

# AN EVALUATION OF TOOLS FOR CONVERTING TEXT TO HYPERTEXT

by

William James Blustein

Department of Computer Science

Submitted in partial fulfillment  
of the requirements for the degree of  
Master of Science

Faculty of Graduate Studies  
The University of Western Ontario  
London, Ontario

December 1993

© William James Blustein 1994

THE UNIVERSITY OF WESTERN ONTARIO

FACULTY OF GRADUATE STUDIES

CERTIFICATE OF EXAMINATION

Advisor

Examining Board

---

---

---

---

The thesis by

William James Blustein

entitled

AN EVALUATION OF TOOLS FOR CONVERTING TEXT TO  
HYPERTEXT

is accepted in partial fulfillment of the

requirements for the degree of

Master of Science

Date 

---

---

Chair of Examining Board

## ABSTRACT

Methods for automatically converting semi-structured text (Usenet messages) into hypertext form using information retrieval methods were investigated. The methods were evaluated using statistical means to determine which will produce hypertext best suited to browsing and searching. Methods were evaluated by comparing the a measure of semantic similarity of all document pairs with the shortest path in a graph formed by hypertext links between those documents.

## ACKNOWLEDGEMENTS

Many people helped me along the way to completing this work. I would like to single some of them out for special attention. Alf Benn helped me understand the matrix math underlying the latent semantic indexing method and helped me with the statistical aspects of the thesis. Mark Bernstein tirelessly answered many of my questions. Sue Dumais supplied the latent semantic indexing package and also answered many questions about the method and information retrieval in general. Marg Lundy provided me with sound advice and helped me to configure the PARAFAC program.

I am especially grateful to my supervisor, Bob Webber, for the countless hours he spent proofreading my text and for suggesting I investigate information retrieval methods.

# TABLE OF CONTENTS

<b>CERTIFICATE OF EXAMINATION</b>	<b>ii</b>
<b>ABSTRACT</b>	<b>iii</b>
<b>ACKNOWLEDGEMENTS</b>	<b>iv</b>
<b>TABLE OF CONTENTS</b>	<b>v</b>
<b>LIST OF TABLES</b>	<b>viii</b>
<b>LIST OF FIGURES</b>	<b>ix</b>
<b>NOTATION</b>	<b>xi</b>
<b>Chapter 1 Introduction</b>	<b>1</b>
1.1 Advantages of Hypertext . . . . .	1
1.2 Conversion of Text to Hypertext . . . . .	2
<b>Chapter 2 Background</b>	<b>3</b>
2.1 Previous Work . . . . .	3
2.1.1 Highly Structured Documents . . . . .	3
2.1.2 The AI Approach . . . . .	4
2.1.3 The IR Approach . . . . .	5
2.2 Information Retrieval . . . . .	5
2.2.1 Performance Evaluation . . . . .	6
2.2.2 Document Models . . . . .	8

2.2.3	IR Techniques . . . . .	9
2.3	Similarity Measures . . . . .	12
2.3.1	In Vector Space . . . . .	13
2.3.2	Other Methods . . . . .	14
<b>Chapter 3</b>	<b>Approach</b>	<b>17</b>
3.1	Experimental Design . . . . .	17
3.1.1	Structural Links . . . . .	17
3.1.2	Semantic Links . . . . .	20
3.1.3	Evaluating Semantic Links . . . . .	22
3.2	Experimental Details . . . . .	26
3.2.1	The List of Term Weights . . . . .	26
3.2.2	The LSA Similarity Measure . . . . .	26
3.2.3	The Shortest Path Measures . . . . .	27
3.2.4	The Comparisons . . . . .	27
<b>Chapter 4</b>	<b>Results</b>	<b>29</b>
4.1	Hypertext Graphs . . . . .	29
4.2	Evaluation Using Density Plots . . . . .	30
4.3	Evaluation Using Correlation Measures . . . . .	36
4.4	Effects of Using Graphs to Approximate Similarities . . . . .	38
4.4.1	Effects of Infinite Path Length . . . . .	38
4.4.2	Effects of the Graph Approximation . . . . .	38
<b>Chapter 5</b>	<b>Conclusion</b>	<b>40</b>
5.1	What Is Needed For Automatic Conversion . . . . .	40
5.1.1	What Are The Choices . . . . .	40
5.1.2	Correlation . . . . .	41
5.1.3	Connectivity . . . . .	41
5.2	Pitfalls . . . . .	41
5.2.1	Representation of Disconnected Components . . . . .	42
5.2.2	Loss of Information . . . . .	42
5.3	Which Method Is Best . . . . .	42

<b>Chapter 6</b>	<b>Future Work</b>	<b>44</b>
6.1	Further Experiments . . . . .	44
6.2	Enhancements to the Experimental Model . . . . .	45
6.3	Enhancements in a Working Prototype . . . . .	45
6.4	Latent Semantic Analysis . . . . .	45
<b>Appendix I</b>	<b>An example hypertext system</b>	<b>46</b>
<b>Appendix II</b>	<b>The Raw Data</b>	<b>48</b>
II-1	What format does the raw data have? . . . . .	48
<b>Appendix III</b>	<b>All-Pairs Shortest Path Algorithm</b>	<b>52</b>
<b>REFERENCES</b>		<b>53</b>
<b>VITA</b>		<b>62</b>

## LIST OF TABLES

2.1	Sizes of Some Typical Datasets . . . . .	7
4.1	Correlation of Entropy-Log Weights with LSA . . . . .	37
4.2	Correlation of Inverse Document Frequency-Log Weights with LSA . . . . .	37
4.3	Correlation of GfIdf-Log Weights with LSA . . . . .	38
4.4	Correlation of Pairwise Similarity for GfIdf and Shortest Path Length (Link Distance) for Graphs of Various Outdegrees . . . . .	39



## LIST OF FIGURES

2.1	An Outline of an Information Retrieval System . . . . .	6
3.1	A sample Usenet header . . . . .	18
3.2	Example of Subject Menu . . . . .	19
3.3	Example of Author Menu . . . . .	19
3.4	A Hypertext Graph . . . . .	23
3.5	Experimental Design . . . . .	25
4.1	Density Plot of link distance computed using the global entropy and local log weight with an outdegree of 1 vs. the semantic similarity computed with LSA . . . . .	31
4.2	Density Plot of link distance computed using the global entropy and local log weight with an outdegree of 2 vs. the semantic similarity computed with LSA . . . . .	32
4.3	Density Plot of link distance computed using the global entropy and local log weight with an outdegree of 3 vs. the semantic similarity computed with LSA . . . . .	33
4.4	Density Plot of link distance computed using the global entropy and local log weight with an outdegree of 4 vs. the semantic similarity computed with LSA . . . . .	34
4.5	Density Plot of link distance computed using the global entropy and local log weight with an outdegree of 5 vs. the semantic similarity computed with LSA . . . . .	35
4.6	Correlations Tested in Experiment . . . . .	39
I.1	Help screen for text-only tkWWW interface . . . . .	47

II.1	An example of why news is an information source . . . . .	49
II.2	Asking for information . . . . .	50
II.3	An article asking for and providing information . . . . .	51
III.1	Modified Floyd-Warshall Shortest Path Algorithm . . . . .	52

## NOTATION

### General

$n$  a non-negative integer value

$\infty$  infinity

$\max_n x$  the maximum value of  $x$  as  $n$  ranges from 1 to its maximum

$\min(a, b)$  the smaller value of  $a$  and  $b$

### Vectors

$\vec{x}$  the vector  $x$

$x_i$  the  $i^{\text{th}}$  element of  $\vec{x}$

$|\vec{x}|$  the magnitude of the vector  $x$ ,  $|\vec{x}| = \sqrt{\sum_{i=1}^n (x_i)^2}$  where  $\vec{x}$  has  $n$  elements

$\|\vec{x}\|$  the dimensionality of the vector  $x$ , i.e., the number of elements in the vector  $x$

### Matrices

$X$  matrix  $X$

$X_{i \times j}$  matrix  $X$ , which has  $i$  rows and  $j$  columns

$X^T$  the transpose of matrix  $X$

$(X)_{ij}$  the element in the  $i^{\text{th}}$  row and  $j^{\text{th}}$  column of matrix  $X$

## Weighted Graphs

$g_{ij}$  the weight of the edge from vertex  $i$  to vertex  $j$  in the graph  $G$

$\text{rows}(G)$  the number of rows in the matrix  $G$ .

## Terms and Documents

$D$  the set of documents in the document collection

$T$  the set of terms in the document collection

$|D|$  the number of documents in the document collection

$|T|$  the number of terms

$|D^t|$  document frequency of  $t$ , i.e., the number of documents containing term  $t$

$f_D^t$  global frequency of  $t$ , i.e., the frequency of term  $t$  in the document collection

$f_d^t$  term frequency of  $t$  in  $d$ , i.e., the frequency of term  $t$  in document  $d$

$d|_t^\lambda$  the document  $d$  with all occurrences of the term  $t$  removed

## Weightings

$w_{td}$  the weight of term  $t$  in document  $d$

## Statistics

$\bar{x}$  the arithmetic mean of a sample of measurements  $x_1, x_2, \dots, x_n$

# *Chapter 1*

## *Introduction*

Hypertext documents are a way of presenting textual information [Con87, Nel87, Eas90]. Unlike traditional documents, such as books, hypertext documents are interactive and record explicitly the relationships between different parts of their text [DeR89]. Hypertext was first described as part of a machine called memex [Bus45] in 1945. The term *hypertext* was coined by Theodore Nelson [Nel87] in the 1960's.

### *1.1 Advantages of Hypertext*

It is much easier to search for text in a hypertext document than in a book [ELK<sup>+</sup>91]. To find details in a traditional book, one reads all of it or searches through the index. Book indexes are not as useful as the distributed indexes (called *links*) available in hypertext documents. Keywords in book indexes are arranged alphabetically and out of context, that is, related topics are not necessarily listed near each other. Not only do hypertext systems allow users to automatically search through text, but hypertext documents have indexes distributed throughout many parts of the text.

In a hypertext document, the reader can make links to annotate and connect parts of the text. The annotations are integrated into the document, allowing the user access to them when browsing and searching. Many hypertext systems allow some links to be shared with other users and some kept private.

Clearly, hypertext systems can be tools for more efficient access to information. A major obstacle to more efficient use of information is the difficulty of converting existing documents into hypertext documents. I have investigated methods of automatically making

these conversions. I present an outline of a method for making such conversions and some suggestions for how such a method can be incorporated into existing hypertext systems.

## *1.2 Conversion of Text to Hypertext*

When people convert unstructured text into hypertext documents, they also have to form links between different chunks of that text. In order to make these links, they must read and understand the text well enough to make connections between the parts. I explore methods for automatic link creation, which do not require a person to read or understand the text in order to build the document — the computer does the work.

The purpose of this thesis is to examine methods for the automatic linking of texts. To that end, I have investigated indexing methods, models of hypertext, and hypertext implementations. I propose a method for producing a collection of hypertext documents that is equivalent to a collection of ordinary documents. I have tested my method with a collection of postings to the Usenet newsgroup comp.graphics and some of its antecedents. The postings in that collection have been converted to an equivalent hypertext form using the Hypertext Markup [sic] Language (HTML) for use in the World Wide Web project (WWW). WWW is a distributed hypermedia system. The Web's collection of hypermedia documents are accessible through client software, called *browsers*, which connect to WWW servers over the Internet. Hypermedia documents in the WWW are marked-up in HTML [BLC], which is similar to SGML [Gol90]. I chose to implement my linking strategies by converting text documents into HTML form.

Almost all previous research on automatically converting text into hypertext documents assumes that the text is highly structured and can thus be parsed by a computer. For example, a dictionary might already have cross-references to synonyms and antonyms. Such cross-references could easily be converted into hypertext links. In such cases, the links between words and entries have already been made by humans, based on an understanding of the text. Although there has been great interest in the automatic conversion of general texts into hypertext documents, little work has been done to date [Rai87, CRM89, Ber90].

## *Chapter 2*

### *Background*

#### *2.1 Previous Work*

Automatic conversion of text into a hypertext document is an interesting open problem [Bal93]. I identify three different approaches: 1) working with only highly structured text; 2) an AI artificial intelligence approach employing semantic parsers and knowledge bases; and 3) an information retrieval (IR) approach involving statistics derived from the text.

##### *2.1.1 Highly Structured Documents*

The Oxford English Dictionary (OED) Project [RT87, RT88] and the conversion of The Engineering Data Compendium [Glu89] are examples of the creation of hypertext links in existing highly structured traditional documents. In the case of the OED Project, typefaces and abbreviations made some of the connections, that would be represented as links, explicit in the original text. Many other links were generated between words based on their occurrence in the body of the definitions. The OED researchers employed a semantic parser to distinguish homonyms during searches.

The Compendium workers used programs to identify possible links in the text. After these links had been identified, people chose which links to include in the hypertext document. They were guided in their choices by a model of what they thought users would want from a hypertext version of an engineering encyclopaedia.

Researchers on the OED Project warn that documents with great structural variance or too much structure are bad candidates for conversion into hypertext documents. Both

research teams warn that if a document has too many links, the users will have difficulty searching it. However, the OED researchers suggest that when an user is just browsing, extra links might make the document more useful. Also they claim that if the system is fast enough, then users would benefit more by exploring many alternative paths through the document than by following the guidance provided by a few well-chosen links [RT88, pp. 877–8].

Other researchers are working with electronic versions of text that have already been structured for use with computers. Part of the MAESTRO Project [Fah88, FB90] is to make links between texts marked-up in SGML [Gol90]. Legal documents with numbered sections, paragraphs, and clauses are a prime example of highly structured text [ACG91]. Some attention has been given to turning text marked-up in *\*roff* format into hypertext [TW86, FPS89] with keywords identified by text highlighting, e.g., italicized words. Programs to convert from a variety of structured formats to a marked-up form are available [BS93].

### *2.1.2 The AI Approach*

The papers illustrating the AI approach [CRM89, Rai87, HR88] deal with text from highly restricted domains. They employ semantic parsers to find relationships in the texts. Among other problems, such systems require grammatically correct texts. These systems use knowledge bases to aid in the parsing of such texts. For each new subject, a new knowledge base must be built.

VISAR [CRM89] is a hypertext system for maintaining a database of journal citations. It is knowledge intensive and requires the user to explicitly customize the knowledge base. The system will only search for concepts that are in its knowledge base, thus it could ignore many useful connections that the user may not have considered. It searches for citations that satisfy certain relationships between concepts that have been specified by the user. This is a significant improvement over systems that use only keywords.

The approach used in the TOPIC system [Rai87, HR88] depends on the use of a semantic parser and explicit domain-specific knowledge bases. TOPIC represents the topical structure of the text in a hierarchical graph, with links based on a semantic understanding of thematically coherent chunks of the text. The graphs thus constructed can be searched at varying levels of detail. Queries result in the automatic generation of hypertext links between nodes of the graph. The system is also designed to answer questions about the text



solely from summaries stored at the upper levels of the hierarchy. The link-making part of TOPIC has been tested on a small number of texts (20 texts, each about 2000–3000 text tokens long). The authors note that their system has not been developed to create links within collections of thematically varied texts [HR88, p.188].

### 2.1.3 *The IR Approach*

My text consists of hundreds of files from many authors. An average file contains about 200 words. Also, the authors of these files did not restrict themselves to a small semantic domain. Files topics range from mathematical models to industrial rumours. The style of these articles varies widely as well. Thus, my initial text is more closely related to the kinds of texts that information retrieval (IR) researchers process, rather than those studied by AI researchers.

One example of the IR approach is the work of Mark Bernstein [Ber90]. His system makes links between similar passages in monographs. His similarity measure is based on a Bloom filter hashing [Blo70] of the individual words and word stems. This approach works with ‘compact, independent hypertext documents which may be considered to address a single subject, and which are intended to be read rather than queried.’ [Ber90, p. 213] He specifically excludes electronic mail and Usenet messages from the program’s domain [Ber90, p. 213].

Bloom filters use multiple hash functions to generate a probabilistic set membership test. They can never falsely report a member as a non-member but will sometimes declare a non-member to be a member. They operate by hashing all keys into the same address space (typically a one-dimensional array). Bernstein’s method is to compare the dot product of vectors thus generated from two passages. The higher the magnitude of the product the greater the similarity. He also normalizes the results with respect to passage length [Ber93].

## 2.2 *Information Retrieval*

Most information retrieval work is done with bibliographic databases in mind [Mea92]. IR is usually thought of as a process of query refinement, in which a user queries a computerized library catalogue to find records of interest. Such systems typically work by comparing an user’s query (often called a *request* in the IR literature) with the records in the collec-

tion. IR techniques are also applied to full text databases in which queries are evaluated with respect to entire documents. This process is illustrated in Figure 2.1<sup>1</sup>. There is a good deal of overlap between hypertext and IR, but the research areas are quite different [Hal88, CNBKO90, You90, Bas91].

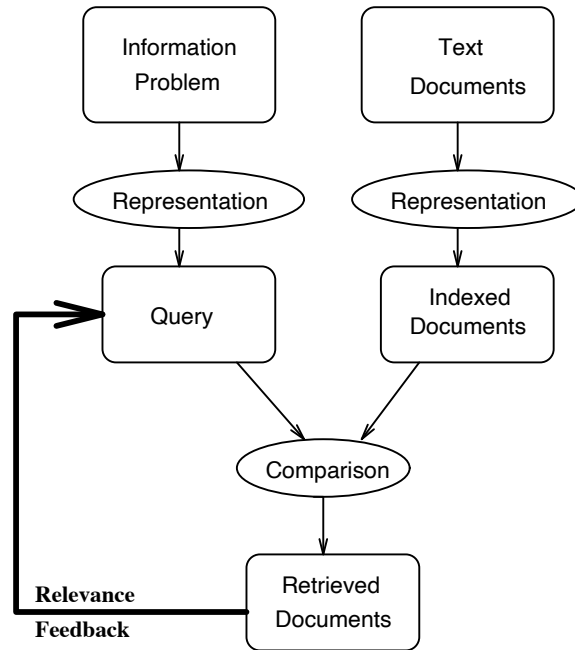


Figure adapted from: Figure 1 of 'Hypertext and Information Retrieval: What are the Fundamental Concepts?' [A panel discussion] W. Bruce Croft, panel proposer. (Hypertext: Concepts, Systems and Applications, 1990).

Figure 2.1: An Outline of an Information Retrieval System

### 2.2.1 Performance Evaluation

There is not yet a satisfactory measure of the indexing effectiveness of an IR system [Mea92, p. 286]. Systems however are often evaluated using measures of *precision* and *recall* [Sha86, Sal68] or measures derived from them [Swe80, Mea92]. Recall is defined as the number of relevant database entries that match the query divided by the number of all database entries. Precision is the number of found records relevant to the query divided by the number of records found.

The relevance of a record to a query is a subjective measure. A document may be

---

<sup>1</sup>The figure is adapted from a similar figure accompanying a panel discussion [CNBKO90].

Collection	Number of	
	Terms	Documents
TIME [Dum91]	10 337	425
MED [DDL+88]	5831	1033
Cranfield [Dum91]	4486	1400
CISI [DDL+88]	1460	5143
ADI [Dum91]	374	82
comp.graphics	2478	1608

Table 2.1: Sizes of Some Typical Datasets

considered relevant to a query even if they have no vocabulary in common. The *vocabulary problem* in IR is that different people will often choose different words with which to express the same concept [FLGD87, KN90] or to index the same document [Jon90, KN90].

Systems are often evaluated by the recall and precision values [Sal92, TS92] for standard queries on standard document collections such as Cranfield [Har92] or MEDLARS [Dum91]. Table 2.1 summarizes characteristics of databases composed of articles from: Time magazine, abstracts in the U.S. National Library of Medicine (MED), abstracts about aeronautics from the Cranfield collection, abstracts about library science and related fields (CISI and ADI), and the collection I use for my tests. Large systems are not always tested using measures of recall and precision.

Scott Deerwester et al. use the terms *polysemy* and *synonymy* to describe the difficulties that make up the vocabulary problem [DDF<sup>+</sup>90]. They use polysemy to refer to the problem that many words have more than one meaning — one needs context to determine the meaning. This can pose a big problem for mechanical indexing methods. By synonymy they mean that many different words can be used to mean the same thing at different levels of detail. For example, ‘machine’, ‘computer’, and ‘calculator’ can all refer to the same thing.

One solution to this problem is to restrict the vocabulary of queries and documents to be unambiguous. This solution however will not work with natural language text and is not likely to be favoured much by users. Current research focuses on how to improve recall given that the vocabulary problem exists.

### 2.2.2 Document Models

Rather than representing documents as sequences of terms in order, many IR systems use a representation of the documents as terms and term frequency counts. This is termed a *model* in IR. Most current work is done with models of documents.

#### *The Vector Model*

Documents are typically modelled as vectors of attributes. Collections of documents are typically modelled as matrices (a vector of document vectors). The attributes are chosen to be useful for the retrieval task, such as boolean values indicating the presence or absence of a term in a document. For example, document  $d$  might be represented by the vector  $\vec{d} = \langle w_{1d}, w_{2d}, \dots, w_{|T|d} \rangle$  where  $|T|$  is the number of distinct terms in the document collection and  $w_{td}$  is 1 if term  $t$  appears at least once in document  $d$  and 0 otherwise.

Although the vector model does not have a strong theoretical basis [RW86, WZW85], it has been used extensively [Sal91]. The model is often used with the implicit assumption that terms are independent [Dum91, WZW85], so  $|T|$  terms can be represented by  $|T|$  orthogonal dimensions.

#### *Tree Models*

Documents may be described by *trees*, i.e., directed acyclic graphs. Tree structures are a natural way to represent the hierarchical nature of many documents. Parse trees, for example, are used to describe text as grammatical phrases consisting of smaller components. Trees can also be used to describe the structure of a book — a book may be composed of sections that are in turn composed of subsections. A particular subsection is part of exactly one section. There may be many trees describing one document since the division into parts is often a matter of interpretation.

Lev Goldfarb [Gol86] has proposed measuring the similarity of documents by the minimal number of steps required to convert the tree describing one document to a tree describing the other. Some related work is being carried out at Queen's University [DB91], where a program has been written to build hypertext links between documents marked-up with an explicit tree structure [Fah88, FB90].

## *Clustering*

There are ways of extracting essential predictors about documents and using them for comparisons that do not rely on an explicit vector space model. Clustering is a method by which documents are grouped into classes by topic [Sal91]. This is accomplished by comparing documents to the pseudo-document that is a perfect representative of a class. This is similar to the vector space concept of a *centroid*, a single document that represents the average values of all the documents in the collection. Frank B. Baker [Bak62] proposed using a probabilistic approach for this. He was not able however to perform rigorous testing of the idea. G. Salton and Chris Buckley [SB90] propose using clustering as part of a linking strategy to improve relevance.

### *2.2.3 IR Techniques*

#### *Choosing Terms*

G. Salton, C. S. Yang, and C. T. Yu [Sal89, SY74] have developed methods for choosing words for indexing documents. The following explanation is derived from Salton [Sal89, Chapter 9]. For a list of individual words in all documents, the method is to select words and collections of words, that discriminate between the documents. Words with very high or very low frequency are eliminated from consideration as indexing terms at the beginning of the process. For the purposes of indexing, low frequency words are replaced by their equivalent word in a thesaurus. For example, ‘acquiescence’ and ‘compliance’ might both be indexed as though they were ‘consent’ [Sal89, after Table 9.5]. Some high frequency words are combined to form low frequency phrases that can be used to better distinguish between documents.

The selection of terms is used to model the document as a boolean vector. The entry for a term in a document vector is an indication of whether or not the term occurs in the document. The use of real valued term weights provides a better way to distinguish between documents. The weights provide a representation of documents in space. Similarity measures are used to determine the distance between vectors in that space.

## Weighting Terms

Assigning weights to terms improves performance considerably [Spa72, Har86]. Karen Sparck Jones [Spa72] was the first to assign weights to terms by their frequency in the document collection. She found substantial improvement over boolean weighting when she assigned each term in a collection  $D$  of documents the weight defined in Equation 2.2-1. S. K. M. Wong and Y. Y. Yao [WY92] performed a more thorough analysis and testing than Sparck Jones and concluded that a probabilistic logarithmic measure performs even better. Their formulae are exceedingly complex and are not reproduced here. Salton notes that such methods are rarely practical [Sal91, p. 975].

Term weighting schemes are used to increase recall and precision [Sha86]. Typically a term's weight in a document is computed from the product of the measure of the term's importance in the document (its *local weight*) and its importance in the entire collection (its *global weight*).

The weighting schemes below are defined using term frequency  $f_d^t$  of term  $t$  in document  $d$ . The number of documents in which term  $t$  occurs is  $|D^t|$ . The global frequency of term  $t$  is denoted by  $f_D^t$ . The number of documents in the collection is  $|D|$ . The number of terms in the collection is  $|T|$ .

*Global Weightings* Some global weighting schemes are:

- Specificity [Spa72]

Sparck Jones found substantial improvement over boolean weighting when she assigned each term in a collection  $D$  of documents the weight of

$$\lceil \log_2 |D| \rceil - \lceil \log_2 f_D^t \rceil + 1 \quad (2.2-1)$$

where<sup>2</sup>  $f_D^t$  is the frequency of term  $t$  in the collection of  $|D|$  documents.

- Inverse Document Frequency (Idf) [Dum91, Sal89]

Frequency-based indexing models are based on the premise that the best indexing terms 'are those that occur frequently in individual documents but rarely in the remainder of the collection' [Sal89, p. 280]. This is the inspiration for the inverse document frequency weighting:

---

<sup>2</sup>A summary of notation used throughout this document begins on page xi.

$$\log_2 \left( \frac{|D|}{|D^t|} \right) + 1 \quad (2.2-2)$$

- GfIdf [Dum91]

$$\frac{f_D^t}{|D^t|} \quad (2.2-3)$$

- entropy [Dum91]

The entropy weighting scheme was developed from information theory [Sal89].

$$1 - \sum_d \frac{\frac{f_d^t}{f_D^t} \log \left( \frac{f_d^t}{f_D^t} \right)}{\log(|D|)} \quad (2.2-4)$$

- term-discrimination value (TDV) [Sal89, pp.281 – 4]

The term-discrimination value is a sophisticated measurement of the distinguishing power of each term. The TDV is a measure of the power of a term to change the density of the document space of a collection.

$$\frac{1}{|D|(|D|-1)} \sum_{d_1 \in D} \sum_{\substack{d_2 \in D \\ d_2 \neq d_1}} (\text{sim}(d_1, d_2) - \text{sim}(d_1|_t^\lambda, d_2|_t^\lambda)) \quad (2.2-5)$$

where  $d|_t^\lambda$  is identical to  $d$  except that it has no occurrences of term  $t$  and  $\text{sim}(d_1, d_2)$  is a similarity measure (see Section 2.3 for some examples).

*Local Weightings* Some local weighting schemes are:

- raw term frequency [Dum91, Har86, Sal89]:

$$f_d^t$$

- Binary or Boolean [Dum91]

$$\begin{cases} 1 & \text{if } f_d^t > 0 \\ 0 & \text{otherwise} \end{cases} \quad (2.2-6)$$

- Log [Dum91, Har86]

$$\log(f_d^t + 1) \quad (2.2-7)$$

Salton and Buckley [SB91, p. 24] have produced weighting formulae for use with an electronic copy of an encyclopaedia. Their formulae are neither local nor global but a combination of the two. They use different formulae for comparing sentences, paragraphs and whole entries. For example, they compute the similarity between two documents,  $d_1$  and  $d_2$ , as the dot product of the two document vectors, where the term weights in the documents are given by Equation 2.2–8.

$$\frac{\left(1 + \frac{f_d^t}{\max_p f_p^t}\right) \log \frac{|D|}{|D^t|}}{\sqrt{\sum_{t \in T} \left(1 + \frac{f_d^t}{\max_p f_p^t}\right)^2 \left(\log \frac{|D|}{|D^t|}\right)^2}} \quad (2.2-8)$$

### *Other Refinements*

IR workers have developed many methods to improve recall by matching terms that are not identical. The best known method is called *stemming* or *morphological analysis*. Using this method, ‘computer’ and ‘computing’ would both be indexed by ‘compute’ [NTB84, Sal89]. Salton et al. have developed a method for choosing which words and groups of words to use for indexing a document collection [SYY74]. They use the discrimination value analysis (Equation 2.2–5).

## *2.3 Similarity Measures*

Similarity measures in IR are usually computed between a query and a document but can also be computed between documents. Most of these methods work with the vector model. The similarity of two documents in the vector model is computed by determining the distance between their corresponding vectors. Since the indexed terms are not necessarily independent the computed similarity measures may not be meaningful. They are often used in practice however.

William Jones and George Furnas [JF87, p. 420] made a geometric analysis of similarity measures which they intended to complement an empirical analysis of similarity measures. They attempted to characterize the iso-similarity contours of the measures to determine the behaviour of the measures. The analyses in Subsection 2.3.1 summarize some of their work. I have excluded the analyses of certain methods since I have not seen them referred



to elsewhere and the analyses show these methods to be impractical.

### 2.3.1 In Vector Space

The weight of term  $t$  in document  $d$  is  $w_{td}$ . The document  $d$  is represented by a vector of weights  $\vec{d} = \langle w_{t_1d}, w_{t_2d}, \dots, w_{t_{|T|}d} \rangle$ , where there are  $|T|$  distinct terms.

#### *The Overlap Measure*

The overlap measure is used in the SMART system [JF87]. It is very sensitive to document length, long and short documents will both produce improper results.

$$\frac{\sum_{t \in T} \min(w_{td_i}, w_{td_j})}{\min \left( \sum_{t \in T} w_{td_i}, \sum_{t \in T} w_{td_j} \right)} \quad (2.3-9)$$

#### *Inner Product*

When representing documents by their vectors an obvious similarity measure is to use the dot product (often called the *inner product* in the IR literature):

$$\vec{d}_i \cdot \vec{d}_j = \sum_{t \in T} w_{td_i} \times w_{td_j} \quad (2.3-10)$$

The inner product is not a good measure of relevance since it is possible for a single term's weight to dominate the entire measure. For example, in a document collection about massage techniques the word 'massage' may appear many times in all the documents. A longer document could easily have more occurrences of the word 'massage' than a short one. The inner product will tend to produce higher similarity scores for long documents than short ones.

#### *The Cosine Measure*

One attempt to correct this problem is the cosine measure (Equation 2.3-11). It is a measure of the cosine of the angle between the two document vectors normalized to between  $-1$  and  $1$ , a value of zero indicates that the two documents are totally unrelated. The cosine measure is essentially the same as the inner product where the vectors have been converted to unit vectors.

$$\frac{\vec{d}_i \cdot \vec{d}_j}{|\vec{d}_i| \times |\vec{d}_j|} \quad (2.3-11)$$

The cosine measure does not have the same problem as the inner product. This measures only the coincidence of terms but does not account for the different term weights in the documents. The inner product measure could be used to distinguish between the cases where two documents shared the same terms but in vastly different weights (one high and one low) and the case where two documents shared the same terms both with high weights. The cosine measure cannot distinguish those cases — it is limited in what it detects but it is hard to fool. Further, terms which are not part of both documents will increase the angle of separation between the vectors thus decreasing the cosine measure.

### *Conclusion*

In practice anything that normalizes for document length, e.g., the cosine measure is acceptable [Ber93]. Susan Dumais [Dum93] has found empirically that the cosine measure works best. I use the cosine measure for my experiments.

### *2.3.2 Other Methods*

#### *Bloom Filter Apprentice*

Bernstein [Ber90, MB93] developed a novel method for using Bloom filters [Blo70] to compute the similarity of passages in monographs (see also Section 2.1.3). His ‘link apprentice’ is part of an interactive program for reading texts, e.g., high school level history textbooks. At the user’s command, it computes a ranked list of the twenty pages that are most similar to the current one. The similarity measure used is the magnitude of the scalar product of the Bloom (filter) vectors for the pages. The filters are built by hashing on each individual word of each page. Bernstein reports great success with the apprentice but has not reported any formal analysis of its performance.

#### *Latent Semantic Analysis*

V. Raghavan and S. K. M. Wong [RW83, RW86] have analysed the theoretical underpinnings of the vector space model. In the model, a collection of documents is represented as a matrix of vectors of terms weights within documents. They suggest that a true matrix

model would allow arithmetic manipulation of the vectors to form new matrices. In 1988 Furnas et al. [FDD<sup>+</sup>88] presented *latent semantic analysis* (LSA).

LSA is a method based on the singular value decomposition (SVD) [FMM77] of a matrix representing a document collection. SVD is used to represent the matrix as a combination of linearly independent factors. The matrix is then approximated by using only the most significant  $k$  factors.

LSA can be used to compare documents for similarity with each other. It can also be used to measure the amount of overlap between single documents and collections. LSA can be used to compare the similarity of terms, ‘lighting’ and ‘shading’ for example. It can also be used to measure the relatedness of a term to a document [DDF<sup>+</sup>90].

The method performs at least as well as standard methods for standard test datasets such as SMART [DDF<sup>+</sup>90, p. 400 – 404]. Dumais reported that a 40% improvement over such standard methods has been achieved by applying LSA to matrices where the entries are a combination of local log weights (Equation 2.2–7, page 11) and global entropy (Equation 2.2–4) weighting. A particular advantage of the LSA method is that the factor representation avoids synonymy problems (see page 7).

Latent Semantic Analysis converts a matrix of term-document vectors into a compact representational model. Latent Semantic Analysis replaces a matrix of term-document vectors with a smaller model that represents the most important factors in the matrix. This model represents each document as a  $k$ -element vector where  $k$  is the number of factors found in the document collection. The model can be represented geometrically in  $k$ -space by  $k$  orthogonal dimensions. SVD is used to determine the factors to include in the model. Words used in the same documents are assumed to be related to each other by the topic of the documents. Those terms are placed close to each other in a representational model. Position in the  $k$ -space serves as a kind of indexing, i.e., the closer two items are in the space the more related they are.

The analysis proceeds by breaking the rectangular term-document matrix  $X$  (Equation 2.3–12) into three special matrices of singular vectors and singular values.

$$\underset{t \times d}{X} = \underset{t \times r}{T_0} \underset{r \times r}{S_0} \underset{r \times d}{D_0^T} \quad (2.3-12)$$

where

$T_0$  is the matrix of left singular vectors

$D_0^T$  is the transpose of the matrix of right singular vectors

$S_0$  is the diagonal matrix of singular values

$r$  is the rank of  $X$

Both the matrices composed of singular vectors have orthonormal columns. The elements of the singular vectors represent the factors extracted by SVD.

The essential factors are represented by the matrix  $\hat{X}$  (Equation 2.3-13) of rank  $k$  in a reduced model. The reduced matrix  $\hat{X}$  is closest ‘in the least squares sense’ [DDF<sup>+</sup>90, p. 398] to  $X$ . The new diagonal matrix  $S$  is obtained by removing rows and columns of zeros from  $S_0$ . The new singular vectors are obtained by deleting the corresponding rows and columns from  $D_0$  and  $T_0$ .

$$X \approx \hat{X} = TSD^T \quad (2.3-13)$$

The matrix  $\hat{X}$  is presumed to represent the important and reliable patterns underlying the data in  $X$  [DDF<sup>+</sup>90, p. 398].

Similarity measures are simply computed from the model. The similarity of two terms,  $i$  and  $j$  is  $(TS^2T^T)_{ij} = (\hat{X}\hat{X}^T)_{ij}$ . The similarity of the terms used in two documents  $u$  and  $v$  is  $(DS^2D^T)_{uv} = (\hat{X}^T\hat{X})_{uv}$ . There is an intuitive connection between the  $T$  matrix and the terms and the  $D$  matrix and the documents.

The parameters of the method are the term-document weight matrix and the number of factors to use in the reduced model. If too many factors are chosen then performance suffers. If too few factors are chosen, then it is likely that some essential factors will not be included [Dum91]. There is no a priori method for choosing the best value of  $k$  although research is being conducted to determine some useful heuristics [DDF<sup>+</sup>90, Dum91].

## *Chapter 3*

### *Approach*

#### *3.1 Experimental Design*

I wrote prototype programs to convert text files containing 1609 Usenet news postings (described in Appendix II) containing 2411 terms into a hypertext document. I converted the text into an equivalent marked-up form using HTML, The Hypertext Markup Language [BLC]. Marked-up documents have all the same content as the original document as well as special symbols representing the logical structure of the document. Additional symbols are used to indicate the presence of hypertext links. The conversion of the documents into a hypertext form requires that links be formed between documents and possibly within documents. I deal with links of two types: structural and semantic.

##### *3.1.1 Structural Links*

Structural links are hypertext versions of the connections that were already present in documents. I have represented the news articles as two logical units: 1) the text body 2) a header containing information about the message and the delivery system used for it. In the case of my data, the ‘References’ field in a header makes an explicit connection between articles [Hor83, HA87]. Figure 3.1 shows a typical message header with its structural links highlighted. Semantic links make connections between documents based on the similarity of their information content that was implicit in the original data collection. The semantic links in my implementation connect articles that contain the same technical terms. Another possible method would be to connect articles with similar content — as Bernstein [Ber90]

```

Relay-Version:  version B 2.10 5/3/83; site utzoo.UUCP
Posting-Version: version B 2.10.2 (Tek) 9/28/84 based on 9/17/84; site orca.UUC P
Path:  utzoo!watmath!clyde![...!orca!warner
From:  [warner@orca.UUCP] ([Ken Warner])
Newsgroups: net.graphics
Subject: Re: [Graphics Standards Information!]
Message-ID: <1414@orca.UUCP>
Date:  Thu, 21-Mar-85 12:56:10 EST
Article-I.D.: orca.1414
Posted:  Thu Mar 21 12:56:10 1985
Date-Received:  Tue, 26-Mar-85 06:32:48 EST
References: <7300008@hp-sdd.UUCP>
Reply-To:  warner@orca.UUCP (Ken Warner)
Organization: Tektronix, Wilsonville OR
Lines:  7
Summary:

```

Figure 3.1: A sample Usenet header

does. Since many Usenet articles contain significant portions of text quoted from other articles, many of the links thus formed would duplicate information already present in the header. The semantic links should reveal information that would not be readily apparent from structural links.

I created prototype programs to convert from files containing Usenet messages to hypertext form with links formed on the author, title and keyword header fields. I chose to implement the subject and author links<sup>1</sup> as links to menus. Links from those menus lead to articles. Figure 3.2 shows a menu of subject links. Figure 3.3 shows a menu of author links.

---

<sup>1</sup>The subject links were based on the 'Subject' or 'Title' fields of the message header. The author link was based on the 'From' field of the header.

## Articles with subject 'Circle Algorithm?'

`net.graphics#1443`

Subject: Circle Algorithm?  
 From: robison@uiucdcsb.CS.UIUC.EDU

`net.graphics#1456`

Subject: Re: Circle Algorithm?  
 From: jutz@pogo.UUCP (Curt Jutzi)

`net.graphics#1461`

Subject: Re: Circle Algorithm?  
 From: keithd@cadovax.UUCP (Keith Doyle)

Figure 3.2: Example of Subject Menu

## Articles with author 'Keith Doyle'

`net.graphics#507`

Subject: Re: Graphics to VCR, Help  
 From: keithd@cadovax.UUCP (Keith Doyle)

`net.graphics#517`

Subject: Elliptical arc algorithm HELP!  
 From: keithd@cadovax.UUCP (Keith Doyle)

`net.graphics#531`

Subject: Re: New IBM PC graphics board  
 From: keithd@cadovax.UUCP (Keith Doyle)

`net.graphics#564`

Subject: Re: Intersect line with polygon  
 From: keithd@cadovax.UUCP (Keith Doyle)

`net.graphics#660`

Subject: Language bindings for GKS and VDI (VDM?)  
 From: keithd@cadovax.UUCP (Keith Doyle)

`net.graphics#1461`

Subject: Re: Circle Algorithm?  
 From: keithd@cadovax.UUCP (Keith Doyle)

Figure 3.3: Example of Author Menu

### 3.1.2 Semantic Links

Having developed a structural linking prototype, I proceeded to experiment with methods for making links from the body of the messages. I have evaluated the application of methods from IR to the problem of creating semantic links.

#### *Purpose*

I designed an experiment to evaluate various semantic linking strategies. An effective strategy would be one that has a minimal number of links between articles with the same ideas, but does not link unrelated articles. The various linking strategies were evaluated by applying the concepts of *link distance* and *semantic similarity*. I define the link distance between two articles as the minimum number of links that must be traversed to navigate from one article to the another, in a specific hypertext system. The semantic similarity is a measure of how close the meanings of two articles are to each other. Semantic similarity is evaluated using methods from IR.

#### *Method*

Terms were chosen from the document collection and several term weighting schemes were applied to them. I simulated the effects of links between terms in different documents using the cosine similarity measure based on those weights. I analyzed the effects of the weighting schemes by comparing the link distances between all pairs of documents using the all-pairs shortest path algorithm and LSA.

There were many implementation decisions to be made. The number of links from a document could be limited by the number of terms in the document or I could allow an arbitrary number of links.

Permitting an unlimited number of links would create documents with an unwieldy number of links [BD91]. An analysis by link distance would be meaningless, since every document would be one link away from every other document. Such a strategy would be equivalent to allowing calls to a search engine from within a hypertext browser. While the idea may have merit, it is certainly not the focus of this investigation.

Having decided to limit the number of links from each document, I had to choose how to implement this decision. The limitation could be an arbitrary number or it could be based on the indexing scheme.



Limiting the number of links from a document based on term weight is the same as classifying every document as belonging to exactly one category and making links only within categories. Since document classification is not the goal of this thesis I did not implement that method. I considered implementing the links in a similar way as the structural links. I then had to decide how to form the links. Limiting the number of documents that a given term could be linked to would lead to documents being grouped by term. For example, if  $n_d$  document links of the  $|D| - 1$  possible are permitted then only  $n_d + 1$  documents will ever be linked on any term.

I chose to limit the number of links from a document to the  $n_d$  most similar documents. In a full implementation, two documents would have to have a term in common to be linked, otherwise there would be nothing to form the link on.

*Term Weights* I tested linking strategies using weighted terms. I compared the performance of schemes in which only the  $n_d$ ,  $1 \leq n_d \leq 5$ , most similar documents were linked. Preliminary experiments with graphs of differing degrees showed little improvement when more than 5 links were allowed from a document. This conclusion was also reached by Jacques Savoy [Sav93, p. 42] in a different experiment (see Section 3.1.3).

First I chose the terms that the links would be formed upon. The terms were weighted using each of the combinations of the following global weights:

1. the GfIdf (Equation 2.2-3, page 11),
2. inverse document frequency (Equation 2.2-2) and
3. entropy (Equation 2.2-4)

in combination with the local log weights.

### *Choosing Terms*

I made a list of all words<sup>2</sup> in all documents. R. E. Webber selected a list of 2478 terms, from a list of 16 076 words, based on his familiarity with the field of computer graphics.

---

<sup>2</sup>By *words* I mean all whitespace delimited strings containing alphabetic characters devoid of leading, trailing and multiple hyphens.

*Stemming* One of Salton's steps is to replace words with their stems (see Section 2.2.3). Stemming can often reduce the number of terms to be considered, but it is computationally intensive and requires an online dictionary. Donna Harman [Har87] found that stemming makes little difference in the performance of IR systems. I do not stem any words for my experiments.

The terms were weighted with a combination of local log and global entropy weighting (see Section 2.2.3) as suggested by Dumais [Dum91]. The list of 2478 terms is effectively represented by a reduced model of 100 factors by the LSA package [DDB]. The number of factors chosen was suggested by Dumais [Dum91].

### 3.1.3 *Evaluating Semantic Links*

LSA seems to have the best performance (in terms of recall and precision) of any IR method for general text [Dum91]. I chose LSA because it performed well for general text and avoids the synonymy problem so common with document collections from many authors. I used the document-to-document similarity measure available with latent semantic analysis (see Section 2.3.2) as an approximation of semantic similarity. My experiments were to determine how well other methods would perform as a linking strategy for a hypertext system.

For the purposes of evaluation, I treated the hypertext links as edges connecting documents in a directed graph. The link distance between two documents is the shortest path between the two nodes representing the documents. For example, in Figure 3.4, the shortest path from document C to document A is 2 (via either document F or B) and the shortest path from document B to document A is 1 (although a longer path via C and F exists). Note that the edges are directed, i.e., a link between two documents in one direction does not imply that there must be a link the other way too.

A directed edge between two documents in the hypertext graph is either present or not. If present, it indicates that at least one link exists between the documents. If there is no edge, then there is no link. For the purposes of the shortest path algorithm (see Appendix III), I assigned an initial weight of infinity to edges corresponding to document pairs with no links between them.

Figure 3.5 (page 25) is an outline of my experiment. A table of weights was constructed for each of the weighting schemes. A hypertext graph was built by linking every document

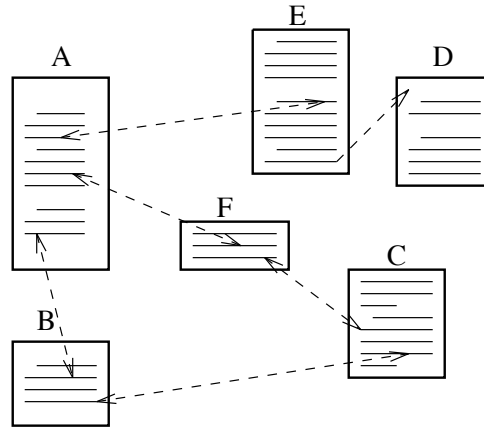


Figure 3.4: A Hypertext Graph

with the  $n_d$  most similar other documents. Similarity was determined by using the cosine measure (Equation 2.3–11). The link distance, defined as the shortest path for all pairs of documents, was then computed. The link distance and the semantic similarity, computed using LSA, were compared. There is a high similarity for documents with low outdegree but a less similarity for documents with high outdegrees. Full details are presented in Chapter 4.

#### *Methods for Evaluating Hypertext*

Although there are many papers about evaluating the user-interfaces of hypertext systems, I found only a couple of papers about evaluating the indexing of hypertext systems. Philippe Aigrain and Véronique Longueville [AL92] are developing methods to evaluate the quality of links in a hypermedia database of images. They compute the link distance as a function  $dp$  using a probabilistic model of user behaviour. For example,  $dp(0.9)$  is the smallest number of links the user must be allowed to traverse, to ensure a 90% probability of navigating from an image  $d_1$  to an image  $d_2$ .

Savoy [Sav93] created an IR system augmented with some hypertext links based on document to document similarity. He uses the poorly performing [JF87, p. 432] Dice measure to judge the similarity of documents and measures performance in terms of recall and precision.

I considered several methods for evaluating the quality of the hypertext graphs that my program produced. I compare the semantic similarity and link distance between all pairs of documents. I used the similarity measures obtained using latent semantic analysis as an approximation of the perfect similarity measure between documents.

Other approaches compare graphs made from LSA with graphs made with other methods. These methods could be used to determine which method performs better than the rest but would not provide an objective measure of performance. Here I outline some possible testing methods. Measuring the correlation between the shortest paths in a graph built from LSA data and one built using another method would be one way to method of evaluation. A t-test could be used to evaluate the significance of the difference between the link distance of every pair of documents in both graphs. Another test would be to compare the count of most similar documents that overlap between graphs weighted with LSA and those weighted with other methods.

I chose to test the correlation between link distances computed by different methods and semantic similarities between pairs of documents. Unlike methods where a graph is built from the LSA data, this method compares the link distance directly to the semantic similarity. This is better because the results do not need to be interpreted to exclude artifacts from the conversions into hypertext graphs. I also made plots of the coincidence of the two measures to see what the distribution was.

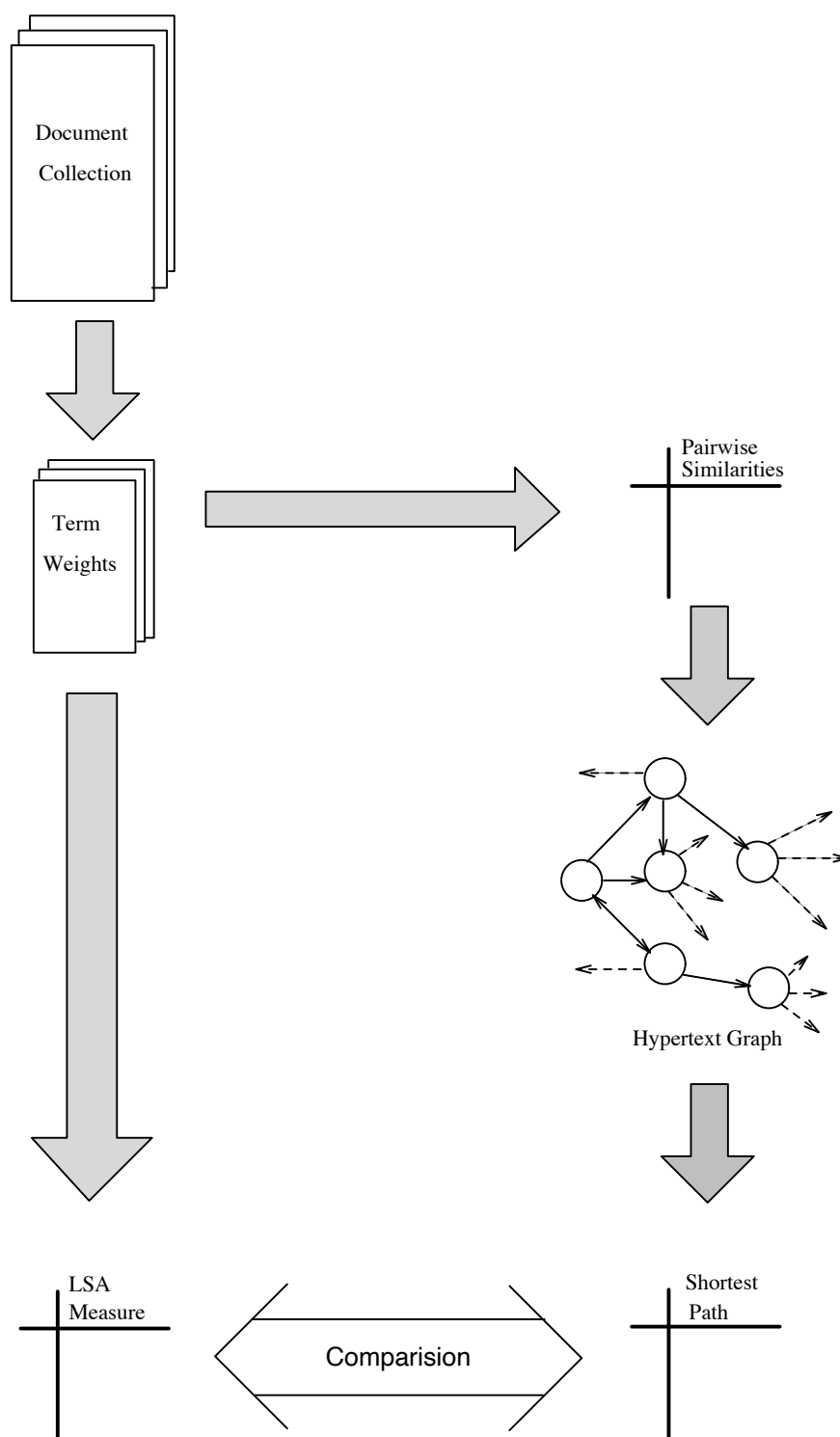


Figure 3.5: Experimental Design

## 3.2 *Experimental Details*

An overview of the major steps undertaken to generate the data for the experiment appears in Figure 3.5 on page 25. The current section is an account of all the steps. The correlations computed in the experiment are presented in Figure 4.6 on page 39.

### 3.2.1 *The List of Term Weights*

Each Usenet article was in a separate file. From every document a list of all words occurring in the body of the article and their frequency of occurrence was produced. Both of the similarity comparisons are done with files of terms and term frequencies. These files and the ordered list of them were produced before the other steps. All the terms in those lists were replaced by their term numbers and all non-terms were deleted from the lists.

### 3.2.2 *The LSA Similarity Measure*

The LSA similarity measure was computed using the term frequency files (see the left-hand side of Figure 3.5). This subsection details the steps taken to compute the LSA similarity measure.

1. Made a list of all non-empty files (filename LSA.list)
2. Concatenated all files into one file (filename DOC) in the order the files appeared in the list (LSA.list)
3. Computed the factor analysis using LSA
  - (a) Ran the pindex program with the following options: `-l 1 -m 50 -nc 0 -nd -w log entropy -P -N -v DOC` which computed the matrix for the factor analysis.
4. Produced a file (filename lsa.syn.data) of document to document similarity measures. This data is represented by the table named ‘LSA Measure’ on the lower left-hand side of Figure 3.5.
  - (a) Ran the syn program  $|D| = 1609$  times with the following options: `-S -d  $d_i$  -S -d a`, where  $d_i = 1 \dots |D|$ , to create the lsi.syn.data file.

### 3.2.3 The Shortest Path Measures

This subsection details what was done to compute the shortest path measures. An overview of this process is presented on the right-hand side of Figure 3.5.

1. Produced files of term weights from all files in the list of non-empty files using the Entropy-Log, Idf-Log and GfIdf-Log weighting schemes (see Section 2.2.3).
2. Computed the cosine similarity measure (Equation 2.3–11) between every pair of documents using the list (LSA.list) to ensure that documents were in the same order as for the LSA computation above.
3. Made graphs with outdegree of 1 to 5.
  - (a) For every document  $d_i$ , selected the five other documents that  $d_i$  has the largest cosine with.
  - (b) Created a graph for each of the outdegree levels using the cosine data.
    - The only edges in the graph of outdegree one connected each document to its most similar neighbour, the only edges in the graph of outdegree two connected each document to its two most similar neighbors, etc.
    - Edges between document  $d_i$  and itself were assigned a weight of zero; directed edges between the documents in the cosine list were assigned a weight of one; all other edges were assigned a weight of infinity.
4. Computed the shortest path between all document pairs using the shortest path algorithm presented in Appendix III.

### 3.2.4 The Comparisons

For each weighting scheme and outdegree, files were created that combined the LSA similarity value and shortest path for every pair of documents. Each record in the files consisted of two fields of data relating to the document pair  $d_i$  and  $d_j$ : the LSA similarity and shortest path between them. Similar files were created from the LSA similarity values and the GfIdf cosine similarity data (see Section 4.4.2). Five files (one for each outdegree) combining the GfIdf cosine similarity data and GfIdf shortest path data were also created. The correlations were computed using these files.

The density plots were also created using these files. The value in each shortest path field was multiplied by  $\pm 0.5$  to create a new file. The new files were used as input to a plotting program to produce Figures 4.1 – 4.5.



## *Chapter 4*

### *Results*

Different weighting schemes yield different metrics for evaluating the similarity of documents. A comparison of various metrics is necessary if we are to choose between methods for converting text to hypertext. Since there is no standard method for comparing two metrics, I tried two different approaches presented below.

#### *4.1 Hypertext Graphs*

The problem of comparing weighting schemes in the context of a hypertext system is made more complex by the necessary introduction of hypertext graphs (see Section 3.1.3). Hypertext graphs are constructed from a ranked list of document to document similarities. For example, in a graph with single outdegree each document is connected to the one other document that most resembles it. The graph built with the same weighting scheme but a higher outdegree will include the first one. Thus the graph with outdegree two will be composed of all documents and edges connecting each document to the two most similar other documents. For every document, one of its two most similar documents will, of course, also be found in the graph with outdegree one. LSA provides a finite measure of the similarity of two documents, there are never two documents that do not have any measure of similarity. Hypertext graphs however, may contain document pairs with no path between them. An important decision is what value to use to represent the ‘infinite’ path length between such documents. In this experiment I chose to use a number one larger than the largest possible path length for the graph,  $|D|$ .

## 4.2 *Evaluation Using Density Plots*

The density plots (Figures 4.1 – 4.5) show the distribution of path lengths against the LSA similarity measure for document pairs that are connected in the hypertext graph. The darker regions correspond to values with many document pairs — the lighter regions correspond to more sparse areas. The plots begin with link distance zero, i.e., documents that had exactly the same terms with the same frequency.

Figure 4.1 was made from a highly structured graph where there is only one edge leaving each vertex. The dark regions at the left of the plot correspond to the number of documents that are a short link distance from each other. In this graph the few documents that are linked have a short path distance between them as we can see from the largest distance shown in the plot. The short path length between documents may be used to argue in favour of creating hypertext systems from such graphs — similar documents are never many steps away from each other.

The plot for the same weighting scheme but with outdegree two (Figure 4.2) is very different. Whereas the longest finite path in the first plot was 5, with increased outdegree the path length extends to 18. This graph is much darker than the previous one. The many dark regions indicate that many document pairs are connected in this graph. Still more documents are connected in the graph where every document has outdegree three (Figure 4.3). The resulting graph is highly structured, although many more documents are reachable than before, the longest path is 48. When the outdegree is increased to four (Figure 4.4) the structure collapses as documents that have similarity to each other are connected directly. The longest path length here is 28, much like the first graph but with many more document pairs linked. The graph with outdegree of five (Figure 4.5) contains paths between all the pairs of documents in the graph of outdegree 4 and additional, less closely related pairs. In this graph the structure has collapsed further resulting in a maximal shortest path length of 22.

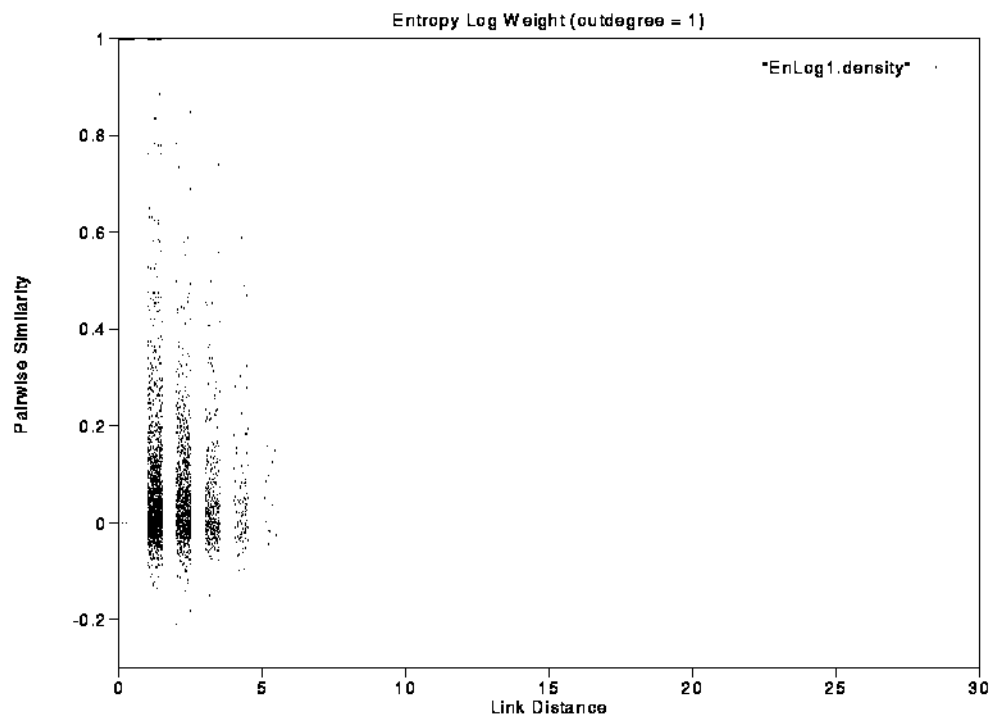


Figure 4.1: Density Plot of link distance computed using the global entropy and local log weight with an outdegree of 1 vs. the semantic similarity computed with LSA

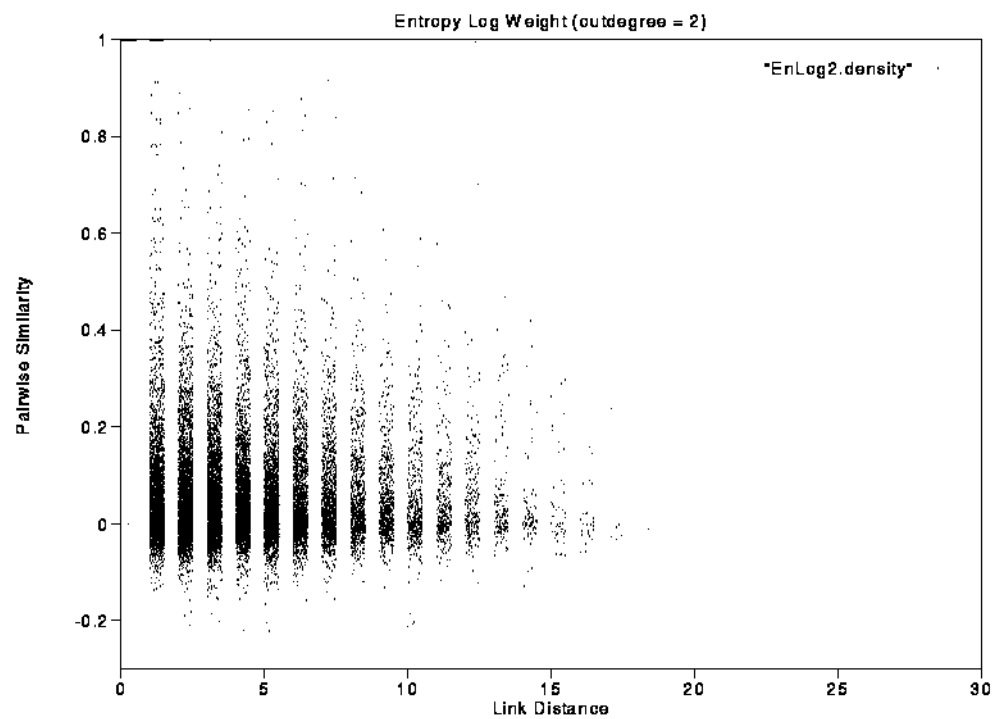


Figure 4.2: Density Plot of link distance computed using the global entropy and local log weight with an outdegree of 2 vs. the semantic similarity computed with LSA

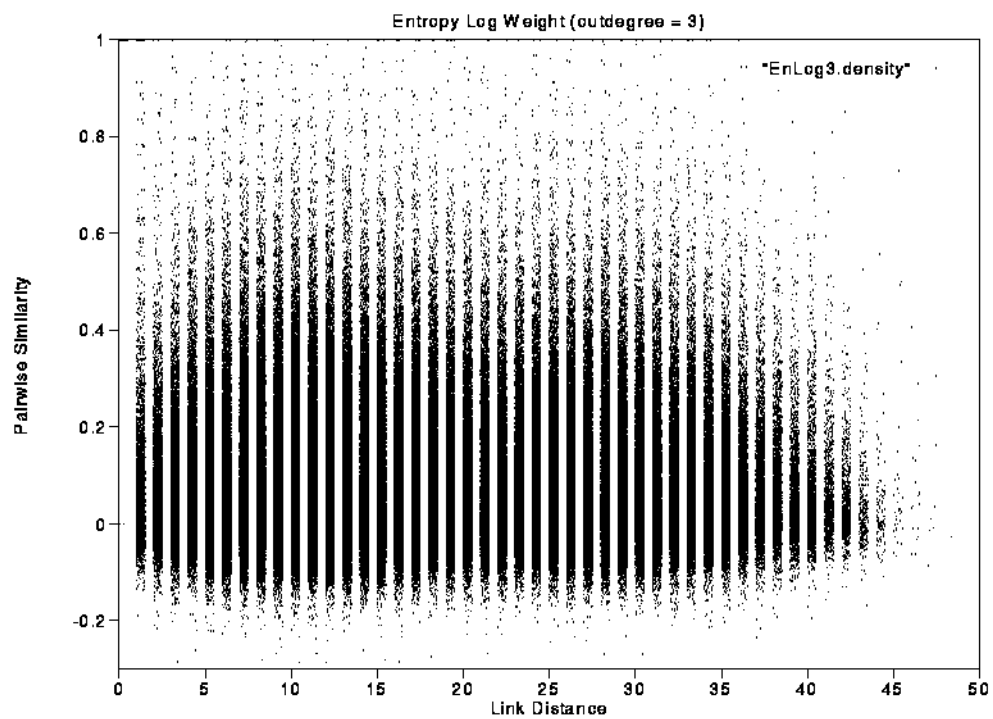


Figure 4.3: Density Plot of link distance computed using the global entropy and local log weight with an outdegree of 3 vs. the semantic similarity computed with LSA

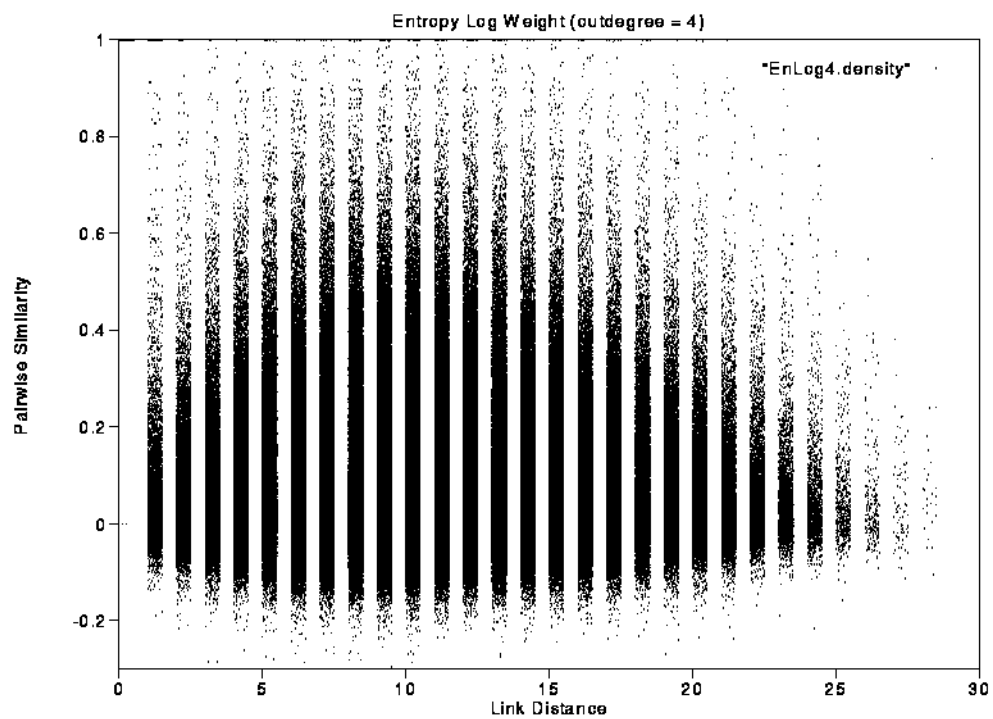


Figure 4.4: Density Plot of link distance computed using the global entropy and local log weight with an outdegree of 4 vs. the semantic similarity computed with LSA

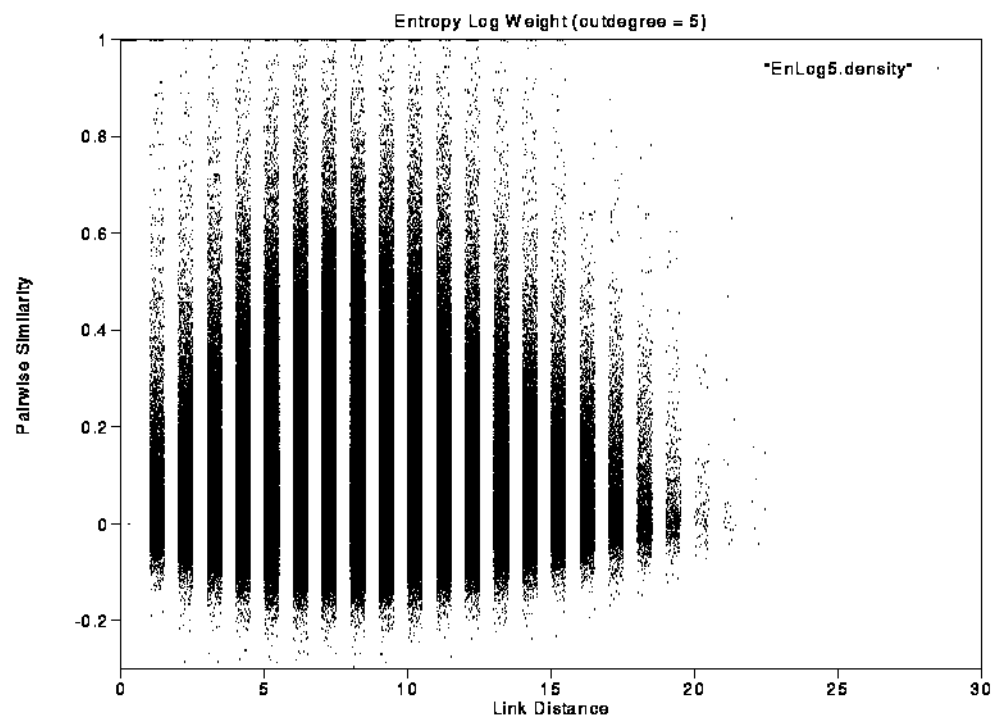


Figure 4.5: Density Plot of link distance computed using the global entropy and local log weight with an outdegree of 5 vs. the semantic similarity computed with LSA

### 4.3 Evaluation Using Correlation Measures

‘The Pearson product moment coefficient of correlation  $r$  [Equation 4.3–1] is a measure of the strength of the linear relationship between two variables  $x$  and  $y$ .’ [MS88, p. 445] It is used to estimate the true correlation for a population from a sample population. In the case of this thesis, the sample is drawn from the document collection and the population is all similar document collections.

$$r = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum(x_i - \bar{x})^2 \sum(y_i - \bar{y})^2}} \quad (4.3-1)$$

I have used the correlation coefficient as a second method for comparing the metrics, with the link distance as  $x$  and the semantic similarity as  $y$ . The LSA similarity measure is a real number while the link distance must be an integer. This difference means that the link distance is at best an approximation of the semantic similarity between two documents. As such, small differences in how the hypertext graph is made can have significant consequences for the comparison of the measures.

Table 4.1 shows the results of the correlation measure for all pairs of documents and for only those pairs that had a path between them for the graph built using a combination of entropy and log weights. The first column of numbers shows the correlation for all<sup>1</sup> pairs. Document pairs that did not have a path between them were assigned a path length of  $|D|$ , which is an upper bound on allowable path lengths in the graph. The second column shows the correlation measures for only those document pairs that had a path between them. These measurements are for the same hypertext graph as was presented in Figure 4.1 on page 31. This figure does not take into account document pairs without links between them.

The final column shows the proportion of pairs in the second column to the first. For example, with an outdegree of 1 less than 0.2% of all document pairs have a path connecting them. In a real hypertext system that would mean that there is no way of navigating from a given document to another, for the vast majority of document pairs. This figure is a concrete way of describing the sparseness of the corresponding graph.

Table 4.2 shows the same measurements but for the graph built using a combination of inverse document frequency and log weights. Similarly, Table 4.3 is the same data for a graph built using GfIdf and log weights. I determined that a major reason many of

---

<sup>1</sup>There were 2 588 881 pairs of documents.



Entropy-Log			
	All Paths	Finite Paths	Finite
1	−0.0054	−0.7458	0.18%
2	−0.0054	−0.3553	0.87%
3	−0.0054	−0.0253	33.90%
4	−0.0054	−0.0227	70.49%
5	−0.0054	−0.0199	79.80%

Table 4.1: Correlation of Entropy-Log Weights with LSA

the correlations were low was the correlation of unconnected documents, so I computed correlations for only those document pairs that had paths connecting them. The smallest dataset was the combination of the Idf and log term frequency weights with an outdegree of 1. It has 4470 pairs of documents.

The correlations have negative values because the greater the semantic similarity between two documents the greater their pairwise similarity. However, the greater the similarity between two documents the shorter the path should be between them, so it is expected that correlations between these tables should be less than zero. Generally, a correlation with magnitude greater than or equal to 0.20 is significant [Ben93].

Idf-Log			
	All Paths	Finite Paths	Finite
1	−0.0054	−0.7323	0.17%
2	−0.0055	−0.2775	1.18%
3	0.0000	−0.0216	47.51%
4	−0.0030	−0.0164	76.59%
5	−0.0067	−0.0200	87.31%

Table 4.2: Correlation of Inverse Document Frequency-Log Weights with LSA

GfIdf-Log			
	All Paths	Finite Paths	Finite
1	−0.0055	−0.6843	0.19%
2	−0.0055	−0.4010	0.68%
3	−0.0047	−0.1473	4.91%
4	−0.0008	−0.0162	44.16%
5	0.0022	−0.0207	62.48%

Table 4.3: Correlation of GfIdf-Log Weights with LSA

## 4.4 Effects of Using Graphs to Approximate Similarities

### 4.4.1 Effects of Infinite Path Length

Clearly small numbers of disconnected graph components can have a great effect on the correlation measure. Also, the effect of using a high value ( $|D|$ ) to represent infinite path length has an effect. Further investigation into the interaction of these factors is warranted. I present some suggestions in Section 6.1.

### 4.4.2 Effects of the Graph Approximation

In order to better understand the relationships between the pairwise similarity measures and shortest path figures, I computed additional similarities between them. Figure 4.6 is a representation of the additional experiments I undertook. In the figure, the grey arrows represent steps in the process of data collection (see Figure 3.5 for the details) and the hollow arrows indicate pairs of tables for which additional correlations were computed.

The correlation measured between the pairwise similarities and semantic similarity for all pairs was 0.0409. This is the ideal upper bound on all the other correlation measures. The correlations between the pairwise similarity measures and the link distances are presented in Table 4.4. The results show a higher correlation between the pairwise similarities and the link distances than between the pairwise similarities and the semantic similarity. Thus the low correlation values computed between the semantic similarity and the shortest path appear to be a result of the loss of detailed information during the conversion of the pairwise similarities to shortest paths.

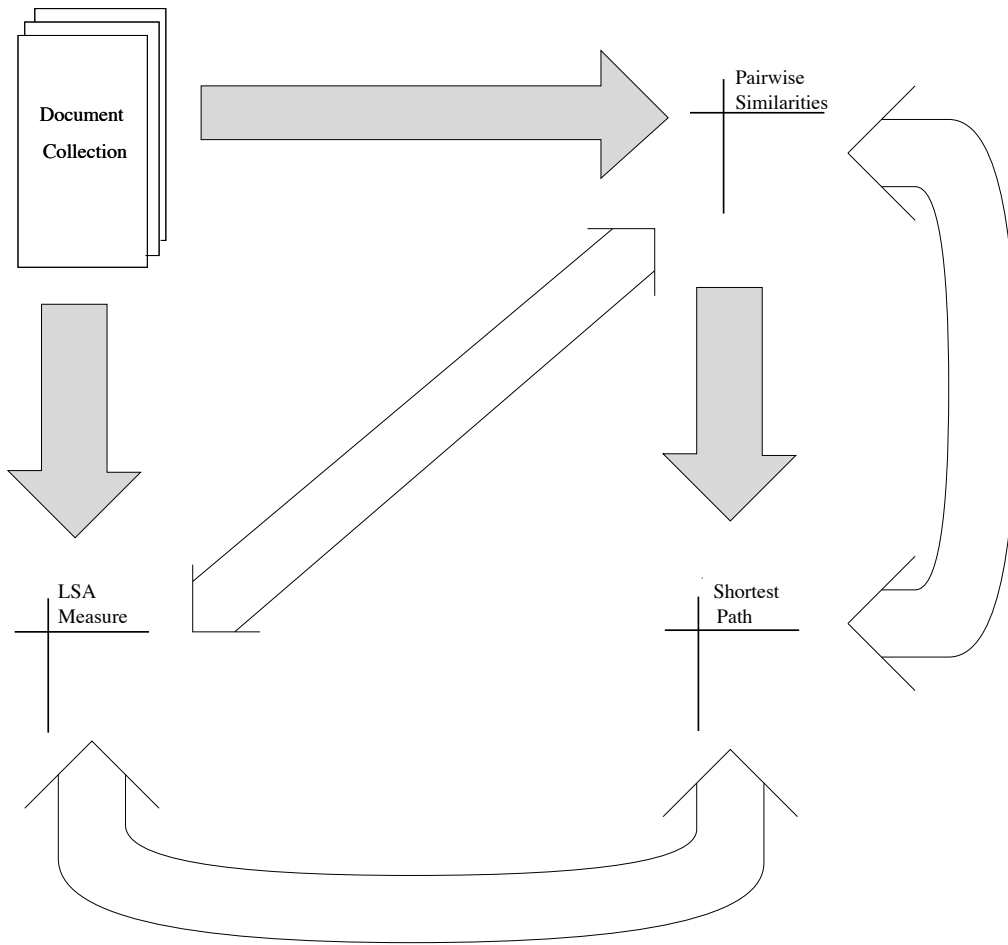


Figure 4.6: Correlations Tested in Experiment

GfIdf-Log Pairwise — Link Distance	
	Correlation
1	−0.0042
2	−0.0113
3	−0.0104
4	−0.0005
5	−0.0193

Table 4.4: Correlation of Pairwise Similarity for GfIdf and Shortest Path Length (Link Distance) for Graphs of Various Outdegrees

## *Chapter 5*

### *Conclusion*

I have examined the application of some methods from information retrieval to the problem of automatically converting text into hypertext. I simulated the creation of hypertext documents from plain text documents by building graphs composed of documents and hypertext links using IR techniques (see Figure 3.5, page 25). I experimented with three weighting schemes and five levels of outdegree. I evaluated the effects of the methods by comparing the link distances in the graphs to the semantic similarities between document pairs and by measuring the proportion of documents that are linked.

#### *5.1 What Is Needed For Automatic Conversion*

##### *5.1.1 What Are The Choices*

Two choices must be made in developing an IR-based method for automatically converting text into hypertext. A method for weighting terms must be chosen and the number of out-links to permit each document to have must be decided upon. Savoy[Sav93, p. 41] reports that allowing a document to have five out-links produces better performance than restricting the number of out-links to one. This is in keeping with my findings. I did not investigate methods with an outdegree higher than five because five seemed sufficient to reach almost all documents with a short enough path length.

There are two considerations in selecting an appropriate weighting scheme: 1) the correlation between the semantic similarity and link distance and 2) the degree of connectivity in the hypertext graph achieved using the scheme.

### 5.1.2 *Correlation*

It is desirable for an automatic system that converts text into hypertext to produce hypertext graphs with short path lengths between similar documents. This corresponds to the notion that when a user is browsing in a hypertext system it should be easy for them to find documents related to the topic they are reading about. A system that wastes the limited number of links available between documents on unrelated or loosely related document pairs is of little use. A system that does not provide links between highly related documents is of even less use. The correlation measure (Equation 4.3–1, page 36) can be used to measure the quality of links a method forges. A danger when using the correlation measure alone is that the value used to represent ‘infinite’ path length may skew the results. I found using the density plots together with the correlation figures more useful than relying solely on the correlation figures.

### 5.1.3 *Connectivity*

If we judge a hypertext system by the correlation measure alone we run the risk of creating hypertext systems no one will want to use. Typically hypertext graphs with single outdegree have very high correlations, in the connected portion, but they connect only a tiny fraction of the documents in the collection. A hypertext system geared towards searching and browsing must have paths between as many nodes as possible. Therefore another important consideration in choosing a method for automatically converting text to hypertext is what fraction of the document collection can be reached from any document. This corresponds to the proportion of document pairs that have paths between them.

## 5.2 *Pitfalls*

In testing simulations of methods for automatic conversion of text into hypertext I came across two difficulties worth noting. Both difficulties have to do with the conversion of pairwise similarity data into shortest path data.

### 5.2.1 *Representation of Disconnected Components*

The method of representing the path length between two documents that are unconnected in the graph can have a tremendous impact on the correlation measure. The correlation measure measures the strength of the linear relationship between the link distance and semantic similarity measures. If a semantic similarity value coincides with more than one link distance value then the magnitude of the correlation will be less than one. The degree to which it is less than one depends on the difference between the distance values and their means. The larger the value used to fill in for missing path lengths in the graph, the more skewed the correlation measure will be. Thus the value chosen to represent a non-existent path length should not be too large. However it must not be so small that it could be mistaken for a true path length. Future experiments should be done using the value exactly one larger than the maximum path length found in the graph.

### 5.2.2 *Loss of Information*

The pairwise similarity measures are the basis for a ranking of document pairs by similarity. In that ranking, fine distinctions can be drawn between different pairs based on the real number value of their similarity scores. When that data is used to build a hypertext graph, the fine distinction between document pairs is lost because in a hypertext system links are either present or not. In this way they correspond to the binary weights presented in Section 2.2.3.

It is not clear how the partial loss of ranking order affects the similarity scores. It is clear that there is more information about document to document similarity in the pairwise similarities than in the shortest path data. The longest path in the graph might be an important indicator of the quality of the hypertext document. Shorter maximal paths will permit more browsing as suggested by Raymond and Tompa [RT88, pp. 877–8].

## 5.3 *Which Method Is Best*

An examination of Tables 4.1 – 4.3 shows that the link distances in hypertext graphs built using a combination of Entropy and local log weights corresponds best with the measure of semantic similarity. However, the percentage of the graph connected for a given outdegree is higher for graphs constructed using Idf and local log weights than the other methods

tested. I believe that from the information available from these experiments, the percentage coverage data is the best upon which to base a decision about the relative quality of the different methods.

Thus, using the current evaluation method, it appears that Idf-Log is the best weighting scheme. This conclusion is based on the current understanding of the relationship between the link distance and semantic similarity measures. Further experiments with other parameters could result in other conclusions. In particular, it seems that the method of dealing with disconnected components tends to skew the correlation measure. Other methods may produce significantly different results.

## Chapter 6

### *Future Work*

#### 6.1 *Further Experiments*

A further investigation into why low correlations were obtained between similarity scores generated by LSA and other weighting schemes is warranted. A study of the differences between the similarity scores and graph representation might be of great interest.

The use of different values for document pairs which have no path between them should be examined. In the current experiment such pairs were assigned a path length of  $|D|$ . Since this practice led to unexpectedly low correlation values for some weighting schemes (see Section 4.4.1, page 38) other measures, such as using 1 plus the maximum path length actually found in the graph, should be tested.

The effect of using LSA to build the hypertext link graph (see the subsection titled *Methods for Evaluating Hypertext* on page 23) should be determined empirically.

In the current experiment every document had the same number of out-links. Perhaps a better experiment would be to limit the maximum number of links that could leave a document but only link documents whose similarity was above a cutoff value.

An investigation of the number of complete subgraphs found in the hypertext graph could be of great use. Every complete subgraph represents a *pocket* or collection of highly related documents. If there are many pockets in the document collection then the value of the correlation coefficient might be misleading: there could be a high correlation even though very few documents were linked. The resulting graph can be visualized as a single strand of string with many small knots in it. Each knot represents a pocket, a region with many interrelated documents. The pieces of string between the knots would represent single



links from a few (possibly only one) documents in each pocket to another pocket.

## 6.2 *Enhancements to the Experimental Model*

Many minor improvements to the method presented could be tested. For example, a method to recognize and correct minor variations in spelling (‘center’ for ‘centre’ and ‘raydiosity’ for ‘radiosity’ for example) could be of benefit when working with unedited text, such as I have. In the case of Usenet postings the possibility of only presenting links to documents outside of the current thread should be considered. This would make it easier for users to locate a few novel connections instead of being overwhelmed by semantic links that duplicate structural ones (see Section 3.1.1, page 17).

## 6.3 *Enhancements in a Working Prototype*

Salton et al.’s method (see Section 2.2.3) for choosing terms should be used in any real implementation. As well, proper nouns should also be indexed if the system is to be useful for browsing.

User-interface issues must be dealt with in a full implementation. A discussion of such issues is beyond the scope of this thesis.

## 6.4 *Latent Semantic Analysis*

Methods for selecting the best number of factors, to use with LSA, will increase its utility as a method of computing similarity.

A method for rolling in new terms and documents into the model without recomputing the SVD would be useful. A method whereby the essential terms are extracted from new docs might help here.

Deerwester et al. [DDF<sup>+</sup>90, p. 405] note that the LSA method does not deal well with words with multiple meanings, e.g., ‘bank’. Such words will be represented by the weighted average of their different meanings. I expect this will be less of a problem with documents from a single domain of discourse than a general collection. A successor method to LSA should not have this property.

## *Appendix I*

### *An example hypertext system*

Hypertext systems are typically implemented in a graphical user interface with multiple windows each of which contain text [Con87]. New windows are brought up as the user selects links to follow. Information can be found either by having the system search for a string or by following links. Some hypertext systems provide visual browsers to show the user a graphical representation of the database they are querying.

The tkWWW program is a browser designed for use with the World Wide Web hypertext project [Kro92, chapter 13]. The World Wide Web project ‘seeks to build a world wide network of hypertext links.’ [Wan92] Figure I.1 contains the list of commands available from a non-graphical interface to the World Wide Web system.

---

(1) Next page	(2) Back page	(3) go Up to previous document	(4) Find keywords in WWW index
(5) List hypertext links	(6) Recall document stack	(7) go to Top of current document	(8) go to End of current document
(9) Go to document at relative address mail	(10) saVe document text in scratchpad	(11) eXit hypertext session	(12) Query hypertext links
(13) Comment to owner of current document.	(15) follow Succeeding article/document reference.	(16) go Home to initial start-up document.	(17) save Anchor (bookmark) for reference.
(18) Do action associated with reference.	(19) collect Waiting (new) items.	(20) Modify hypertext.	(21) get Instructions (help) on this program.
(?) help	(-) esCHYPERTEx	(++) homebase	

---

Figure I.1: Help screen for text-only tkWWW interface

## *Appendix II*

### *The Raw Data*

There is much useful information available in Usenet (such as, Article II.1), but most of it is not easily accessible. Without proper indexing and searching techniques, the information is difficult to find and use. In 1989, Andrew Tanenbaum estimated that Usenet was probably the largest computer network in the world [Tan89]. Messages sent over Usenet form part of an important information resource. With an average of 16 Mb of new messages available from Usenet every day, it is clear that anyone wanting to find material will need some form of cross-indexing and searching. A hypertext system could provide that and more. Current technology forces users to view messages in one of a few rigidly defined ways. However there would be great benefits to viewing messages in other orders. The idea of a dynamic version of Usenet was suggested as early as 1987 [Fic87].

It is convenient to use articles collected from Usenet as raw data for my thesis, but the results of my thesis project should apply equally well to a wide variety of unstructured texts.

#### *II-1 What format does the raw data have?*

News postings (like those found in the following figures) consist of two parts: 1) a message consisting of text and 2) a header containing information about the message and the delivery system used for it [Hor83, HA87]. The header often contains lines tagged ‘Subject’, ‘Summary’, and ‘Keywords’ to aid in indexing. However, in practice these lines are seldom useful because they are maintained haphazardly.

Figures II.1, II.2 and II.3 are examples of commonly occurring types of messages.

Path: aramis.rutgers.edu!rutgers!ames!cit-vax!polecat!hemphill  
 From: hemphill@polecat.caltech.edu (Scott Hemphill)  
 Newsgroups: sci.math,comp.graphics  
 Subject: Re: Request for information on Monte Carlo Methods for Integral Equations  
 Summary: Monte Carlo references  
 Keywords: Rendering, Integral Equations  
 Message-ID: <2996@cit-vax.Caltech.Edu>  
 Date: 9 Jun 87 20:49:10 GMT  
 References: <180@dana.UUCP>  
 Sender: news@cit-vax.Caltech.Edu  
 Reply-To: hemphill@csvgax.Caltech.EDU (Scott Hemphill)  
 Organization: California Institute of Technology  
 Lines: 25  
 Xref: aramis.rutgers.edu sci.math:972 comp.graphics:627

In article <180@dana.UUCP> mlp@dana.UUCP (Mark Patrick) writes:  
 >I am trying to develop an understanding of stochastic techniques and how  
 >they are applied to computer graphics (in particular radiosity, and solutions  
 >to the general rendering equation). I have a pure maths and computer science  
 >background but not much exposure to probability theory or solving integral  
 >equations. I would be interested in hearing about text books or papers  
 >describing the Monte Carlo method for solving integral equations.  
 >

With a Math/CS background I recommend

R. Y. Rubenstein, *Simulation and the Monte Carlo Method*,  
 Wiley, New York, 1981.

It contains an excellent summary of methods used for random variate generation, and good information on variance reduction techniques.

For applications to computer graphics, I would start with  
 James T. Kajiya, "The Rendering Equation", SIGGRAPH '86,  
 pp. 143-150.

Scott Hemphill  
 hemphill@csvgax.caltech.edu  
 ...!seismo!cit-vax!hemphill

Figure II.1: An example of why news is an information source

Relay-Version: version B 2.10 5/3/83; site utzoo.UUCP  
 Posting-Version: version B 2.10.2 9/5/84; site swcadvax.UUCP  
 Path: utzoo!watmath!clyde!burl!ulysses!mhuxr!ihnp4!...!rich  
 From: rich@swcadvax.UUCP (Jeanne Rich)  
 Newsgroups: net.graphics,net.micro,net.wanted  
 Subject: Interactive drawing tool for IBM PC-AT  
 Message-ID: <143@swcadvax.UUCP>  
 Date: Fri, 11-Jan-85 18:32:53 EST  
 Article-I.D.: swcadvax.143  
 Posted: Fri Jan 11 18:32:53 1985  
 Date-Received: Mon, 14-Jan-85 03:39:37 EST  
 Distribution: net  
 Organization: Intel Corp., Hillsboro, OR  
 Lines: 21  
 Xref: watmath net.graphics:593 net.micro:9046 net.wanted:5529

I am looking for an interactive drawing program currently running on the IBM PC-AT under DOS or XENIX that will produce a drawing data file in some known format.

The following are some requirements for the program:

- Uses a mouse or tablet for input.
- Has a similer interface to MACDRAW.
- Will put the drawing produced by the program in some sort of known data format (ie:VDM format).
- Must be available now.

I would like very much to hear about any commercial or academic programs that meet this requirement.

---Jeanne Rich  
 Intel Corp.  
 Software CAD Group  
 Tools Team  
 (503) 640-7787.  
 ...!decvax!decwrl!sequent!ogcvax!omovax!swcadvax!rich

Figure II.2: Asking for information

Relay-Version: version B 2.10 5/3/83; site utzoo.UUCP  
 Posting-Version: nyu notesfiles V1.1 4/1/84; site csd2.UUCP  
 Path: utzoo!...mit-eddie!godot!harvard!seismo!cmcl2!csd2!hummel  
 From: hummel@csd2.UUCP  
 Newsgroups: net.graphics  
 Subject: Image processing on the IBM PC  
 Message-ID: <49200001@csd2.UUCP>  
 Date: Wed, 31-Oct-84 21:14:00 EST  
 Article-I.D.: csd2.49200001  
 Posted: Wed Oct 31 21:14:00 1984  
 Date-Received: Sat, 3-Nov-84 20:16:30 EST  
 Organization: New York University  
 Lines: 16  
 Nf-ID: #N:csd2:49200001:000:831  
 Nf-From: csd2!hummel      Oct 31 21:14:00 1984

I received some information from a company in Quebec that makes a couple of IBM PC compatible boards for image processing. The more interesting board, the "Oculus 200," is a 512 by 512 pixel eight bit frame store, with a seven bit frame grabber (digitizer), and a video out for a monochrome monitor. The cost is roughly \$2K. For 24 bit color, three boards (and three slots) would be needed. There doesn't seem to any provision for 8 bit color. There are some interesting software options, in the \$500 - \$1500 range, including an image of character data to ascii file converter. They don't give error rates or processing times.

The company is CORECO, Inc. Has anyone purchased one of these boards, and have any experience or comments about using them?

Bob Hummel  
 (hummel@nyu.ARPA)  
 (...ihpn4!cmcl2!csd2!hummel)

Figure II.3: An article asking for and providing information

## Appendix III

### *All-Pairs Shortest Path Algorithm*

```
SHORTESTPATH( $G$ )
1.  $size \leftarrow \text{rows}(G)$ 
2. for  $k \leftarrow 1 \dots size$ 
3.   for  $i \leftarrow 1 \dots size$ 
4.     for  $j \leftarrow 1 \dots size$ 
5.        $g_{ij} \leftarrow \min(g_{ij}, g_{ik} + g_{kj})$ 
6.     endfor
7.   endfor
8. endfor
end.
```

Figure III.1: Modified Floyd-Warshall Shortest Path Algorithm [CLR91, pp. 558 – 563]

$G$  is an  $n_d \times n_d$  adjacency matrix for a directed weighted graph.

The initial edge weights are one of: zero, one or infinity. All elements  $g_{ii}$  have initial weight 0. All elements for which there is an edge between vertices  $i$  and  $j$  have weight 1.

After the final iteration, the value of each element  $g_{ij}$  is the length of the shortest path from vertex  $i$  to vertex  $j$  and vice versa.

The algorithm has a tight asymptotic bound  $\Theta(n_d^3)$ .



## REFERENCES

- [ACG91] Maristella Agosti, Roberto Colotti, and Girolamo Gradenigo. A Two-Level Hypertext Retrieval Model for Legal Data. In *SIGIR '91*, pages 316 – 325, 1991.
- [ACM85] ACM-SIGIR. *Research and Development in Information Retrieval Eighth Annual International ACM SIGIR Conference*. ACM, 5 – 7 June 1985.
- [ACM87] ACM. *Hypertext '87 Papers*, The University of North Carolina, Chapel Hill, North Carolina, 13 – 15 November 1987. Association for Computing Machinery.
- [ACM88] ACM. *Proceedings of the ACM Conference on Office Information Systems*, Palo Alto, CA, 23 – 25 March 1988. SIGOIS Bulletin v.9 #2 – 3 (April – July 1988).
- [ACM89] ACM. *Hypertext 89 Proceedings*, Pittsburgh, Pennsylvania, 5 – 8 November 1989. Association for Computing Machinery.
- [ACM91] ACM. *Hypertext '91 Third ACM Conference on Hypertext Proceedings*. Association for Computing Machinery, 15 – 18 December 1991.
- [AL92] Philippe Aigrain and Véronique Longueville. Evaluation of navigational links between images. *Information Processing & Management*, 28(4):517 – 528, July – August 1992.
- [Bak62] Frank B. Baker. Information retrieval based upon latent class analysis. *Association for Computing Machinery Journal*, 9:512 – 521, 1962.

- [Bal93] V. Balasubramanian. List of Open Issues in Hypertext/Outline of Review. File `//eies2/eies2/www/WWW/bala/issues` in the World Wide Web hypertext database, October 1993. Telnet address 128.235.1.43, login id `www`.
- [Bas91] Reva Basch. Books Online: Visions, Plans, and Perspectives for Electronic Text. *Online*, July 1991.
- [BCSR91] A. Bookstei, Y. Chiaramella, G. Salton, and V.V. Raghava, editors. *SIGIR '91 Proceedings of the 14th Annual International ACM/SIGIR Conference on Research and Development in Information Retrieval*, Chicago, Illinois USA, 13 – 16 October 1991. ACM SIGIR, The Association for Computing Machinery.
- [BD91] Emily Berk and Joseph Devlin, editors. *Hypertext/Hypermedia Handbook*. Software Engineering. Intertext Publications, McGraw-Hill Publishing, 1991.
- [Ben93] Alfred Benn. Personal communication, 17 December 1993.
- [Ber90] Mark Bernstein. An Apprentice That Discovers Hypertext Links. In *Hypertext: Concepts, Systems and Applications*, pages 212 – 223, 1990.
- [Ber93] Mark Bernstein. Personal communication. e-mail, 8 June 1993.
- [BL89] Ken Bice and Clayton Lewis, editors. *CHI '89 'Wings For The Mind'*, Austin, Texas, 30 April – 4 May 1989. ACM SIGCHI, Addison-Wesley.
- [BLC] Tim Berners-Lee and Daniel Connolly. *Hypertext Markup Language A Representation of Textual Information and Meta Information for Retrieval and Interchange*. World Wide Web, 15 March 1993 version edition. This hypertext document has WWW address `http:info.cern.ch/hypertext/WWW/MarkUp/MarkUp.html`. It can be read by connecting to `info.cern.ch` (numeric address 128.141.201.74) with the telnet protocol.
- [Blo70] Burton H. Bloom. Space/Time Trade-offs in Hashing Coding with Allowable Errors. *Communications of the ACM*, 13(7):422 – 426, July 1970.
- [BP88] Christine L. Borgman and Edward Y.H. Pai, editors. *Information & Technology Planning for the Second 50 Years Proceedings of the 50th Annual Meeting*

*of the American Society for Information Science*, volume 25, Atlanta, Georgia, 23 – 27 October 1988. Learned Information, Inc.

- [BS93] Richard Brandwein and Mike Sendall. HTML converters. WWW address <http://info.cern.ch/hypertext/WWW/Tools/Filters.html>, 1993.
- [Bus45] Vannevar Bush. As we may think. *The Atlantic*, 1945. As reprinted by Nelson [Nel87].
- [CLR91] Thomas H. Cormen, Charles E. Leiserson, and Ronald L. Rivest. *Introduction to Algorithms*. McGraw-Hill, 1991.
- [CNBKO90] W. Bruce Croft, Marie-France Bruandet Nicholas Belkin, Rainer Kuhlen, and Tim Oren. Hypertext and Information Retrieval: What are the Fundamental Concepts? In *Hypertext: Concepts, Systems and Applications* , pages 362 – 366, 1990.
- [Con87] Jeff Conklin. Hypertext: An Introduction and Survey. *IEEE Computer*, 20:17 – 41, 1987.
- [CRM89] Peter Clitherow, Doug Riecken, and Michael Muller. VISAR: A System for Inference and Navigation in Hypertext. In *Hypertext '89 Proceedings* , pages 293 – 304, 1989.
- [DB91] Nicholas Duncan and David T. Barnard. The document-to-document correction problem. Technical Report 91-315, Department of Computing and Information Science, Queen’s University at Kingston, July 1991.
- [DDB] Scott Deerwester, Susan Dumais, and Michael Berry. LSI package. copyright ©1990 Bell Communications Research, Inc.  
Provided by Susan Dumais <dumais@bellcore.com> Bell Communications Research, 445 South St., Morristown, NJ 07960, USA.
- [DDF<sup>+</sup>90] Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. Indexing by Latent Semantic Analysis. *Journal of the American Society for Information Science*, 41:391 – 407, September 1990.

- [DDL<sup>+</sup>88] Scott Deerwester, Susan Dumais, Thomas Landauer, George Furna, and Laura Beck. Improving information retrieval with latent semantic indexing. In *ASIS '88 Proceedings of the 51st Annual Meeting* , pages 36 – 40, 1988.
- [DeR89] Steven J. DeRose. Expanding the Notion of Links. In *Hypertext '89 Proceedings* , pages 249 – 255, 1989.
- [Dum91] Susan T. Dumais. Improving the retrieval of information from external sources. *Behavior Research Methods, Instruments, & Computers*, 23(2):229 – 236, 1991.
- [Dum93] Susan T. Dumais. Personal communication. e-mail, 5 May 1993. Dumais is the corresponding author on many papers about latent semantic analysis (LSA) and a co-author of others [FDD<sup>+</sup>88, DDL<sup>+</sup>88, DDF<sup>+</sup>90, Dum91].
- [Eas90] Steve M. Easterbrook. What is Hypertext? In *Text Retrieval The State of the Art* , pages 119 – 137, 1990.
- [ELK<sup>+</sup>91] Dennis E. Egan, Michael E. Lesk, R. Daniel Ketchum, Carol C. Lochbaum, Joel R. Remde, Michael Littman, and Thomas K. Landauer. Hypertrext for the electronic library? CORE sample results. In *Hypertext '91 Proceedings* , pages 299 – 312, 1991.
- [Fah88] Eanass Fahmy. Programmatically generating connections in document forests. Master's thesis, Queen's University, Kingston, Ontario, Canada, 1988. Department of Computing and Information Science.
- [FB90] Eanass Fahmy and David T. Barnard. Adding hypertext links to an archive of documents. *The Canadian Journal of Information Science*, 15(3):25 – 41, September 1990.
- [FDD<sup>+</sup>88] George W. Furnas, Scott Deerwester, Susan T. Dumais, Thomas K. Landauer, Richard A. Harshman, Lynn A. Streeter, and Laren E. Lochbaum. Information retrieval using a singular value decomposition model of latent semantic structure. In *SIGIR '88*, Grenoble, France, 1988.
- [Fic87] David K. Fickes. Usenet in Hypertext form. Usenet Message-ID <8711130839.AA02525@bucsb.bu.edu>, posted to comp.society.futures,

November 1987. Obtained using anonymous ftp from site `nl.cs.cmu.edu` (numeric address `128.2.222.56`) in directory `/usr/toad/hypertext`. File name `hyperusenet.proposal.1`.

- [FLGD87] G.W. Furnas, T.K. Landauer, L.M. Gomez, and S.T. Dumais. The vocabulary problem in human-system communication. *Communications of the ACM*, 30(11):964 – 971, November 1987.
- [FMM77] George E. Forsythe, Michael A. Malcolm, and Cleve B. Moler. *Computer Methods for Mathematical Computations*, chapter 9. Least Squares and the Singular Value Decomposition, pages 192 – 239. Prentice-Hall, first edition, 1977.
- [FPS89] Richard Furuta, Catherine Plaisant, and Ben Shneiderman. A Spectrum of Automatic Hypertext Constructions. *Hypermedia*, 1(2):179 – 195, 1989.
- [Gil90] Peter Gillman, editor. *Text Retrieval The State of the Art*. Institute of Information Scientists, Taylor Graham, 1990. Proceedings of ‘The User’s Perspective’ (1988) and ‘Text Management’ (1989).
- [Glu89] Robert J. Glushko. Transforming Text Into Hypertext for a Compact Disc Encyclopedia. In *CHI ’89 ‘Wings for the Mind’*, pages 293 – 298, 1989.
- [Gol86] Lev Goldfarb. Metric data models and associated search strategies. *SIGIR Forum*, 20(1 – 4):7 – 11, Spring – Summer 1986.
- [Gol90] Charles F. Goldfarb. *The SGML Handbook*. Clarendon Press, 1990.
- [HA87] M. Horton and R. Adams. Standard for interchange of USENET messages. Request for Comments 1036, Internet Engineering Task Force, December 1987. Obsoletes RFC 850 [Hor83].
- [Hal88] Frank G. Halasz. Reflections on Notecards: Seven Issue for the Next Generation of Hypermedia Systems. *Communications of the ACM*, 31(7):836 – 852, July 1988.
- [Har86] Donna Harman. An experimental study of factors important in document ranking. In *SIGIR ’86*, 1986.

- [Har87] Donna Harman. A failure analysis on the limitations of suffixing in an online environment. In *SIGIR '87*, 1987.
- [Har92] Donna Harman. Evaluation issues in information retrieval. *Information Processing & Management*, 28(4):439 – 440, July – August 1992.
- [Hen90] Diane Henderson, editor. *Information in the Year 2000: From Research to Applications Proceedings of the 53rd ASIS Annual Meeting*, volume 27, Toronto, Ontario, 4 – 8 November 1990. Learned Information, Inc.
- [Hor83] Mark R. Horton. Standard for interchange of USENET messages. Request for Comments 850, Internet Engineering Task Force, June 1983. Updated by RFC 1036 [HA87].
- [HR88] Udo Hahn and Ulrich Reimer. Automatic Generation of Hypertext Knowledge Bases. In *Proceedings of the ACM Conference on Office Information Systems*, pages 182 – 188, 1988.
- [JF87] William P. Jones and George W. Furnas. Pictures of relevance: A geometric analysis of similarity measures. *Journal of the American Society for Information Science*, 38(6):420 – 442, November 1987.
- [Jon90] Kevin P. Jones. Natural-language processing and automatic indexing: a reply. *The Indexer*, 17(2):114 – 115, October 1990.
- [KN90] C. Korycinski and Alan F. Newell. Natural-language processing and automatic indexing. *The Indexer*, 17(1):21 – 29, April 1990.
- [Kro92] Ed Krol. *The Whole Internet User's Guide & Catalog*. O'Reilly & Associates, Inc., 1992.
- [LH85] Margaret E. Lundy and Richard A. Harshman. *Reference Manual for the PARAFAC Analysis Package*. Scientific Software Associates, 48 Wilson Avenue, London, Ontario, N6H 1X3, Canada, 1985. PARAFAC is available via anonymous ftp from `phobos.ssc1.uwo.ca`.
- [MB93] Elli Mylonas and Mark Bernstein. A literary apprentice. Submitted to *Computing in the Humanities*, 1993.

- [Mea92] Charles T. Meadow. *Text Information Retrieval Systems*. Academic Press, 1992.
- [MS88] William Mendenhall and Terry Sincich. *Statistics for the Engineering and Computer Sciences*. Dellen Publishing Company, second edition, 1988.
- [Nel87] Theodor Holm Nelson. *Literary Machines*. Privately published, The Distributors, 702 South Michigan, South Bend IN 46618 USA, 87.1 edition, 1987.
- [NTB84] G.Th. Niedermair, G. Thurmair, and I. Büttel. Mars: A retrieval tool on the basis of morphological analysis. In *Research and Development in Information Retrieval* , 1984.
- [Rab86] Fausto Rabitti, editor. *1986 — ACM Conference on Research and Development in Information Retrieval*. ACM-SIGIR, ACM, 8 – 10 September 1986.
- [Rai87] Rainer Hammwöhner and Ulrich Thiel. Content Oriented Relations between Text Units — a Structural Model for Hypertexts. In *Hypertext '87 Papers* , pages 155 – 176, 1987.
- [RT87] Darrell R. Raymond and Frank Wm. Tompa. Hypertext and the New Oxford English Dictionary. In *Hypertext '87 Papers* , pages 143 – 153, 1987.
- [RT88] Darrell R. Raymond and Frank Wm. Tompa. Hypertext and the Oxford English Dictionary. *Communications of the ACM*, 31(7):871 – 879, July 1988.
- [RW83] Vijay V. Raghavan and S.K.M. Wong. A critical analysis of vector space model for information retrieval. In *Research and Development in Information Retrieval* , 1983.
- [RW86] Vijay V. Raghavan and S.K.M. Wong. A critical analysis of vector space model for information retrieval. *Journal of the American Society for Information Science*, 37(5):279 – 287, September 1986.
- [Sal68] Gerard Salton. *Automatic Information Organization and Retrieval*. McGraw-Hill computer science. McGraw-Hill Book Company, 1968.
- [Sal89] G. Salton. *Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer*. Addison-Wesley, 1989.

- [Sal91] Gerard Salton. Developments in Automatic Text Retrieval. *Science*, 253, 30 August 1991.
- [Sal92] Gerard Salton. The state of retrieval system evaluation. *Information Processing & Management*, 28(4):441 – 449, July – August 1992.
- [Sav93] Jacques Savoy. Effectiveness of information retrieval systems used in a hyper-text environment. *Hypermedia*, 5(1):23 – 46, 1993.
- [SB90] Gerard Salton and Chris Buckley. Approaches to global text analysis. In *ASIS '90 Proceedings of the 51st Annual Meeting*, pages 228 – 233, 1990.
- [SB91] Gerard Salton and Chris Buckley. Automatic text structuring and retrieval — experiments in automatic encyclopedia searching. In *SIGIR '91*, pages 21 – 30, 1991.
- [Sha86] W. M. Shaw, Jr. The foundation of evaluation. *Journal of the American Society For Information Science*, 37(5):346 – 348, September 1986.
- [Spa72] Karen Sparck Jones. A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, 28(1):11 – 21, March 1972.
- [SRA90] N. Streitz, A. Rizk, and J. André, editors. *Hypertext: Concepts, Systems and Applications*, The Cambridge Series on Electronic Publishing. INRIA, France, Cambridge University Press, November 1990. Proceedings of the First European Conference on Hypertext.
- [Swe80] John A. Swets. Effectiveness of information retrieval methods. In Belver C. Griffith, editor, *Key Papers in Information Science*, pages 349 – 367. Knowledge Industry Publications, Inc., White Plains, New York, 1980.
- [SYY74] G. Salton, C.S. Yang, and C.T. Yu. A theory of term importance in automatic text analysis. TR 74-208, Department of Computer Science, Cornell University, July 1974.
- [Tan89] Andrew S. Tanenbaum. *Computer Networks*. Prentice Hall, second edition, 1989.



- [TS92] Jean Tague-Sutcliffe. The pragmatics of information retrieval experimentation, revisited. *Information Processing & Management*, 28(4):467 – 490, July – August 1992.
- [TW86] Randall H. Trigg and Mark Weisner. TEXTNET: A Network-Based Approach to Text Handling. *ACM Transactions on Office Information Systems*, 4(1):1 – 23, January 1986.
- [vR84] C.J. van Rijsbergen, editor. *Research and Development in Information Retrieval*, British Computer Society Workshop, King’s College, Cambridge, 2 – 6 July 1984. Proceedings of the third joint BCS and ACM symposium.
- [Wan92] Joseph Wang. ANNOUNCING: tkWWW 0.3 Alpha. Usenet Message-ID <1992Aug30.232318.13727@athena.mit.edu>, cross-posted to comp.lang.tcl, alt.hypertext, comp.windows.x and comp.lang.sgml, August 1992.
- [WY92] S.K.M. Wong and Y.Y. Yao. An information-theoretic measure of term specificity. *Journal of the American Society for Information Science*, 43(1):54 – 61, January 1992.
- [WZW85] S. K. M. Wong, Wojciech Ziarki, and Patrick C. N. Wong. Generalized vector space model in information retrieval. In *SIGIR ’85*, 1985.
- [You90] Laura De Young. Linking Considered Harmful. In *Hypertext: Concepts, Systems and Applications*, pages 238 – 249, 1990.
- [YR87] C.T. Yu and C.J. Van Rijsbergen, editors. *SIGIR ’87 Proceedings of the Tenth Annual International ACM SIGIR Conference on Research & Development in Information Retrieval*. ACM-SIGIR, ACM, 3 – 5 June 1987.

## VITA

NAME	William James Blustein
PLACE OF BIRTH	Montréal, Québec, Canada
YEAR OF BIRTH	1967
POST-SECONDARY	The University of Western Ontario
EDUCATION AND	London, Ontario, Canada
DEGREES	1992 BSc (Honours)
	The University of Western Ontario
	London, Ontario, Canada
	1994 MSc
HONOURS AND	Special University Scholarship
AWARDS	The University of Western Ontario
	1993
RELATED WORK	Teaching Assistant
EXPERIENCE	The University of Western Ontario
	1990 – 1993
PUBLICATIONS	Michael A. Bauer, J. Michael Bennett, Scott T. Feeney, James Blustein, Richard McBride <i>UWO Dept. of Com- puter Science Technical Report #279 Replication Strategies for X.500: Experiments with a Prototype X.500 Directory</i> , October 1990 (ISBN 0-7714-1243-6).
	James Blustein, <i>LINKAGE Help Manual</i> , available from Drs. R. G. Korneluk and A. E. Mackenzie, Molecular Genetics, CHEO, 1501 Smyth Rd., Ottawa, Canada.

James Blustein, *How to Make LINKAGE Pedigrees*, available from Drs. R. G. Korneluk and A. E. Mackenzie, Molecular Genetics, CHEO, 1501 Smyth Rd., Ottawa, Canada.