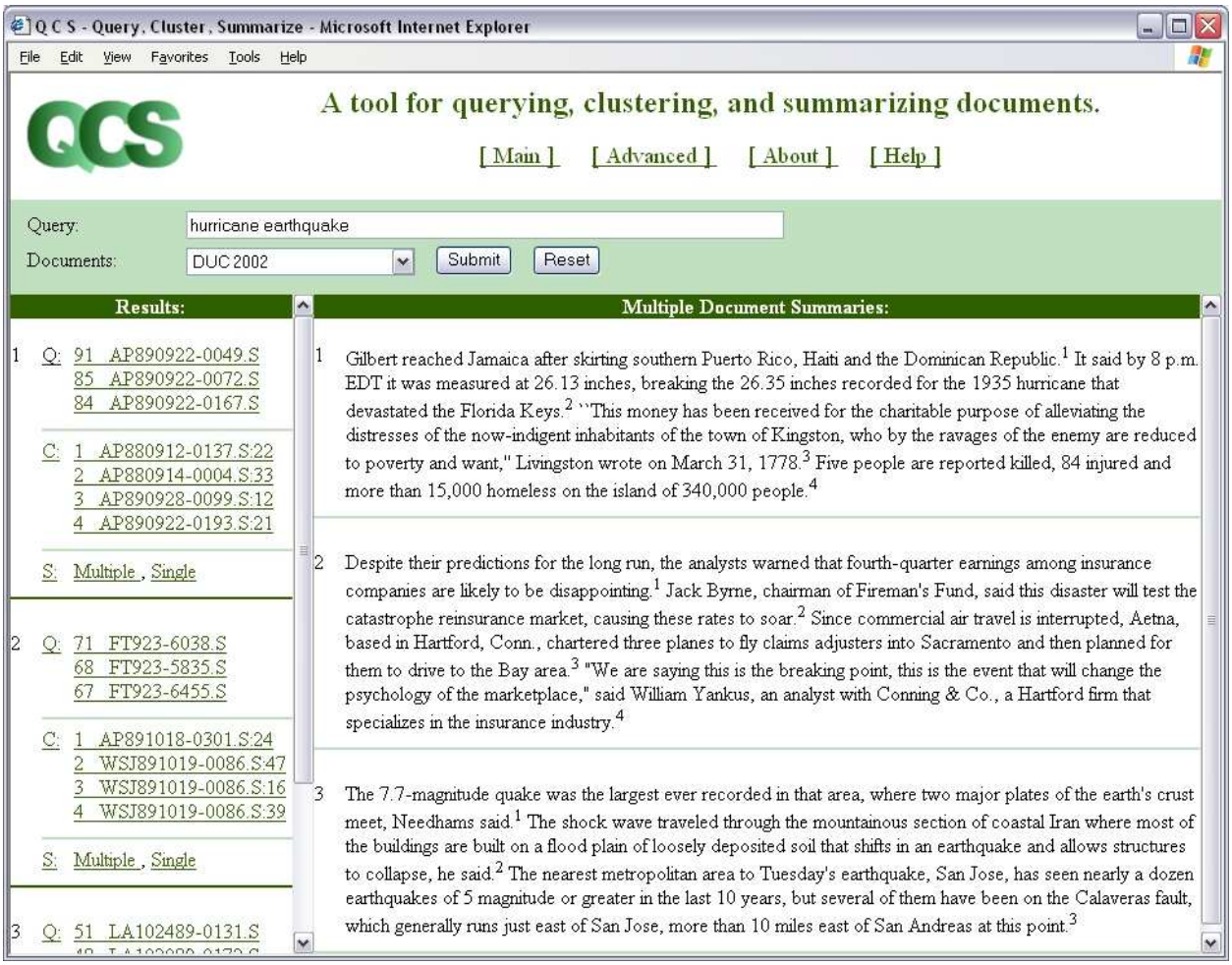

Figure 3.2: The QCS v1.0 user interface

The dynamic HTML pages are generated by Java servlets which are deployed via a Apache Tomcat Java Server (v.4.1.12). The interface between the QCS server (consisting of all of the C/C++ code) is handled using the Java Native Interface (JNI) in QCS v1.0. JNI allows Java servlets to interact with C/C++. For QCS, this allows the computationally intensive code to be developed in C/C++, which can be highly optimized on a given hardware platform, while still allowing for the greatest amount of portability for the user interface.

A screen shot of the QCS v1.0 user interface is presented in Figure 3.2. There are three main frames of the interface: the query form, the navigation bar, and the results frame. The query form contains an input field for entering a query and field for selecting the document set on which to perform the query. Currently, only the DUC 2002 document set is available for online use. The navigation bar contains links to the documents and is organized to reflect the output from the querying, clustering and summarization tools. For each cluster, query scores for the top 3 scoring documents are displayed and contain hyperlinks to the documents, links to the documents containing the sentences used in the multi-document summary are presented along with the index of the sentence within the original document,

and links to view the multi-document and single-document summaries are presented. The results frame is where all information requested through the navigation bar is presented. The default output placed in the results frame are multi-document summaries that are produced by the QCS system.

At the time of this writing, the online version of QCS v1.0 can be found at the URL

`http://stiefel.cs.umd.edu:8080/qcs/index.html`

As this is an evolving project, some of the features presented in this section may be phased out or not available at different times. Users should be aware of this when using the online version.

Chapter 4

Validation

Validation of the QCS system has proven to be a formidable task. On the one hand, validation of the implementations of the GTP/PGTP and gmeans have already been presented in [1, 2] and [5, 6], respectively. As of the dates on these cited references, these implementations have been shown to perform as well if not better than the state-of-the-art methods of query-based IR and clustering. On the other hand, very little research has focused on the combination of query-based IR and clustering, and thus no robust, systematic methods exist for validating results using this combination.

As for summarization, the DUC conferences currently provide the only means for large-scale evaluation. The evolving HMM+QR system has consistently performed as one of the best summarization systems during previous DUC evaluations [8, 15]. However, many of the participants of these conferences as well as experts in the field agree that there is much work ahead for the summarization research community before robust, systematic, automatic evaluation methods are developed for validating summarization algorithms. Currently, the output of summarization tools are evaluated at the DUC conferences by humans.

The inclusion of the full example a single query run through QCS v1.0 as presented in Sections 2.3.3, 2.4.3, and 2.5.3, can act as a form of validation for the interested reader. The incorporation of this type of subjective response into the development process of IR systems is the focus of research in the area of relevance feedback. With feedback from various users being incorporated into it, an IR system can bias its future output to reflect the users' collective measure of relevance. Although this approach lends little to improving methods for code validation, it does aid in determining which factors of an IR system may impact users the most. In this author's opinion, the purpose of code validation is to provide reproducible tests for determining the accuracy and stability of the implementation of an algorithm. To this end, as far as summarization is concerned, any method that can help an IR system progress toward reproducible user satisfaction will aid in the understanding and development of robust methods of validation.

Plans for the QCS system include adding a mechanism for capturing relevance feedback from users. However, as presented in the following section, there are several ideas for the future of the QCS system, and it is unclear which new features will aid most in achieving the system's goals. For now, more "validation" of QCS v1.0 can be performed by the reader by visiting the URL specified in Section 3.5.

Chapter 5

Future Directions

There are certainly many different directions that are possible in the future development of the QCS system. Optimization of the code through the use of more persistent variables is one direction that had to be put off until after the completion of QCS v1.0. Due to inconsistencies in how GTP, gmeans and the HHM+QR tools access the documents, the term-document matrix is not loaded as a persistent data array. Changing this would require significant modifications of the existing code. Since most of the code was not written by this author, this modification alone could require an amount of time equal to the development cycle of QCS v1.0.

An initial Application Programming Interface (API) has been developed to function as a wrapper around the various components of the QCS system. In the future, there may be more efficient or accurate means to solve the type of problems the QCS system is designed to address. Thus, creating modular code and developing a robust API to allow for future integration with other IR systems has been a major focus in the development of the QCS system. Incorporating a different algorithm for any one of the components of the QCS system would provide insight on the robustness of the modularity and scalability of the system.

5.1 Querying

Modifications to the parsing and indexing routines of the current QCS system could improve the performance of the querying tool. Specifically, explicit term stemming, query expansion, and using phrases as terms (such as full person names, company names, countries, etc. that always contain more than one word) have been shown in the IR literature to help improve query-based IR tools.

5.2 Clustering

Since only one of the distance measures is currently being used in gmeans, plans are in the works to incorporate the others for testing. A more rigorous analysis of the range for the number of starting clusters as well as for the upper limit on the number of clusters is also planned.

As a possible alternative to gmeans, a support vector machine (SVM) could be implemented to perform the categorical clustering of the query results. As there are several implementations of SVM's currently available, this could serve as a test of the modularity of the QCS system as described above.

5.3 Summarizing

The process of creating a summary document for each of the k topic clusters (where k is the number of partitions in the final iteration of the clustering algorithm) is inherently parallelizable. If the number of topic clusters is greater than the number of processors being used, then the work can be divided among the processors so that each processor has roughly the same amount of topic clusters to process. This is most likely not the optimal assignment of work in the context of load balancing, but it will be the simplest to implement and act as a good starting point for a parallel implementation of the algorithm described below.

Work has already begun in preparing the existing code for parallelization.

5.4 User Interface

The current interface to the QCS system does not allow the user to choose any of the algorithmic parameters specified throughout this report. Detailed analysis via parameter estimation techniques could highlight those parameters which most greatly affect the quality of the multi-document summaries. Once this is accomplished, fields for specifying these important parameters could be added to the user interface. As is typical with many IR tools, the plan is to include a link to a more advanced interface to include these parameter fields, so as to not hinder users that are not concerned with changing any of the parameters from their default values.

Chapter 6

Conclusions

QCS v1.0 is presented as a tool for improving the efficiency of online scientific literature searches. Certainly, the ideas presented here can easily be extended for general online searches as well.

Results of using QCS v1.0 on the DUC 2002 document set illustrate that it is possible to produce multi-document summaries of a set of documents returned by a query, and that these summaries can roughly cover the major independent topics of that set of documents.

The QCS system has been developed as a completely modular tool, so that as improvements are made in the areas of query, clustering, and summarizing documents, new methods can easily be integrated into the system. Furthermore, it has been developed as a client-server application in which the client can be run from any platform that can process HTML documents, which currently includes most major computing platforms.

The hope is that the QCS system will be useful for anyone who wants to know everything about something, but doesn't have the time to read everything ever written.

Appendix A

Stop Words Used in QCS v1.0

a	about	above	accordingly	across	after
afterwards	again	against	all	allows	almost
alone	along	already	also	although	always
am	among	amongst	an	and	another
any	anybody	anyhow	anyone	anything	anywhere
apart	appear	appropriate	are	around	as
aside	associated	at	available	away	awfully
b	back	be	became	because	become
becomes	becoming	been	before	beforehand	behind
being	below	beside	besides	best	better
between	beyond	both	brief	but	by
c	came	can	cannot	cant	cause
causes	certain	changes	co	come	consequently
contain	containing	contains	corresponding	could	currently
d	day	described	did	different	do
does	doing	done	down	downwards	during
e	each	eg	eight	either	else
elsewhere	enough	et	etc	even	ever
every	everybody	everyone	everything	everywhere	ex
example	except	f	far	few	fifth
first	five	followed	following	for	former
formerly	forth	four	from	further	furthermore
g	get	gets	given	gives	go
gone	good	got	great	h	had
hardly	has	have	having	he	hence
her	here	hereafter	hereby	herein	hereupon
hers	herself	him	himself	his	hither
how	howbeit	however	i	ie	if
ignored	immediate	in	inasmuch	inc	indeed
indicate	indicated	indicates	inner	insofar	instead
into	inward	is	it	its	itself
j	just	k	keep	kept	know
l	last	latter	latterly	least	less
lest	life	like	little	long	ltd
m	made	make	man	many	may
me	meanwhile	men	might	more	moreover
most	mostly	mr	much	must	my

myself	n	name	namely	near	necessary
neither	never	nevertheless	new	next	nine
no	nobody	none	noone	nor	normally
not	nothing	novel	now	nowhere	o
of	off	often	oh	old	on
once	one	ones	only	onto	or
other	others	otherwise	ought	our	ours
ourselves	out	outside	over	overall	own
p	particular	particularly	people	per	perhaps
placed	please	plus	possible	probably	provides
q	que	quite	r	rather	really
relatively	respectively	right	s	said	same
second	secondly	see	seem	seemed	seeming
seems	self	selves	sensible	sent	serious
seven	several	shall	she	should	since
six	so	some	somebody	somehow	someone
something	sometime	sometimes	somewhat	somewhere	specified
specify	specifying	state	still	sub	such
sup	t	take	taken	than	that
the	their	theirs	them	themselves	then
thence	there	thereafter	thereby	therefore	therein
thereupon	these	they	third	this	thorough
thoroughly	those	though	three	through	throughout
thru	thus	time	to	together	too
toward	towards	twice	two	u	under
unless	until	unto	up	upon	us
use	used	useful	uses	using	usually
v	value	various	very	via	viz
vs	w	was	way	we	well
went	were	what	whatever	when	whence
whenever	where	whereafter	whereas	whereby	wherein
whereupon	wherever	whether	which	while	whither
who	whoever	whole	whom	whose	why
will	with	within	without	work	world
would	x	y	year	years	yet
you	your	yours	yourself	yourselves	z
zero					

Appendix B

Example Query Results

Query: hurricane earthquake

Score	*Subject Line*
.90	Hurricane Latest In String Of Disasters To Hit Historic City
.85	Hurricane Forecasters Carry On Amid Chaos
.85	Forecasting Aided By Supercomputers, But Still An Uncertain Science
.84	Killer Storm Hits South Carolina Coast
.83	Scientists: Warming Trends Could Mean Fiercer Hurricanes
.82	City Sends Money To Charleston In Repayment Of 211-Year-Old Debt
.82	150,000 Take Off As Hugo Takes Aim At Ga., Carolina
.82	Loss Of Life Low Because People Were Prepared
.81	Hurricane Gilbert Heading For Jamaica With 100 Mph Winds
.80	Gilbert: Third Force 5 Hurricane This Century
.80	Markets: Hurricane Rages But Dollar Storm Blows Out
.80	Officials Ride Out Storm In Historic City Hall
.79	Hurricane Hits Jamaica With 115 Mph Winds; Communications Disrupted
.78	Gilbert Reaches Jamaican Capital With 110 Mph Winds
.77	Hugo'S Path Of Destruction
.77	Killer Hurricane Roarstoward Mexico With 175 Mph Winds
.75	Cleaning Up After Andrew
.74	After Battering Yucatan, Gilbert Strengthens, Heads For Gulf Coast
.74	Bay Area Quake
.74	What Makes Gilbert So Strong?
.73	Gulf Coast From Mexico To Louisiana Braces For Hurricane Gilbert
.73	The Bay Area Earthquake: Bush Makes Sure He Acts Speedily On Quake Crisis
.73	Floods From Hurricane Kill 10 Police In Monterrey, Overturn Buses
.72	Hurricane Gilbert Heads Toward Dominican Coast
.71	Hurricane Batters Southern Us But Lets Insurers Off Lightly
.68	Storms Batter Yucatan; Thousands Flee
.68	Ga Says Hurricane Claims Could Reach 'Up To Dollars 40M'
.67	Us Insurers Face Heaviest Hurricane Damage Claims
.67	Gilbert Slams Into Yucatan Peninsula
.63	Hurricane Damage Put At Dollars 20Bn As 2M People Told To Leave Homes
.58	Gilbert Downgraded To Tropical Storm; Floods And Tornadoes Feared
.57	Expert Suggests Planets Affect Drought
.55	The Bay Area Earthquake: Insurance Firms Expected To Pay Billions Of Dollars
.54	Relief Pours Into Bay Area
.54	Hurricane Hugo May Be Most Costly Storm Ever

Query: hurricane earthquake

Score	*Subject Line*
.51	A Special Session, With Speed
.48	Several Insurers' Net To Be Cut In Quarter By Hurricane Hugo
.48	Rushdie Donates Quake Aid; Iran Says He'S Still Doomed
.48	The Bay Area Quake; Pressure Points
.47	Hugo To Be Costliest Storm For Insurers But No Early Rate Increase Is Expected
.47	The World Series; Oakland Athletics Vs. San Francisco Giants
.47	Area Where Earthquake Hit Seen As Highly Probable In 1988 Report
.47	San Diegans Await Word On Iran Relatives
.45	County'S Iranians Offer Aid In Quake
.45	Iran Earthquake Caught Victims Asleep In Fragile Homes
.45	Iranian Media Low Key In Reporting On Killer Quake
.44	Scientists Last Year Had Pinpointed Tuesday'S Site As Likely To Be Jolted
.44	Earthquake Rocks San Francisco
.43	Iran Earthquake Caught Victims Asleep In Fragile Homes
.39	Contributions For Iranian Relief Light, Expected To Pick Up
.37	Iran Accepts U.S. Help; Earthquake Death Toll Hits 35,000
.37	While La Awaits 'The Big One,' Another Killer Quake Hits San Francisco
.37	Earthquake Rocks San Francisco
.36	Many Homeowners Not Insured; Analysts Say Disaster May Benefit Insurers
.35	Group Claims Iran Underreporting Earthquake Casualties
.35	Soviet Trucks Carry Quake Aid To Iran
.35	Tons Of Relief Supplies Arriving, But Distribution Difficult
.34	Radical Paper Blames U.S., In Part, For Quake Damage
.34	Friendly, Hostile Countries Send Aid To Iran
.34	U.S. Puts Rancor Aside, Aids Quake Relief Effort
.33	Five Decades Of Major Earthquakes Worldwide
.32	Feds, State Promise Huntsville: 'Whatever's Needed Will be Done'
.32	Death Toll Rises To 981, Legislator Beaten To Death
.28	The Bay Area Quake; What Next?
.28	Parents Of Peace Corps Worker Find Out Son Is Safe
.25	Quake Along India-Nepal Border Reportedly Leaves More Than 200 Dead
.22	100 Feared Dead In Earthquake That Hits Nepal And Eastern India
.21	Death Toll At 749 In Quake
.17	Earthquake Death Toll Rises As Rescue Teams Reach Remote Areas
.17	Rescuers Cope With Rains, Aftershocks
.16	One-Third Of Nepal Devastated By Quake
.16	Unsafe Buildings Bulldozed As Recovery From Killer Quake Continues
.14	Powerful Storm Pushes North, Brings Floods And Destruction To Northeast
.07	Federal Relief Ok'D; Governor Visits The Grieving
.07	Guatemalan Ferry Sinks; 59 Dead, Six Missing
.06	In Iran, Saddam Is Generalissimo, Foster Child And Frankenstein
.06	Huntsville Picks Up Pieces One Day After Killer Tornado
.04	Gorbachev, Thatcher Talk 4 Hours
.03	Elite Field Enhances Marathon Top Women Eager To Run In S.F. Event
.03	Bush Names Deane Hinton As Ambassador To Panama
.03	Bush Picks Contra Aid Figure As Envoy To Mexico
.03	Asian Translators Rushed In To Help Parents Of Children In Shooting
.03	Critics Call Battleship'S 16-Inch Guns Outmoded, Dangerous
.03	Wen Jiabao, Spokesman Comment On Flood Situation

Appendix C

Example Cluster Results

Cluster: 1 **Number of Documents:** 17 **Mean Query Score:** .74

Score	*Subject Line*
.83	Scientists: Warming Trends Could Mean Fiercer Hurricanes
.81	Hurricane Gilbert Heading For Jamaica With 100 Mph Winds
.80	Gilbert: Third Force 5 Hurricane This Century
.80	Markets: Hurricane Rages But Dollar Storm Blows Out
.79	Hurricane Hits Jamaica With 115 Mph Winds; Communications Disrupted
.78	Gilbert Reaches Jamaican Capital With 110 Mph Winds
.77	Killer Hurricane Roarstoward Mexico With 175 Mph Winds
.75	Cleaning Up After Andrew
.74	After Battering Yucatan, Gilbert Strengthens, Heads For Gulf Coast
.74	What Makes Gilbert So Strong?
.73	Gulf Coast From Mexico To Louisiana Braces For Hurricane Gilbert
.73	Floods From Hurricane Kill 10 Police In Monterrey, Overturn Buses
.72	Hurricane Gilbert Heads Toward Dominican Coast
.68	Storms Batter Yucatan; Thousands Flee
.67	Gilbert Slams Into Yucatan Peninsula
.58	Gilbert Downgraded To Tropical Storm; Floods And Tornadoes Feared
.57	Expert Suggests Planets Affect Drought

Cluster: 2 **Number of Documents:** 17 **Mean Query Score:** .72

Score	*Subject Line*
.90	Hurricane Latest In String Of Disasters To Hit Historic City
.85	Hurricane Forecasters Carry On Amid Chaos
.85	Forecasting Aided By Supercomputers, But Still An Uncertain Science
.84	Killer Storm Hits South Carolina Coast
.82	City Sends Money To Charleston In Repayment Of 211-Year-Old Debt
.82	150,000 Take Off As Hugo Takes Aim At Ga., Carolina
.82	Loss Of Life Low Because People Were Prepared
.80	Officials Ride Out Storm In Historic City Hall
.77	Hugo'S Path Of Destruction
.71	Hurricane Batters Southern Us But Lets Insurers Off Lightly
.68	Ga Says Hurricane Claims Could Reach 'Up To Dollars 40M'
.67	Us Insurers Face Heaviest Hurricane Damage Claims
.63	Hurricane Damage Put At Dollars 20Bn As 2M People Told To Leave Homes
.55	The Bay Area Earthquake: Insurance Firms Expected To Pay Billions Of Dollars
.54	Hurricane Hugo May Be Most Costly Storm Ever
.48	Several Insurers' Net To Be Cut In Quarter By Hurricane Hugo
.47	Hugo To Be Costliest Storm For Insurers But No Early Rate Increase Is Expected

Cluster: 3 **Number of Documents:** 11 **Mean Query Score:** .44

Score	*Subject Line*
.51	A Special Session, With Speed
.48	The Bay Area Quake; Pressure Points
.47	The World Series; Oakland Athletics Vs. San Francisco Giants
.47	Area Where Earthquake Hit Seen As Highly Probable In 1988 Report
.45	Iran Earthquake Caught Victims Asleep In Fragile Homes
.45	Iranian Media Low Key In Reporting On Killer Quake
.44	Scientists Last Year Had Pinpointed Tuesday'S Site As Likely To Be Jolted
.44	Earthquake Rocks San Francisco
.43	Iran Earthquake Caught Victims Asleep In Fragile Homes
.37	While La Awaits 'The Big One,' Another Killer Quake Hits San Francisco
.36	Many Homeowners Not Insured; Analysts Say Disaster May Benefit Insurers

Cluster: 4 **Number of Documents:** 5 **Mean Query Score:** .38

Score	*Subject Line*
.74	Bay Area Quake
.54	Relief Pours Into Bay Area
.37	Earthquake Rocks San Francisco
.28	The Bay Area Quake; What Next?
.03	Bush Names Deane Hinton As Ambassador To Panama

Cluster: 5 **Number of Documents:** 11 **Mean Query Score:** .36

Score	*Subject Line*
.48	Rushdie Donates Quake Aid; Iran Says He'S Still Doomed
.47	San Diegans Await Word On Iran Relatives
.45	County'S Iranians Offer Aid In Quake
.39	Contributions For Iranian Relief Light, Expected To Pick Up
.37	Iran Accepts U.S. Help; Earthquake Death Toll Hits 35,000
.35	Group Claims Iran Underreporting Earthquake Casualties
.35	Tons Of Relief Supplies Arriving, But Distribution Difficult
.34	Radical Paper Blames U.S., In Part, For Quake Damage
.34	U.S. Puts Rancor Aside, Aids Quake Relief Effort
.33	Five Decades Of Major Earthquakes Worldwide
.06	In Iran, Saddam Is Generalissimo, Foster Child And Frankenstein

Cluster: 6 **Number of Documents:** 3 **Mean Query Score:** .27

Score	*Subject Line*
.73	The Bay Area Earthquake: Bush Makes Sure He Acts Speedily On Quake Crisis
.03	Bush Names Deane Hinton As Ambassador To Panama
.03	Bush Picks Contra Aid Figure As Envoy To Mexico

Cluster: 7 **Number of Documents:** 9 **Mean Query Score:** .21

Score	*Subject Line*
.32	Death Toll Rises To 981, Legislator Beaten To Death
.28	Parents Of Peace Corps Worker Find Out Son Is Safe
.25	Quake Along India-Nepal Border Reportedly Leaves More Than 200 Dead
.22	100 Feared Dead In Earthquake That Hits Nepal And Eastern India
.21	Death Toll At 749 In Quake
.17	Earthquake Death Toll Rises As Rescue Teams Reach Remote Areas
.17	Rescuers Cope With Rains, Aftershocks
.16	One-Third Of Nepal Devastated By Quake
.16	Unsafe Buildings Bulldozed As Recovery From Killer Quake Continues

Cluster: 8 **Number of Documents:** 4 **Mean Query Score:** .13

Score	*Subject Line*
.32	Feds, State Promise Huntsville: 'Whatever's Needed Will be Done'
.14	Powerful Storm Pushes North, Brings Floods And Destruction To Northeast
.07	Federal Relief Ok'D; Governor Visits The Grieving
.06	Huntsville Picks Up Pieces One Day After Killer Tornado

Cluster: 9 **Number of Documents:** 3 **Mean Query Score:** .04

Score	*Subject Line*
.07	Guatemalan Ferry Sinks; 59 Dead, Six Missing
.03	Asian Translators Rushed In To Help Parents Of Children In Shooting
.03	Critics Call Battleship'S 16-Inch Guns Outmoded, Dangerous

Cluster: 10 **Number of Documents:** 2 **Mean Query Score:** .03

Score	*Subject Line*
.04	Gorbachev, Thatcher Talk 4 Hours
.03	Wen Jiabao, Spokesman Comment On Flood Situation

Appendix D

Example Summarization Results

Cluster: 1 **Number of Documents:** 17 **Mean Query Score:** .74

Gilbert reached Jamaica after skirting southern Puerto Rico, Haiti and the Dominican Republic. Pereira, who spoke by telephone from Mexico City, said heavy rain was falling over the peninsula and communications with Cancun and Cozumel were out. The hurricane center said Gilbert was the most intense storm on record in terms of barometric pressure. Jamaican Prime Minister Edward Seaga said late Tuesday that at least six people were killed, and an estimated 60,000 were left homeless in "the worst natural disaster in the modern history of Jamaica".

Cluster: 2 **Number of Documents:** 17 **Mean Query Score:** .72

Such increased demand for reinsurance, along with the losses the reinsurers will bear from these two disasters, are likely to spur increases in reinsurance prices that will later be translated into an overall price rise. For example, insurers may seek to limit their future exposure to catastrophes by increasing the amount of reinsurance they buy. Nationwide Insurance Co., a mutual company based in Columbus, Ohio, said Hugo "is the single largest claims disaster" it has seen in its 63-year history. Hugo could have a marginal impact on third-quarter income at Travelers Corp., Aetna Life & Casualty Insurance Co. and Chubb Corp., according to industry analysts.

Cluster: 3 **Number of Documents:** 11 **Mean Query Score:** .44

The 7.7-magnitude quake was the largest ever recorded in that area, where two major plates of the earth's crust meet, Needhams said. The shock wave traveled through the mountainous section of coastal Iran where most of the buildings are built on a flood plain of loosely deposited soil that shifts in an earthquake and allows structures to collapse, he said. The nearest metropolitan area to Tuesday's earthquake, San Jose, has seen nearly a dozen earthquakes of 5 magnitude or greater in the last 10 years, but several of them have been on the Calaveras fault, which generally runs just east of San Jose, more than 10 miles east of San Andreas at this point .

Cluster: 4 **Number of Documents:** 5 **Mean Query Score:** .38

An Anheuser-Busch Inc. brewery in Fairfield, 35 miles northeast of San Francisco, interrupted normal operations to fill cans with drinking water instead of beer for free shipment to relief agencies, the company said. Coast Guard offices across the West Coast provided San Francisco with as many boats and helicopters as could be spared, said Coast Guard spokeswoman Elizabeth Neely. "We all know the horrors of an earthquake and we express our deep-felt sympathy for the people of the San Francisco Bay area".

Cluster: 5 **Number of Documents:** 11 **Mean Query Score:** .36

The Indian government said it would donate medicine, blankets and other relief supplies worth more than $500,000. John Paul also sent a telegram to the papal nuncio, or diplomatic representative, in Tehran saying the pope "is praying fervently for the wounded and the families of the victims". The newspaper was implying that with these actions the United States had blocked the diversion of funds to earthquake safety measures. China's Red Cross said it was sending $106,000 worth of relief supplies, state-run television said Saturday. It called on Iranians to reject relief offers by the United States and "other governments whose hands are stained with the blood of the Iranian people".

Cluster: 6 **Number of Documents:** 3 **Mean Query Score:** .27

President Bush has decided to nominate John D. Negroponte, a veteran diplomat who helped direct U.S. aid to Nicaraguan rebels, to the key position of ambassador to Mexico, Administration officials said Thursday . In the aftermath of the California earthquake, President Bush and his aides flew into a whirlwind of earthquake-related activity yesterday morning. Helms questioned the appointment of a Democrat to the post, and Lugar expressed concern that Aronson had no experience with the economic issues facing South America, they said . President Bush has named career diplomat Deane Hinton as ambassador to Panama, the White House announced Tuesday.

Cluster: 7 **Number of Documents:** 9 **Mean Query Score:** .21

It was the deadliest quake to strike the subcontinent since 1950, when, by official count, 1,500 people were killed in the eastern Indian state of Assam. Indian officials said Wednesday that at least 196 died in India in Sunday's disaster. Prime Minister Marich Man Singh Shrestha visited Dharan, one of the hardest-hit towns, and surveyed heaps of bricks that were once three-story and five-story houses. Tremors flattened more than 25,000 houses in India, officials said, and the Bhoothai Balan River flooded nearly 50 villages in the Madhubani district. Officials in both countries expected the death toll to rise as relief workers reached remote towns that have been inaccessible since Sunday's devastating quake.

Cluster: 8 **Number of Documents:** 4 **Mean Query Score:** .13

Marilyn Quayle, wife of Vice President Dan Quayle, praised the heroism of this tornado-ravaged northern Alabama city's residents during a visit Saturday. Federal aid was approved Friday for tornado-ravaged Huntsville, where 17 died and several hundred were injured or lost their homes, and Gov. Guy Hunt toured hospitals and visited with grieving families. A violent storm that spun tornadoes across the South and Midwest blew north Thursday, knocking a cafeteria wall down on top of lunching schoolchildren in upstate New York. Danny Cooper, director of the Alabama Emergency Management Agency, said three people were seriously injured in an earlier tornado that hit a home in Clay County in eastern Alabama.

Cluster: 9 **Number of Documents:** 3 **Mean Query Score:** .04

The Navy also on Thursday placed a moratorium on firing the big guns, found only on the Iowa and its three sister battleships, the New Jersey, the Wisconsin and the Missouri. The Navy said Thursday that none of the guns in the Iowa turret had been fired before the explosion took place. In a larger sense, critics say the battleships themselves, once the mainstay of gunboat diplomacy, have outlived their purpose. Missiles, the backbone of the modern Navy's attack capability, have integrated propellants. "These are the only guns in the Navy that still used bagged powder".

Cluster: 10 **Number of Documents:** 2 **Mean Query Score:** .03

While inspecting flood control and relief work in Jiangxi Province, through which the Ganjiang flows, Wen Jiabao, alternate member of the Political Bureau and member of the Secretariat of the CPC Central Committee, said the flooded areas should work hard to mitigate losses through their own efforts and with support from various departments. Meanwhile, 740 million yuan in loans and another 50 million yuan in donations have also been sent to flooded areas in Guangdong Province. BFN [Text] Beijing, June 26 (XINHUA) – Floods have all receded along Xijiang, Beijiang, Xiangjiang and Ganjiang except at a section of the Xijiang River in Wuzhou city, Guangxi, a spokesman from the State Flood Control and Drought Relief Headquarters announced today.

Bibliography

[1] M. W. Berry, S. T. Dumais, and G. W. O'Brien. Using Linear Algebra for Intelligent Information Retrieval. *SIAM Rev.*, 37(4):573–595, 1995.

[2] M. W. Berry and D. I. Martin. Parallel SVD for Scalable Information Retrieval. In *Proceedings of the International Workshop on Parallel Matrix Algorithms and Applications*, Neuchatel, Switzerland, 2000.

[3] J. M. Conroy and D. P. O'Leary. Text Summarization via Hidden Markov Models and Pivoted QR Matrix Decomposition. Technical report, University of Maryland, College Park, March, 2001.

[4] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. Indexing by Latent Semantic Analysis. *Journal of the American Society of Information Science*, 41(6):391–407, 1990.

[5] I. S. Dhillon, J. Fan, and Y. Guan. Efficient Clustering of Very Large Document Collections. In V. K. R. Grossman, C. Kamath and R. Namburu, editors, *Data Mining for Scientific and Engineering Applications*. Kluwer Academic Publishers, 2001. Invited book chapter.

[6] I. S. Dhillon and D. S. Modha. Concept decompositions for large sparse text data using clustering. *Machine Learning*, 42(1):143–175, Jan 2001.

[7] D. M. Dunlavy. An Information Retrieval System for Improving Efficiency in Scientific Literature Searches. *QCS IR System Project Proposal.* September, 2002.

[8] D. M. Dunlavy, J. M. Conroy, J. D. Schlesinger, S. A. Goodman, M. E. Okurowski, D. P. O'Leary, and H. van HalterenS. Performance of a Three-Stage System for Multi-Document Summarization. In *Proceedings of the Document Understanding Conference*, 2003.

[9] T. Dunning. "Accurate Methods for Statistics of Surprise and Coincidence". *Computational Linguistics*, 19:61–74, 1993.

[10] W. N. Francis and H. Kucera. *Frequency Analysis of English Usage: Lexicon and Grammar*. Houghton Mifflin Company, Boston, MA, 1982.

[11] T. G. Kolda and D. P. O'Leary. A Semidiscrete Matrix Decomposition for Latent Semantic Indexing in Information Retrieval. *ACM Trans. Inf. Sys.*, 16(4):322–346, 1998.

[12] C.-Y. Lin and E. Hovy. "The Automatic Acquisition of Topic Signatures for Text Summarization". In *Proceeding of the Document Understanding Conference*, 2002.

[13] A. Mikheev. "Tagging Sentence Boundaries". In *Proceedings of the First Meeting of the North American Chapter of the Association of Computational Linguistics (NAACL)*, pages 264–271, Seattle, WA, 2000. Morgan Kaufmann.

[14] L. R. Rabiner. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. *Proceedings of the IEEE*, 77:257–285, Jan 1989.

[15] J. Schlesinger, J. Conroy, M. Okurowski, and D. O'Leary. "Machine and Human Performance for Single and Multidocument Summarization". *IEEE Intelligent Systems*, 18(1):46–54, Jan/Feb 2003.

[16] J. D. Schlesinger, M. E. Okurowski, J. M. Conroy, D. P. O'Leary, A. Taylor, J. Hobbs, and W. H. T. Wilson. Understanding Machine Performance in the Context of Human Performance for Multi-document Summarization. In *Proceedings of the Workshop on Automatic Summarization*, 2002.