2004

# Implementation and analysis of a direct multidisplay repository

Jinghao Miao
*Iowa State University*

# Implementation and analysis of a direct Multidisplay repository

by

Jinghao Miao

A dissertation submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Major: Computer Engineering

Iowa State University

Ames, Iowa

2004

Graduate College
Iowa State University

This is to certify that the Doctoral dissertation of

Jinghao Miao

has met the dissertation requirements of Iowa State University

Signature was redacted for privacy.

Major Professor

Signature was redacted for privacy.

For the Major Program

## DEDICATION

I would like to dedicate this thesis to my parents who have been supporting me throughout my Ph.D. study. Without their support and encouragement I would not have been able to complete this work. I would also like to thank my friends for their support during the writing of this work.

# ACKNOWLEDGEMENTS

Many people helped in diverse ways while I have pursued this research and my degree in Computer Engineering at Iowa State University. First and foremost, I greatly appreciate my major professor Dr. Daniel Berleant, for his guidance in my professional devvelopment, his patience and support, and his encouragement of my own independent research interests. I would also like to thank each member of my committee — Dr. Julie Dickerson, Dr. Srikanta Tirthapura, Dr. Patrick Patterson and Dr. Lee Honeycutt — for their unique efforts and contributions to my research. I am particularly greatful to my best friend, Hao-Li Wang, for his loving assistance, efforts and inspiration throughout the writing of this thesis; I learned a great deal from him about how to carry out effective and efficient research and how to make the research more enjoyable as well.

A lot of students participated in my user study survey, all of whom contributed to completing my research. I appreciate their help. I also appreciate Dongping Xu who offered references for my research, and friends who have supported me over the years during my stay at Iowa State University.

Most of all, I would like to thank my parents for their continued support over the years. I am well aware that you do not fully understand what I do, or why I do it, but you have each been completely supportive for me, even with thousand of miles continuing to separate us.

# TABLE OF CONTENTS

**ACKNOWLEDGEMENTS** . . . . . . . . . . . . . . . . . . . . . . . . . iv

**LIST OF TABLES** . . . . . . . . . . . . . . . . . . . . . . . . . . . ix

**LIST OF FIGURES** . . . . . . . . . . . . . . . . . . . . . . . . . . x

**ABSTRACT** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . xiii

**CHAPTER 1. OVERVIEW** . . . . . . . . . . . . . . . . . . . . . . . 1

1.1 Introduction to MultiBrowser . . . . . . . . . . . . . . . . . . . . 2

1.2 From Paragraph to Document Networks . . . . . . . . . . . . . . . 3

**CHAPTER 2. Background and Related Work** . . . . . . . . . . . . . . 4

2.1 Background . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 4

2.2 Related Work . . . . . . . . . . . . . . . . . . . . . . . . . . . . 5

2.2.1 User Models and Information Customization . . . . . . . . . . . . 5

2.2.2 Visual Interfaces to Facilitate Document Retrieval . . . . . . . . . 11

2.3 Testbed - MultiBrowser . . . . . . . . . . . . . . . . . . . . . . . 14

2.3.1 Overview of MultiBrowser . . . . . . . . . . . . . . . . . . . . . 14

2.3.2 Design of MultiBrowser . . . . . . . . . . . . . . . . . . . . . . 16

**CHAPTER 3. FORMALIZATION OF THE MULTIBROWSER INTERFACE** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 18

3.1 Multidisplay and Monodisplay . . . . . . . . . . . . . . . . . . . . 18

3.1.1 Definitions . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 19
3.1.2 Formalizing Multidisplay . . . . . . . . . . . . . . . . . . . . . . . . 19
3.2 Direct & Indirect Display, and Prefetch . . . . . . . . . . . . . . . . . . . 20
3.3 Direct Display and Multidisplay Combined . . . . . . . . . . . . . . . . . 21
3.4 Direct Multidisplay and Prefetch . . . . . . . . . . . . . . . . . . . . . . 22
3.4.1 The issue of space available per item . . . . . . . . . . . . . . . . 24

CHAPTER 4. DIRECT MULTIDISPLAY IN MULTIBROWSER . . 26
4.1 Phase 1: Document-Scale Processing . . . . . . . . . . . . . . . . . . . . 26
4.2 Phase 2: Paragraph-Scale Processing . . . . . . . . . . . . . . . . . . . . 27

CHAPTER 5. A NORMALIZED GAUSSIAN CATEGORY MODEL FOR COLOR REPRESENTATION IN MULTIBROWSER . . . . 29
5.1 Gaussian Distribution . . . . . . . . . . . . . . . . . . . . . . . . . . . . 29
5.2 Normalized Gaussian Method . . . . . . . . . . . . . . . . . . . . . . . . 29
5.3 Color Generation in MultiBrowser with the Modified Gaussian Method . 30
5.4 Determining the Parameter $\sigma$ . . . . . . . . . . . . . . . . . . . . . . . 32
5.5 Color Saturation . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 33

CHAPTER 6. EXPERIMENTS . . . . . . . . . . . . . . . . . . . . . . . 35
6.1 User Impressions . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 35
6.2 Experiments on Color Bars . . . . . . . . . . . . . . . . . . . . . . . . . 36
6.2.1 Experiment A: Response Speed . . . . . . . . . . . . . . . . . . . 37
6.2.2 Experiment B: Response Consistency . . . . . . . . . . . . . . . . 37
6.3 Discussion: Color Bars . . . . . . . . . . . . . . . . . . . . . . . . . . . 38

CHAPTER 7. LINKING STRUCTURES IN MULTIBROWSER . . . 42
7.1 Background Knowledge . . . . . . . . . . . . . . . . . . . . . . . . . . . 42
7.1.1 P-Graphs . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 42

7.1.2 D-Graphs . . . . . 43

7.1.3 Centrality and Dispersion . . . . . 45

7.2 P-graph Generating Algorithms . . . . . 46

7.2.1 Basic P-Graph Algorithm . . . . . 46

7.2.2 2-Clan P-Graph Algorithm . . . . . 46

7.2.3 Modified 2-Clan P-Graph Algorithm . . . . . 48

**CHAPTER 8. LINKING STRUCTURE EXPERIMENTS** . . . . . 49

8.1 Experiment 1: P-Graph Types . . . . . 49

8.2 Experiment 2: Document Retrieval Systems . . . . . 52

8.3 Experiment 3: Topics . . . . . 55

8.4 Results . . . . . 57

**CHAPTER 9. NAVIGATION IN RING-STRUCTURED REPOSITORIES** . . . . . 60

**CHAPTER 10. SOCIAL FILTERING IN MULTIBROWSER** . . . . . 63

10.1 What Is Social Filtering? . . . . . 63

10.2 Related Work . . . . . 64

10.2.1 Tapestry . . . . . 64

10.2.2 GroupLens . . . . . 65

10.3 Social Filtering Implementation in MultiBrowser . . . . . 66

10.3.1 Important Concepts . . . . . 66

10.3.2 Information Rating Prototype . . . . . 67

10.3.3 Summary . . . . . 68

**CHAPTER 11. CONCLUSION AND FUTURE WORK** . . . . . 69

11.1 Conclusions . . . . . 69

11.2 Future Work . . . . . 70

11.2.1 Brief Introduction . . . . . . . . . . . . . . . . . . . . . . . . . . . . 70

11.2.2 Internal Hyperlink Consideration . . . . . . . . . . . . . . . . . . . 70

11.2.3 Color Bars as Representations of Document Content . . . . . . . 71

**APPENDIX A. USER STUDY SURVEY SHEETS** . . . . . . . . . . . 73

**APPENDIX B. USER STUDY RESULTS** . . . . . . . . . . . . . . . . . . 78

**APPENDIX C. COMPILING AND RUNNING MULTIBROWSER** . 82

**BIBLIOGRAPHY** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 84

# LIST OF TABLES

Table 3.1 Examples of direct monodisplay, indirect monodisplay, direct multidisplay, and indirect multidisplay. . . . . . . . . . . . . . . . . 21

Table 6.1 Analysis of responses to the request, "comment on the density of relevant information appearing, on average, on the screen when MultiBrowser is used vs. when ordinary display is used." . . . . 36

Table 6.2 Analysis of responses to the request, "comment on the density of relevant information appearing, on average, on the screen when MultiBrowser is used vs. when ordinary display is used." . . . . 36

Table B.1 The times to complete the similarity judgment tasks in *Experiment 2* of the user study. . . . . . . . . . . . . . . . . . . . . . . . 80

Table B.2 The answers of 6 different subjects to a specific question $Q$. . . . 80

Table B.3 Conversion of the orginal answers. . . . . . . . . . . . . . . . . . 80

Table B.4 Standard deviations of the results. . . . . . . . . . . . . . . . . . 81

# LIST OF FIGURES

Figure 2.1 Traditional information pipeline (after [5]). . . . . . . . . . . . . 6
Figure 2.2 New information pipeline ([5]). . . . . . . . . . . . . . . . . . . . . 6
Figure 2.3 A sample MultiBrowser screen. . . . . . . . . . . . . . . . . . . . . 15
Figure 2.4 An enlarged view of a "FIND SIMILAR" link in the MultiBrowser view. . . . . . . . . . . . . . . . . . . . . . . . . . 16
Figure 5.1 The curve for the modified Gaussian distribution. . . . . . . . . 32
Figure 6.1 Hybrid of TileBars and a simplified version of Byrd's speckled scroll bar visualization. (Thanks to Dongping Xu [6].) . . . . . . 41
Figure 7.1 A paragraph contains links to the five most similar other paragraphs. . . . . . . . . . . . . . . . . . . . . . . . . . . . . 43
Figure 7.2 P-graph for documents $A$ and $B$. . . . . . . . . . . . . . . . . . . . 44
Figure 7.3 D-graph for documents $A$ and $B$ (and $C$). . . . . . . . . . . . . . 45
Figure 7.4 Four types of indirect inter-paragraph links. . . . . . . . . . . . . 47
Figure 7.5 Situations depicted in Figure 7.4 with the new, derived paragraph links added. . . . . . . . . . . . . . . . . . . . . . . . . . . . . 48
Figure 8.1 Weights of documents in the D-graph of the *powered parachuting* repository resulting from the Basic P-graph. . . . . . . . . . . 50

Figure 8.2 Weights of documents in the D-graph of the *powered parachuting* repository resulting from the 2-Clan P-graph. . . . . . . . . . . . 51

Figure 8.3 Weights of documents in the D-graph of the *powered parachuting* repository resulting from the Modified 2-Clan P-graph. . . . . . 51

Figure 8.4 Document weights implied by the 2-Clan P-graph generated for a repository of documents returned by Altavista. . . . . . . . . . 52

Figure 8.5 Authority and Hub values for each document in the same document set as in Figure 8.4. Values were generated using HITS on the Basic P-graph. . . . . . . . . . . . . . . . . . . . . . . . . . 53

Figure 8.6 PageRank scores for each document in the same document set as in Figure 8.4. Values were generated using PageRank algorithm on the Basic P-graph. . . . . . . . . . . . . . . . . . . . . . . . 53

Figure 8.7 Weights of documents, ordered high-to-low, from the D-graphs of each of five *vegetarian cooking* repositories derived from the return lists of five Web search engines. The D-graphs were derived from the Basic P-graphs of the repositories. . . . . . . . . . . . . 54

Figure 8.8 Weights of documents, ordered high-to-low, from the D-graphs of each of five *vegetarian cooking* repositories derived from the return lists of five Web search engines. The D-graphs were derived from the 2-Clan P-graphs of the repositories. . . . . . . . . . . . 54

Figure 8.9 Weights of documents, ordered high-to-low, from the D-graphs of each of five *vegetarian cooking* repositories derived from the return lists of five Web search engines. The D-graphs were derived from the Modified 2-Clan P-graphs of the repositories. . . . . . 55

Figure 8.10 Weights of documents, ordered high-to-low, from the D-graphs of each of five *powered parachuting* repositories derived from the return lists of five Web search engines. The D-graphs were derived from the Basic P-graphs of the repositories. . . . . . . . . . . . . 56

Figure 8.11 Weights of documents, ordered high-to-low, from the D-graphs of each of five *powered parachuting* repositories derived from the return lists of five Web search engines. The D-graphs were derived from the 2-Clan P-graphs of the repositories. . . . . . . . . . . . 56

Figure 8.12 Weights of documents, ordered high-to-low, from the D-graphs of each of five *powered parachuting* repositories derived from the return lists of five Web search engines. The D-graphs were derived from the Modified 2-Clan P-graphs of the repositories. . . . . . . 57

Figure 8.13 Log-log plot of document weight vs. document rank. . . . . . . . 58

Figure 9.1 Ring structuring of a hypertext repository classifies documents into the kernel, near ring, or far ring. . . . . . . . . . . . . . . . 61

Figure 9.2 A sample of a paragraph with explicit annotations of ring information. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 62

Figure 10.1 User rating information. . . . . . . . . . . . . . . . . . . . . . . . 68

# ABSTRACT

When several information items are presented simultaneously, the potential exists for users to interact with information in a rich and rewarding way. The MultiBrowser system supports such interaction via an information foraging style of hypermedia browsing. Multiple windows, colored bars that provide visual cues, and automatically inserted hyperlinks among the paragraphs in a document set contribute to the structure of repositories created by MultiBrowser. These repositories are browsable over the Web using an ordinary Web browser.

Using these repositories as a platform, we investigate hypertext repositories with automatically generated hyperlinks among paragraphs. The algorithms that generate the links yield results with significant common characteristics despite large differences among the algorithms. Furthermore, different repositories generated from the return lists of different search engines show significant common characteristics. Finally, repositories generated for different domains also have common characteristics. This suggests that these characteristics are pervasive properties of judiciously retrieved document sets. The common characteristics revolve around a tendency for some documents (called the kernel) to be destinations of a disproportionately high fraction of the hyperlinks in the repository. An emergent property of such repositories with practical significance is that browsing activities will have a tendency to trap users within the kernel, perhaps without them realizing it. To enable users to choose to avoid being trapped, we propose a ring structure for hypertext repositories. This structure is exposed to the user by annotations

to the hyperlinks.

# CHAPTER 1. OVERVIEW

Information flooding over the World Wide Web (WWW) has motivated researchers and developers to greatly improve intelligent information access, manipulation, and presentation techniques, to facilitate effective use of these resources. The recent hypermedia "revolution" is one step toward realizing the full potential of reader interaction with the circumscribed pool of information items represented in documents. Earlier steps include the inventions of writing, the scroll, the book, and the printing press. Now consider a vision for interacting with texts: a repository of text that, together with a suitable inference engine and user interface, acts as a discussion partner for the user. The user asks a question or makes a comment, and the system response is to present the most appropriate passage in the repository. Alternatively, the system response might be derived from the user interface rather than directly from the text itself, such as a request for clarification or some other meta-comment. A response could also contain both a passage and a meta-comment, such as a description of the relevance of the passage to the question. This would be useful for example if the relevance was analogical. Such a system would allow a user to have a "conversation" with a text.

Taking this idea further, a system for dialoguing with documents could provide several responses to a user's input simultaneously. Why be limited to just one? Using such a system would be roughly analogous to having a discussion with a board of experts. In this case the expertise resides in the combination of the repository and the document dialogue system.

This chapter will give a brief overview of the MultiBrowser system, an interactive hypermedia browsing system.

## 1.1 Introduction to MultiBrowser

The intriguing ultimate goal for text presentation and interaction with users can serve now as motivation for constructing experimental systems that move toward that goal. The system we focus on, MultiBrowser, responds to user actions with multiple passages from within a repository of hypertexts. This strategy is a potentially useful way to support the preference of readers of on-line text to scan and pick out passages rather than reading in a traditional sense (Nielsen 1999 [42]). The style of interaction that occurs in MultiBrowser is information foraging (Pirolli and Card 1995 [46], Pirolli 1998 [47], and Wexelblatt and Maes 1999 [62]), rather than information retrieval. The concept of information foraging is proposed as a paradigm for navigation different from the more goal-oriented activity of the traditional information retrieval paradigm [5] [22] [25]. It has been suggested that this concept can help in understanding how to create new interactive information system designs ([22] [46]). Information foraging is a relatively undirected kind of browsing. Foraging is an appropriate model of behavior when goals are vague, a situation that often occurs in practice, as when one "surfs" the Web with a "browser", "skims" a book, or "browses" library shelves. Such foraging activities can play an important role because the traditional information acquisition model of information retrieval with highly focused goals may not always be the most appropriate one. In MultiBrowser, users browse by foraging within an automatically generated hypermedia structure that links paragraphs that are automatically determined to be similar.

## 1.2 From Paragraph to Document Networks

To address the goal of information foraging, we have been exploring information foraging within sets of documents whose URLs are provided by Web search engines in response to queries [8]. For a search engine return list, each paragraph in each document is processed, and the 5 other paragraphs in the repository that are most similar to it are found. The distance metric we used for comparing paragraphs is the cosine measure in the space of strings of 5 characters, or 5-grams [13] [40]. This approach has been shown to produce results competitive with expressing texts as weighted vectors of words [40].

In this technique, each paragraph is expressed as a vector of 5-grams, with each 5-gram weighted by the number of times it occurs in the paragraph. Each vector is then normalized and mapped to a point in 5-gram space [13] [2]. Points are compared using the cosine similarity measure [13] [2]. For each paragraph, the 5 other most similar paragraphs are identified regardless of what documents they are in. Based on these similarity computations, different algorithms for inserting hyperlinks among paragraphs can be used, and their properties compared. That is the subject of part of my research.

For each paragraph, a 6-targeted hyperlink is added, from it to each of those five similar paragraphs and itself. The intent is to support simultaneous display of multiple related paragraphs in separate windows [7] [22] [51]. This is consistent with a long-running current of thought in the hypertext field typified by such early and influential works as [22] [28] [50] [51] [55]. Therefore the entire repository contains a network of paragraphs connected by these hyperlinks. The properties of these link networks form the major part of the present investigation.

# CHAPTER 2. Background and Related Work

## 2.1 Background

A key goal mentioned in Chapter 1, for a system to provide multiple responses simultaneously, can be met with a multiple window architecture. The term "collage" for describing such a display seems to have originated with Kaltenbach and Frasson (1989 [28]), who describe a system for presenting mathematical proofs which was later extended to presenting hypertext (Kaltenbach et al. 1991 [29]). At least two studies suggest that the non-overlapping, space-filling windows strategy is preferable to overlapping windows for displaying related texts. In one, Bly and Rosenberg (1986 [7]) showed that such a windowing strategy supports faster identification of task-relevant paragraphs than an overlapping window strategy when it is possible to present the bulk of the source material without overlapping. In another, Kandogan and Shneiderman (1997 [31]) investigated completion times for 21 task conditions, finding that their "Elastic Windows" windowing method enabled faster task completion than overlapping windows. Collage-based displays are supported by a long tradition in hypertext systems. The NoteCards system (Halasz 1988 [22]) and the Sun version of Hyperties (Shneiderman 1987 [50]; Lifschitz and Shneiderman 1987 [38]) are among the early such systems. MultiBrowser uses collages that are tiled (space filling and without overlaps). Some notable other works using such collages include the Krakatoa Chronicle online newspaper (Kamba et al. 1995 [30]), the VOIR interface which, like Krakatoa, is based on the newspaper metaphor (Golovchinsky and Chignell 1997 [19]), and Elastic Windows (Kandogan and Shneiderman 1997 [32]).

## 2.2 Related Work

### 2.2.1 User Models and Information Customization

#### Introduction

The quantity of information flooding the Internet makes navigating the web to find desired information a frustrating, labor-intensive and time-consuming task. Consider the case in which we use WWW search engines to acquire information on chronic headache. We might find over 10,000 references on headache, but the information is disorganized and does not reflect specifics of interest. Many of the references may be testimonials, advertisements and general medical information about headaches, but not specific to one's symptoms. Thus we might actually fail to find the information we want. Thus, information overload requires us to develop innovative technologies to remedy the problem. According to some reports [11] there are a number of possible solutions such as scenario-based proxies, content-sensitive navigation and matching, and user models. Information customization, derived from the above technologies, involves presentation of digital information in a form suited to the customers' needs. Its need arises in part as a result of information overload and is becoming more important as a way to make existing information more useful.

#### The Information Pipeline

Traditional information artifacts progress in stages [5] (see Figure 2.1). Information users have access to various types of electronic information such as bulletin boards, websites and so forth. Traditional access tools to such information does involve a certain degree of information filtering, *e.g.*, keyword matching, however, this kind of preprocessing and filtering is limited because, for example, it only gives the right document but not the *right information* from the document. This is not very efficient in meeting users' specific information needs. Thus before information is presented to end users, a new stage in the traditional information pipeline can be helpful (Figure 2.2). The concept of user models is introduced next before examining information customization further.

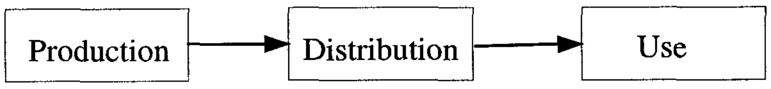


Figure 2.1 Traditional information pipeline (after [5]).

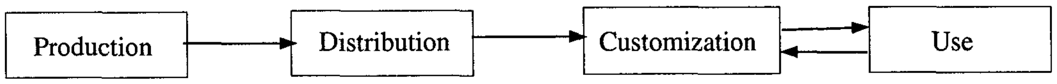


Figure 2.2 New information pipeline ([5]).

### User Models

In an interactive system, a model of the user can be useful. This means that in the course of the interaction with the user the system should represent facts or assumptions about various aspects of the user that might be relevant in the task domain at hand. These include in particular [35]

- the user's goals;
- the plans with which the user wants to reach his/her goals; and
- the knowledge (beliefs) of the user about a particular domain.

The set of these facts or assumptions forms the user model of the system if they can be separated by the system from the rest of the system's knowledge. The ability to automatically draw inferences about the user's beliefs, goals and plans enables the system to reason about a situation from the user's point of view. In turn, this helps the system to be able to exhibit cooperative dialog behavior. For instance, it can enable the system to

- take into account the goals and plans for which the user makes a query, and supply additional relevant information, if necessary;
- take into account what the user probably already knows or does not know about a situation, and thus avoid redundancy and incomprehensibility in its answers and explanations; and
- detect wrong beliefs of the user and inform the user about them.

Because each individual user has potentially unique concerns, it is necessary that a user model contain a representation of characteristic information about the user as well as a description of the user's query intention and goals. These models will help interpret the content of a user query and effectively customize results by guiding the query facilities in deriving the answer.

Previous work on user modeling typically follows one of three user classification methods:

1. nondomain-specific methods, which characterize users by the extent of their knowledge,
2. domain-specific methods, which use stereotypes to describe general groups of users, and

3. multidimensional approaches, which combine nondomain and domain-specific techniques.

However, these approaches do not fully exploit domain-specific knowledge and are often limited to a single application domain and/or a small set of user classes. There are some remedies that could partially solve these problems by constructing more domain-specific knowledge. Then, in addition to a user model, a new constraint model is added to customize the information retrieval process. The constraint selection module consists of a set of predefined rules that select a set of attributes based on the user profile to add additional constraints to the original user query. The constraint rules are different for each query function. In the following, we discuss the use of the user model in different query functions [35]:

- *Resource Selection and Filtering* — The data sources for answering queries are selected on the basis of the user's profile. Given data sources that are available as input for information customization, additional attributes are used to classify those data sources, such as the goal of the data source (*e.g.* educational, reference, etc.), the source's expertise level (novice, expert), and the source's target audience (children, adults). The constraint selection module specifies what set of user attributes (*e.g.* age, gender, and education level) are used to choose the data sources. Constraints for resource selection are then generated based on the mapping of the user model to the data source attributes.

- *Content Correlation* — To provide relevant information to the user, additional constraints are added to the query. The constraint selection module maps a user model to a list of attribute fields that specify which attributes and values are to be used as constraints. Then the user model stores information on how the data types should be correlated; this information is supplied by a domain expert to the user model.

- *Approximate Matching* — The approximate matching process references the user model to guide relaxation. If no approximate matching methodology is specified by the user in a query, then the system uses the defaults defined within the user model.

- *Result Ranking* — Query results are ranked using a nearness measure, such as weighted relaxation error for approximate matching, appropriateness of resource, user preferences (*e.g.*, diagrams are preferred over text), etc.

- *Visualization* — the interface between users and applications takes into account the user to determine an effective means of data presentation. Default user interfaces are stored in the user model, but are easily adapted and customized at the individual user modeling level.

### Information Customization

Due to the inherent deficiencies of uncustomized information discussed earlier, the stage of customization should be added before the information is presented to end users. The output of this stage is then compatible with the users' point of view. Information is customized when transformed into an extract or other form that gives end users what is needed at a given moment. The process include editing, navigation link establishment, use of browsing software, structure or keyword analysis, information visualization and other means. All these processing steps affect users' interactions.

There are many issues in the customization of information using user models [27]:

- How to model the user? This could be done through either explicit or implicit methods based on the user model concepts discussed above.
- What information is needed to describe a user for information retrieval purposes? From the user models we can extract both direct and indirect information for each specific information retrieval purpose.
- How to represent, use and share the user model? Accurate user information can be collected through a shared retrieval system to shorten the information gathering period.

Since users' personal information and their query requirements are reflected in a customized information retrieval system, it is more likely that the results retrieved through the new information pipeline will meet user needs. There can be several ways to use the user model in an information retrieval system. Generally speaking, these can be done on either the server side or the client side. This will not be discussed in detail in this report. An important issue in information customization is language processing. There are many algorithms concerning this technology and user modelling is one of them.

The goal of information customization is to tailor an information artifact to a customer's specific needs at a given moment. Needs may be changing over even brief periods of time. Information customization must be sensitive to the quickly changing nature of the user's needs.

### User Models and Query Optimization

There are a number of ways to optimize queries based on user models. The result from the optimization will give optimal queries which carry user profiles for better retrieval. Rocchio's algorithm [53] is one of the ways to focus on this goal. It uses the feedback query concept to represent user profiles based on users' original queries. The optimized queries have a remarkable performance in information retrieval. The experiments were conducted in 1997 and improvements of this algorithm were also proposed by Singhal and his team members that year [53]. In SPIRE 2000, Tmar and Boughanem [58] proposed that using learned profiles in Mercure's model could achieve better results. This suggests combining the above two modeling methods in a system. One way to achieve that is to sequentially use the outputs from Rocchio's algorithm as the inputs for Mercure's models.

### 2.2.2 Visual Interfaces to Facilitate Document Retrieval

#### Brief Overview

Researchers in information retrieval (IR) have long searched for ways to make their systems more accessible to end users and to develop new ways for users to explore data. Visualization techniques (computer based methods to display large quantities of information graphically) appear promising as a means for achieving both goals. Nowadays research in IR has moved from the area of bibliographic records to full text and image retrieval out of databases that may contain hundreds of thousands of documents. Web retrieval, for example, frequently results in the identification of tens of thousands of "relevant" documents. Research on Visual Information Retrieval Interfaces (*VIRIs*) aims at providing the user with a method of organizing and refining such masses of data, a task that is virtually impossible with the traditional list of documents returned.

Many VIRI systems try to categorize sets of documents that are closely related, and display the documents in a well organized manner so that users are able to efficiently identify the desired information with ease. There are a number of ways to present the information retrieved depending on the VIRI used. These displays may be iconic, a landscape, a numerical array, or take on some other structure. And if possible, two or three dimensional displays can be applied to assist efficient and helpful web document retrieval. Advanced ways may also include user manipulations of those displays based on various search results.

The university of Pittsburg is now developing a set of VIRIs. The primary interface being studied is VIBE (Visual Information Browsing Environment). Others include GUIDO (Graphical User Interface for Document Organization), BIRD (Browsing Interface for Retrieving Documents), and DARE (Distance-Angle Retrieval Environment). Each interface has its own distinctive features and functionalities and each is implemented in several versions to facilitate studying the usability of its specific features.

### Sample Interfaces

**Bird and J. BIRD — A Query Formulation Tool.** BIRD (Browsing Interface for Retrieving Documents) was developed by Kim (1997 [33]), a Ph.D. student at University of Pittsburgh. Its primary function is to support query formation. Boolean operators are notoriously difficult to use correctly, even for users who are mathematically trained. Thus users may have to make multi-level decisions and keeping track of those decisions might be complicated. BIRD provides three basic operations:

1. separate a set of documents into subsets,
2. merge selected subsets and,
3. manually add/delete individual documents into the set.

Documents are vectorized with weights representing the importance of the index terms in them. Then they are put into separator boxes according to the index terms, after which the selected documents can be examined in detail or moved into categorizer boxes. Trash boxes are used to store unwanted documents. Documents in trash boxes can have their index terms hidden in term pools. There are also clipper boxes which are used to clip a specific document. Though users can only use two index terms to separate documents at a time, by iteratively breaking down sets of documents and merging them, complex sets of documents can be constructed. The J. BIRD interface is a simplification of BIRD and was achieved by representing documents as virtual sheets of paper. The funnel was replaced by a sorter and the matrix by 2 bins indicating "matches"and "does not match." To use the device a user selects a query term, then drags the stack of unsorted papers to the sorter. The results are displayed as proportionally sized stacks in the output bins.

**GUIDO — A Visual Tool for Retrieving Documents.** GUIDO [43] uses an iconic interface to display large number of documents. This is becoming increasing important in current computing and information systems. It allows users to visually perceive the relationships among documents and between document and query terms. The concept of reference points is introduced in this method to indicate the relevance relationships between users' interests and the documents. Based on previous work that focused on query vectorization, GUIDO adds more reference points such as user profiles, document clusters and so on. Because the reference points are often based on users' interests, they can also be called points of interests (POIs). GUIDO uses spatial vectors rather than commonly used vector space models. When multiple POIs are applied, users will be able to see a full set of documents in the form of icons spatially organized according to POIs. Due to the small scope of users' interests, all these icons have to be filtered. Currently GUIDO uses two dimensions with only two POIs and allows users to add a new POI, changing the filter accordingly. GUIDO display documents in icons or small rectangles. Users do not need to know the underlying mathematical formulations, but can visually see the document sets and how relevant they are.

**VIBE and WebVIBE — An Inferencing Tool.** Korfhage and Olsen developed VIBE (Visual Information Browsing Environment) at the University of Pittsburgh. VIBE [44] represents points of interest (POIs) or query terms as nodes. Documents are represented by various sized rectangles suspended within the surface bounded by the POIs. The internal representation of a document is a term vector, which is usually weighted, and the interface position of the document is calculated as the vector sum of the components indicated by the POIs. Seven special-purpose modes are hidden in the menu system. WebVIBE retains much of the appearance of the original VIBE but the metaphor has been changed to a naïve physics view of magnetism. The user is called upon to view the POIs as magnets and the documents are suspended in a magnetic force field. The position of a particular document can be thought of as being determined by the relative "pull" of terms that are present in the document.

## 2.3 Testbed - MultiBrowser

### 2.3.1 Overview of MultiBrowser

The design of MultiBrowser builds on the multiple windows idea with novel features. Each window in a MultiBrowser display is associated with a color bar. The color bar summarizes the location, in the space of all documents in the repository, of the document for which the window provides a view. The motivation for color bars is to provide quickly and easily perceived cues about what documents are similar to what other documents. Another novel feature is the hyperlink associated with each paragraph. This hyperlink is annotated to show how many of the target windows will be views into documents that are in a kernel of core documents in the repository, how many are close to the kernel, and how many are peripheral. In MultiBrowser, clicking on a hyperlink brings up a collage of several non-overlapping, hypermedia-containing sub-windows. Thus links are concep-

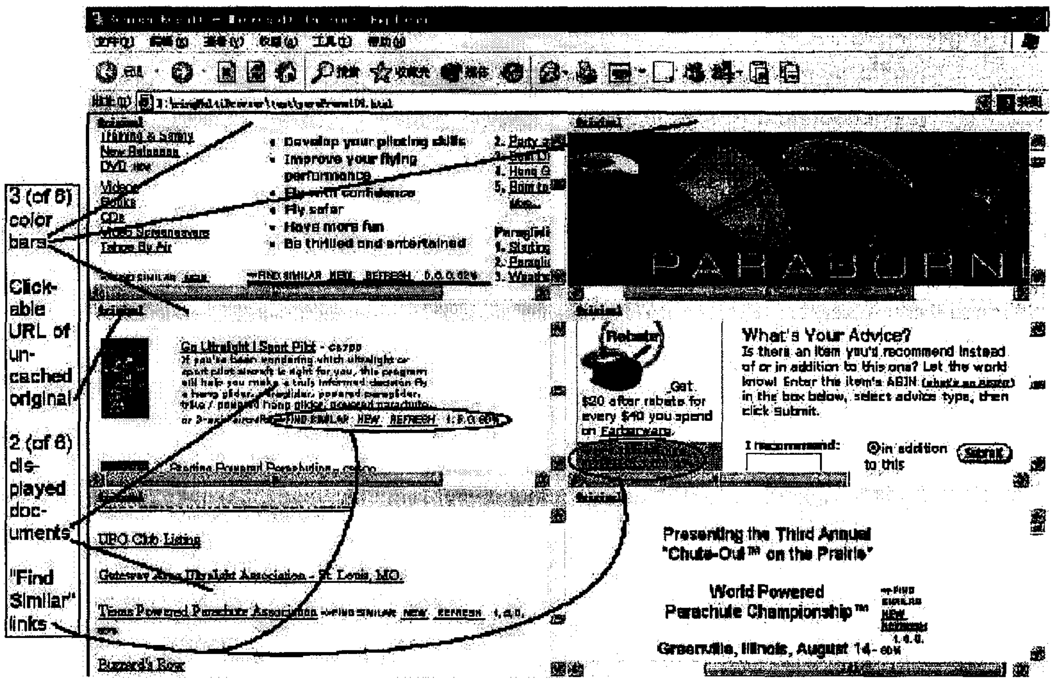
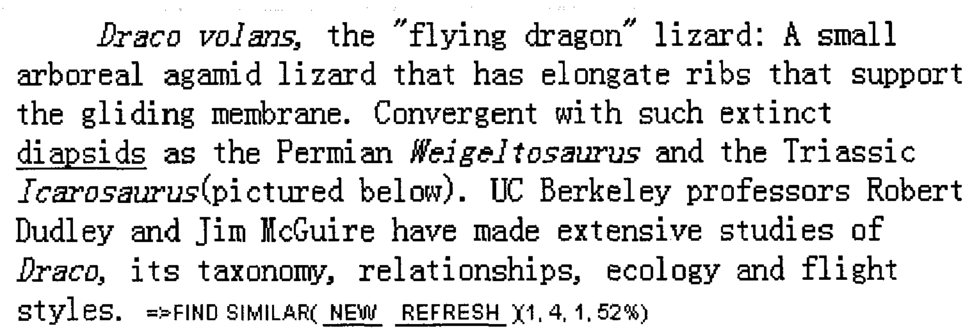

3 (of 6) color bars
Click-able URL of un-cached original
2 (of 6) dis-played doc-uments
"Find Similar" links
What's Your Advice?
Presenting the Third Annual "Chute-Out™ on the Prairie"
World Powered Parachute Championship™
Greenville, Illinois, August 14-

Figure 2.3 A sample MultiBrowser screen.

tually one-to-many. Each collage provides a view of the repository. Figure 2.3 shows a sample screen from a repository created from documents obtained by the Web search engine query "powered parachuting" (no endorsement of any company is intended). Figure 2.4 shows an enlarged view of a "FIND SIMILAR" link. The 4 numbers associated with each "FIND SIMILAR" link in the Figure state, for its associated paragraph, how many of the six windows it points to will contain documents with, respectively, many incoming paragraph links, a modest number($n$), few($f$), and a composite "%" ranking equal to $100 - 8n - 16f$. The annotations will be explained in the following chapters.

Unlike the Avant Browser (http://www.avantbrowser.com), which supports overlapping windows as part of the browser software, the MultiBrowser windowing interface is defined strictly within the repository itself, using standard HTML commands. HTML



*Draco volans*, the "flying dragon" lizard: A small arboreal agamid lizard that has elongate ribs that support the gliding membrane. Convergent with such extinct diapsids as the Permian *Weigeltosaurus* and the Triassic *Icarosaurus*(pictured below). UC Berkeley professors Robert Dudley and Jim McGuire have made extensive studies of *Draco*, its taxonomy, relationships, ecology and flight styles. =>FIND SIMILAR( NEW  REFRESH )(1, 4, 1, 52%)

Figure 2.4 An enlarged view of a "FIND SIMILAR" link in the Multi-Browser view.

<frameset>... ... </frameset> commands are used to specify windowed text for display in a standard browser.

### 2.3.2 Design of MultiBrowser

As described above, MultiBrowser clusters the documents in the repository into three groups. The $k$-means algorithm, a standard clustering algorithm, is used [60]. The distance metric we use is the cosine measure in the space of strings of 5 characters, or 5-grams. A Gaussian-like function of distances of each document to the centroids of the three clusters is used to compute intensities of red, green, and blue (RGB) components, which are combined into a color for a color bar associated with each document. The color bar summarizes the document in the sense that the user can visually compare the similarities of the documents in the different sub-windows by comparing their color bars. This may help them decide which window to read next, for example. Each color bar also contains a link that, when clicked, loads the corresponding document into the full browser frame for a closer look. After a color bar has been computed for each document, the documents are segmented into paragraphs and the set of paragraphs is processed. Each paragraph is mapped to a normalized point in 5-gram space, and points are compared using the cosine similarity measure mentioned earlier. For each paragraph, the five other most similar paragraphs are identified, regardless of what documents they are in, so that all six paragraphs can later be displayed simultaneously in the six sub-windows in response to a user click on a "FIND SIMILAR" link following the given paragraph. Thus each link in essence has six targets.

In the following chapters, I will explain in detail the implementation of MultiBrowser and the functions of each element in this browsing system.

# CHAPTER 3. FORMALIZATION OF THE MULTIBROWSER INTERFACE

The MultiBrowser interface uses what we will call *multidisplay*, because each sub-window shows information separately (Figure 2.3). Collectively the sub-windows provide a view of the repository. The goal is to provide a personalized (*i.e.* customized [5]) view that meets the needs of a particular user at a particular moment. Browsing involves interactive navigation through the space of these views. This navigation includes clicking, most often on a hyperlink compiled into the repository which points to a new view of the repository consisting of a different set of sub-windows. Navigation also includes scrolling within a given sub-window, which can be expanded to the full browser window by clicking the color bar of the sub-window. The MultiBrowser interface design is based not only on multidisplay but also on direct display (presentation of actual information, rather than metadata such as phrases or icons representing the actual information).

## 3.1 Multidisplay and Monodisplay

This chapter formalizes the intuition that simultaneous display of $n$ items rapidly increases the information retrieval precision of the overall display for even modest values of $n$. The concept of information prefetch is then used to further analyze the MultiBrowser user interface design.

### 3.1.1 Definitions

Define *multidisplay* as *a non-overlapping, space-filling collage of two or more presented information items.*

Define the opposite of multidisplay as monodisplay: a *monodisplay* is *the presentation of one item.*

While multidisplays show more than one item at a time, monodisplays can show more of a single item in the same display area. Each can be preferable depending on circumstances, hence MultiBrowser's support of multidisplay and monodisplay.

### 3.1.2 Formalizing Multidisplay

Let $P_1$ be the probability that the item most likely to be of interest to the user, is of interest. For example, if the items are displayed because they match a query, given a reasonable ranking algorithm this would be the one with the highest rank. The probability that this item is *not* of interest is therefore $1 - P_1$. Similarly, let $P_n$ be the probability that the $n^{th}$ most likely item to be of interest, is. The probability that it is not is thus $1-P_n$. Then $p$ *(no displayed item is of interest)* $= (1-P_1)(1-P_2)...(1-P_n)$. Define $P$ as

$$\begin{aligned} P &= p\ (\textit{at least one displayed item is of interest}) \\ &= 1-(1-P_1)(1-P_2)(1-P_n) \end{aligned} \tag{3.1}$$

$P$ rapidly approaches 1 as $n$ increases, assuming that all of the displayed items have relevance probabilities that are not near 0. Call this the non-small precision assumption. The non-small precision assumption will usually hold for $P_1$ because content is usually viewed because of its likely interest. On the other hand if few relevant items exist, then $P_n$ might be close to 0 for $n$ above some value $k$. Then a multidisplay of over $k$ items would not be helpful. Under the non-small precision assumption, the multiplicative character of Equation 3.1 implies that increasing $n$ by even a little moves $P$, the chance that at least one displayed item will be of interest, substantially closer to 1. This will be significant for users, unless $P$ is already near 1 (which is a desirable situation).

*Example 1.* Suppose $P_1 = 0.4, P_2 = 0.35, P_3 = 0.3, \ldots$. Then by Equation 3.1, increasing $n$ from 1 to 2 increases the probability of an item being of interest from .4 to .61, over 50%, while increasing $n$ from 6 to 7 has little to recommend it since this would only slightly increase the probability of an item being of interest, from .861 to .875 or just 1.6%.

*Conclusion.* The formalization, and *Example 1*, show that displaying a modest number of items is sufficient for most values of $P_n$.

*Potential application.* Search engines could improve the usefulness of their returned lists by varying the number of returned items that are displayed based on their quality. In cases where fewer items are displayed, more information about each could be presented.

## 3.2 Direct & Indirect Display, and Prefetch

Define *direct display* as *the presentation of actual content.* Direct display contrasts with indirect display: define *indirect display* as *the presentation of metadata.*

*Metadata*, a library science term, refers to brief characterizations whose principal use is facilitating access to more detailed information (Smith 1996 [54]). An example of metadata is the title+passage material in a Web search engine return list. A similar concept in the IR domain is sometimes described as *document surrogate material* (Hearst 1999 [25]). The direct and indirect display concepts can be extremes of a continuum, as shown by the following examples.

- If only part of an information item, such as a long document, can be displayed then that direct display is also an indirect display describing the rest.

- Blurring the distinction between direct and indirect display may be a design objective, as in Zellweger et al.'s (1998 [[63]]) fluid links.
- User needs can vary so that, for example, the display of a bibliographic reference is indirect if one needs the material itself, but direct if one merely needs to know its author.

Typically direct display gives immediate access to content, while indirect display provides abstraction and summarization.

## 3.3 Direct Display and Multidisplay Combined

A display can be mono- or multi-, and simultaneously direct or indirect, making four combinations (Table 3.1).

Table 3.1 Examples of direct monodisplay, indirect monodisplay, direct multidisplay, and indirect multidisplay.

| | **Direct** | **Indirect** |
|---|---|---|
| **Mono-** | Typical editing of a document | "Press Ctrl-Alt-Delete to begin." |
| **Multi-** | Displays of several windows | Directories; presentations of lists of links |

*Direct multidisplay* is most useful when there is both a limited number of items to display, and sufficient room to display them. Increasing display capacity, which makes it possible to show more and larger items, is possible because ongoing trends of improvement in network bandwidths and display hardware allow ever higher resolutions at lower cost [59]. For example, electronic paper has recently been marketed and electronic wallpaper is becoming plausible as a future technology [14]. Trends in display hardware capacity are significant because even modest %/year increases imply availability of displays with several times the capacity of present displays in the not-too-distant future. Thus the appetite of direct multidisplay for display space will become less of a problem over time. At the same time the summarizing function of the metadata in indirect displays, which is cognitively important, is feasible to retain: it takes relatively little space, and so may simply be presented along with the direct display it corresponds to. These considerations suggest a trend toward use of direct multidisplay, at least until human perceptual capacity becomes a bottleneck.

## 3.4 Direct Multidisplay and Prefetch

One use for direct multidisplay is reducing the number of interface actions. Actions such as moving and clicking a mouse take time and attention that could otherwise be put to other uses, and a multidisplay requires fewer such actions than a monodisplay. For example, in common Web search engines, accessing the content of some URL in a return list requires, (1) inferencing about the content from the metadata in the list, and (2) a click to view the item chosen. To view a second item requires (3) clicking the "back" button on the browser to return to the list, then repeating steps (1) and (2). After viewing a few items, returning to a previously viewed one that needs to be viewed again, an event familiar to many readers, annoyingly requires either remembering which one it was, or rechecking one or more to find it again. The need for such interface actions should be minimized. The following analysis explores *prefetch*, one way that direct multidisplay can conserve user interface actions.

Suppose a system displays at least two items and, after the user views one, views a second. Let this be called a *prefetch*, because the second viewed item was displayed before the first was read. Prefetch can only occur when direct multidisplay is used, because with monodisplay a second item is not even displayed, and with indirect display no item(s) at all are displayed, only their metadata. We wish to characterize the likelihood of successful prefetch in a direct multidisplay.

Let $n$ be the number of items in a multidisplay. *Prefetch* occurs when two things are true: (1) some displayed item is of interest now, and (2) upon finishing with that item one of the remaining $n-1$ displayed items will be of interest. Given some item currently of interest in a multidisplay, define $Q_2$ as the probability of the most likely of the remaining $n-1$ items to be of interest after the viewer finishes the current item. The subscript "$_2$" is used instead of "$_1$" for $Q$ because a reasonable guess for the value of $Q_2$ would be that it is close to $P_2$ as defined in Section 4.1.2. Define $Q_3$ similarly as the probability of the second most likely of the remaining items to be of interest after the viewer finishes the current item, and analogously for $Q_4, ..., Q_{n-1}$. Then

$$\begin{aligned} P_{none} &= p\ (none\ of\ the\ remaining\ n-1\ items\ will\ be\ of\ interest\ next) \\ &= (1-Q_2)(1-Q_3)(1-Q_4)...(1-Q_n) \qquad (3.2) \end{aligned}$$

Therefore the probability that the prefetch occurs when a first item is of interest, denoted $P'$, is

$$\begin{aligned} P' &= p\ (prefetch\ occurs \mid a\ first\ item\ is\ of\ interest) \\ &= 1 - P_{none} \\ &= 1-(1-Q_2)(1-Q_3)(1-Q_4)...(1-Q_n) \qquad (3.3) \end{aligned}$$

The last step is to remove the condition that a first item is of interest, which has probability $P$. This is done by multiplying by $P$ (see Equation 3.1), giving Equation 3.4.

$$p\ (prefetch\ occurs) = P \times [1-(1-Q_2)(1-Q_3)(1-Q_4)...(1-Q_n)] \qquad (3.4)$$

As in Equation 3.1, the non-small precision assumption implies that increasing the value of $n$ offers initially large but rapidly diminishing returns.

*Example 2.* Suppose $P = 0.861$ (the result of *Example 1*). Suppose also that $Q_2 = 0.3, Q_3 = 0.25, Q_4 = 0.2, \ldots$. For $n = 2$,

$$p\ (prefetch\ occurs) = 0.861 \times [1 - (1 - 0.3)] = 0.26.$$

For $n = 3$ however,

$$p\ (prefetch\ occurs) = 0.861 \times [1 - (1 - 0.3) \times (1 - 0.25)] = 0.41.$$

This is a sizeable improvement of 58%. By comparison, the improvement from increasing $n$ from 6 to 7 is only 2.4%.

Since increasing $n$ to high values will usually not result in much improvement, even a lengthy list of links, using indirect display to cram as many as possible onto a screen, would likely have only incremental benefit, and even then the prefetch is quite incomplete since only links and not content are presented.

### 3.4.1 The issue of space available per item

The more items displayed the less space is available for each. When this space is too small to display the interesting part of an item, or worse, enough of it to determine if it is in fact of interest, the gain of displaying more items competes with the loss of not displaying enough of each, making the display an indirect multidisplay, at least for that user. One way to alleviate this problem is to be judicious about what in the item is displayed. For example the Google search engine displays only a phrase or so of each retrieved item, but these phrases are chosen for relevance to the query that retrieved them. Another approach would be to vary the size of the different windows depending on content so as to display enough to meet the user's needs. If the size of the portion of a document that should be displayed could be determined, then a generalized form of Equation 3.1 could be stated and optimized. One approach to determining the portion to display would use automatically derived theme shifts [41] that define natural boundaries of document segments.

# CHAPTER 4. DIRECT MULTIDISPLAY IN MULTIBROWSER

MultiBrowser constructs hypermedia repositories that are structured to support information foraging using a standard Web browser. MultiBrowser takes as input either a list of URLs, or a search engine query which it passes to a Web search engine to get a list of URLs. It then constructs a repository from the listed documents, in two phases as follows.

## 4.1 Phase 1: Document-Scale Processing

MultiBrowser begins by clustering the documents into three groups, using the standard $k$-means algorithm. The distance metric used is the cosine measure in the space of 5-grams, or strings of 5 characters (Damashek 1995 [13]). Mayfield and McNamee (1998 [40]) compared this metric with use of full words and found that 5-grams were competitive [40]. For each document, the inverses of its distances to the centroids of the three clusters, arbitrarily labeled the red, green, and blue clusters, are used to compute intensities of red, green, and blue (RGB) components of a composite color. The composite color is presented in a color bar directly above the document when it is displayed in a window (Figure 2.3). The color bar provides a summary of the document's location in the space of documents. That in turn allows the user to visually compare its similarity to documents in other windows by comparing their color bars, which they may wish to do in deciding which window to read next. Each color bar also contains a hyperlink that, when clicked, loads the corresponding document into the full browser frame for a closer look. Further discussion regarding color bars is given in Chapter 5.

## 4.2 Phase 2: Paragraph-Scale Processing

After a color bar has been computed for each document, the documents are segmented into paragraphs and the set of paragraphs is processed. As in the case of documents, each paragraph is mapped to a normalized point in 5-gram space, and points are compared using the cosine similarity metric. For each paragraph, the most similar other paragraphs are identified, regardless of what documents they are in, so that later the display can show the paragraph and the other paragraphs most similar to it, each in its own window. This will occur in response to a user click on a "FIND SIMILAR" link (Figure 2.3).

Immediately preceding each paragraph an anchor (HTML <a name=...> tag) is inserted. Immediately after each paragraph a "FIND SIMILAR" link is inserted which points to a unique HTML file containing a <frameset...> tag specifying three, four or six windows, depending on the value of the *windowNum* configuration parameter provided to the system at repository creation time. Figure 2.3 shows the case of six windows. The number of windows is specified because presumably the optimal number is not the same under all circumstances; research is needed regarding the optimal number of windows given the display and information environment. Each window displays a document starting at one of the anchors preceding a paragraph in the document. The top left window redisplays what was in the window containing the "FIND SIMILAR" link that was just clicked. The other $windowNum - 1$ windows display documents starting at the anchors preceding each of the $windowNum - 1$ paragraphs in the repository that were rated as most similar to the one whose "FIND SIMILAR" link was clicked. Thus each link in essence has *windowNum* targets. The value of such multi-tailed links was explored as early as the 1980s [55]. Each window is scrollable, allowing users to move to anywhere in its document. At its top is the color bar computed for the document.

The system is designed to be accessed using a standard Web browser. If the user clicks "NEW" in the "FIND SIMILAR" link, a new set of windows is shown in a new browser frame that pops up over the current browser frame, thereby caching the previous browser frame on-screen behind the new one for easy access later. If instead the user clicks "REFRESH" in the "FIND SIMILAR" link, the new set of windows replaces the current display, which is moved to the browser's history list. (History navigation has been shown to be valuable [10] [56].)

A repository cannot currently be created in real time. However, it can be created overnight, over a lunch break, etc., so that in cases where this delay is acceptable a custom repository can be created. Alternatively, WWW *alcoves*, pre-computed repositories on specialized topics, can be created and placed on the Web. Ultimately, real-time repository creation would be useful. One approach to this would be to allow a user to start foraging after just one or two documents have been incorporated into the repository, while simultaneously adding more documents at the same time as the user continues foraging.

# CHAPTER 5. A NORMALIZED GAUSSIAN CATEGORY MODEL FOR COLOR REPRESENTATION IN MULTIBROWSER

## 5.1 Gaussian Distribution

The usual normal (Gaussian) distribution in one variable ($X$) is given by Equation (5.1)

$$Gaussian(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{x-\mu}{\sigma})^2} \tag{5.1}$$

where $\sigma$ is the standard deviation and $\mu$ is the mean or expected value. This normal curve is a probability density function, so it has the special property that $\int G(x) = 1$. In this equation, $x - \mu$ is the Euclidean distance of the point $x$ in a one-dimensional space to the mean value $\mu$.

## 5.2 Normalized Gaussian Method

Lammen's dissertation [37] used a normalized Gaussian Category Model to represent color perception. His method was motivated by general results given by Shepard [49]. To use this approach here, we drop the factor $\frac{1}{\sigma\sqrt{2\pi}}$ because we do not need $\int G(x)$ to be 1. This leads to the variation of the Gaussian function in Equation (5.2).

$$N(x) = e^{-\frac{1}{2}(\frac{x-\mu}{\sigma})^2} \tag{5.2}$$

This function has some interesting properties as a color space model:

- The maximum value occurs at $\mu$, regardless of the value of $\sigma$.
- The value of the function decreases monotonically (but not linearly) as a function of the distance from $\mu$.
- It is strictly positive everywhere.
- It has only 2 mathematical parameters, $\sigma$ and $\mu$.

## 5.3 Color Generation in MultiBrowser with the Modified Gaussian Method

From the normal Gaussian distribution, we have derived a method to describe how close two values are to each other. The equation is given as follows:

$$Close(x) = e^{-\frac{1}{2}(\frac{x}{\sigma})^2} \tag{5.3}$$

where $Close(x)$ is the closeness between values. Here $x$ is the angle between two document 5-gram vectors, in radians.

Now let us see how the modified method can be applied to our MultiBrowser system. A set of document is clustered into three clusters of vectors in 5-gram space. Three centroids, one for each of three primary colors $R$, $G$ and $B$ are represented by three $N$-dimensional vectors. The angles between a document vector and the three centroid vectors can be calculated as follows.

Suppose a document has a 5-gram vector $X$, and one of the three centroids has a 5-gram vector $C$, that is,

$$X = (x_1, x_2, ..., x_D)$$

$$C = (c_1, c_2, ..., c_D)$$

where D is the hash table size which determines the number of elements in these vectors. First these two vectors are normalized to have length = 1. That is,

$$v_i = \frac{x_i}{\sqrt{\sum_{i=1}^{D} {x_i}^2}}$$

and

$$w_i = \frac{c_i}{\sqrt{\sum_{i=1}^{D} {c_i}^2}}$$

where $V$ and $W$ are the normalized vectors satisfying $\sum_{i=1}^{D} {v_i}^2 = 1$ and $\sum_{i=1}^{D} {w_i}^2 = 1$. Then a value $S$ is calculated using the following equation:

$$S = \sum_{i=1}^{D} v_i \times w_i \tag{5.4}$$

$S$ falls in the interval [0, 1], and is the cosine of the angle between vectors $V$(or $X$) and $W$(or $C$). Then an arccosine function is simply applied to obtain the angle value, in radians, between the two vectors.

Next we will see how to use the angles to generate colors. For example, suppose that a document is represented by the vector $X$. The centroids are represented as vector $R$, $G$ and $B$. Let the angle between $X$ and $R$ be $\alpha$, the angle between $X$ and $G$ be $\beta$, and the angle between $X$ and $B$ be $\gamma$. Thus $Close(\alpha)$, $Close(\beta)$ and $Close(\gamma)$ describe how close the document $X$ is to the centroids $G$, $R$ and $B$ respectively, and can be used to generate three component colors of the composite color of the color bar for document $X$.

This satisfies our color scheme constraints, and can replace the cosine-based metric in computing the the color bar color. Meanwhile, by modulating the value of $\sigma$ we can affect the size and dispersion of the color space in our color scheme. The non-linear decrease in the function with distance from a centroid reflects the general shape of psychological categories as discussed by Shepard [49]. However we need to provide a value for parameter $\sigma$.

## 5.4 Determining the Parameter $\sigma$

Equation 5.3 tells us that when $x = 0$, indicating two overlapping vectors, $Close(x) = Close(0) = 1$ which is the maximum value. On the other hand, when $x = \frac{\pi}{2}$ (indicating two orthogonal vectors), $k = Close(x) = Close(\frac{\pi}{2})$ which is the minimum possible value $k$. In Equation (5.3), that minimum is determined by parameter $\sigma$.

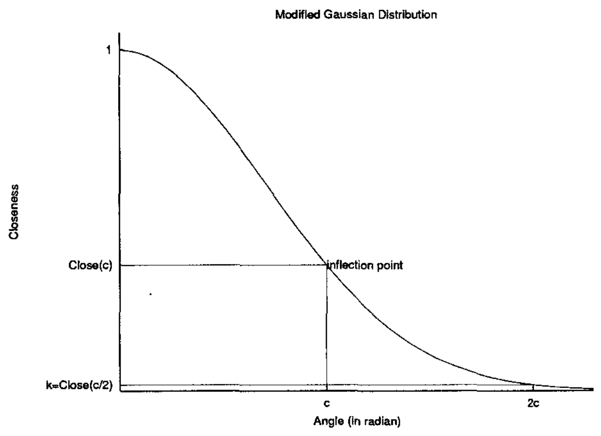


Figure 5.1 The curve for the modified Gaussian distribution.

Figure 5.1 shows the curve for the modified Gaussian distribution. Suppose the point ($c$, $Close(c)$) on the curve is the inflection point. The value $k$ is determined by $Close(2c) = k$. The reason to choose that particular way to determine $k$ is because we wish a steep curve throughout the closeness range, which is the interval [$k$, 1] for the angle, whose range is [0, $\frac{\pi}{2}$]. Therefore we set $Close(2c) = Close(\frac{\pi}{2})$, implying that

$2c = \frac{\pi}{2}$, that is,

$$c = \frac{\pi}{4}. \tag{5.5}$$

The point $(c, Close(c))$ is the inflection point on the curve because the second derivative of the function there is 0:

$$\frac{d^2(Close(x))}{dx^2} = 0,$$

that is,

$$\begin{aligned} \tfrac{1}{\sigma^2}e^{-\frac{x^2}{2\sigma}}(\tfrac{x^2}{\sigma^2} - 1) &= 0 \\ \tfrac{x^2}{\sigma^2} - 1 &= 0 \\ x^2 &= \sigma^2. \end{aligned}$$

Thus we have that when $x = \sigma$, the point $(\sigma, Close(\sigma))$ is the inflection point.

So we know that

$$\sigma = c. \tag{5.6}$$

Equations (5.5) and (5.6) imply that $\sigma = \frac{\pi}{4}$ and $k = Close(\frac{\pi}{2})$.

## 5.5 Color Saturation

The calculation just described and performed three times, once for each of the three distances of a document to the three centroids The three resulting values of $k$ will be utilized to generate the three component colors of the document's color bar. We know that $Close(x) \in [k, 1]$ and the color component values falls between 0 and 255. A stretch transformation, Equation (5.7) is used map a value in the interval of $[k, 1]$ to a value in the interval of [0, 255].

$$Saturation(p) = 255 \times \frac{p-k}{1-k} \tag{5.7}$$

where $p = Close(x)$ and $k = Close(\frac{\pi}{2})$. Thus we can generate three components as follows.

Suppose $R$, $G$ and $B$ are the three centroid vectors. Let the angle between the document $x$ and $R$ be $\alpha$, the angle between $x$ and $G$ be $\beta$, and the angle between $x$ and $B$ be $\gamma$. The three component values $R(x)$, $G(x)$ and $B(x)$ are calculated as follows:

$$\begin{aligned} R(x) &= Saturation(Close(\alpha)) \\ G(x) &= Saturation(Close(\beta)) \\ B(x) &= Saturation(Close(\gamma)) \end{aligned} \tag{5.8}$$

Thus a color is formed based on the three component values, and will be used as the color of the color bar for that document to help users visually compare the similarities in color of different documents.

# CHAPTER 6. EXPERIMENTS

Two experiments were carried out. The first experiment was to check on user impressions of the MultiBrowser system. Then, since the purpose of the color bars is to support document similarity judgments, we investigated the ability of subjects to use color bars to make such judgments. (The experiment sheets are in Appendix A.)

## 6.1 User Impressions

Checking whether users could comprehend and use the MultiBrowser interface architecture could be addressed with a small group. We used six volunteers. None had prior experience with MultiBrowser. Half browsed a MultiBrowser repository on powered parachuting and half one on vegetarian cooking. Repositories were configured to display with six windows. Each subject spent several minutes browsing at leisure following a short training session, then wrote free form answers to questions on their impressions.

*Results.* Subject responses were content analyzed, a standard social sciences method (Evans 2003 [15]), by examining them for key words and phrases relating to three questions (Tables 6.1-6.2). Table 6.1 shows that subjects tended to consider the density of information displayed by MultiBrowser to be comparatively high. Table 6.2 shows that all subjects considered MultiBrowser's "FIND SIMILAR" links to be an efficient way to find relevant information (though we present no objective evidence that they actually are).

Table 6.1 Analysis of responses to the request, "comment on the density of relevant information appearing, on average, on the screen when MultiBrowser is used vs. when ordinary display is used."

| **Density of Information** | **# of Answers** |
|---|---|
| Too high | 2 |
| Above normal | 2 |
| A cceptable | 2 |
| Below normal or too low | 0 |

Table 6.2 Analysis of responses to the request, "comment on the density of relevant information appearing, on average, on the screen when MultiBrowser is used vs. when ordinary display is used."

| **Efficiency** | **# of Answers** |
|---|---|
| High, or above normal | 4 |
| Efficient | 2 |
| Low, or below normal | 0 |

## 6.2 Experiments on Color Bars

In an information foraging environment, automatic assessments of text similarity can help when an interesting text inspires the desire to read other, similar texts. The paragraph hyperlinks in MultiBrowser support navigating among similar paragraphs. MultiBrowser also supports navigating among similar documents by including a color bar for each window (Figure 2.3). Clicking the color bar of a window brings up the document in a large, single-window display. The user can also condition a decision about what paragraph hyperlink to click by taking the color bar of the containing document into account. Because the purpose of the color bars is to facilitate judgments of similarity among documents, the speed and consistency of such judgments were investigated.

### 6.2.1 Experiment A: Response Speed

This experiment compared the color bar metadata with a typical example of text metadata, the short title+passage descriptions provided by AltaVista. In this experiment two document sets, of ten each, were identified. One set was from the powered parachuting repository and the other was from the vegetarian cooking repository. Eighteen subjects were tested, nine on each set. For testing purposes all document content was removed, leaving only the color bars. Then for each color bar $b$ (representing a document) in a set, subjects attempted to identify which two of the other color bars in the set were most similar to $b$. The AltaVista-provided title+passage metadata were used as a control condition under the assumption that they were representative of non-query-dependent textual metadata representations of documents. While viewing this metadata subjects were asked, as in the color bar condition, which two others in the set seemed most similar to $b$ for each metadata item $b$.

*Experiment A results* (shown in Appendix B). Every subject was faster on the color comparison task than on the title+passage task ($p < 0.00001$, two-tailed sign test). On average subjects completed the color comparison task $2.9x$ faster.

### 6.2.2 Experiment B: Response Consistency

Consistency across subjects makes sense as an important aspect of the quality of similarity judgments. Thus this experiment assessed the consistency across subjects of their similarity judgments about color bars. A tendency for subjects to agree that particular documents are similar based on presented metadata suggests that the metadata (either color or the title+passage control condition) supports similarity judgments that have objective reality. On the other hand, lower agreement across subjects suggests less ability of the metadata to support objective similarity judgments, forcing subjects to either make subjective judgments or at best to rely on subjectively chosen criteria. As in the response speed experiment, each comparison problem involved choosing the two color bars most similar to the color bar of a document $b$, or (the control condition) picking the two most similar AltaVista-provided title+passage excerpts to that of $b$. Each of 9 subjects performed both the experimental and control tasks for each comparison problem they solved. None of the subjects had used MultiBrowser.

*Experiment B results* (shown in Appendix B). In 30 of the 32 comparison problems subjects had a higher rate of agreement with each other about which two color bars were most similar to a given color bar than they did about which two title+passages were most similar to a given title+passage. Thus color bars support judgments of document similarity that are more consistent across subjects than those supported by title+passage excerpts ($p < 0.0001$, two-tailed sign test). There are however a few subtleties in these results. These are discussed next.

## 6.3 Discussion: Color Bars

The color bars technique is not limited to MultiBrowser. It could also be used, for example, as a clue in search engine return lists and other situations where it is desirable to show similarities among members of a set visually. Although color bars are different from the standard title+passage data, both are ways of representing documents with metadata. Therefore a comparison of their abilities to represent documents is desirable. Despite their large differences, it is possible to do such a comparison for well-defined measures. The two measures we tested were (1) the speed, and (2) the consistency across users, of similarity judgments. Color bars were found to support similarity judgments among documents that were both faster and more consistent across subjects than the traditional title+passage excerpts provided by AltaVista.

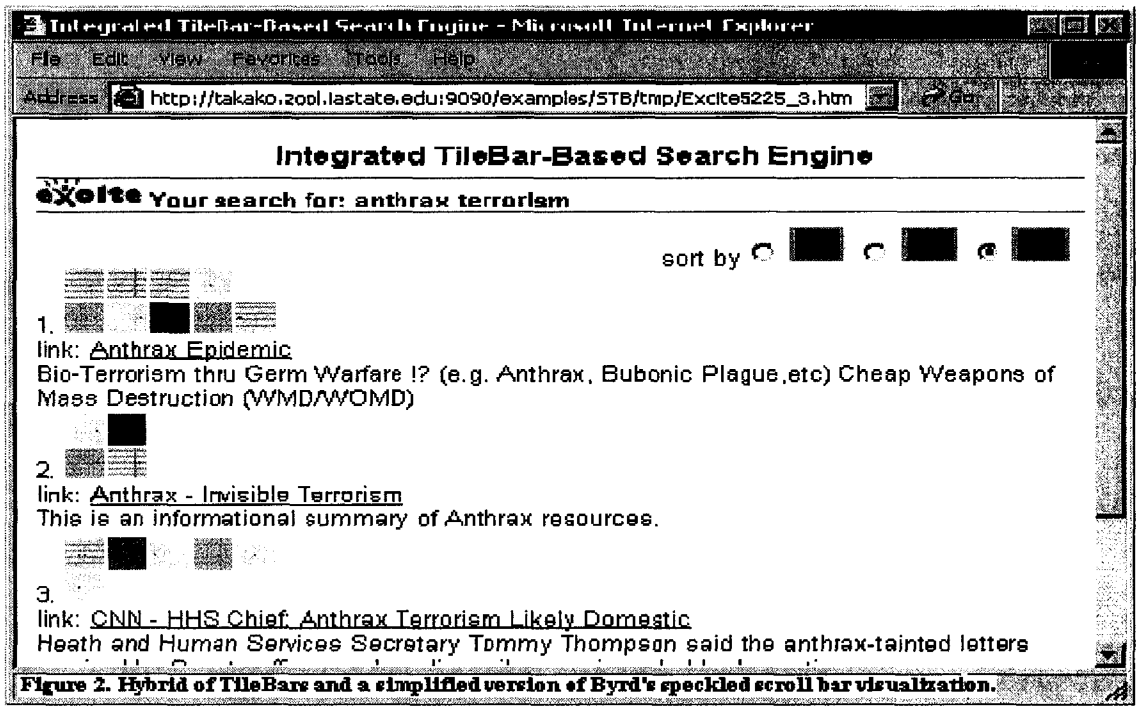
The issue of consistency across subjects of similarity judgments needs further comment. The fact that subjects tended to agree with each other more on similarity judgments of color bars than of title+passages does not prove that their similarity judgments were more correct for color bars. In principle, they could agree while being incorrect. To resolve this question would require a gold standard for correct similarity judgments. The cosine measure on document 5-gram vectors, used for measuring text similarity in the work reported here, is often considered reasonable but other reasonable measures could be proposed that might give significantly different results. Indeed, assessments of document similarity that account for the range of cognitive processes that could legitimately underlie a judgment of similarity probably need to provide answers that are no more specific than a range of possibilities. Thus, it is unclear how to obtain a gold standard that would be able to legitimately classify any given subject's similarity judgment as incorrect.

We also did not address the questions of whether some other method of generating text metadata would lead to different results, or of whether color blindness, present in different varieties and in different degrees in a significant minority of the population, was present in any subjects or affected the results. Nevertheless, the results do suggest color bars as a design space alternative to the typical title+passage metadata for the purpose of supporting fast, consistent document similarity judgments by users.

Although color bars out-performed text metadata in supporting speed and consistency in similarity judgments, the fact that the color assignment method requires the document set to be divided into three clusters, one for each of the three primary colors, deserves further comment. In particular, what happens if using the most natural clustering, however that might be defined, would result in other than three clusters?

Except in mathematically unusual cases, if the most natural grouping is one cluster, then some reasonable way to partition that cluster into two or more clusters can be chosen. If there are two clusters, then one of them can similarly be segmented into two or more new clusters, making a total of three or more. If there are four or more, then some pair of clusters is at least as reasonable to merge into one as any of the other pairs, enabling reducing the total number of clusters. Repeating the process outlined will eventually yield three clusters as desired.

Other bar-based document visualizations have also been described. TileBars (Hearst 1995 [24]) is a well-known example. The speckled scroll bar visualization of Byrd (1999 [9]) is another example. These visualizations contain internal structure representing facts about the internal structures of documents they represent. In our lab we have shown the feasibility of combining these concepts in a browsing system (Figure 6.1). The result is a TileBar in shades of gray, augmented with colored speckles — a speckled TileBar. To combine that with the color bar concept as well is clearly possible: the gray levels in a speckled TileBar would need to be replaced with brightness levels of the color assignment algorithm of Chapters 4 and 5.


Integrated TileBar-Based Search Engine - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address http://takako.zool.iastate.edu:9090/examples/STB/tmp/Excite5225_3.htm
Go
sort by

Integrated TileBar-Based Search Engine

excite Your search for: anthrax terrorism

1.
link: Anthrax Epidemic
Bio-Terrorism thru Germ Warfare !? (e.g. Anthrax, Bubonic Plague,etc) Cheap Weapons of Mass Destruction (WMD/WOMD)

2.
link: Anthrax - Invisible Terrorism
This is an informational summary of Anthrax resources.

3.
link: CNN - HHS Chief: Anthrax Terrorism Likely Domestic
Heath and Human Services Secretary Tommy Thompson said the anthrax-tainted letters

Figure 2. Hybrid of TileBars and a simplified version of Byrd's speckled scroll bar visualization.

Figure 6.1 Hybrid of TileBars and a simplified version of Byrd's speckled scroll bar visualization. (Thanks to Dongping Xu [6].)

# CHAPTER 7. LINKING STRUCTURES IN MULTIBROWSER

As described in Chapters 2 and 3, each paragraph in a document has a six-target hyperlink pointing to five other similar paragraphs and itself. These outgoing hyperlinks connect paragraphs into a directed graph. The properties and characteristics of the linking structures among paragraphs will be further analyzed and discussed in this chapter.

## 7.1 Background Knowledge

It is important that we distinguish between graphs of links among paragraphs (P-graphs) and graphs of links among documents (D-graphs). P-graphs are generated first from the repositories, and D-graphs from the P-graphs. The concepts of P-graph and D-graph will be explained next.

### 7.1.1 P-Graphs

A P-graph describes the hyperlinks among paragraphs in a repository. It is a directed graph in which each node represents a paragraph. In this investigation, paragraph $i$ in a particular document has hyperlinks to 6 paragraphs, itself and the 5 others computed as most similar to it. A part of the P-graph containing node $i$ is illustrated in Figure 7.1. The directed links indicate hyperlinks from paragraph $i$ to the most similar paragraphs

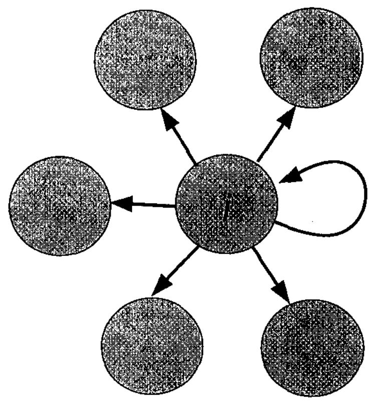


Figure 7.1 A paragraph contains links to the five most similar other paragraphs.

in the repository regardless of what documents these paragraphs are in. In the system that creates the repositories, these are implemented as a single multi-target hyperlink; clicking it brings up a six-window display that simultaneously shows the six targets.

### 7.1.2 D-Graphs

A D-graph is a directed graph showing which documents contain paragraph link(s) to paragraph(s) in which other documents. In a D-graph, each node represents a document in the repository. Directed links between nodes have weights calculated based on the number of paragraph links between the documents, as follows.

Let $A$ and $B$ be two documents, and let $i$ and $j$ be paragraphs, $i$ in A and $j$ in B. If there is a directed link from $i$ to $j$ in the P-graph, we construct a directed link from $A$ to $B$ in the D-graph. The weight of the $A \rightarrow B$ link is the total number of directed links in the P-graph from paragraphs in $A$ to paragraphs in $B$.

Figures 7.2 and 7.3 give an example. Suppose we have a P-graph where document $A$ has 4 paragraphs and document $B$ has 3, and the links between paragraphs in both documents are as in Figure 7.2. Then the corresponding D-graph is shown in Figure 7.3.

A link weight is calculated for each link in the D-graph by adding up all of the links in the P-graph that connect a paragraph in $A$ to a paragraph in $B$. A document weight is then defined as the sum of the link weights of its incoming links. For example, document $A$ has weight 2, while $B$ has weight 4. The bigger a link weight, the more related to each other, in some sense, the two linked documents are. The bigger a document weight, the more central, in some sense, the document is within the repository. Thus the number of paragraph links in the P-graph supporting a given document link in the D-graph is encoded in the weight of the document link, though paragraph links themselves are unweighted.

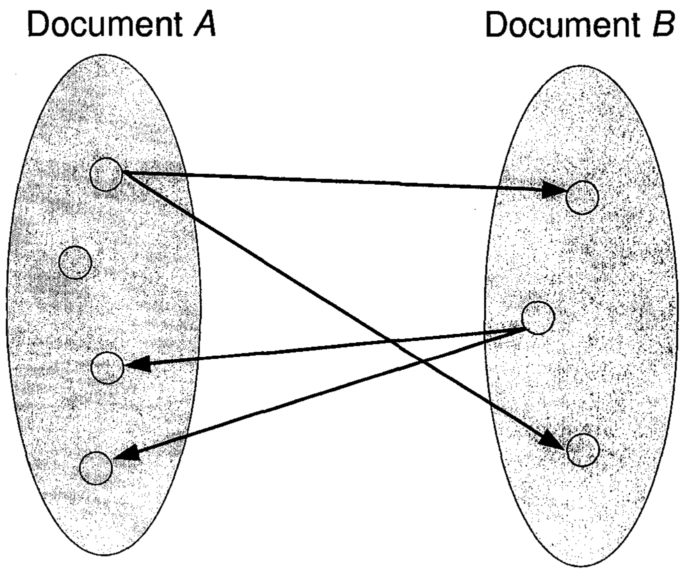


Figure 7.2 P-graph for documents $A$ and $B$.

The P-graphs and D-graphs both represent opportunities for navigation in the repository. A link in a P-graph represents a real hyperlink associated with a paragraph displayed in a web browser. A single click on the paragraph brings up a display of the different destination paragraphs in separate windows. Assuming an entire document is

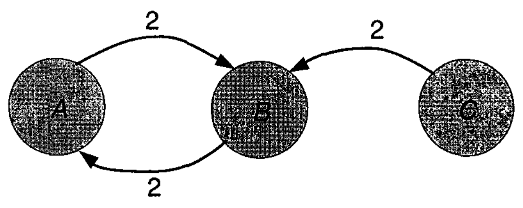


Figure 7.3 D-graph for documents $A$ and $B$ (and $C$).

accessible from any paragraph in it (*e.g.* by scrolling the window containing the paragraph to see other parts of the document), then a link in a D-graph is followed when any inter-paragraph link connecting one document to another is followed.

### 7.1.3 Centrality and Dispersion

If some documents in the D-graph have relatively high weight and others have low or even zero weight, then a user browsing within the repository will have a tendency, perhaps without even realizing it, to move into a "core" of high-weight documents. Low-weight documents on the other hand will tend to be visited rarely or not at all. We call this phenomenon trapping. If a repository has a strong tendency toward trapping it is considered to have high *centrality.* However, if documents tend to have similar weights, the repository is considered to have high *dispersion.*

Our goal is to support effective navigation. This suggests avoiding trapping, while at the same time enabling the user to move to documents that are central to a repository if they wish. A good D-graph is one that has sufficient centrality to draw a user toward the core (the most "relevant" documents) and sufficient dispersion to prevent other documents from being inaccessible to a greater or lesser degree.

## 7.2 P-graph Generating Algorithms

Motivated by the need to generate D-graphs with a desirable balance between centrality and dispersion, we investigated three P-graph generation algorithms to use as input to the D-graph generation process described earlier. These P-graph algorithms are described in this section. The next section investigates the D-graphs that are implied by these P-graphs.

### 7.2.1 Basic P-Graph Algorithm

This is the 5-gram based algorithm described in Section 2. The other two P-graph algorithms are based on further processing of the Basic P-graphs produced by the Basic P-graph algorithm.

### 7.2.2 2-Clan P-Graph Algorithm

An $N$-clan [48] [57] is a graph in which each node has a path of up to length N to every other node, and all these paths contain only nodes in the clan. Such clans can appear as subgraphs of a larger graph. Previous work [57] has shown that clan graphs can help in constructing a collection of high quality. We are interested in 2-clan graphs here. Why 2-clan graphs in particular? Figure 7.4 graphically depicts four types of indirect inter-paragraph relationships that can be derived from the Basic P-graph. Figure 7.4($a$) is simply the relationship stated in the Basic P-graph. Figure 7.4($b$) shows a Basic P-graph in which paragraph $A$ is related to paragraphs $B$ and $C$. This suggests that $B$ and $C$ are related to each other by virtue of being related to $A$. Figure 7.4($c$) shows a Basic P-graph in which paragraphs $B$ and $C$ are related to $A$. This suggests that that $B$ and $C$ are also related to each other by virtue of being similar to $A$. Figure 7.4($d$) shows a limited (two-link) transitivity relationship in a Basic P-graph that suggests paragraph

$C$ is related to paragraph $B$ because $C$ is related to $A$ and $A$ is related to $B$. The links shown in Figure 7.4, augmented with the new relatedness links, are shown in Figure 7.5. By replacing the structures in Figure 7.4, wherever they appear in a Basic P-graph, with the corresponding structures in Figure 7.5, a 2-Clan P-graph is formed.

The relationships in Figure 7.5 are motivated by the 2-clan graph concept. Other relationships derivable from 3-or-more-clan graphs would be more remote and their value might be presumed more questionable.

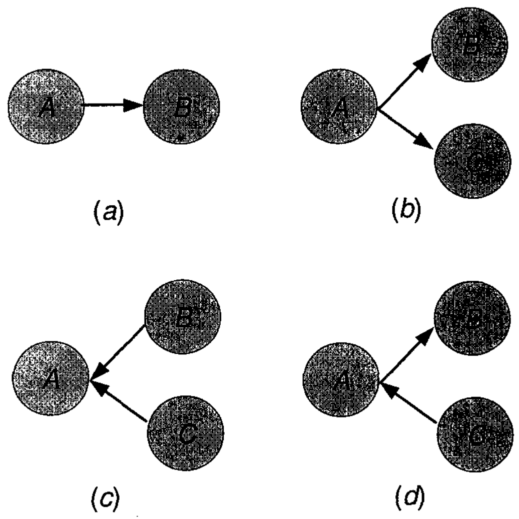


Figure 7.4 Four types of indirect inter-paragraph links.

The 2-Clan P-graph may be used to generate a new D-graph, which will likely contain more edges than the D-graph based on the Basic P-graph because some documents are likely to be linked by paragraph connections in the 2-Clan P-graph that were not linked by paragraph connections in the Basic P-graph.

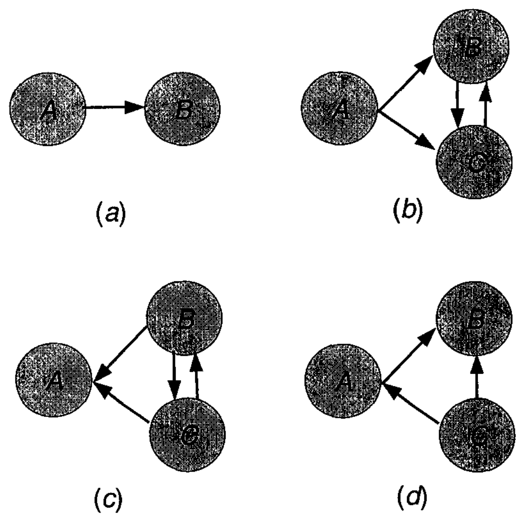


Figure 7.5 Situations depicted in Figure 7.4 with the new, derived paragraph links added.

### 7.2.3 Modified 2-Clan P-Graph Algorithm

We also investigated the D-graphs implied by a P-graph formed by adding to the Basic P-graph only those links newly implied by Figure 7.4(c). The resulting P-graph will be called a Modified 2-Clan P-graph. Modified 2-Clan P-graphs tend to have more links than the corresponding Basic P-graphs, but fewer than the corresponding 2-Clan P-graph.

# CHAPTER 8. LINKING STRUCTURE EXPERIMENTS

I have now introduced three types of P-graphs: Basic P-graphs, 2-Clan P-graphs, and Modified 2-Clan P-Graphs in the previous chapter. Each of these implies D-graphs. We investigated properties of these D-graphs in three different experiments: P-graph type (*Experiment 1*); document retrieval system (*i.e.* web search engine, *Experiment 2*); and topic (*Experiment 3*). Experiment 1 also provides a comparison with the well-known HITS and PageRank algorithms [34] [45], suggesting that the 2-Clan P-graph approach can provide similar results but at lower computational cost.

## 8.1 Experiment 1: P-Graph Types

This experiment investigated how different P-graph algorithms influence centrality and dispersion in the D-graphs they imply. The document set used was that returned by the query *powered parachuting*, submitted to the Altavista search engine. The Basic P-graph, 2-Clan P-graph and Modified 2-Clan P-graph algorithms were applied and the corresponding D-graphs were generated. Based on the resulting D-graphs, documents were ranked according to their weights. Figure 8.1 shows the document weights implied by Basic P-graphs. Figures 8.2 and 8.3 show the document weights implied by 2-Clan P-graphs and Modified 2-Clan P-graphs respectively.

Figures 8.1-8.3 all have curves that decay in a quasi-exponential manner. This indicates relatively high centrality, implying a tendency for users to perhaps unwittingly be

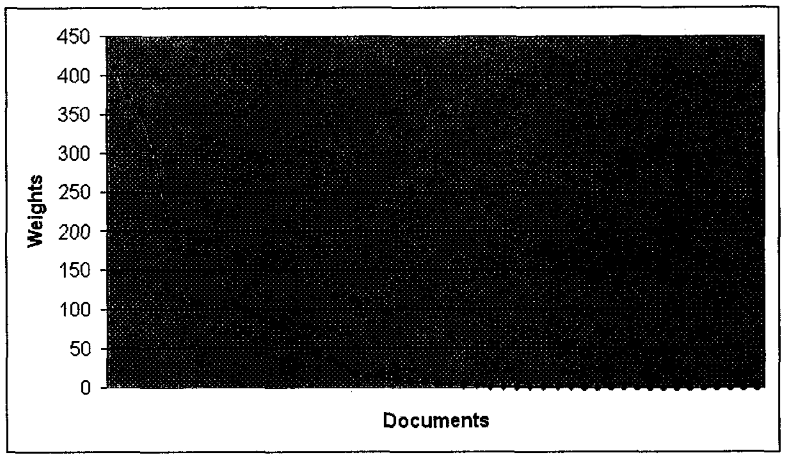


Figure 8.1 Weights of documents in the D-graph of the *powered parachuting* repository resulting from the Basic P-graph.

guided by the repository structure toward browsing documents with high weights, and away from browsing documents with low weights.

Two well-known ranking algorithms, HITS and PageRank, were also implemented and used to order the documents based on the Basic P-graph. The purpose was to compare their results to the Basic, 2-Clan, and Modified 2-Clan P-graph algorithms. Figure 8.4 shows the histogram of the document weights implied by the 2-Clan P-graph method. Figure 8.5 shows the Hub and Authority scores of the documents implied by the HITS algorithm. Figure 8.6 shows the PageRank scores of the documents implied by PageRank algorithm. In the three graphs the same repository as in Figures 8.1-8.4 was used, but only half of the documents, those with the highest weights, are shown.

A comparison of Figures 8.4, 8.5 and 8.6 shows that the relative weights of the documents in the D-graph implied by the 2-Clan P-graph are very similar to the relative document weights implied by the HITS and PageRank algorithms as applied to the Basic P-graph. Yet, the 2-Clan P-graph algorithm is computationally more efficient because it generates document weights in one pass, while the HITS and PageRank algorithms

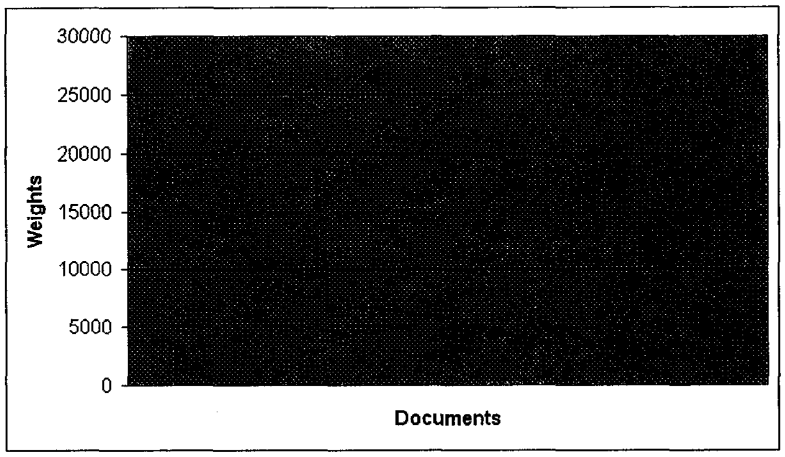


Figure 8.2 Weights of documents in the D-graph of the *powered parachuting* repository resulting from the 2-Clan P-graph.

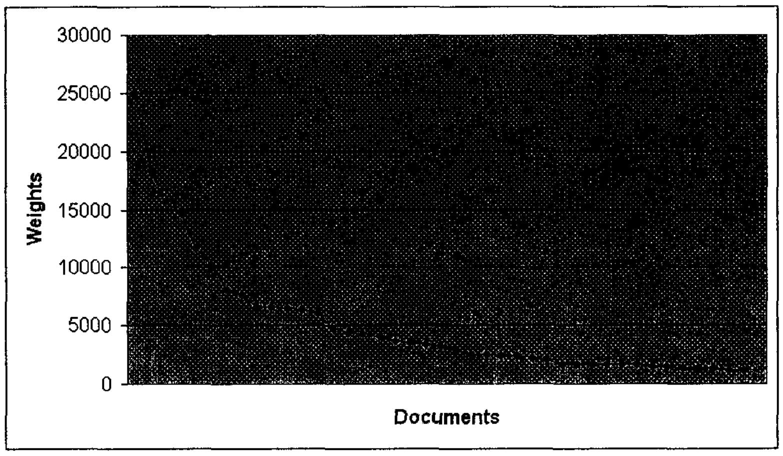


Figure 8.3 Weights of documents in the D-graph of the *powered parachuting* repository resulting from the Modified 2-Clan P-graph.

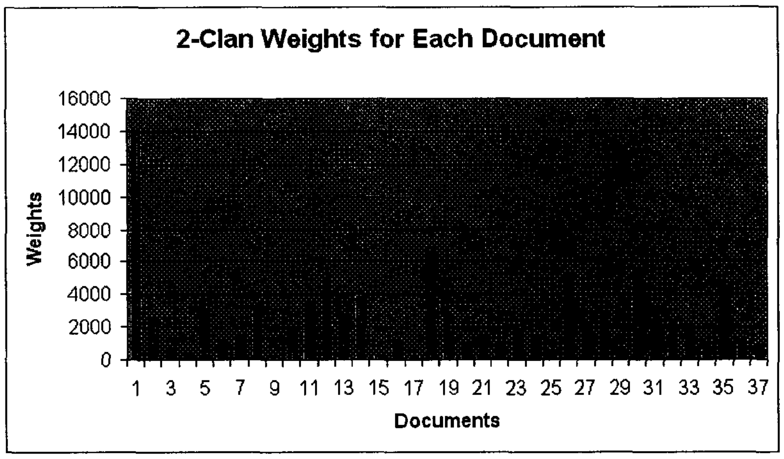


Figure 8.4 Document weights implied by the 2-Clan P-graph generated for a repository of documents returned by Altavista.

are iterative.

## 8.2 Experiment 2: Document Retrieval Systems

This experiment compared different search engines in terms of the centrality and dispersion in D-graphs generated from their returned document lists. We selected five popular search engines, Google, Altavista, Hotbot, Lycos and Ask Jeeves, to generate repositories based on the query *vegetarian cooking*. As in the previous experiment, we applied the Basic P-graph, 2-Clan P-graph and Modified 2-Clan P-graph algorithms to generate P-graphs and their implied D-graphs, and then computed the weights of the documents from each D-graph. Figures 8.7 through 8.9 show the results. These figures show that the search engines all return document lists with similar characteristics. In particular, they lead to D-graphs whose document weights, when ordered high-to-low, give curves with high centrality and thus low dispersion.

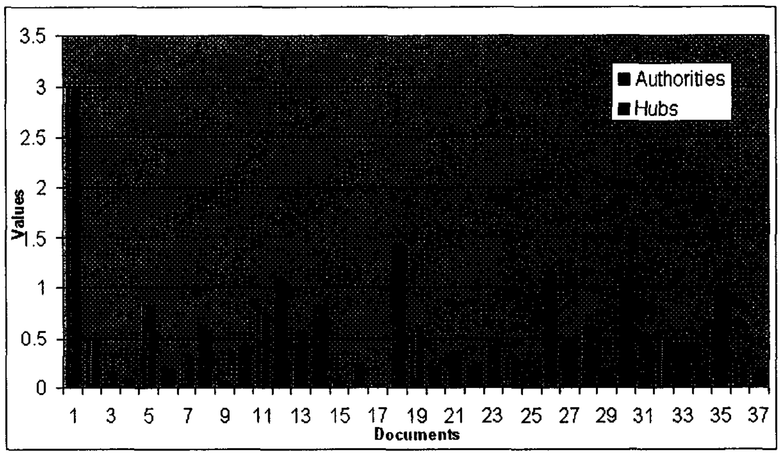


Figure 8.5 Authority and Hub values for each document in the same document set as in Figure 8.4. Values were generated using HITS on the Basic P-graph.

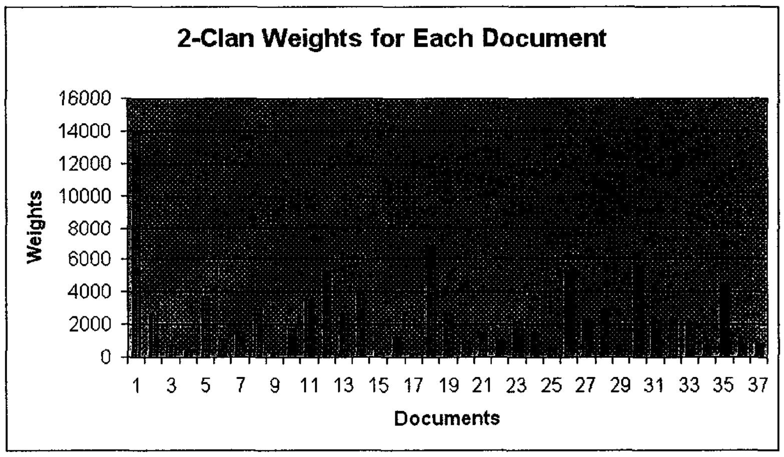


Figure 8.6 PageRank scores for each document in the same document set as in Figure 8.4. Values were generated using PageRank algorithm on the Basic P-graph.

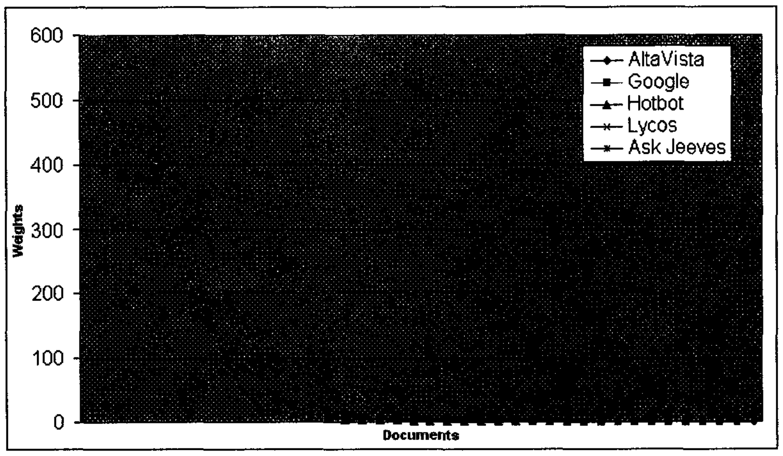


Figure 8.7 Weights of documents, ordered high-to-low, from the D-graphs of each of five *vegetarian cooking* repositories derived from the return lists of five Web search engines. The D-graphs were derived from the Basic P-graphs of the repositories.

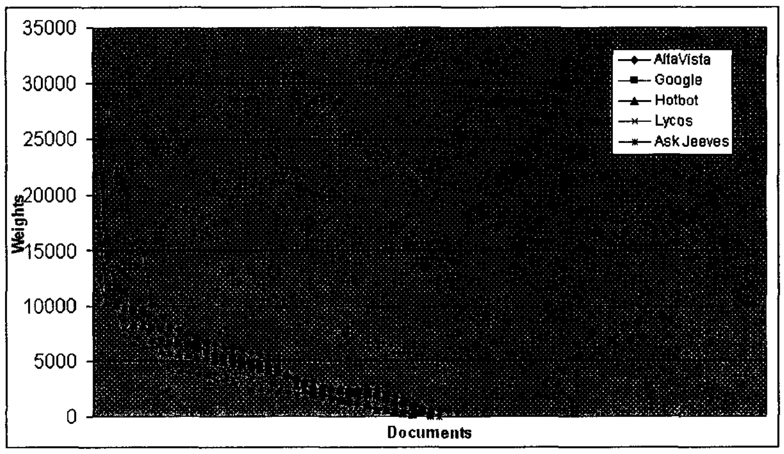


Figure 8.8 Weights of documents, ordered high-to-low, from the D-graphs of each of five *vegetarian cooking* repositories derived from the return lists of five Web search engines. The D-graphs were derived from the 2-Clan P-graphs of the repositories.

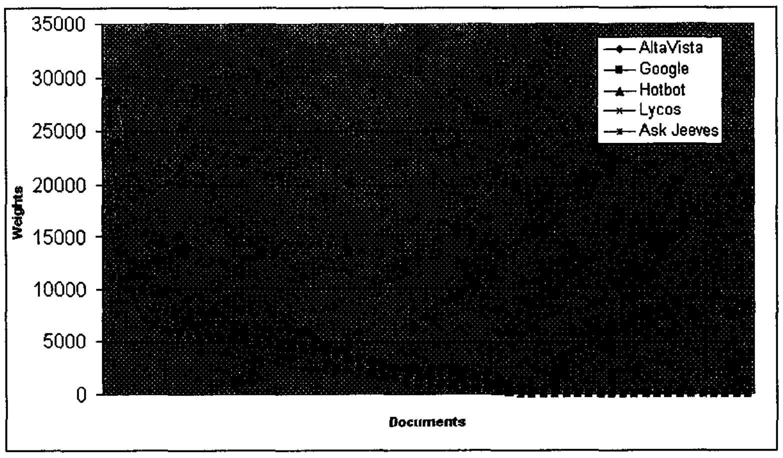


Figure 8.9 Weights of documents, ordered high-to-low, from the D-graphs of each of five *vegetarian cooking* repositories derived from the return lists of five Web search engines. The D-graphs were derived from the Modified 2-Clan P-graphs of the repositories.

## 8.3 Experiment 3: Topics

This experiment examined if quasi-exponential curves (and therefore high centrality and low disperson) occur across different topics. Repositories were generated based on documents returned in response to two different queries, powered parachuting and vegetarian cooking, to five search engines. The two queries had widely varying total numbers of relevant documents returned, one almost 100 times the other. Thus these repositories sample the diversity of repositories in both topic and total number of relevant documents. Figures 8.10 through 8.12 show the results from the powered parachuting query using different paragraph linking algorithms. A comparison with Figures 8.7 through 8.9 shows that both queries resulted in repositories with the same quasi-exponential centrality and dispersion properties.

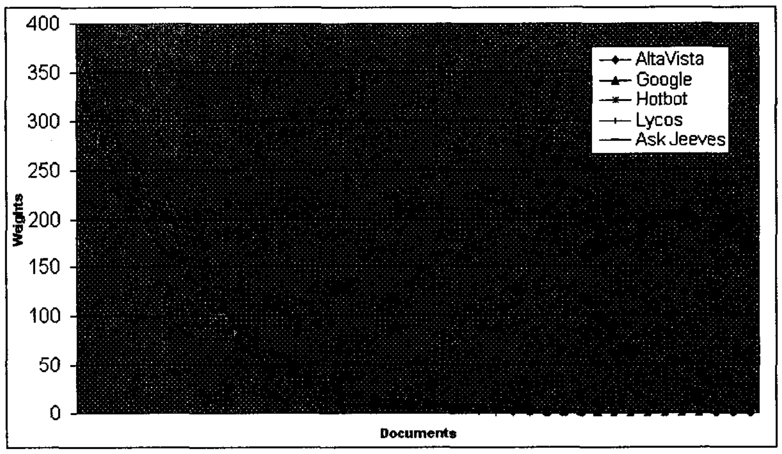


Figure 8.10 Weights of documents, ordered high-to-low, from the D-graphs of each of five *powered parachuting* repositories derived from the return lists of five Web search engines. The D-graphs were derived from the Basic P-graphs of the repositories.

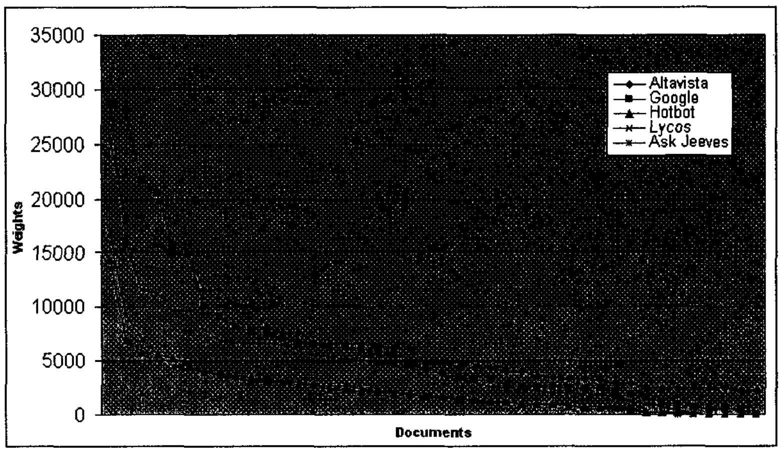


Figure 8.11 Weights of documents, ordered high-to-low, from the D-graphs of each of five *powered parachuting* repositories derived from the return lists of five Web search engines. The D-graphs were derived from the 2-Clan P-graphs of the repositories.

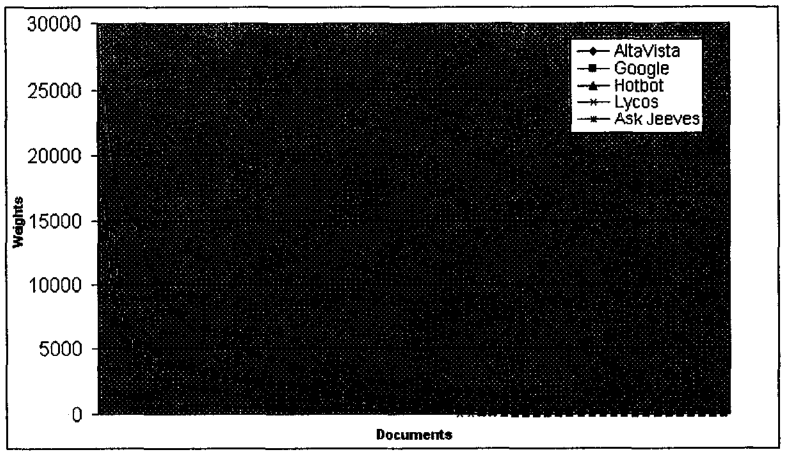


Figure 8.12 Weights of documents, ordered high-to-low, from the D-graphs of each of five *powered parachuting* repositories derived from the return lists of five Web search engines. The D-graphs were derived from the Modified 2-Clan P-graphs of the repositories.

## 8.4 Results

The three experiments indicate a tendency toward quasi-exponential curves that occurs across varied conditions. In particular, it occurred across two contrasting document sets, across multiple search engines and across different P-graph algorithms.

The quasi-exponential curve shapes suggest a Zipf-like relationship [64] [65] [66]. This relationship describes the empirical observation of constant slope for a log-log plot of rank vs. value of ranked parameter. In this case the rank is obtained by ordering documents from highest to lowest weight, and the value of the ranked parameter is therefore document weight. To see how well the Zipf relationship applies here, we took the 30 curves in Figures 8.7 through 8.12 and normalized each curve to have the same height. Then the average height for each $x$-axis value was computed over all 30 curves. Finally the resulting curve was plotted on a log-log scale. Figure 8.13 shows the result. The results show a curve of fairly constant slope over most of its span, but with a

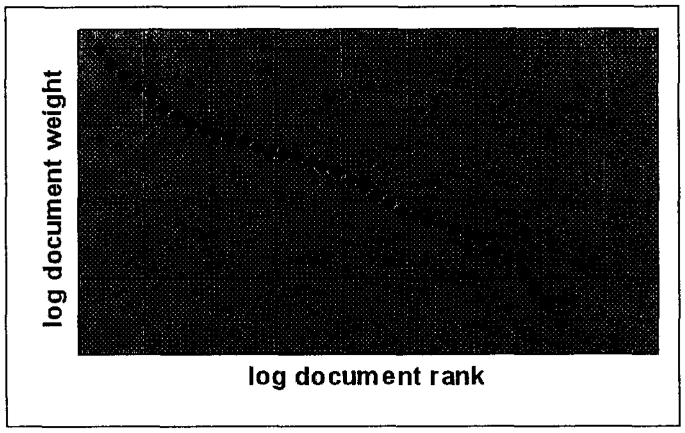


Figure 8.13 Log-log plot of document weight vs. document rank.

noticeable drop-off toward the end where the graininess of the data is largest since document weights are low (a zero weight could not even be plotted on a log scale). Because of the general pervasiveness of Zipf-type curves in text, it is not surprising that it appears here, and it is likely that it applies in other hypertext scenarios as well. This underlines the need to watch for high centrality and consequently trapping in hypertext repositories, and to enable users to avoid trapping when they wish to.

The high degree of centrality in the relevant figures suggests that in each case, a kernel of documents that are most similar has been identified. They presumably reflect the focus of the repository, with other documents being both figuratively and literally in the tail of the curve. However from an information foraging point of view, high centrality implies that users, as they follow links, are likely to move toward and into those documents with the highest weights, and to do so without necessarily realizing it [16] [17]. They may thus fail to be exposed to the rich variety available in the repository. Under the information foraging paradigm, this is a problem. It is exacerbated by the very low; sometimes even zero weights of many of the documents, which tends to make them inaccessible, again preventing the user from experiencing the richness of the repository.

The next section proposes a solution, currently implemented in our MultiBrowser system [6].

# CHAPTER 9. NAVIGATION IN RING-STRUCTURED REPOSITORIES

We propose a ring-structured solution to support an information foraging style of navigation in repositories with high centrality. The ring structure consists of a kernel containing the documents with highest weights, and two concentric "rings" around it containing documents of intermediate and low weights, respectively (Figure 9.1).

During the process of navigation, the user should be presented with links that are annotated to indicate whether they point to documents in the kernel, near ring or far ring. The user is thus able to decide which links to follow when foraging. The kernel intuitively would be associated with documents of high relevance to whatever criterion or query was used to obtain documents for the repository. The near ring intuitively would contain documents that are likely to be of interest if foraging beyond the kernel is desired. The far ring would contain documents peripherally related to the focus of the repository and thus most likely to be of interest when foraging goals are more expansive or undirected.

Suppose each paragraph in a repository has six outgoing links, 5 to other paragraphs and 1 to itself. These links might, as in the MultiBrowser system [6], be implemented as a 6-target hyperlink that, when clicked, brings up a 6-window display of all six link targets. Let $k$ be the number of targets to paragraphs in kernel documents. Likewise, let $n$ and $f$ be the numbers of targets to paragraphs contained in near and far ring

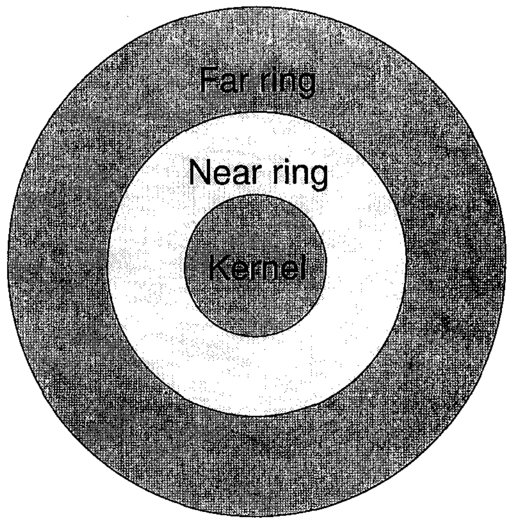


Figure 9.1 Ring structuring of a hypertext repository classifies documents into the kernel, near ring, or far ring.

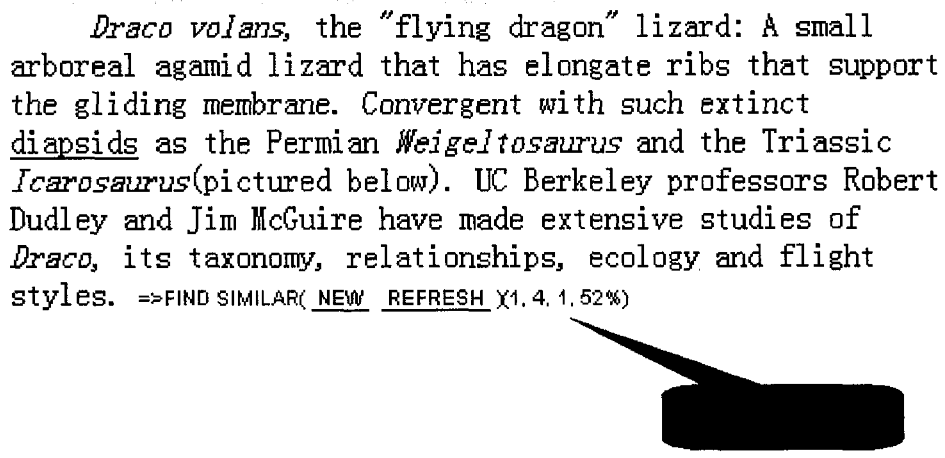
documents respectively. Then we have the following equation:

$$k + n + f = 6.$$

Clicking the 6-target hyperlink brings up a 6-window display showing $k$ paragraphs in kernel documents, $n$ paragraphs in near-ring documents, and $f$ in far-ring documents. We introduce a parameter $r$ paragraphs to describe the degree to which the 6-target hyperlink emanating from a paragraph tends to direct browsing toward the kernel:

$$r = \frac{100 - 8\times n - 16\times f}{100} \tag{9.1}$$

Figure 9.2 shows an example of a paragraph with explicit annotations of its ring information. It contains a tuple of four elements. In the example, $k = 1$, $n = 4$ and $f = 1$, which means that out of 6 outgoing links, 1 points to a kernel document, 4 points to near ring documents and 1 points to a far ring document. The value $r$ is calculated in Equation 9.1 and displayed in Figure 9.2 in the annotated tuple. It describes the degree to which the paragraph tends to lead the navigator to the kernel.



*Draco volans*, the "flying dragon" lizard: A small arboreal agamid lizard that has elongate ribs that support the gliding membrane. Convergent with such extinct diapsids as the Permian *Weigeltosaurus* and the Triassic *Icarosaurus*(pictured below). UC Berkeley professors Robert Dudley and Jim McGuire have made extensive studies of *Draco*, its taxonomy, relationships, ecology and flight styles. =>FIND SIMILAR( NEW REFRESH )(1, 4, 1, 52%)

Figure 9.2 A sample of a paragraph with explicit annotations of ring information.

If $n = 0$ and $f = 0$ that means all targets are paragraphs contained in kernel documents, and $r = 1$. If $n = 6$ and $f = 0$ that means all targets are paragraphs contained in near ring documents and $r$ is close to 0.50, indicating neither a tendency to direct browsing toward the kernel nor away from it. If $n = 0$ and $f = 6$, all targets are paragraphs contained in far-ring documents and $r$ is close to 0, indicating a 6-target hyperlink that points away from the kernel. The trapping problem is alleviated because hyperlinks are annotated with numbers that give the guidance needed for a user to forage without being trapped in a small subset of the documents.

# CHAPTER 10. SOCIAL FILTERING IN MULTIBROWSER

## 10.1 What Is Social Filtering?

*Social Filtering,* also called collaborative filtering, is an emerging technique for dealing with a large amount of information in various information environments with different information structures, where information is filtered for a user based on the likes of other people with similar tastes. Originally, Malone [39] characterized social filtering as describing social connections between people. Nowadays, in many systems a variation is used: user profiles are compared with each other and weighted for their degree of similarity. Groups of similar profiles are formed and users belonging to one group will be served the same set of objects. Therefore, social filtering systems require a critical mass of participants and objects to work well.

Collaborative filtering systems work by including people in the filtering system. After all, we might expect people to be better at evaluating documents than a computed function. Current automatic filtering systems attempt to find articles of interest to their user, often using some scoring function to evaluate features of the documents and returning the documents with the highest scores. People can effortlessly evaluate features of a document that are important to other people, but would be difficult to detect automatically.

When a reader is accessing a document, it is normally impossible to tell who was the first to access it or whether it is the most commonly used reference in the system.

Collaborative filtering alleviates this in part by associating with computer documents the history of their usage. Analogously, Hill et al. [26] make the observation that the objects we use in our everyday lives accumulate wear and tear as a normal part of their use: pages in books become wrinkled, bindings creased, and margins smudged with fingerprints [3]. The objects with more wear are the more commonly used ones, and further the wear acts as an index to relevant information inside the object. An example is the way reference books sometimes open to commonly used pages when dropped on a desk. Giving searchers access to this usage history lets them take advantage of the type of subtle hints that we commonly use when making read/don't read decisions in the real world.

Much of the work on social filtering for the Web has focused on so-called explicit methods, which means that readers annotate a document by simply adding an explicit rating value to it. A distinct drawback of this method is that it needs extra efforts from raters to evaluate the web pages. This may cause extra costs and issues of mutual trust between raters. In contrast, the implicit approach requires no extra efforts from the raters, but has the disadvantage that the rating information provided tends to be simpler and, potentially, of lower value. Thus researchers are attempting to find some middle way between explicit and implicit approaches with low raters' efforts and high rating value.

## 10.2 Related Work

### 10.2.1 Tapestry

*Tapestry*, developed by Xerox [18] was the first system to support collaborative filtering. It allows its users to annotate the documents they read. Other Tapestry users can then retrieve documents to read based not only on the content of the documents themselves, but also on what other users have said about them. Tapestry provides free text annotations as well as explicit "like it"and "hate it" annotations so users can pass judgments on the value of the documents which they read.

In its current incarnation, Tapestry suffers from two distinct problems. The first problem is the size of its user base. Because Tapestry is based on a commercial database system it cannot be given away freely. Further, Tapestry was not designed for use by large numbers of people at distributed sites. Both these factors combine to limit the pool of potential Tapestry users to researchers at Xerox PARC. Based on anecdotal evidence, this pool does not seem large enough to support a critical mass of users. The vast majority of documents go unannotated, so there is little collaborative information to use when filtering. The second problem with Tapestry is the means by which users enter filters into Tapestry. One common interface to Tapestry requires users to specify requests for information in the form of queries in an SQL-like language. Writing such a query requires the user to have a firm sense of what types of articles he wants to read, which is a hindrance to exploration of new areas and makes it hard to browse the available information for serendipitous hits.

### 10.2.2 GroupLens

GroupLens [36] is for filtering Usenet news postings. In this system, communities of users rank the articles they read on a numerical scale. The system then finds correlations between the ratings different users have given the articles. It uses an implicit measure by recording readers' reading time to give an evaluation value such that the information is gathered cost-free. Meanwhile, the issue of trust is resolved through community and social interaction. People learn of one another's interests and experiences, and reputations develop that establish the value of the rating assisted by a statistical method for combining historical matches.

GroupLens does achieve the goal of relevant implicit filtering. However, there are numerous reasons why this method cannot be simply ported to various other information environments, such as the Web. In general, GroupLens records the time spent reading a Usenet posting as an implicit measure of the reader's valuation of it. In principle, therefore, Usenet ratings can be gathered cost-free. Usenet has many features essential for social filtering. First, Usenet is established on the concept of community. News groups provide memberships which is crucial to collaborative activities [61]. Second, news groups are designed to be interactive with news readers. This makes the information accessible to recipients, and thus makes the rating of information relatively easy to implement. In contrast, the Web is founded on an abstract information model, where information is organized as a site, rather than as a community or a group. Furthermore, the Web is inherently less interactive than news groups: information generation and consumption are mostly clearly separated.

## 10.3 Social Filtering Implementation in MultiBrowser

### 10.3.1 Important Concepts

MultiBrowser supports paragraph similarity analysis and could potentially incorporate social filtering. Elements that are conceptually important in implementing social filtering, and may be considered are:

- readers — whoever reads a paragraph or the document should be able to leave feedback on the quality of the information. The types of feedbacks could be simply a binary sign (+/-) indicating if the object is good or bad. Or feedback could be a score computed through a scoring algorithm based on various parameters. Feedback could also be textual, such as a guestbook.

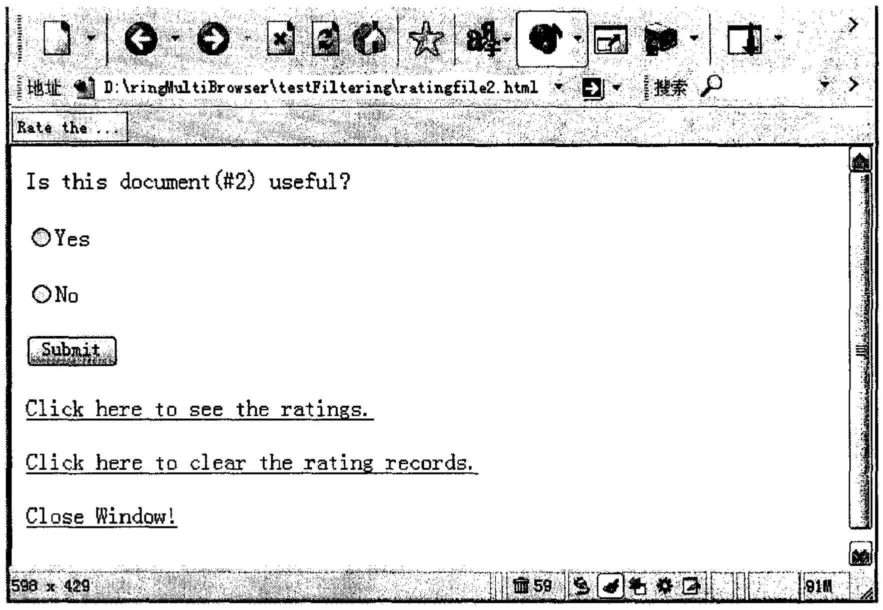
- objects — the unit of analysis could be either a paragraph or a document depending on which the reader is interested in. Our current system uses an n-gram based 2-clan graph algorithm to assess paragraph similarity, which proves to be able to generate topically coherent sites [57]. The advantage of this algorithm is to easily incorporate user profiles, which is an essential feature in social filtering.

- time — when a particular object is accessed and how long the user has spent on it. This can be an important factor in determining users' reading behaviors. This can be implemented implicitly using the system clock.

### 10.3.2 Information Rating Prototype

We have built a prototype implementation of social filtering in MultiBrowser, shown in Figure 10.1. Each document is associated with a form for its rating information. Each user is able to rate this document using this form. Currently the rating information is simply binary (good or bad), but could potentially incorporate more complex rating information. Users can tell the system if the document is good or bad in terms of their reading interests. At the same time, users can read the rating information left by previous users. Such feedback information in this example is how many users think the document is good and how many other users think the document is bad. The percentage of "good" evaluations is given as well.

This simple prototype of social filtering implementation in MultiBrowser makes gathering rating information easy and straightforward. Meanwhile, each object retains explicit raw rating information for future use.



Is this document(#2) useful?

○Yes

○No

Submit

Click here to see the ratings.

Click here to clear the rating records.

Close Window!

Figure 10.1 User rating information.

### 10.3.3 Summary

Regardless of which approach is applied, the passive mode takes control over the construction of the ratings, and the users can then use them for their future MultiBrowser-related activities. An advantage of this is that users do not have to be trained on the techniques for controlling rating.

In contrast, the active mode requires each user to distribute rating information among a group of users voluntarily. This means the raters are required to be highly responsible and trustworthy. Otherwise untrustworthy rates can adversely affect the system, negating the value of the rating information.

# CHAPTER 11. CONCLUSION AND FUTURE WORK

## 11.1 Conclusions

MultiBrowser is an information foraging system as well as a first step toward a much more futuristic vision: a system that supports written conversational dialogues between a user and a repository in accordance with a "panel-of-experts" metaphor.

From an analysis of MultiBrowser, quasi-exponential Zipf-type curves were found to characterize automatically generated paragraph-linked hypertext repositories under varied conditions. These curves depict document weights showing that some documents in a repository of query-retrieved documents are the destinations of many more links than other documents, and other documents are the destinations of very few links. Although the full extent to which these results generalize has not been determined, their consistent appearance across varied experimental conditions suggests that the degree to which the phenomenon generalizes to hypertext in general is significant and may be pervasive. Given hypertext repositories with this quasi-exponential property, it is important to enable users to choose a foraging strategy that will allow them, for example, to avoid becoming trapped, perhaps unwittingly, within the repository kernel. A ring-like partitioning of the repository in which documents are categorized into the kernel, the near ring, or the far ring, as well as annotations to hyperlinks that provide numerical clues, has been implemented to provide users with the required information to inform their browsing activities.

## 11.2 Future Work

### 11.2.1 Brief Introduction

There are a number of potentially useful goals for further work in addition to the research on visualizations noted above. One is real-time operation, important because it can be inconvenient to have to wait for the system to process a set of documents into a repository. An algorithm for doing this would process documents into the repository one at a time, and allow the user to browse the repository at the same time as it is gradually increasing in size. The same idea could also apply to adding a newly available document to a mature repository. Another goal is to allow updating of a repository by replacing a document in it with a new version of the same document. One way to approach this would be to decompose it into a document deletion operation preceding the document addition just described.

In addition, two important issues and goals derived from them are considered and introduced next.

### 11.2.2 Internal Hyperlink Consideration

The internal, author-generated hyperlinks that documents typically have seem likely to contain valuable information for repository building that can supplement the 5-gram based analysis. Here is how we might design an algorithm which incorporates both the internal hyperlink information and the 5-gram based "FIND SIMILAR" links already implemented.

First, apply the HITS algorithm to the internal hyperlinks pre-existing among documents that are to be processed into the repository. The authority score of each document is thus obtainable. We also would follow the current document weight generation algorithm and use the weights of the incoming D-graph links for each document (see Figure

7.3).

Suppose the authority scores of documents fall in the interval $[A_{min}, A_{max}]$ where $A_{min}$ is the minimal authority score calculated in the repository and $A_{max}$ is the maximal authority score calculated. For a particular document $D$, its authority score is $A_D \in [A_{min}, A_{max}]$. Suppose the weight of incoming D-graph links for each document falls in the interval $[L_{min}, L_{max}]$ where $L_{min}$ is the minimal value calculated and $L_{max}$ is the maximal value calculated for any of the documents. The document $D$ has a weight for its incoming D-graph link equal to $L_D \in [L_{min}, L_{max}]$. Then an appropriate arithmetic combination of $A_D$ and $L_D$ will be made to generate a new document weight for further analysis.

### 11.2.3 Color Bars as Representations of Document Content

We have found that color bars are superior to typical title+passage metadata in two different aspects. One is that color bars support faster similarity judgment than title+passage metadata. The other is that the results of color bar comparisons are more consistent across subjects than those of title+passage task comparisons.

However, an important issue not dealt with is whether the results of color bar comparisons are consistent with the *true similarity judgements* reflecting the actual human perception of similarities between documents after they are read, the so called "gold standard." An experiment might be helpful in determining this and will be introduced here.

Suppose we have $N$ color bars representing $N$ documents $(D_1, D_2, ..., D_N)$, and $M$ subjects $(S_1, S_2, ..., S_M)$ participating in the user study. Each subject decides which 2 other documents are most similar to a given document by judging which 2 other color bars are most similar in color to the color bar of the given document. The subjects' answers are then summarized.

Then the same group of $M$ human subjects will be given some time to read the same $N$ documents that are used to generate color bars for the color bar comparison task. They then designate which two documents among the $N$ documents are most similar in content to the given document $D_i$ for $1 \leq i \leq N$.

Finally the two sets of results, one for color bar comparisons and one for comparisons of actual full documents, can be statistically compared. The degree to which the color bar results agree with the "gold standard" is a measure of the ability of color bars to support document similarity judgments.

A similar experiment but using title+passage metadata instead of color bars can be performed. Then, color bars and title+passage metadata can be compared to see which is better able to support document similarity judgments that match the gold standard.

# APPENDIX A. USER STUDY SURVEY SHEETS

Two user studies were carried out to test the usability of MultiBrowser. *Experiment 1* was to check users' general impression on MultiBrowser. *Experiment 2* was to examine the ability of color bars in MultiBrowser to support similarity judgments. The materials used are shown next.

## Experiment 1

*Protocol:*

The user study is conducted as follows:

- Read the instructions to the user.
- Ask him or her if he or she has any question. If so, answer questions.
- Bring up the first page of the repository that he or she will use.
- Let him or her get familiar with the system.
- Give him or her enough time as he or she wants to write down the comment.
- Observe as the user attempts to answer questions related to operations on repositories.

*Directions:*

This is part of our research project on Direct Multidisplay systems. This system processes web sites or other collections of documents on the web, adding numerous "FIND SIMILAR" links that will be helpful in finding similar paragraphs in the documents. When clicked, a "FIND SIMILAR" link brings up a 6-frame window. Each frame contains a document at the site, initially showing a pertinent passage in it. Other parts of the document can be scrolled to.

It will take about 30 minutes to complete the questions. If you have any question, please don't hesitate to contact Dr. Berleant, and you may leave in the middle if you wish.

You will have the opportunity to browse two repositories that have been created on two different topics: powered parachuting and vegetarian cooking. Then I will ask you some questions about your experiences. Your thoughts, opinions and suggestions will be greatly appreciated.

Now you can surf the repositories. If you have any question, please feel free to ask.

*Questions:*

Next I will ask some questions on your general impression of the system:

1. Comment on the density of relevant information appearing, on average, on the screen when multidisplay is used vs. when ordinary display is used.

2. Comment on the efficiency of finding relevant information using "find similar" links.

3. Comment on the styles of information display with 6 sub-frames appearing in the main window.

4. Comment on ways the system might be updated to make it more usable.

## Experiment 2

*Protocol*

The user study is conducted as follows:

- Read the instruction to the user;
- If the user has questions concerning the instructions, answer them;
- Let the user get familiar with the interfaces of the system;
- Let the user answer questions;
- Record the start time and end time of each section in the study;
- Briefly explain what the color bars are and what for;
- Give the user enough time to do the user study and write down the comments if he or she wants to.

*Directions*:

This is part of our research project on Direct Multidisplay systems. This study will take about 30 minutes. If you have any question, please don't hesitate to contact Dr. Berleant, and you may leave in the middle if you wish. Your thoughts, opinions and suggestions will be greatly appreciated.

*Part 1*: Look at the screen. See the 10 sub-windows, labeled "color bar for document 1" through "color bar for document 10." Notice each has a color bar at the top. You will be asked to say which color bars are similar to a given color bar. Do you have any questions? (Answer any questions they have.)

Please observe the screen and answer these questions:

1. Which two color bars are most similar to the color bar for document 1?

2. Which two color bars are most similar to the color bar for document 2?
3. Which two color bars are most similar to the color bar for document 3?
4. Which two color bars are most similar to the color bar for document 4?
5. Which two color bars are most similar to the color bar for document 5?
6. Which two color bars are most similar to the color bar for document 6?
7. Which two color bars are most similar to the color bar for document 7?
8. Which two color bars are most similar to the color bar for document 8?
9. Which two color bars are most similar to the color bar for document 9?
10. Which two color bars are most similar to the color bar for document 10?

*Part 2*: Now, look at a different screen. This screen shows a list of documents and brief information about each, from a Web search engine. Please answer the following questions based on the information you see on the screen.

1. Which two documents are most similar in content to document 1?
2. Which two documents are most similar in content to document 2?
3. Which two documents are most similar in content to document 3?
4. Which two documents are most similar in content to document 4?
5. Which two documents are most similar in content to document 5?
6. Which two documents are most similar in content to document 6?
7. Which two documents are most similar in content to document 7?

8. Which two documents are most similar in content to document 8?

9. Which two documents are most similar in content to document 9?

10. Which two documents are most similar in content to document 10?

# APPENDIX B. USER STUDY RESULTS

The results of the two user study experiments are given in this appendix. Part 1 will show the results of users' general impression about MultiBrowser. Part 2 will show the results of users' operations on MultiBrowser to examine its capabilities.

## Response Speed Results

The times to complete tasks were recorded and are shown in Table B.1.

## Results of Response Consistency

The answers of subjects were nominal data. To statistically analyze the data with standard methods, the following steps had to be taken:

1. Calculate the numbers of occurrences of various answers;
2. Assign orders to the numbers of occurrences in increasing order;
3. Compute standard deviations of the numbers of the occurrences.

For example, Table B.2 shows the answers of 6 subjects to a specific question $Q$. Here is how to analyze the data in Table B.2 using a standard statistical method.

1. Calculate the numbers of occurrences:

$$\# \, of \text{ ``}A\text{''}s = 1$$

$$\# \, of \text{ ``}B\text{''}s = 3$$

$$\# \, of \text{ ``}C\text{''}s = 2$$

2. Assign orders to the numbers of occurrences in an increasing order:

$$1 \Rightarrow \text{``}B\text{''}$$

$$2 \Rightarrow \text{``}C\text{''}$$

$$3 \Rightarrow \text{``}A\text{''}$$

   Now the answers to the question $Q$ can be converted into the format shown in Table B.3.

3. Now we have an array of converted answers: $(1, 1, 3, 1, 2, 2)$. The standard deviation of the array can be calculated, which is 0.816. The concept of standard deviation tells us that the smaller the standard deviation, the more consistent the answers.

We applied the same method to the results of similarity judgments. Table B.4 shows the standard deviations of the results in two different types of tasks: the color bar task and the title+passage task.

Table B.1 The times to complete the similarity judgment tasks in *Experiment 2* of the user study.

| **Color Bar Tasks (in sec.)** | **Title+Passage Tasks (in sec.)** |
|---|---|
| 3 | 12 |
| 2 | 4 |
| 2 | 8 |
| 2 | 6 |
| 1 | 5 |
| 1 | 5 |
| 2 | 9 |
| 2 | 7 |
| 2 | 4 |
| 2 | 5 |
| 2 | 4 |
| 1 | 9 |
| 2 | 10 |
| 1 | 3 |
| 2 | 4 |
| 1 | 3 |
| 1 | 3 |
| 1 | 4 |

Table B.2 The answers of 6 different subjects to a specific question $Q$.

| **Subjects** | **1** | **2** | **3** | **4** | **5** | **6** |
|---|---|---|---|---|---|---|
| **Answers to the question $Q$** | B | B | A | B | C | C |

Table B.3 Conversion of the orginal answers.

| **Subjects** | **1** | **2** | **3** | **4** | **5** | **6** |
|---|---|---|---|---|---|---|
| **Converted Answers** | 1 | 1 | 3 | 1 | 2 | 2 |

Table B.4 Standard deviations of the results.

| Subjects | Color Bar Tasks | Title+Passage Tasks |
|---|---|---|
| 1 | 0.669 | 1.642 |
| 2 | 0.965 | 1.303 |
| 3 | 0.866 | 2.001 |
| 4 | 1.165 | 2.734 |
| 5 | 1.030 | 1.643 |
| 6 | 1.337 | 0.996 |
| 7 | 1.642 | 1.679 |
| 8 | 0.965 | 1.643 |
| 9 | 0.622 | 1.712 |
| 10 | 0.996 | 2.089 |
| 11 | 0.669 | 1.435 |
| 12 | 0.522 | 1.927 |
| 13 | 0.522 | 1.732 |
| 14 | 0.669 | 2.875 |
| 15 | 0.778 | 2.109 |
| 16 | 0.965 | 1.676 |
| 17 | 0.669 | 1.337 |
| 18 | 0.778 | 0.650 |
| 19 | 0.522 | 1.730 |
| 20 | 1.030 | 1.379 |
| 21 | 0.669 | 1.642 |
| 22 | 0.965 | 1.303 |
| 23 | 0.866 | 2.001 |
| 24 | 1.165 | 2.734 |
| 25 | 1.030 | 1.643 |
| 26 | 1.337 | 0.996 |
| 27 | 0.669 | 1.435 |
| 28 | 0.522 | 1.927 |
| 29 | 0.522 | 1.732 |
| 30 | 0.669 | 2.875 |
| 31 | 0.778 | 2.109 |
| 32 | 0.965 | 1.676 |

# APPENDIX C. COMPILING AND RUNNING MULTIBROWSER

MultiBrowser is written in Java. It can run on different platforms such as Windows, Linux and so on. This appendix introduces how to generate and browse repositories with MultiBrowser.

## Repository Generation

There are three different ways to generate a MultiBrowser repository:

1. A user can input one or more keywords (e.g. powered parachuting) which are used as query terms to AltaVista to find documents for the repository;
2. A user can input an HTML document (e.g. a URL of a web page) which is used as a source of links to documents to be placed in the repository; and
3. A user can input a single document, which becomes the only document in the repository.

The above three ways are numbered. Each user is prompted for a choice by simply typing the number representing the way in which he wants to generate the repository.

Then the system will prompt for the location on the hard disk where the user wants to store the repository.

The system will keep prompting for various other parameters for generating repositories, such as the threshold of the similarity measurement above which two similar texts are deemed effectively identical and hence redundant, the number of documents to be retrieved and proceed into the reository, and how many sub-windows are to be displayed (currently MultiBrowser supports 4 and 6 sub-windows).

After the user gives values for all the required parameters, the system will automatically download documents from the Internet, process and reformat them, and finally create the repository.

## Browsing Repositories

Each repository has a starting page which is usually a list of all the documents in the repository. A user can simply click any of the listed documents to access the repository.

Once multi-window displays are visible, the user is able to click any "FIND SIMILAR" link to navigate to some other paragraphs which are similar in content to the paragraph that is being read.

The user can also use the color bar on top of the sub-window containing this document to judge similarity between documents in case if it is desired to read more documents similar in content.

The color bar contains a hyperlink pointing to the original web page. Thus the user can get a current, 1-window display of a particular document.

# BIBLIOGRAPHY

[1] Ballim, A. and Y. Wilks. “Beliefs, stereotypes and dynamic agent modeling,” *User Modeling and User-Adapted Interaction*, vol. 1, pp. 33-65, 1991.

[2] Baeza-Yates, R. “Similarity in two dimensions,” *In COOCON '98*, LNCS 1449, Taipei, Taiwan, August 1998. Springer Verlag, 319-328.

[3] Benford, F. “The Law of Anomalous Numbers,” *Proc. Amer. Phil. Soc.* 78, 551-572, 1938.

[4] Berghel, H., D. Berleant, T. Foy and M. McGuire. “Cyberbrowsing: Information Customization on the Web,” *Journal of the American Society for Information Science*, Vol. 50, 1999.

[5] Berleant, D. and H. Berghel. “Customizing Information: Part 1,” *Computer*, *27*(9): 96-98. "Part 2,” *Computer*, *27*(10): 76-78, 1994.

[6] Berleant, D., J. Miao, Z. Gu and D. Xu. “Towards Dialogues With Documents: MultiBrowser,” *Proceedings of International Conference on Information Technology (ITCC '04)*, April 2004.

[7] Bly, S.A. and J. K. Rosenberg. “A Comparison of Tiled and Overlapping Windows,” *Proceedings of ACM Conf. Human Factors in Computing Systems (CHI '86)*, 1986, 17: 101-106.

[8] Brin, S. and L. Page. "The Anatomy of a Large-Scale Hypertextual Web Search Engine," *Proceedings of the WWW 7 Conference*, 1998, 107-117.

[9] Byrd, D. "A Scrollbar-Based Visualization for Document Navigation," *Fourth ACM Conference on Digital Libraries (DL '99)*, 1999, 122-129.

[10] Catledge, L.D. and J.E. Pitkow. "Characterizing Browsing Strategies in the World-Wide Web," *Computer Networks and ISDN Systems*, 1995 *27*(6): 1065-1073.

[11] Chu, W.W., D.B. Johnson and H. Kangarloo. "A Medical Digital Library to Support Scenario and User-Tailored Information Retrieval," *IEEE Transactions on Information Technology in Biomedicine*, Vol. 4 Issue 2 , June 2000, pp. 97 -107.

[12] http://www.cs.usyd.edu.au/~bob/CandD.html (as of March 2004).

[13] Damashek, M. "Gauging Similarity With N-Grams: Language-Independent Categorization of Text," *Science*, 1995 *10*(2)267: 843-848.

[14] E Ink Corp. http://www.eink.com (as of March 2004).

[15] Evans, W. "Content Analysis Resources," http://www.car.ua.edu/ (as of December 2003).

[16] Foltz, M.A. "Designing Navigable Information Space," M.S Thesis, Dept. of Electrical and Computer Engineering, MIT, 1998.

[17] Furnas, G.W. "Effective View Navigation," *Proc. ACM Conf. Human Factors in Computing Systems (CHI '97)*, 1997, 367-374.

[18] Goldberg, D., B. Oki, D. Nichols and D.B. Terry. "Using Collaborative Filtering to Weave an Information Tapestry," *Communications of the ACM*, December 1992, Vol 35, No12, pp. 61-70.

[19] Golovchinsky, G. and M.H. Chignell. "The Newspaper as an Information Exploration Metaphor," *Information Processing & Management*, 1997 *33*(5): 663-683.

[20] http://alexia.lis.uiuc.edu/~twidale/irinterfaces/bib-main.html (as of March, 2004).

[21] Gu, Z. "Direct Multidisplay for Web Document Repositories," Master's thesis, Dept. of ECpE, Iowa State Univ., 2000.

[22] Halasz, F.G. "Reflections on Notecards: Seven Issues For the Next Generation of Hypermedia Systems," *Communications of the ACM*, 1988, *31*(7): 836-852.

[23] Hara, Y. and K. Watanabe. "Hypermedia Research at C&C Research Labs, NEC USA," *CHI'97 Electronic Publications: Organizational Overview*, 1997.

[24] Hearst, M. "TileBars: Visualization of Term Distribution Information in Full Text Information Access," *Proc. CHI '95*, 1995, 59-66.

[25] Hearst, M. "User Interfaces and Visualization," *Chapter 10 in R. Baeza-Yates and B. Riveiro-Neto, eds.*, *Modern Information Retrieval*, ACM Press, 1999, pp. 257-339.

[26] Hill, W., J. Hollan, D. and T. McCandless. "Edit Wear and Read Wear," *Proc. CHI'91 Human Factors in Computing Systems* (Monterey, 1992), Addison-Wesley, pp. 3-9.

[27] Huang, X., G.I. McCalla, J.E. Greer and E. Neufield. "Revising Deductive Knowledge and Stereotypical Knowledge in a Student Model," *User Modeling and User-Adapted Interaction*, vol. 1, pp. 87-115, 1991.

[28] Kaltenbach, M. and C. Frasson. "Dynaboard: User Animated Display of Deductive Proofs in Mathematics," *International Journal of Man-Machine Studies*, 1989, 30: 149-170.

[29] Kaltenbach, M., F. Robillard, and C. Frasson. "Screen Management in Hypertext Systems with Rubber Sheet Layouts," *Proceedings Hypertext '91*, 1991, 91-105.

[30] Kamba, T., K. Bharat, and M.C. Albers. "The Newspaper on the Web," *Proc. WWW4* (Boston MA, November 1995).

[31] Kandogan, E. and B. Shneiderman. "Elastic Windows: A Hierarchical Multi-Window World-Wide Web Browser," *ACM Symposium on User Interface Software and Technology*, 1997.

[32] Kandogan, E. and B. Shneiderman. "Elastic Windows: Evaluation of Multi-Window Operations," *Proc. ACM Conf. Human Factors in Computing Systems (CHI '97)*, 1997.

[33] Kim, H. and R.R. Korfhage. "BIRD, a Browsing Interface for the Retrieval of Documents," *In Proceedings of the 1994 IEEE Symposium on Visual Languages*, Los Alamitos, IEEE Computer Society Press.

[34] Kleinberg, J. "Authoritative Sources in a Hyperlinked Environment," *Proc. ACM-SIAM Symposium on Discrete Algorithms*, 1998, 46: 604-632.

[35] Kok, A.J. "A Review and Synthesis of User Modeling in Intelligent Systems," *Knowledge Eng. Rev.*, vol. 1, no. 1, pp. 21-47, 1991.

[36] Konstan, J., B. Miller, D. Maltz, J. Herlocker, L. Gordon and J. Riedl. "GroupLens: Collaborative Filtering for Usenet News," *Communications of the ACM*, March, 1997, pp. 77-87

[37] Lammens, Johan M. "A Computational Model of Color Perception and Color Naming," *Ph.D. Dissertation at State University of New York at Buffalo*, June 1994.

[38] Lifschitz, K. and B. Shneiderman. "Window Control Strategies for On-Line Text Traversal," http://www.cs.umd.edu/hcil/members/bshneiderman/umlpapers/articles.html (as of March 2004).

[39] Malone, T.W., et al. "Intelligent Information-Sharing Systems," *Communications of the ACM(1987)* Vol. 30, No. 5, pp. 390-402.

[40] Mayfield, J. and P. McNamee. "Indexing Using Both N-Grams and Words," *NIST Special Publication 500-242: The Seventh Text Retrieval Conference (TREC 7)*, 1998, 419-424.

[41] Miller, N.E., P.C. Wong, M. Brewster, and H. Foote. "TOPIC ISLANDS — A Wavelet-Based Text Visualization System," *IEEE Visualization '98*, 1998.

[42] Nielsen, J. "User Interface Directions for the Web," *Communications of the ACM (Jan. 1991)*, 1991, 41: 65-72.

[43] Nuchprayoon, A. and R.R. Korfhage. "GUIDO, a Visual Tool for Retrieving Documents," *In Proceedings of the 1994 IEEE Symposium on Visual Languages*, Los Alamitos, IEEE Computer Society Press. pp.64-71.

[44] Olsen, K.A., R.R. Korfhage, K.M. Sochats, M.B. Spring and J.G. Williams. "Visualization of a Document Collection: the VIBE System", *Information Processing & Management*, 29(1), 69-81, 1993.

[45] Page, L., S. Brin, R. Motwani and T. Winograd. "The PageRank Citation Ranking: Bringing Order to the Web," *Stanford Digital Library Technologies Project*, 1998.

[46] Pirolli, P. and S. Card. "Information Foraging in Information Access Environments," *Proc. ACM Conf. Human Factors in Computing Systems (CHI '95)*, 1995, 51-58.

[47] Pirolli, P. "Exploring Browser Design Trade-Offs Using a Dynamical Model of Optimal Information Foraging," *Proc. CHI '98*, 1998.

[48] Scott, J. "Social Network Analysis: A Handbook," *Sage Publications, Inc.*, Thousand Oaks, CA, 1991.

[49] Shepard, Roger N. "Toward a Universal Law of Generalization for Psychological Science," *Science*, 1987, 237:1317-1323.

[50] Shneiderman, B. "User Interface Design for the Hyperties Electronic Encyclopedia," *Proc. Hypertext '87*, 189-194.

[51] Shneiderman, B., C. Plaisant, R. Botafogo, D. Hopkins, and W. Weiland. "Designing to Facilitate Browsing: A Look Back at the Hyperties Workstation Browser," *Hypermedia*, 1991, *3*(2): 101-117.

[52] Shneiderman, B., D. Byrd, and B. Croft. "Sorting Out Searching, a User-Interface Framework for Text Searches," *Communications of the ACM*, 1998, emph41(4): 95-98.

[53] Singhal A., M. Mitra, and C. Buckley. "Learning Routing Queries in a Query Zone," *Proceedings of SIGIR '97, 20th ACM International Conference on Research and Development in Information Retrieval*, 1997.

[54] Smith, T.R. "The Meta-Information Environment of Digital Libraries," *D-Lib Magazine, Corp. for National Research Initiatives*, http://www.dlib.org/back.html, July/August 1996.

[55] Stotts, D. and R. Furuta. "Petri-Net-Based Hypertext: Document Structure with Browsing Semantics," *Transactions on Information Systems*, 1989, *7*(1): 3-29.

[56] Tauscher, L. and S. Greenberg. "Revisitation Patterns in World Wide Web Navigation," *Proc. ACM Conf. Human Factors in Computing Systems (CHI '97)*, 1997, 399-406.

[57] Terveen, L. and W. Hill. "Constructing, Organizing, and Visualizing Collections of Topically Related Web Resources," *ACM Trans. on CHI '99"*, 1999, *6*(1): 67-94.

[58] Tmar, M. and M. Boughanem. "Learning Profile in Routing: Comparison between Relevance and Gradient Back-Propagation," *Proceedings of the Seventh International Symposium on String Processing Information Retrieval (SPIRE '00)*, 2000.

[59] United States Display Consortium. *Display Trends Newsletter,* http://www.usdc.org/newsroom/newsletters_downloads/displaytrends2.PDF and http://www.usdc.org/newsroom/newsletters_downloads/ppdisplaytrends.PDF (as of March 2003).

[60] Weisstein, E.W. "K-Means Clustering Algorithm," *From MathWorld–A Wolfram Web Resource*, http://mathworld.wolfram.com/K-MeansClusteringAlgorithm.html (as of March 2004).

[61] Wellman, B. and M. Gullia. "Net Surfers Don't Ride Alone: Virtual Communities as Communities," P. Kollock and M. Smith (Eds.), *Communities in Cyberspace: Perspectives on New Forms of Social Organization.* Berkley: University of California Press, 1997.

[62] Wexelblatt, A. and P. Maes. "Footprints: History-Rich Tools for Information Foraging," *Proc. CHI '99*, 1999, 270-277.

[63] Zellweger, P., B.-W. Chang, and J. Mackinlay. "Fluid Links for Informed and Incremental Link Transitions," *Proc. Hypertext '98*, Pittsburgh, June 20-24, 50-57.

[64] Zipf, G.K. *The Psychobiology of Language*, 1935, Houghton Mifflin.

[65] Zipf, G.K. "The Meaning-Frequency Relationship of Words," *The Journal of General Psychology*, 1945, 33: 251-256.

[66] Zipf, G.K. "The Repetition of Words, Time-Perspective, and Semantic Balance," *The Journal of General Psychology*, 1945, 32: 127-148.