Automatic text summarization as applied to information retrieval:

Using indicative and informative summaries

Min-Yen Kan

Submitted in partial fulfillment of the

requirements for the degree

of Doctor of Philosophy

in the Graduate School of Arts and Sciences

COLUMBIA UNIVERSITY

2003

©2003

Min-Yen Kan

All Rights Reserved

Abstract

Automatic text summarization as applied to information retrieval:
Using indicative and informative summaries

Min-Yen Kan

I identify weaknesses with the standard "ranked list of documents" information retrieval user interface by examining the search process as performed in the traditional library by professional librarians and catalogers. I distill these processes into a list of core strategies which can be effectively fulfilled by multidocument summaries which assist in both the searching and browsing process. This thesis implements such automatic text summarization components to create an alternative method of presenting search results coming from IR frameworks.

As a post-processor of results coming from a search framework, CENTRIFUSER implements these principles by producing both informative and indicative summaries that aid the user in information seeking tasks. CENTRIFUSER uses novel techniques in analyzing source articles as a nested tree of topics, which allows the system to compare and contrast discussions of common topics across documents, and to identify rare topics. Documents similar in topic distribution are grouped together to enable faster and more accurate relevance judgment.

A novel contribution in CENTRIFUSER is the focus on generating indicative summaries. I analyze two sources of indicative summaries – online public access

catalog summaries as well as annotated bibliography entries – by examining guidelines for writing such summaries and by cataloging types of information used in actual summary corpora. The study reveals that metadata, such as the purpose or audience of a resource, are important inclusions in indicative summaries. By using the study's results, I derive an algorithm that enables CENTRIFUSER to author indicative summaries that both utilize and include metadata, a novel contribution in the summarization field.

To enhance the quality and the variety of summaries that are produced, I have employed novel techniques in natural language generation. The system analyzes documents using a two-part method: high-level content planning deduces what semantic predicates to include and where to place them, and a low-level realization model computes the most appropriate phrasing for each predicate using both local as well as global context.

Contents

List of Figures **ix**

List of Tables **xv**

Chapter 1 Introduction **1**

1.1 Background 1

1.1.1 Problem statement 3

1.2 Centrifuser: Automatic text summarization 4

1.3 Contributions 7

1.4 Guide to the thesis 9

Chapter 2 Background **12**

2.1 Informational reference interviews 13

2.1.1 Strategy 1. Clarify and elicit the user's information need . . 16

2.1.2 Strategy 2. Form a search plan 17

2.1.3 Strategy 3. Evaluate documents for initial relevance 18

2.1.4 Strategy 4. Form a user model 19

2.1.5 Adapting strategies to the online IR context 20

2.2 Cataloging resources 22

2.2.1 Characteristic 1. Write descriptions to assist relevance judgments 24

2.2.2 Characteristic 2. Clearly differentiate bibliographic items shown . . . 26
2.2.3 Characteristic 3. Organize the listing to assist search . . . 28
2.2.4 Characteristic 4. Support alternative search methods . . . 29
2.2.5 Characteristic 5. Use uniform descriptions . . . 30
2.3 Discussion . . . 31
2.3.1 Related work in visualizing search results . . . 32
2.3.2 An argument in favor of text user interfaces . . . 34
2.4 Aspects of summarization . . . 35

Chapter 3 Creating document topic trees **38**

3.1 Document topic trees . . . 39
3.1.1 Translation from rich markup . . . 41
3.1.2 Spatial layout recognition . . . 43
3.1.3 Topic segmentation . . . 44
3.2 Improving linear segmentation: SEGMENTER . . . 44
3.2.1 Extracting tokens . . . 46
3.2.2 Weighting term occurrences . . . 47
3.2.2.1 Linking distance . . . 48
3.2.2.2 Assigning weights . . . 49
3.2.2.3 Zero sum normalization . . . 51
3.2.2.4 Finding local maxima . . . 52
3.2.3 Algorithm training . . . 53
3.2.4 Evaluation . . . 53
3.2.5 Building the document topic tree from linear segments . . . 55
3.3 Combining spatial and lexical features: CLASP . . . 56
3.3.1 Line style annotator module (`Layser`) . . . 59

3.3.2 Header priority module (`Header`) . . . 62
3.3.3 Lexical cohesion module (`Cohesor`) . . . 63
3.3.4 Feature weighter (`Combiner`) . . . 65
3.4 Using CLASP to derive the document topic tree . . . 66
3.4.1 Style catalog . . . 67
3.5 Evaluation of CLASP . . . 68
3.6 Conclusion . . . 71

Chapter 4 Creating composite topic trees 73
4.1 Related work . . . 75
4.2 Text type . . . 76
4.2.1 Document granularity . . . 78
4.3 Creating the composite topic tree (CTT) . . . 81
4.3.1 Design feature 1: Structural similarity metric . . . 81
4.3.2 Design feature 2: Using relative topic level (RTL) . . . 83
4.3.3 Design feature 3: Using a three tiered merging approach . . . 84
4.3.3.1 Multi-instance document identification and merging 85
4.3.3.2 Main cross document merging . . . 87
4.3.3.3 Hierarchical merging . . . 88
4.4 Evaluation setup . . . 89
4.4.1 Results . . . 91
4.5 Conclusion . . . 94

Chapter 5 Indicative and informative summaries for searching and browsing 96
5.1 Integration with offline processing . . . 97
5.2 Query mapping . . . 98

5.3 Supporting browsing with navigation links and extracted similarities . . . 99
5.3.1 Navigation links . . . 100
5.3.2 Informative synopsis based on similarities . . . 102
5.4 Supporting searching with generated indicative differences . . . 108
5.4.1 Generalizing to multiple documents . . . 109
5.4.2 Generalizing to interactive queries . . . 110
5.4.3 Algorithm overview of CENTRIFUSER's indicative group summaries . . . 111
5.4.4 Refining topic types . . . 113
5.4.4.1 Document categories . . . 114
5.5 Time complexity of CENTRIFUSER . . . 117
5.6 A note on support of level of detail . . . 119
5.7 Conclusion . . . 119

Chapter 6 Indicative summarization 121

6.1 Introduction . . . 121
6.2 Metadata as potential summary content . . . 123
6.2.1 Cataloging metadata standards . . . 124
6.2.2 Prescriptive summary guidelines . . . 127
6.2.3 Descriptive summary corpora analysis . . . 128
6.2.3.1 Online public access catalog summaries on consumer health information . . . 129
6.2.3.2 Annotated bibliographic entries . . . 129
6.2.3.3 Discussion . . . 131
6.3 Content planning in CENTRIFUSER's indicative group summaries . . . 133
6.3.1 Content planning . . . 134

6.3.1.1 Inter-category plan 135
6.3.1.2 Intra-category plan 136
6.3.2 Sentence planning and lexical choice 138
6.3.2.1 Organizing metadata messages into sentences . . . 138
6.3.2.2 Generating referring expressions 139
6.3.2.3 Lexical choice 141
6.3.2.4 Surface realization 142
6.4 Conclusion 142

Chapter 7 Statistically informed generation for natural language 144
7.1 Introduction 144
7.2 Training annotation of summary corpora 146
7.2.1 Automatic annotation using machine learning 147
7.2.1.1 A new feature: Genericity 148
7.2.1.2 Discussion 150
7.3 Learning for the content planner 150
7.3.1 Content determination 151
7.3.2 Content ordering 153
7.3.3 Evaluation of the global ordering 155
7.3.4 Stochastic content plan comparison with rule-based content plan 157
7.4 Learning for surface realization 157
7.4.1 Delimiting attribute values from associated text 160
7.4.1.1 Discussion 163
7.4.2 Comparison versus annotation of only attribute values . . . 165
7.5 Regeneration using resources 166
7.5.1 Parse tree transformation to functional description 169
7.5.1.1 Evaluation 172

7.5.2 Unifying associated texts and attributes 174
7.5.3 Aggregation module 175
7.5.4 Stylistic evaluation module 179
7.5.4.1 Identify and encode linguistic features 180
7.5.4.2 Predict features using machine learning 182
7.5.4.3 Score candidate texts 183
7.5.5 Evaluation 184
7.6 Relation to CENTRIFUSER 185
7.7 Conclusion 187

Chapter 8 Evaluation **189**
8.1 Methods 190
8.2 Results 195
8.2.1 Qualitative usability analysis 196
8.2.2 Comparative ratings of all interfaces 198
8.2.3 Demographics 199
8.3 Discussion 200

Chapter 9 Conclusion **205**
9.1 Main contributions 205
9.1.1 Assessing the fit between user needs and the standard ranked list 206
9.1.2 Document and composite topic trees 207
9.1.3 Topically targeted summarization and metadata integration 208
9.1.4 Stochastic natural language generation 208
9.2 Deliverables 209
9.3 Limitations and future work 211
9.4 Revisiting the information seeking process in the digital library . . . 216

Appendix A Survey of information retrieval displays **241**

A.1 Survey methods 243

A.1.1 Gopher 243

A.1.1.1 Archie, Veronica and Jughead 245

A.1.2 WAIS 245

A.1.3 World Wide Web 246

A.1.3.1 AltaVista 248

A.1.3.2 Google 249

A.1.3.3 Yahoo! 250

A.1.3.4 About.com 251

A.1.4 Domain specific search engines 251

A.1.4.1 Northern Light 252

A.1.4.2 Westlaw 253

A.1.4.3 OVID 254

A.1.4.4 EDGAR 255

A.2 Survey results 257

Appendix B Sample document on Angina **260**

Appendix C Building a "One POS per word" dictionary from COMLEX Syntax **271**

C.1 Step 1 - Maximizing coverage 272

C.2 Step 2 - Exception list 272

C.3 Exceptions 274

Appendix D Composite topic tree evaluation materials **275**

D.1 Guidelines for evaluation 276

D.2 Evaluation composite topic trees 278

D.2.1 Consumer healthcare documents on diseases 280

D.2.2 Consumer travel brochures 281

Appendix E On the collection of annotated bibliographic entries 284

E.1 Annotated bibliography language resource 286

E.1.1 Collection methodology 286

E.1.2 Encoding the XML bibliographic entry corpus 287

E.1.3 Semantic annotation of metadata 289

E.1.4 Listing of semantic metadata predicates 290

E.2 Corpus attributes 293

Appendix F CENTRIFUSER evaluation materials 296

F.1 Guidelines for evaluation 297

F.2 Screenshot of the evaluation interface 300

F.3 Ranking survey 301

F.4 Demographic survey 303

Index 305

List of Figures

1.1 A CENTRIFUSER summary generated from summarizing the first ten hits from Google on the query "Angina". 4

3.1 A sample document topic tree, encoding the *Merck Manual of Medicine*'s page on angina. The structure of the document is similar to the NHLBI document given in Appendix B. 40

3.2 Distribution of selected terms from a text, as shown by TextTiling. Taken from Hearst (1994). Reproduced with permission. 45

3.3 SEGMENTER architecture. 46

3.4 Partial term list from the `termer` module of SEGMENTER from the example sentences shown above. 48

3.5 A term, "angina", and its occurrences and assigned term types (marked as `term`) per paragraph in a hypothetical document. Each column indicates a sentence, with the number of occurrences indicated by numbers, and linkages between occurrences indicated by '`x`'s. 50

3.6 The term, "angina", from Figure 3.5 and its links and score assignment to paragraphs. 52

3.7 Flattening a binary tree derived from hierarchical agglomerative clustering. 56

3.8 CLASP system architecture. 58

3.9 Extending CLASP system architecture to work with richer text formats. 59
3.10 Five translated rules from `Combiner`. 67
3.11 Excerpt of a CLASP run on the example angina document given in Appendix B, showing the processes the lead up to `Combiner` process and subsequent transformation into a document topic tree. 68
3.12 Converting a document structure tree to features. 71

4.1 Excerpt of an automatically constructed topic tree for consumer healthcare documents on diseases. 74
4.2 Excerpts of sample consumer health document topic trees (in outline form) containing disease information, used in part to construct the composite topic tree shown in Figure 4.1. 78
4.3 Granularity levels, from (l) most general to (r) most specific. Relative Topic Levels (RTL) will be discussed in Section 4.3.2. 80
4.4 RTL in action. 1) original configuration, 2) after topic merging, 3) propagation and identification of document type. Nodes not essential to the RTL propagation process omitted due to space considerations. 85
4.5 Pseudo code for multiple instance document merging. 87
4.6 Pseudo code for main cross document merging. 88
4.7 Pseudo code for the final HAC step. 90
4.8 Three steps in the top-down hierarchical merging process. Topic nodes under consideration for merging indicated by the black ring. . 91
4.9 Excerpt of the evaluation outline for patient information on heart diseases. The entire evaluation outlines for both domains is replicated in Appendix D. 92

5.1 CENTRIFUSER's overall architecture. 98

5.2 CENTRIFUSER's browsing components: navigational links (top); bottom; multidocument summary (bottom). 100
5.3 Navigation browsing scope, indicated by the dashed outline, as illustrated before and after browsing to the "Surgery" topic. Resulting navigation bar shown below the tree. 101
5.4 Navigation control construction under different space limitations: a) with less space, b) with more space. 103
5.5 A pictorial representation of how *relevant*, *irrelevant*, *intricate* topic types are defined by the interaction of a document topic tree and two different queries, for k=2. 104
5.6 Aligning topics from the composite tree and individual topic tree. Typicality scores are then propagated across. "R"are and "T"ypical topic labels used by the indicative summarization process, discussed later in Section 5.4.3. 105
5.7 The sample summary from Figure 1.1, highlighting the generated indicative summary in the bottom half categorizes documents by their difference in topic distribution. 111

6.1 Sample CENTRIFUSER indicative group summaries, as shown earlier in Figure 5.7, repeated here for convenience. 133
6.2 Indicative summary content plan, solid edges indicate moves in the sample summary in Figure 6.1. 134
6.3 Predicates instantiated for the *atypical* document category for the summary in Figure 6.1. 135
6.4 Intra-category discourse plan, solid edges indicate moves in the atypical document category. The final choice on which obligatory structure to use is decided later during realization. 137

6.5 A document cluster consisting of three documents with salient metadata from a sentence planning viewpoint. Possible messages listed on the right. 138
6.6 A document cluster with salient metadata with discourse units selected as ovals, as planned by the sentence planner. Possible aggregated messages on the right. 139
6.7 Compaction operation in referring expression generation. 140
6.8 Reordering operation in referring expression generation in a document category consisting of four titles. 141
6.9 Sentence plan for the atypical document category. 142

7.1 Overview of the SIGNAL architecture for text generation. 146
7.2 Statistical content determination example. Starred entries are metadata predicates that could automatically be determined to be salient. 152
7.3 Examples of learned elements from the content planner. The content plan is a collection of probabilistic ordering constraints. 153
7.4 Highest agreement full orderings of the indicative summary metadata predicates using harmonic penalties. Predicates are swappable where "|" occurs. 155
7.5 Statistical content ordering of the *Guide* example. 157
7.6 Examples of learned elements from the surface realizer for the *Audience* predicate. The surface realizer consists of attribute values (underlined), and associated text that convey the predicate's semantics. 158
7.7 Instances of the *Readability* predicate, with associated text (top) and standing alone (bottom). 160
7.8 A portion of the list of stemmed lexical dependencies for various predicates, sorted by order. 161

7.9 An example functional description used in SURGE. Reproduced with permission from Elhadad and Robin (1996). 168
7.10 Same input as Figure 7.9, in LISP form. 169
7.11 Sentences in which the `animate` attribute value does not change the surface form. 170
7.12 Minimal pairs in which the `animate` attribute value changes the surface form. 170
7.13 A joke relying on different circumstantial interpretations. 170
7.14 Translation of two associated texts (in parse tree form) by the translator to non-deterministic functional descriptions. 172
7.15 Portions of filled associated text (l), showing `signalPredicate` and `cat` fields unified from source attribute values (r). 175
7.16 An aggregation example for an FD in the *Guide* example, through two iterations. 176
7.17 The compilation of aggregation statistics of two multi-predicate sentences from the 100-entry corpus. Semantically aligned parse trees are used to generate the analysis, but the annotated text is shown here for clarity. 177
7.18 Portions of FDs showing conveyed predicates and their use in valid FD sequences for summaries of the *Guide*. 178
7.19 Machine learning architecture for the features of the target predicate, with sample tuples shown. 183
7.20 Translated machine learned rule set (from native `ripper` format) for the relative clause feature. 183
7.21 Summaries of varying quality for the *Guide* at different lengths. . . 184

8.1 Excerpt of a coded transcript of a subject while examining the CENTRIFUSER system. 195

8.2 Quantitative evaluation based on raw data in Table 8.2. 199

C.1 Demoted nouns. 273
C.2 Promoted discourse markers. 274
C.3 Demoted *Adjectives*, reclassified as *Other*. 274

D.1 Guidelines for the CTT evaluation process, part 1. 276
D.2 Sample output used in instructions. 277
D.3 Critiquing guidelines in the CTT evaluation process. 278

E.1 Sample excerpt from an annotated bibliography entry. 285
E.2 Relation of micro and macro collection attributes 289
E.3 Portion of the annotated bibliographic entry from Figure E.1, represented as structured fields in our XML corpus. 289

F.1 Guidelines for the evaluation process, part 1. 297
F.2 Guidelines for the evaluation process, part 2. 298
F.3 Guidelines for the evaluation process, part 3. 299
F.4 Screenshot of the evaluation interface. 300
F.5 First page of the comparative ranking survey. 301
F.6 Second page of the comparative ranking survey. 302
F.7 First page of the demographic survey. 303
F.8 Second page of the demographic survey. 304

List of Tables

2.1 A comparison of the standard ranked list and CENTRIFUSER's alternative interface in modeling techniques used by information professionals. 32

3.1 Overview of weighting and linking scheme used in SEGMENTER; starred scores to be established by zero sum procedure, explained later. 50

3.2 Evaluation results on precision and recall scales. 54

3.3 Sample line and calculated `Layser` feature values. Document category features not shown. 61

3.4 The inventory of logical line styles in the `Layser` and `Combiner` modules. 62

3.5 Final `Header` features, calculated as the difference between initial `Layser` features. 63

3.6 Lexical cohesion features and values calculated for sample source and target words used in the `Coheser` module. 66

3.7 Glass-box, intrinsic evaluation of CLASP. Figures reflect precision versus group majority. 69

3.8 Black-box, extrinsic evaluation of CLASP via genre classification. Figures reflect precision versus group majority. BoW and KC figures are for both with and without the addition of CLASP features. . . . 71

4.1 Instances and sample substructure topic nodes for different text types. 79
4.2 Errors reported in the CTT construction module as reported by two reference librarians. 93

5.1 Conditions used to categorize documents into document types. . . . 114

6.1 Metadata fields or categories in MARC, ANSI and Dublin Core standards. 126
6.2 Prescribed features of annotated bibliographies from several sources. 128
6.3 Distribution of metadata features in library catalog summaries of consumer healthcare publications. 130
6.4 Distribution of topical metadata in the 100-sentence annotated corpus.131
6.5 Distribution of non-topical metadata in the 100-sentence annotated corpus. 132
6.6 Document categories and their distribution of topic types. 135

7.1 Summary semantic annotation accuracy over the 100 entries annotated corpus, using 5-fold cross-validation. Features are cumulative left to right. 148
7.2 Coverage of different orderings. 156
7.3 Random sample of the recovered associated texts, simplified from their parse trees. Those marked with a star were judged incorrect. . 163
7.4 Precision (by type) of predicate attribute value/associated text labeling, categorized by predicate frequency. Starred predicates (without values) are not amenable to the attribute value/associated text paradigm. 164

7.5 Parse tree to functional description translation accuracy. 173
7.6 List of features used in the stylistic evaluation module, with sample values for two example FDs from Figures 7.14 and 7.16. 181
7.7 Evaluation measures for the stylistics module. 185
7.8 SIGNAL modules that utilize the training corpus. 188

8.1 Frequency of positive (+) and negative (-) coded verbal comments made by subjects regarding the usability and content of the four systems tested. 197
8.2 Evaluation raw ranks. 202
8.3 Evaluation summed general demographics. Weights used for averages shown in parentheses. Only non-zero summed lines shown. . . 203
8.4 Evaluation summed computer demographics. Weights used for averages shown in parentheses. Only non-zero summed lines shown. . . 204

A.1 Features present in document descriptions across surveyed IR systems. 257
A.2 Features present in output interfaces aside from the document descriptions, in the surveyed IR systems. 258

C.1 Conflation of POS in COMLEX to term finding gross POS. 273

D.1 Node types produced by the algorithm for the two text types in the CTT evaluation. 279

E.1 Feature distribution across entries of different document lengths. Frequency of document feature given as entry, average relative position of feature given in parentheses (0 indicates the beginning of the entry, 1, the end of the entry). Document features listed in order of descending frequency in the annotated corpus. 295

Acknowledgments

This thesis would not have been possible without the support, direction and love of a multitude of people. First, I have been truly blessed to have two very wonderful women guiding me on the path of scholarship, whose skills have greatly complemented each other and gave me the unique chance to explore the ramifications of my work in different areas. Kathy McKeown, who led me through the beginnings of a project that was to become my thesis, and then let me lead my thesis work once I knew where I was going. Judith Klavans, for encouraging me to balance my Ph.D. career in all aspects of academia, research, family and friends.

I have been blessed to have had many supporting my endeavors for scholarship since the very beginning of this work, playing multiple roles for which I am greatly thankful for:

My parents Yuen-Han and Kit-Keung Kan, my sister Min-Ying for the moral support and constant irritation of asking about when I would be finished with the dissertation;

My thesis committee members: Luis Gravano, Inderjeet Mani and Tomek Strzalkowski for their critical reading of the thesis and giving positive criticism and asking insightful questions that enabled me to clarify claims and contributions which needed additional coverage in the thesis;

The CENTRIFUSER evaluation team: Desmond Jordan, David Kaufman, André Kushniruk, Mark LaFlamme and Vimla Patel without whose help my thesis' hypothesis would be untested and the system sorely missing the needs of its target audience;

Also thanks to Michael Elhadad, Marti A. Hearst and Jacques Robin for their kind permission in allowing me to reproduce their figures (Figures 3.2 and 7.9) for use in this thesis.

The Columbia Digital Libraries group, both from the STIMULATE 2 and

PERSIVAL projects: Eugene Agichtein, Shih-Fu Chang, Jim Cimino, Shahram Ebadollahi, Noémie Elhadad, Steve Feiner, Carol Friedman, Michel Galley, Alejandro Jaimes, Stephen Johnson, Vasileios Hatzivassiloglou, Panos Ipeirotis, Elizabeth LaRue, Simon Lok, Eneida Mendonça, Smaranda Muresan, Seungyup Paek, Carl Sable, Sergey Siegelman, Justin Starren, Greg Whalen, Yoon-Ho Seol and Kazi Zaman;

My officemates over the many Ph.D. years: Hongyan Jing, Barry Schiffman, Melania Degeratu and Jackson Liscombe for putting up with a smelly, stinky officemate that worked at odd hours at odd things;

My collaborators from industry and universities: Martin Braschler and Peter Schaüble from Eurospider AG, Branimir Boguraev, Roy Byrd, Herb Chong, Mary Neff and Yael Ravin from IBM T. J. Watson Labs, Marilyn Walker from AT&T's Shannon Research Labs, Janyce Wiebe of University of Pittsburgh, Slava Katz of New York University and Nina Wacholder of Rutgers University, for letting see the opportunities of natural language processing in the real world;

The natural language group, both past and present: Valerie Barr, Sasha Blair-Goldensohn, John Chen, Liz Chen, Kris Concepcion, Peter Davis, Pablo Duboue, Kathy Dunn, Dave Elson, Dave Evans, Elena Filatova, Julia Hirschberg, Gabriel Illouz, Hongna Lu, Sameer Maskey, Ani Nenkova, Shimei Pan, Shmuel Popper, Rebecca Passonneau, Dragomir Radev, Owen Rambow, Jay Sandhaus, Andy Schlaikjer, James Shaw, Eric Siegel, Simone Teufel, Jennifer Venditti, Wubin Weng, Brian Whitman and Hong Yu;

My project supervisors: Mukesh Dalal, Pascale Fung and Michelle Zhou for letting me get a glimpse of what research life was and whetting my appetite for it, and my project students and collaborators: Himani Naresh and Susan Lee;

The Computer Science Department's front (and often, back) office staffers: Rosemary Addarich, Alice Cueba, Twinkle Edwards, Mel Francis, Genevive

Gouborne, Pat Hervey, Mary McKenzie, Matthew Schlager and Martha Zadok for pulling strings for me in times of need;

and finally my friends Mary Carmel, Yi Fang, Lucy Joseph, Kenric Leung, Caroline Naguwa, Kaming Wu and countless others for showing love and support in both times of great happiness and in times of deep depression. I have made it this far only because you were there to bring me back to level no matter the depth of pain.

I am also grateful for the comments from the anonymous reviewers of the papers I have had the privilege of publishing in conferences and workshops. Much of the material in this thesis is based upon work support by the National Science Foundation under grants No. IRI 96-19124 and IRI 96-18797 (STIMULATE) and IIS-9817434 (DLI2). Some parts of the work were also supported by a grant from Columbia University's Strategic Initiative Fund sponsored by the Provost's Office. Any opinions, findings, and conclusions or recommendations expressed in this dissertation are mine alone and do not necessarily reflect the views of any of the above funding agencies.

To my parents, Kit-Keung Kan and Yuen-Han Kan.

To my most cherished sister, Min-Ying.

To Michael Hauben, may he rest in peace.

Chapter 1

Introduction

1.1 Background

Before the advent of personal computers and the emergence of the World Wide Web as a global digital library, people with a research question had only the option to go to a traditional library to find an answer. Entering the library, they might find many shelves of materials. How would one locate which book might meet their information need? The central catalog was one source that could fulfill this need. The catalog is a central repository of information about every book in the library, where books are represented by surrogate cards that contain information about the physical location and notes about the book. Cards were often duplicated such that the entries could be stored in different orders in multiple catalogs: by subject, author or title.

The World Wide Web has since altered the form and function of the library. People with research questions can still go to a library and consult the catalogs[1] for help, but more and more frequently will relegate the physical time-consuming search as a backup method. As a first pass for filling their information need, a person might

[1] At present, catalogs are more likely to be in digital form, such as in the Online Public Access Catalog (OPAC) form than in physical card catalog form

execute a general electronic search using related terms to the question or browse at particular domain-specific sites. In many cases, a query is entered using key terms and a ranked list of documents is returned to the user. Users can then retrieve documents that interest them by clicking on their links.

One can draw some parallels between these two processes of search in the traditional and digital libraries. Instead of shelves of books one now accesses digital documents. These documents are housed at different sites, as they might be on different shelves in a physical library. The information retrieval search engine's surrogate is the library's catalogs, although the two perform different tasks. The library catalog organizes the books by index attributes, such as title, author and subject, but cannot find books directly by searcher supplied terms and queries if they are not in the controlled vocabulary of subjects or in authority files (Hodges, 2000). Conversely, many search engines do not have explicit means of searching by a particular index attribute, as the time needed to edit and compile these resources is often prohibitive.

In the physical library, a researcher can always ask another person for assistance in locating some materials. One can ask others who are also searching for similar information or an information specialist, such as a reference librarian. The former method has been brought into the framework of information retrieval engines as collaborative filtering, in which previous sequential searchers are saved as history and offered to the user as potential alternate searches to try (Goldberg et al., 1992; Resnick et al., 1994). The latter method, focusing on the techniques of the reference librarian, is a central theme of this thesis.

A key place where the application of techniques from information and library sciences can yield substantial improvement in the search process is in displaying search results to the user. Thus, one can decompose the automated information retrieval (IR) system into a system consisting of two parts: a **framework** that

selects documents that are relevant to a search query and an **interface** that displays information about the retrieved documents to the user. Much of the design of current IR user interfaces has been driven by the capabilities of the corresponding frameworks.

Current search engines display information in what I define as a *standard ranked list* interface[2]: a ranked list of documents, ordered by relevancy. Each titled entry is accompanied by a short textual description. Long lists of relevant documents are split into multiple, browsable pages and a text box containing the original query allows the user to refine or change the search.

1.1.1 Problem statement

In this thesis, I hypothesize that the standard ranked list interface can be further augmented with abilities to synthesize useful summary information. I examine how the IR interface can be modified to match user needs and how the modification affects the information retrieval framework. Basically, I investigate how techniques in automatic text summarization can expose document structure in terms of topics to the user.

Hypothesis: **Automatic text summarization can provide a information retrieval interface that tailors results to user needs.**

What form the interface takes and which techniques of text summarization are employed are central topics of this thesis. As a means of addressing these problems, I have turned to information and library science, examining the strategies of the reference librarian and cataloger. In doing so, I have cataloged a set of strategies that the reference librarian employs to assist the information seeking process. I argue that using these techniques as a basis for formulating an IR interface would

[2]In Appendix A, I give the exact definition of the standard ranked list, as formulated by examining several different state of the art search engines, both domain-independent ones as well as domain-specific ones.

provide an alternative approach in visualization.

1.2 Centrifuser: Automatic text summarization

In this thesis, I develop such a fully-implemented, alternative information retrieval interface, called Centrifuser. It performs multidocument summarization of similar documents (the exact notion of similarity will be discussed later in Chapter 4). Figure 1.1 shows a Centrifuser summary for a sample query "Angina" (a heart condition).

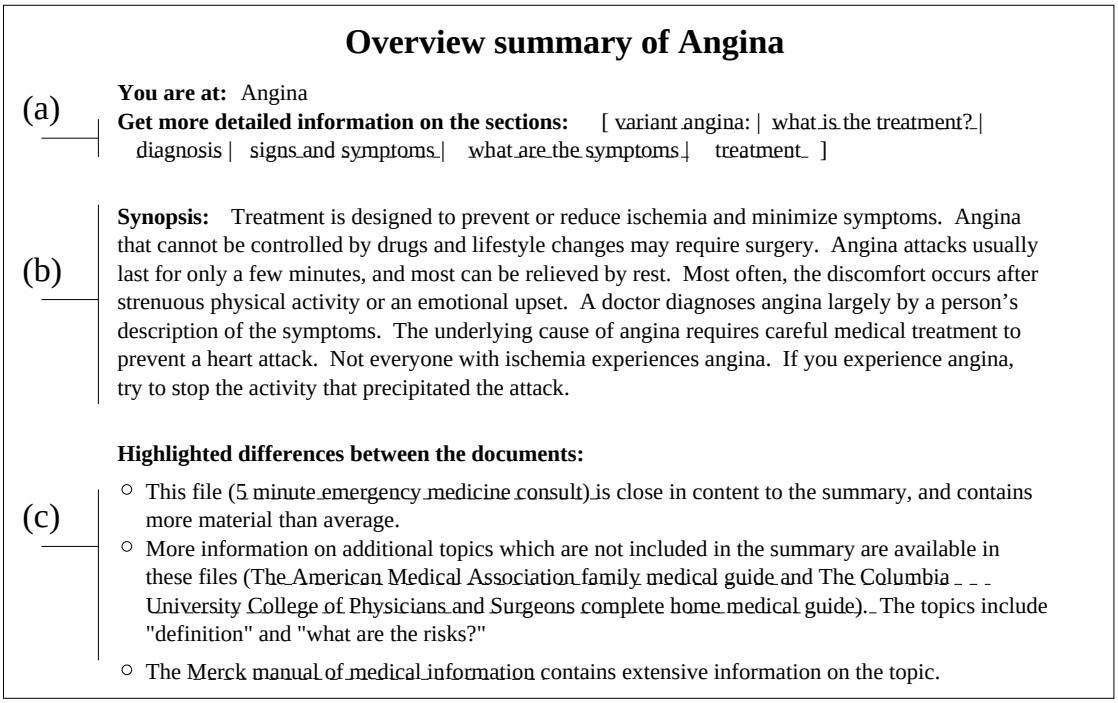


Figure 1.1: A Centrifuser summary generated from summarizing the first ten hits from Google on the query "Angina".

Centrifuser provides three different types of information as output, shown in Figure 1.1. At the top of the output are (a) **navigation links** that allow the user to navigate to broader and narrower subjects related to the original query. In

the middle, (b) an **informative extract** organizes sentences from the search results that represent their similarities. Finally, the bottom contains (c) **indicative summaries** which group the search results into several text bullets. The texts for the bullets are completely computer generated and based on differences in the document's topical structure and distribution. These text bullets utilize significant document features (metadata) about the text. In this thesis, I also explore how these indicative text bullets can be automatically generated using a variety of sentence structures as those found in human written summaries, to prevent the system from generating tedious and repetitive summary texts.

CENTRIFUSER creates these multidocument summaries by using similarities and differences between the documents (Mani and Bloedorn, 1999). The same paradigm has been used for the visualization of retrieval results in graphical user interfaces (e.g., BEAD (Chamlers and Chitson, 1992)). CENTRIFUSER differs from previous research in multidocument text summarization in that the system also compares documents against a model of expected information for documents. By this, I mean that CENTRIFUSER learns information about typical document length, organization and subtopics for particular types of documents, and codes this knowledge in a resource called the Composite Topic Tree (CTT), further discussed in Chapter 5. For this example, a CTT derived for patient information documents on medical conditions shows that symptoms, diagnosis, treatment, and links for more information are typical subtopics and usually occur in that specific order. This type of information enables the system to detect common or rare subtopics in documents which can be conveyed to the user.

To provide a sense of how CENTRIFUSER functions, I summarize the steps that CENTRIFUSER takes to construct its summary.

1. A set of topically related documents is selected by a search engine framework. These documents constitute the input to CENTRIFUSER .

2. Each document in the set is analyzed to identify its structure, in terms of subtopics, as will be discussed in Chapter 3. These analyses form the basis for transformation of each document into a hierarchical tree of topics. For example, a document containing patient information for a particular disease often consists of subtopics including symptoms, diagnosis and treatment.

3. In a related process outside of this summarization overview, CTTs are constructed by merging together many of these document topic trees. This process will be detailed in Chapter 4.

4. A user query is matched to a particular subtopic in each document. This match divides the document topic structure into three different subtopic regions to use for tailored summarization: the relevant region, the irrelevant (usually broader) region, and the intricate (too narrow) region. This is used as the basis for comparison across documents.

5. The relevant regions are processed in two ways. First, relevant regions are linked across documents by subtopic. For example, all symptom topics are linked. The topics are clustered (Hatzivassiloglou et al., 2001) and similar propositions are identified. Representative sentences from each topic are collected and put together to form an informative extract, which is shown in Figure 1.1 as item (b). Second, relevant region subtopics are classified by statistical methods to determine importance, some resulting in the most common or typical nodes, and the others being classified as rare or atypical.

6. Ratios of the number of typical, rare, intricate and irrelevant topics are computed and used to determine membership of each document into one of seven possible document category. These categories reflect the topical distribution of each document. For example, a document might focus on common subtopics such as symptoms and treatments with little or no information on

the more uncommon subtopic, prognosis. The algorithms that are responsible for these previous three items will be detailed in Chapter 5.

7. Group summaries are generated for each of the seven categories that have been assigned a document, shown as item (c) in Figure 1.1. The primary goal of these summaries is to indicate any outstanding features about the group of documents, rather than extracting text directly from them. Thus, these indicative summaries also include metadata about different types of content (e.g., if the document contains figures or tables). These indicative summaries are a key focus of this thesis and are discussed in more detail in Chapter 6. I further investigate how a stochastic corpus analysis of human indicative summaries can be generated by the computer to mimic their variety and fluency in 7.

Thus, CENTRIFUSER uses the topical structure of related documents to compute a summary. These summaries are tailored to the user query. For example, if the user wants to know about symptoms, then a summary which contains more information on symptoms will be generated.

1.3 Contributions

This thesis makes four contributions to the field of Natural Language Processing and related field of Information Retrieval. I briefly summarize these contributions below:

- **Analysis of how current information retrieval technology fits user information needs.** By reviewing library and information science literature and analyzing a corpus of catalog materials, I compile an inventory of different types of user information needs. I assess to what extent the current

standard ranked list interface fulfills the requirements of each of these individual user needs, and make recommendations on how to remedy the observed deficiencies.

- **Text segmentation and topic spotting.** Hierarchical text segmentation and topic spotting are necessary prerequisites of the text summarization algorithms presented in this thesis. I address both text segmentation and topic identification within structurally similar documents as part of this thesis' summarization algorithm. I have developed and enhanced the algorithms for computing topic similarity and the matching of equivalent topics, which are also necessary for multidocument summarization.

- **Automatic text summarization.** I contribute new methods in summarization that take into account the hierarchical nature of expository text in the form of topic trees. The topic tree structure allows CENTRIFUSER to generate summaries at different levels of granularity or on specific portions of the document in a principled way. My focus on producing indicative summaries that include metadata in a flexible framework differentiates my contribution from previous work.

- **Natural language generation.** I show how training on semantically annotated material can be used in conjunction with corpus analysis techniques to automatically derive a natural language generation system that mimics its training corpus. I use this system to generate indicative summaries of single documents that mimic existing indicative summaries. This system introduces a framework that shows how natural language generation can be used to generate new texts that both mimic and invent new variations of summary texts.

1.4 Guide to the thesis

The remainder of this thesis is organized as follows: Chapter 2 examines the techniques of information seeking and presentation as performed in the physical library by information professionals, and shows how their techniques reveal strengths and weaknesses of the current standard IR interface. In the main portion of the thesis, Chapters 3 through 7, I discuss my implementation of an automatic text summarization approach that addresses the weaknesses that I identified. In these main chapters, I detail the implementation of CENTRIFUSER, an automatic text summarization system that summarizes multiple documents that are of similar "text types" (to be defined then). Chapters 8 and 9 evaluate CENTRIFUSER in use in real-world scenarios, discuss the results and offer some conclusions. I expand upon this outline below:

Chapter 2 presents information from related research on information seeking and presentation. I first examine the field of information and library science, examining the flow of questioning and interaction between reference librarians, researchers and consumers with their information needs. I distill nine different strategies which librarians use in interacting with users directly as well as indirectly (through the compilation of cataloging materials). I show how the standard ranked list interface fails to adequately model the techniques used by librarians. On this basis, I make recommendations for how a system should address these weaknesses.

Chapters 3 through 7 comprise the main body of the thesis research. I present CENTRIFUSER, an automatic text summarization system that implements the techniques detailed in Chapter 2. CENTRIFUSER is limited to summarizing documents that are of the same domain and genre (a text type), which is further defined in Chapter 4. I present the details of the system modules in four chapters.

Chapter 3 illustrates how the system analyzes input articles to build a hi-

erarchical representation of the major topics and subtopics in each article. The development of the *document topic tree* neatly captures a document's discussion at varying levels of granularity. I show how to compute this document structure from both plain text as well as from more richly marked-up document formats.

In Chapter 4, I develop the notion of expected information for a particular text type. I define what text type means: as an intersection between text genre and knowledge domain. I show how to compute a tree model of expected information, the *composite topic tree*, by aligning sample document topic trees of the same text type to produce information about the prominence, ordering and expected contents and formatting of specific topics within these similar documents.

Chapter 5 brings together the document and composite topic trees modules of CENTRIFUSER, and shows how they perform their tasks in responding to an online query, to construct the three different sources of output by reprising the stages used to produce the sample CENTRIFUSER run shown earlier in Figure 1.1.

Chapter 6 details the algorithm used by CENTRIFUSER to produce its group indicative summaries, a type of summary that is well-suited for information retrieval presentation since it is specifically structured for relevance judgment. I begin by reviewing prescriptive guidelines for how professionals write indicative summaries. I detail both a preliminary and a more detailed corpus study of two types of indicative summaries and demonstrate an algorithm for authoring indicative summaries from metadata.

I improve upon the indicative summary generation in Chapter 7, in which I present the SIGNAL system, a statistically informed NLG system. Using the same indicative summary corpus compiled in the previous chapter, I induce constraints for generating new indicative summaries. This introduces corpus-training natural language generation for the purpose of summarization. I show how the process is divided up into two major components: one for determining the contents of a sum-

mary and their ordering and another for determining how each piece of information is realized as an English word, phrase or sentence.

Chapter 8 presents the system-wide evaluation of CENTRIFUSER on patient healthcare information documents. I detail how CENTRIFUSER was evaluated using a "think aloud" protocol and the methods used for evaluation. I present the raw evaluation data and interpret the results, which show how my initial hypothesis is supported.

Chapter 9 concludes the thesis with supporting materials, and reviews user needs in IR and how CENTRIFUSER is able to fulfill these needs. I discuss the implications of the thesis and recommend areas for future research to broaden the scope of the thesis and its possible application to general search engine technology.

Chapter 2

Background

Having described the standard ranked list interface in Chapter 1, I now seek out expertise on alternative methods in presenting its information. In this chapter, I do this by examining information retrieval from a perspective of cognitive engineering. Examining the information seeking process from a cognitive engineering perspective means decomposing the search process into steps and strategies carried out by people as they attempt to find information and answer questions. There is vast literature on the subject of knowledge seeking, and in this portion of the thesis I will concentrate on how the field of information and library science in particular has viewed the problem of information seeking, which I will overview and use as a basis for the critique of the standard ranked list and to establish guidelines for the construction of an alternative information retrieval interface.

In the traditional library, librarians often assist users in their searches in two types of scenarios. In the first scenario, a librarian interactively assists a client at the reference desk in an *information reference interview*. In the second scenario, a librarian is not available to assist the user directly. In this case, users can consult static library resources to search the library, such as online public access catalogs, subject guides and other bibliographic resources. I now examine

the different strategies employed by these information professions in the both online and offline scenarios, with particular attention to how weaknesses in the standard IR interface can be addressed and briefly describe how CENTRIFUSER addresses these areas. I then review related work and conclude by assessing how different aspects of text summarization are important to the online search process.

2.1 Informational reference interviews

Library patrons often come to the reference desk in search of many different things. Katz (1987) breaks this down into four groups of queries: directions to the restroom (directional queries), instructions on photocopier use (instructional queries), requests to borrow books (circulation queries), and assistance with finding information (informational queries).

Among other studies of reference desk activities, Sutton and Holt (1995) cataloged the distribution of these types of queries. The first three categories of dialog were found to account for over half of all questions asked, and can be effectively handled by any trained staff in the library. However, it is the fourth and final type, the *informational reference interview*, that calls on the knowledge and experience of the professional reference librarian. The informational reference interview is a process that can be factored into four steps (Katz, 1987):

1. Communicating the information need;
2. Selecting a search strategy;
3. Executing the search;
4. Assessing the relevance of the results.

Katz begins his discussion on this subject by stating that "the most common complaint heard among reference librarians about their work is that few people

know how to ask reference questions" (pg. 16). Thus, the first and possibly most difficult part of the informational reference interview is to find out what information the user wants.

A primary difficulty that is reported in the literature is that users often underspecifytheir information needs. An example of this is when a user comes to the reference desk with a very specific information need (e.g., recent years' statistics on number of people in America afflicted with high blood pressure) but only communicates a part of it to the librarian in their query (e.g., "I need information on high blood pressure"). When this occurs, and unexpected results are returned, one finds that a) users made a tacit assumption that the librarian understood the specifics implied by the request, or b) users lacked the terminology to correctly express their information need.

Strategies for addressing this issue can also be separated along this distinction. When librarians feel that the query is underspecified because its purpose or scope is not clear, they may want to explicitly ask for the user's intentions ("Are you looking for information on high blood pressure for a report?"). Users may be sensitive about their needs (e.g., if the information is for a family member who is ill) and thus questions of this sort need to be phrased in a manner that make it clear that a response is optional and that the purpose of the question is to target relevant sources (Jennerich and Jennerich, 1987). When librarians feel that the query is underspecified because the user does not possess the knowledge to properly express his needs, they can enumerate possible subtopics or facets of the user's query ("Well, there's information about high blood pressure on a number of subtopics, including symptoms and treatments.") (Sutton and Holt, 1995). This cooperative type of response can aid users in better defining their query so that the librarian can understand and address their needs.

Reference librarians also benefit from other channels of information in de-

termining their user's needs besides verbal communication. Attire, age and other nonverbal channels of information can help the librarian assess probable information needs. If a young person asked a librarian about the example question on high blood pressure, he would likely receive a different initial answer than if an adult wearing a lab coat and stethoscope posed the same question.

Once the needs of the user have been successfully communicated, the librarian's task is to translate the information need into a search plan. What information sources should be examined? Should a general catalog search be done or should specific reference materials for a particular subject area be consulted? Which materials may yield higher quality results? Which searches are easiest and least costly to perform? Based on the knowledge of the scope and type of information needed by the user, the librarian can assist in formulating an appropriate search plan.

In the course of, or at the end of the search process, the librarian may need to evaluate the materials produced by the search. Often, these search results are better assessed by the librarian than the user. Librarians are often able to quickly match the genre of information needed to the specific type of user. Does the user want information found in newspaper clippings, trade magazines or scholarly journals? Other questions center around the documents to be evaluated. Are the documents authorities on the topic? Are they highly regarded by scholars in the field? Are they up-to-date resources for the topic? In this regard, the librarian often acts as an information filter, narrowing the choices down to a manageable set of alternatives (2-5 sources). Occasionally, librarians are called on to recommend their favorite source and in these cases a single best choice is picked from all the alternatives.

To summarize, there are many aspects of the reference interview that exhibit specific strategies that would be desirable to adapt to an online IR setting. These include the following (non-exhaustive) features that I have discussed:

1. Reference librarians may first need to clarify and elicit the user's information

needs;

2. Reference librarians may need to decide which information sources to consult based on an assessment of the user's need;

3. Reference librarians must be able to evaluate potential sources during the formulation of the search process and during retrieval;

4. During an interview, reference librarians also form and take advantage of a user model and his information need.

I will examine how the standard ranked list fares in fulfilling these four criteria in turn in the next subsections.

2.1.1 Strategy 1. Clarify and elicit the user's information need

A standard ranked list gives a summary (first n sentences or QWIC – query word in context) on the top hits individually. These extracts, along with the documents' titles, can help a user decide whether specific documents match the intended usage of his query terms. Although the primary function of these extracts and titles is to determine the relevance of the particular document, the context they provide can also assist users in specifying a narrower or related query. The context provided by these texts may contain cues that focus and remind the user of their exact information need. Because of the "probe and revise" nature of many online IR-based queries, the document extract and titles in the ranked list may better assist serendipitous or tangential information retrieval (Ford, 1999; Toms, 2000) than fulfilling the initial probe queries of a user.

In an alternative IR interface, explicit assistance in clarifying queries can be provided; the interface can be designed purposely to aid the user in formulating

related and narrower queries; both to aid searching by refining and further detailing underspecified queries and to aid browsing in giving related queries to execute and view. In Chapter 5, I will show how CENTRIFUSER's navigation bar is explicitly built to provide context information to aid the query reformulation process.

2.1.2 Strategy 2. Form a search plan

In the physical library during the reference interview, the librarian formulates a search procedure based on the criteria the user provides. This search procedure may touch upon resources physically located in different places or which differ in methods of access, and thus is an important part of the search process because it impacts the time efficiency of the search: choosing a bad search strategy in the physical library can result in wasted time consulting and locating unnecessary resources. The parallel problem in the online context is metasearching. Metasearch systems need to evaluate multiple resources and their access costs – known as the database selection problem (Gravano and García-Molina, 1995; Callan, Lu, and Croft, 1995; French and Powell, 2000). In the online context, this selection process is hampered by different access methods and limited metadata, which have been the focus of mediation schemes (Gravano et al., 1997).

However, in the majority of online IR systems, only a single (possibly huge) database is consulted, thus alleviating most of the need to use this strategy. If the search framework allows modification or prioritization of resources in the search process, then it is logical to provide this interface to user when needed. As stated previously, I have separated the multifaceted concept of IR into the search framework and user interface. Forming a search plan is a strategy that is not applicable to the user interface, but to the search framework, and thus is beyond the scope of this thesis.

2.1.3 Strategy 3. Evaluate documents for initial relevance

The ranked list user interface explicitly states that the first document is the most relevant to the query. Ordering the remaining documents in descending relevance implies the basic strategy of trying the first document and proceeding to the following documents next. This assists users if the query is fully specified. The document titles serve as a first pass summary of document relevance, with more detail provided by the document extracts. Ordering the list of documents by computed relevance and having two levels of text (title and extract) assist the user in making a content based relevance judgment.

However, in order to optimize the process of relevance assessment, the interface must be informed what criteria are used to assess relevance and to show these types of information accordingly. This is variable – it will differ per query and per user – but one can note some trends used to make relevance assessment. A document's content is often used to make relevance assessment, and as such, information on the individual topics and aspects of the query should be highlight and shown. The returned documents need to be categorized to show which aspects of the query they cover, and whether they exhaustively cover all aspects of the query. Highlighting the unique aspects or subtopics covered in each document is essential for a searcher to quickly locate and zoom in on the appropriate document(s).

Documents are sometimes very similar or equivalent to others (i.e., duplicates or format permutations between documents); thus, a system can group or merge identical or similar documents together to reduce the time factor in assessing document relevance. The idea here is that if one document in the group is relevant, the rest are relevant. Similarly, if a group of similar documents is considered relevant, the system should choose an exemplar document or two to represent the entire group to reduce the cognitive load on the user in choosing between alternatives. In Chapter 5, I will show how CENTRIFUSER determines unique and

salient aspects of each document and discuss how document clustering is done to reduce the cognitive load on the user.

2.1.4 Strategy 4. Form a user model

The standard ranked list interface does not form a user model or adapt itself to the history of the user's interaction with the system. In many ways, this is a simple and appropriate strategy, to provide a consistent and stable behavior for a given query, irrespective of the user. I will return to address consistency of interface in Characteristic 5.

In an optimal user interface, a user model that may be accumulated can assist the information seeker in different ways. There is much related work and I will only summarize it here. In general, there is a division between user models for individuals and collaborative (global) ones. In individual user models, the system gathers data about the individual user as she interacts with the system. Information stored can include the queries executed by the user and follow-up actions. This store of information can be used for different purposes, including word sense disambiguation in queries with multiple senses and suggesting related queries. In a collaborative user model, the individual user model is further augmented with the additional history provided by other users of the system. These users leave their "information trails" (Pikrakis et al., 1998; Konstan et al., 1997) as they traverse the system in their search process. These trails can help in many similar ways: suggesting revised or related queries, adjusting the relevance ranking of documents. In this model, the individual benefits from not only from her interactions with the system, but also from her peers (hence, "collaborative" interfaces).

User models of information seeking can be broken into individual facets. Belkin and his colleagues (1994) illustrate four dimensions to information seeking that they derived by observing patrons in their activities: method of interaction,

goal of interaction, mode of retrieval, and resource considered. Their dimension of *method of interaction* corresponds to the main and basic distinction to user modeling community: specified by the two modes of searching and browsing. Is the user searching for a specific fact, detail or an answer to a properly qualified question? Or is the user interested in the topic but at a general level, and would be happy browsing an overview, perhaps with a little peripheral information? In an optimal IR interface, both modes should be actively supported. The standard ranked list interface explicitly supports search with its document listing with titles and document extracts, but it does not address the needs of browsers in any form. An alternative to the standard interface which embeds hyperlinks to hand-crafted or edited category pages, or which suggest related topics, partially address the needs of a browser, but leave information for the browser at a one-click distance. In CENTRIFUSER, both searching and browsing modes of information seeking are explicitly supported through the two different types of summaries provided.

2.1.5 Adapting strategies to the online IR context

In summary, with respect to the informative reference interview strategies, an optimal user interface would directly address three of the four strategies that have been discussed. Strategy 2 (Forming a search plan) is not applicable, except in the distributed metasearching context. The standard ranked list performs two of three of these functions, but performs them rather haphazardly. It performs the functions adequately when the query is fully specified, but when the user query is underspecified, recovery or guidance could be handled better. In contrast, CENTRIFUSER handles all three of these strategies explicitly.

In adapting these techniques to a new generation of interfaces such as CENTRIFUSER– it is important to note that these reference interview strategies cannot be brought wholesale into the online context. They must be adapted to

operate within the confines of online IR. Reference interviews take place in the physical world, where speech and other nonverbal cues aid the librarian in constructing a user model and in understanding and presenting feedback to the user. In the online context, users are limited to using keyboard and mouse for input, which exacerbates the communication problem. Although average query length is growing, a number of studies still show that average query length online is under three words (2.2 words (Cutting, 1997), 2.8 words (Karlgren, 1999)), which is in contrast to the much greater average number of words initially spoken in the reference interview process. Since input is so difficult to provide, users often underspecify their information need. This makes the need for assessing relevance and clarifying queries even more important in the online context.

In an online search, users have also come to expect near-instantaneous feedback during the entire search process. Thus, search requests are often iterative because there is little time cost; initial searches often probe for terms to use in follow-up searches. This is in contrast to the distinct, segmented steps of the reference interview. It should be noted that librarians have started offering users of the digital library comprehensive assistance to online search and feedback during the search process by establishing virtual reference desks and "Ask-A" services. These services allow a user to ask a question and receive an answer in email that is compiled by a reference librarian. Although the returned results are often helpful and studies have shown (Wagner, 2002; Wang, 2002) that the slow average response time of a day or more is not satisfactory to most online users. The implication of this is clear: that at all times during the course of a search in the digital library, the user interface must support all of subtasks simultaneously and efficiently – from information need clarification and feedback to the evaluation of results. CENTRIFUSER performs these strategies online and with an efficiency that allows users to receive immediate feedback during the course of their search session.

2.2 Cataloging resources

When reference librarians are not present to actively assist users in fulfilling their information needs, users can go to passive resources for assistance. Both librarians and catalogers compile complementary types of resources to assist users in finding information on their own. These compiled materials are essentially written forms of the information retrieval process. They differ from their informational reference interview counterpart in that they are static and cannot provide interactive assistance to the user; their texts have been written even before the user first encounters them. I investigate how two different types of passive resources, cataloging records and subject guides, are structured to assist information seeking behaviors.

The term *bibliography* was originally used to describe work that dealt with collection and inventory of books. Bibliography thus described books in such a way to differentiate books from each other; to assign a unique identifier to certain editions or folios of printed matter. The ISO (1958) has modernized this idea by stating that bibliographic reference consists of "a set of data sufficiently precise and detailed to enable a publication or part of a publication to be identified." In this sense, the cataloger presents enough details about a resource that could be used to identify it or enumerate it as part of a large collection; saving the editorial remarks about subject content for specialist librarian who would be more qualified to make such a judgment (Stokes, 1969).

Two directions have emerged from the study of bibliography, which entail different uses of the original bibliographic material:

1. Following the function of subject specialist comes the notion of finding aids: these are *subject guides* and *annotated bibliography entries*, which evaluate resources to assist users in determining each resource's relevancy. To perform this, the subject guides and annotated entries have to include a range of information that will enable different users to make relevance judgments

suitable for their specific query and scenario. Large-scale efforts to incorporate electronic versions of these finding aids have been undertaken to provide the online patron with similar resources (Higgins, 1997).

2. Following the function of inventory and book identification comes the notion of *cataloging*, which seeks to disambiguate resources from another by uniquely identifying each particular resource – (Stokes, 1969) defines its purpose as assisting a reader in locating a resource. With such a purpose, the contents of individual entries becomes less significant than its relationship to the whole catalog.

These two directions often are merged together as the general field of bibliography. Based on an informal corpus study and a review of library science literature, I enumerate five characteristics of bibliographies that information professionals use which would be beneficial to model in the online information retrieval setting.

Specifically, the cataloger should:

1. Write cataloging descriptions to assist relevance judgments;
2. Clearly differentiate the bibliographic items;
3. Organize the bibliographic listing to assist a specific method of search;
4. Support alternative search methods or orthogonal classification axes in the catalog;
5. Use uniform descriptions in both ordering and content.

I now examine these characteristics individually and assess the performance of the standard ranked list on each criterion, and again point forward to how CENTRIFUSER will address these characteristics.

2.2.1 Characteristic 1. Write descriptions to assist relevance judgments

The *Guide to Reference Books* (Balay, 1996), perhaps the reference librarians' most valuable tome, states that its annotations "enable users of the *Guide* to understand a source's scope and visualize its arrangement and features so that it can be used more easily at the shelf. In writing annotations, contributors were asked to consider a [book]'s purpose, audience, scope, coverage, arrangement and special features, and were invited to comment on its useful in reference." This description points out that meta-information such as purpose, audience, and coverage are important features in providing a description for the user (even when the user is a librarian herself), in addition to direct information on source content.

The descriptions must vary according to the field of knowledge of the bibliography. In natural sciences, the importance of periodicals (journals), interdependence between sciences, dependency on most recent materials, and explosion of the amount of papers published all affect the bibliographical system (Henkle, 1950). In social sciences, the differentiation between statistics and primary sources of information from the conclusions or the discussions drawn is an additional important distinction (Taeuber, 1950). Layne's (1998) study found out what attributes in art are generally used for relevance assessment, by studying abstracts of art history papers. She discovered that fields such as *Artist* or *Creator* (including modifiers like "15th cent", *Kind of Work* (or genre), *Date*, *Subject*, and *Place* were most often mentioned. Other fields such as *Style*, *Technique* or *Material* and *Related Literary Work* had lower frequencies. In any case, it is easy to see that specific areas of expertise or domains have different information needs to judge relevancy.

Clapp (1950) goes further to define what users expect of bibliography in terms of subject guides and annotations: "In the simplest and most general terms, we expect that bibliography should lead the inquirer, through channels as well-

defined as are the entrances to harbors, to the particular record or records of communication which contain the information or other matter which he seeks. Specifically, one desires – even if one has been led to expect – that bibliography should have assembled all the material relevant to the subject of the search; that it should provide comprehensive coverage to the extent that comprehensive coverage is needed, yet permit selectiveness where that is required....Bibliography should, when these services are needed, evaluate, characterize or criticize the sources of information which it lists in order to narrow down the search or to produce a synthesis or provide a conspectus. Still another expectation...is that bibliography should in some manner signalize new contributions to learning so as to call them to attention of workers in other fields where these contributions might be employed, adapted or developed." (pp. 17-18). From the above description, one sees that these guides and entries also include subjective evaluation on the part of a subject expert, the field specialist librarian.

As such, it is clear that relevance is not always assessed purely by textual content. In cases of rich media such as HTML and in the context of the web, users may base or judge relevance by auxiliary criteria. These criteria can include a wide array of document metadata features: formatting (favoring plain text over rich markup for printing), included media types (includes pictures to go with the news story), size (favoring shorter documents to reduce downloading times), and various types of metadata (e.g., Who was the author? Was the story published in a reputable source?)

At this point it is clear that descriptions of bibliographic items should include other features of the document besides the content. In the standard ranked list, content itself is handled by the document extracts, either in context (i.e., QWIC concordances) or simply by the first n lines or sentences. What the standard system does not handle is providing descriptions that give other forms of information

well. These other pieces of information often can be thought of as **metadata** – information about the document. It is often the case that metadata is not easily derivable from the bibliographic item's text body itself. Authority, audience, purpose, format, readability, length and author are among the types of metadata that can be used for relevance judgment. The standard ranked list does give several crucial pieces of metadata: including the date indexed, the title (when available) and the location of the item. These pieces of information have been shown to improve retrieval effectiveness and satisfaction.

However, in many digital libraries, metadata information is meticulously recorded and encoded in electronic catalogs. These pieces of metadata are often given on the detailed record of a particular bibliographic item, but not in the ranked list summary of the documents matching a search. The ranked list interface does not display this information even if it is available, and needs to be modified to give this information to the user in a compact method. In an optimal IR interface, available metadata information should be displayed to help users whose relevance judgments require these criteria besides pure content. A central focus of CENTRIFUSER is to incorporate such metadata into the summary process, to be discussed in detail in Chapter 6. In cases outside of the digital library where such metadata may not be available, certain types of metadata can be inferred by automatic processes, further discussed in Chapter 3.

2.2.2 Characteristic 2. Clearly differentiate bibliographic items shown

The bibliographic description also highlights the editorial aspect of descriptive annotation, which focuses on the qualities of the whole resource collection, rather than individual entries; that is, entries in a subject guide or annotated bibliography listing should contain resources that cover different areas of expertise or are

appropriate for different purposes.

Thus, differences between resources should be made clear. Many subject guides list several different resources for an area. Multiple resources may be listed for completeness. This follows from the bibliographic function of compiling inventories. Often, different resources are listed because their are authoritative for separate subtopics (e.g., "*A* contains more information about treatment options, but consider using *B* if you are trying to find information about procedures for diagnosis") or better suited for certain purposes (e.g., "*A* is better for its elegant pictures, but *B* contains a much better narrative").

There is ample support for this strategy in the literature. For example, Lubetzky (1953) states this as his second principle of cataloging: "requir[es] the cataloguer to relate the given work to the other works of the author and the other editions of the work...so that he could select the edition most useful or suitable for his purposes, or another edition if one wanted is not in the library."

Redundant entries that cover the same purpose inhibit a searcher from finding a suitable resource as the same judgment call must be made multiple times. (Kraeling, 1950), page 112, states this very clearly: "It is rather that of the dozen or so titles that do come under his scrutiny: a large number may be quite irrelevant, being either too general or specific to do him any real service." It is thus important as an editor to group redundant resources and omit irrelevant ones.

The standard ranked list interface has evolved to detect and remove duplicates and to cluster similar pages by several different methods (Monge and Elkan, 1997; Yan and García-Molina, 1995). In web searches, an additional method is to group relevant pages from the same base web site together. The analogous grouping in the context of the digital library would be to group items that come from the same directory/folder or subject or collection.

Aside from duplicate detection or clustering grouping by similarity, the stan-

dard interface explicitly offers the ranking of documents as the sole distinction between search results. The document titles and summaries are again called to serve as implicit cues for the differences among the bibliographic items. In the optimal IR interface, distinctions between the returned results need to be presented clearly and explicitly. A default organization of the results would differentiate the returned items by the query's subtopics and list the documents in order of comprehensiveness. If many documents pertain to a particular subtopic of the query, the same problem occurs, to which re-clustering the documents by other criteria may be meaningful. The point is that although the standard interface of listing documents by relevance works well when there are few documents that are clearly better than the rest, it can be difficult for the user to select among several highly ranked alternatives. In contrast, CENTRIFUSER seeks to differentiate returned items from a search as much as possible, to assist the user in choosing an appropriate document to retrieve or alternative course of action. This is done with document clustering by subtopic distribution, further discussed in Chapters 5.

2.2.3 Characteristic 3. Organize the listing to assist search

Different subject guides that address different areas are often organized differently to best serve their consumer population. An interesting example of this is the *Art History Resources on the Web* subject guide (Witcombe, 2002), which organizes art history resources at its top level in chronological order for Western art, but geographically for most other regions. The organization shows that even within a particular field there can be orthogonal ways of structuring information that are equally valid.

In support of this, Stokes' *Function of Bibliography* (1969), page 127, tells the cataloger of an annotated bibliography to arrange the annotated bibliography by subject area, since searchers are primarily interested in resources on a certain

subject. However, he cautions that the classification and assignment of resources to particular subjects should only be determined after a full review of all resources to be included in the bibliography. Stokes elaborates on this matter by mentioning that the main organization of such resources has to be matched with the inventory to be cataloged (pg. 106): atlases have specifics in their color plates, and botanical books have illustrated images of plants as well as their scientific descriptions. The axis for primary organization differs among subject areas. Travel guides are often organized by geographical area; the same organization would not work for art, where period and artist axes make more sense.

The standard interface fulfills this characteristic, organizing the returned items by computed document relevance. However, as catalogers have observed, results pertaining to different types of queries are better served with domain-specific sorting criteria. As a result, the standard interface can still be improved to better fulfill this characteristic. Queries (or the items that they retrieve) can be classified and assigned to a subject category and the associated primary sorting criteria for documents given. CENTRIFUSER sorts the items by their typicality (closeness to the average returned item) which presents a more informed sort order than computed relevance.

2.2.4 Characteristic 4. Support alternative search methods

Many bibliographic resources (especially subject guides) also support alternative search methods or orthogonal classification axes. Large subject guides – such as the aforementioned *Guide* – in addition to being logically organized, as per Characteristic 3, often also have an alphabetized index to support keyword search.

Some resources provide multiple indices on their information content (e.g., Chinese dictionaries that allow indexed search by radical, pronunciation and number of strokes) that enable multiple points of access to the same information. These

access methods correspond to different, regular facets of each resource. As mentioned above, a bibliographic item may be sought by a user for a number of different purposes. These purposes may correspond to certain metadata (as in Characteristic 1) and thus a logical organization to support would be the ability to sort or group the documents by their metadata attributes (e.g., by length or by media type).

The standard ranked list sorts by computed topic relevance and does not allow resorting or grouping by any other characteristics. Relevance judgments that base on other criteria outside of topic (i.e., closeness to the query) are not supported. More advanced interfaces often have a query syntax that allows search to be restricted to documents with certain metadata value (e.g., belonging to a certain web site). However, this is a feature of the search framework itself, rather than an ability of the display interface. An optimal interface would allow the returned items of a search to be resorted or grouped by any pertinent metadata or topical aspect. CENTRIFUSER currently does not have the ability to sort by other criteria besides the default topical organization, but this would be simple to add to the system.

2.2.5 Characteristic 5. Use uniform descriptions

Individual descriptions must also be consistent across the bibliographic resource. Uniformity allows an information seeker to learn a single model of presentation and expect certain information within an entry to be present in a specific order and position. This allows the information seeker to quickly locate the piece of information desired. (Stokes, 1981), pg. 346, mentions that "within a single catalog or bibliography there must obviously be uniformity; this is essential". (Harmon, 1989) echoes this point by saying that "Consistency of format is one hallmark of good bibliographic practice". In fact, when the formatting convention is rigorously followed, a change in the format or ordering can be employed to call attention and highlight a particular outstanding characteristic of the bibliographic item.

The standard ranked list follows this characteristic well. Each piece of information associated with the bibliographic item is located in a fixed position, when available. Each item's relevance score, title, document extract, location, size and date indexed appears in a fixed order. Appropriate alternative text is offered for the values of missing fields (e.g., in web searches, the location is often substituted for documents without a title). In CENTRIFUSER, the interface also presents results in a systematic fashion. The navigation bar, document extract, and indicative summaries are given in fixed order. Information within the summaries also conform to domain-specific orderings are are expressed using natural language with variation recovered from a corpus study.

2.3 Discussion

Table 2.1 recaps the nine strategies and characteristics from field of information and library science and summarizes both the standard ranked list and CENTRIFUSER perform on these criteria. An optimal information retrieval interface would satisfy all of these criteria. However, pursuing some of these criteria to their logical optimum can impinge on the performance of other criteria. An example of this is enabling customization of the display by means of user profiling or modeling. A search interface might allow a user to display only fields that the user indicates are useful. Customization implies optimizing the display for a specific user, which means that the uniformity characteristic would be potentially be violated. This can cause problems when two users of the system who have very different customization settings meet, since they will have no common reference point. This situation, called *overcustomization*, has negative effects both for usability (Konomi et al., 1997) as well as software efficiency (Dieckmann and Hölzle, 1997). An appropriate balance between these features needs to be negotiated in the interface design.

Technique	Standard interface explicitly performs this technique?	Centrifuser handles this technique?
Informational Reference Interview		
S1. Clarify user's information need	No, lacks important explicit cues	Yes, exposes subtopics to help clarify query
S2. Form a search plan	*Not applicable*	*Not applicable*
S3. Evaluate documents for initial relevance	No, lacks ability to group related documents and highlight unique features	Yes, clusters related documents together
S4. Form a user model	No	Partially; supports both browsing and searching
Subject Guides and Annotated Bibliographies		
C1. Write descriptions to assist relevance judgment using metadata	Partially, but does not show many types of metadata	Yes, reports metadata in addition to content
C2. Differentiate items shown	No, does not differentiate documents by content or by metadata features	Yes, reveals unique aspects of each document cluster
C3. Organize listing to assist search	Yes, by general notion of relevance	Yes, by order of typicality
C4. Support alternative search methods	No	No, but easy to add support for
C5. Use uniform descriptions	Yes	Yes, and uses natural language variation to alleviate tedium

Table 2.1: A comparison of the standard ranked list and CENTRIFUSER's alternative interface in modeling techniques used by information professionals.

2.3.1 Related work in visualizing search results

From Table 2.1, it is easy to identify two primary areas that would improve the interface with respect to these criteria: exposing subtopic structure and metadata as well as grouping related documents together. These two issues below have been the subject of a number of studies on graphical user interfaces. I briefly discuss these two issues in the context of selected relevant literature.

1. **Expose subtopic structure and metadata attributes to the user** - The TileBars system (Hearst, 1995) exposes the document structure with

respect to a user's query by showing pictorially where search terms occur in the document, and their frequency of occurrence. This allows the user to visualize which terms contribute to the calculated relevance of the document and acts as an indicator for document structure. The TileBars system treats documents as a string of words, whereas CENTRIFUSER improves upon this and handles hierarchical discourse structure commonly found in expository texts. TileBars has the advantage in that the graphical representation of the document explicitly shows the length of the documents and facilitates navigation to specific points in the documents. In addition to document length, CENTRIFUSER supports the display of other types of metadata in a standard textual framework, such as writing style, readability, and embedded content types.

Several approaches map documents into a two-dimension space for topic visualization, including Topic Islands (Miller et al., 1998), which uses wavelet analysis, as well as DEPICT (Rushall and Ilgen, 1996), which uses self-organizing maps. However, the axes of these spaces do not correspond directly to locations or linear position in the documents and can lead to confusion in their interpretations. The Interactive Document Map is a similar method that is based on physical model of springs and particles (Zizi and Beaudouin-Lafon, 1995).

In CENTRIFUSER, topic structure is encoded as a hierarchical tree. Visualizing such tree structures has been a focus of research user interface design. AlgoNet2 (Shippey et al., 1996), renders such a structure as a clickable graph-based tree, and Cat-a-cone (Hearst and Karadi, 1997) renders it similarly in a 3D cone form.

Closest in spirit to CENTRIFUSER's textual descriptions of the hierarchical nature of topics are the SuperBook user interface and its variants (Egan et

al., 1989). SuperBook's expandable table of contents panel allows users to see how their query relates to specific topics and shows their context within the hierarchical topic structure of the document. CENTRIFUSER improves on this interface by giving natural language summaries that take up less room and integrates topical segmentation techniques that can produce the necessary hierarchical "table of contents" in the form of topic trees (discussed in the next chapter).

2. **Cluster and group results to highlight salient features** - The Scatter-Gather paradigm (Pirolli et al., 1996) clusters documents (gathering) and allows the user to select interesting clusters to be reclustered (scattering) in an iterative manner to refine the query. CENTRIFUSER differs from this approach by having a hierarchy of topics computed per document type (discussed further in Chapter 4).

   The VSComp system (Liu, Zhao, and Yi, 2002) compares two web sites against each other, by comparing their individual web pages. Sites' web pages are hierarchically grouped and similarities and differences are indicated, to aid a web site administrator to visualize parts of their site that are superior or deficient in comparison to a competitor's. CENTRIFUSER generalizes this type of comparison to intra-document topic structure and additional handles multiple documents (not just two).

2.3.2 An argument in favor of text user interfaces

In the next four chapters, I will show how CENTRIFUSER specifically addresses the two areas by taking the internal structure of the documents into account and using clustering of these subtopic structures to differentiate documents and highlight unique aspects of the documents. The system interface uses text rather than

graphical and iconic displays to visualize the information in contrast to the systems mentioned in the last section. Text-centric user interfaces are advantageous in that they can be readily understood by lay users. Additionally, text can be customized to serve different user models: to offer overview summaries for browsers and to highlight differences between documents for searchers. Recent work (Morse, Lewis, and Olsen, 2002) compares textual, iconic, graphical and "spring"-type displays in effectiveness, in which icons and text were found most effective in completing a task based evaluation[1]. Although Morse and colleagues found that their text interface was disliked, I believe this may be due to the format of the text presented, not the medium of text itself.

2.4 Aspects of summarization

A central tenet of this thesis is that automatic text summarization is well suited to assist the display of search results. It is well understood that people use documents for different purposes. It is likewise the case that there should be different types of summaries to support these purposes. As such, it is necessary to examine what type of summarization would best suited for this task. I examine three specific dimensions of automatic text summarization (ATS) that are of particular importance for the purpose of summarizing documents for online search and retrieval:

1. **Favoring multidocument over single document summarization**: Most ATS systems function in the single document context, where a single document is condensed to a shorter form. For a good overview of single document summarization technology, see (Paice, 1990). In the context of information retrieval, one has multiple documents that are returned by a single search re-

[1] "Spring" type displays are two- or three-dimensional displays of documents in which the document's location is based on pairwise similarity with others. A document's strength with respect to other documents is calculated using a quadratic force spring equation (Hooke's law).

quest. To generate a single output that summarizes the salient points across these multiple documents is more difficult. Since the documents are related by a common query, they likely contain similar content; thus a system cannot simply concatenate many single document summaries together, because repetition of salient points would result. If ATS is to be a successful methodology for information retrieval, a system that can handle repetition in multiple documents is a prerequisite. CENTRIFUSER's components examine similarity and differences among each document's structure, so it is designed specifically to handle multiple documents.

2. **Favoring query-based over generic summarization**: ATS systems often produce generic summaries that highlight the most salient points of a given text. However, in the online search and retrieval context, an ATS system has access to the query given by the user and should adapt its output to suit the user's declared information need. There are many instances when a rational IR framework finds query keywords in only a subsection of a larger document. Showing this relationship between the query terms and the document has been proven to be an important factor (Amitay (2001) shows the difference between AltaVista – which does generic summarization by reporting the n-top sentences of a document – and Google – which reports sentences in QWIC format). Query-based summarization has also been a focus of research in graphical user interface design (Hearst, 1995); query-based text summarization is the logical parallel in ATS. While it would be acceptable to store generic document summaries and present them in an IR system, a more favorable approach is to produce per-query customized summaries. This latter approach is the method used by CENTRIFUSER.

3. **Informative versus indicative summarization**: Informative summaries provide information on the salient aspects of a document, seeking to cover as

many topics as possible. These summaries omit detail or supporting information and just cover the most important points of the document. Summaries of this type often are used in place of the document as an overview, and are suitable for fulfilling a user's information need if they are browsing for information or have a general interest in the subject of the document. An example of this is an article abstract that mentions the problem, methodology and results of the article.

Indicative summaries, on the other hand, are meant to hint at the contents of the document. In the IR context, indicative summaries play an interesting role because they help the user in judging the relevance of the document, and in determining whether to consider full-text retrieval. They assist a user who is searching for information and has a specific information need. One can think of the indicative summary as a vehicle for routing a user to a specific document in the query result set.

The type of summary that should be produced by an ATS system for IR depends on the scope of the user's information need. Indicative summaries are more important to searchers, while informative summaries are more important to browsers. Both modes of information access will occur in the digital library, so unless the user explicitly gives a preference, both modes should be utilized. An ATS system for IR can either match together two separate systems that construct indicative and informative summaries, or try to implement a single system that produces both. CENTRIFUSER produces both indicative summaries for searchers and informative summaries for browsers using the same underlying framework.

In the upcoming chapters, I will detail the alternative search interface, CENTRIFUSER, which builds on and incorporates automatic, multidocument, query-based indicative and informative text summarization.

Chapter 3

Creating document topic trees

Beginning with this chapter, I will show how the algorithms I developed and implemented in CENTRIFUSER extend the capabilities of the IR user interface to meet the criteria set forth in Chapter 2. CENTRIFUSER is a system that is designed to give both informative and indicative summaries, given a set of similarly structured source documents and a user query. The input requirements are easily satisfied by an IR framework's ranked list of documents in response to a user query. In order to support finding of similarities and differences across documents, CENTRIFUSER needs to be able to compare parts of the source documents with each other, as per Characteristic 2 (clearly differentiate items). For example, being able to direct a user to a document that contains the most comprehensive "treatment" topic for a question on angina implies 1) that treatment topics are identified throughout the multiple source documents, and 2) that these identified topics can be compared against each other to identify which one is the most thorough.

For both tasks, the comparison of individual words across documents does not yield a good solution to the problem, nor does comparing whole documents versus each other. The granularity is respectively too fine (at the word level) or too coarse (at the document level). In this chapter, I introduce a hierarchical framework

based on variously scoped granularity levels that lie between the word and document levels. This topic granularity based framework enables comprehensive similarity and difference computation and addresses problems similar to the scenario above.

In this chapter, I first describe the data structure that I compute to hold a document's topical information in a structured format – the document topic tree – and provided an overview of three different methods that can be used to construct it. I then describe two innovations that I have designed, implemented and evaluated to improve the accuracy of the linear segmentation approach to this problem.

3.1 Document topic trees

A topic is important to us as a granularity level; it is smaller than a document and larger than individual sentences or paragraphs. A suitable representation for documents for this task is one where documents contain a number of topics, structured into a hierarchy. It is a good match for expository documents since many of them are also hierarchically organized. Each document is then represented by a *document topic tree*, which breaks the document into a hierarchical tree of subtopics. This *document topic tree* can be defined as a data structure that allows navigation of a document by its sections. These trees contain subsections of sections and may embed information about contained items, such as figures, tables, hyperlinks or other different types of metadata as per Characteristic 1 (e.g., style: in prose or listed in bulleted form). They may be used directly by an end user for document access, or indirectly through other applications. Figure 3.1 shows the translation of a document from the National Heart, Lung and Blood Institute (the document is reproduced in Appendix B) into a document sample topic tree. The sample tree is a document about angina and has been enriched with formatting and style metadata for each subtopic.

A *logical structure tree* (Summers, 1995) is a related and often equivalent

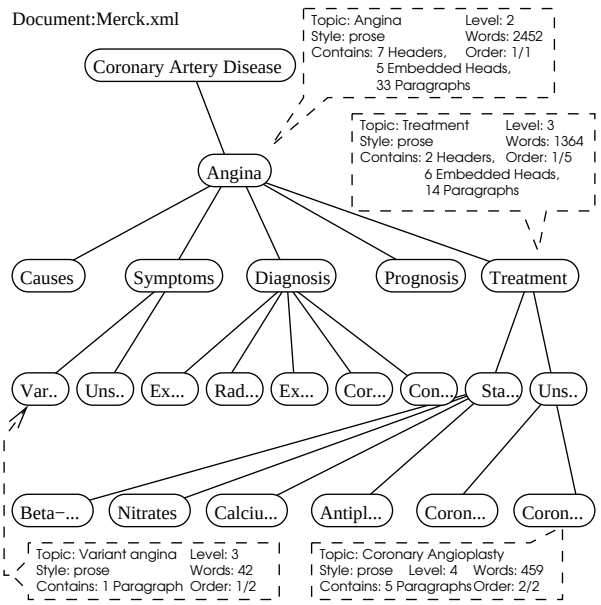


Figure 3.1: A sample document topic tree, encoding the *Merck Manual of Medicine*'s page on angina. The structure of the document is similar to the NHLBI document given in Appendix B.

term to the document topic tree. In her thesis, Summers (1998) differentiates logical structure (e.g., sections) from physical structure (e.g., pages). In her definition, a logical structure (pg. 3):

> "consists of a hierarchy of segments of the document, each of which corresponds to a visually distinguished semantic component of the document. Ancestry in the hierarchy corresponds to containment among document components."

Ancestry in the document topic tree also corresponds to containment, but in my definition, a topic tree does not require any visual separability. I argue that expository texts that have impoverished text formatting indeed do have logical structure, although it may not be visually recoverable or obvious. In these cases, logical structure can be recovered by an analysis of the content of the discourse.

Rhetorical Structure Theory (RST) (Mann and Thompson, 1988) also can be used to build a structural representation of the document. Marcu (1997a; 1997b) operationalized RST into an automated method utilizing discourse information in cue phrases. The RST tree structure of a document captures the discourse and suasive properties of the text, and does not capture the topical aspects of the document. A document's RST tree thus complements its topic tree, exposing different aspects of the same text. The former provides the rhetorical aspects without content; the latter, the topical aspect without discourse. The two trees often coincide and overlap, thus the two approaches can be combined. I return to how this is handled in my pre-processing in Section 3.3.3.

I consider the translation of a raw document to a structured tree a process to which three general methodologies can be applied: 1) translation from rich markup, 2) spatial layout recognition, and 3) topic segmentation.

3.1.1 Translation from rich markup

Many textual documents today are enriched beyond plain text and take advantage of word processing software's abilities to enhance plain text with layout, font and alignment features. Other features, including non-textual features (such as the embedding of tables, charts, images and animations) are also becoming commonplace as the ability to create, store and embed these other media objects becomes easier.

Inferring document structure from these richly formatted documents is often a straightforward process of examining hierarchically marked headers. In the hypertext markup language (HTML), tags <H1> through <H6> define a set of headers that are ordered in descending importance. A simple yet effective way to construct a document topic tree then consists of using the headers to create the tree, with <H1> tags being the highest, superordinate subtopics, and recursing downward to the <H6> tags which would be the lowest, innermost nested subtopics. This would

define a six-level deep tree at its most comprehensive configuration.

With the advent of cascading style sheets (CSS) in HTML and eXtensible Style Language (XSL) for XML, efforts have been made to separate logical form and ordering of a document from its physical appearance. As these technologies become more standardized and embedded within authoring tools, it is likely that these simple strategies will work more effectively and cover a larger proportion of the document topic tree conversion cases. In these cases, the logical structure tree (Summers, 1998) and topic tree are equivalent, as the visual distinction between semantic units can be arbitrarily specified with style sheets that maps different types of units to different physical formats. Building such document topic trees from richly marked data is well-defined and largely consist of software engineering and "crosswalking" between standards.

Hyperlinks in today's medium of the World Wide Web presents a more complex challenge that blurs the difference between single documents and sections. Many web sites are designed with the limitations of the average consumer's computer capacity in mind. These limitations on screen size and connection speed cause authors to structure their information differently. Indeed, good web design espouses these considerations (Ivory and Hearst, 2002), and forces many authors with logically longer pieces of information to section them up into smaller chunks, that are connected via "next page" links and/or other types of navigation hyperlinks.

Inferring document structures in the hyperlink context requires recognition of these features and parity between single document (containing all of the information on a single page) and website (containing all of the information split apart into separate pages connected by links) treatment. This is a topic that is being examined by content generation and user interfaces (Fields and Merriam, 1997) as well as by internal site link structure analysis.

3.1.2 Spatial layout recognition

It is generally acknowledged that richly formatted documents often contain many cues about the importance of certain sections of text. However, it is not always the case that one has access to the document in a convenient, digitalized format. One may only have access to a hardcopy version, which is often the case for documents that were published before the advent of document processing tools. This can also occur when a richly formatted text was originally accessible, but which was transported or exported to an agent only capable of receiving the material in a lower base form (e.g., "save as text" for insertion into an email). In these cases, documents are often scanned in and optical character recognition (OCR) employed to recover the raw text. These tools often present their output in the same page layout as in the original (e.g., raw text in two columns). In addition to the raw text, OCR methods allow the recovery of font sizes and families, as well as the spatial coordinates of the text blocks on the page.

To recover the document structure in these cases, spatial and font features of the text or scanned image are often employed. This is an principle problem in the field of document analysis and recognition and a plethora of work. Watanabe (1999) offers a survey of the techniques in this area, incorporating techniques that use OCR as well as ones that work directly from the image data such as (Niyogi, 1994). Within the domain of recognizing document structure from OCR'ed or raw text, (Hu, Kashi, and Wilfong, 1999) and (Summers, 1998) give techniques for finding contiguous blocks of text and using logical reading patterns to correlate text blocks as a continuous article, thus restoring the article as a complete logical unit. In Section 3.3, I discuss how such techniques, coupled with approaches in topic segmentation (discussed next) can be combined within a machine learning framework to yield favorable results.

3.1.3 Topic segmentation

When presented with raw text with unmarked or absent topical headers, neither of the above techniques – translation from rich markup and spatial layout recognition – can be applied. In this section, I consider the problem of defining topic boundaries in these raw text documents that do not have markup or formatting cues. Topical segmentation can be used in these cases to impose a topic hierarchy on the stream of text. I follow a two-step approach to first impose a partitioning of the text into discrete topical segments (linear segmentation), and then utilize related work to structure these units into a hierarchical tree to form the document topic tree (hierarchical agglomerative clustering).

Hearst (1994) observed that different topics are expressed with a different vocabulary. For example, an article on science may contain adjacent sections on astronomy and biology. Two adjacent topics shift in their vocabulary profile (Youmans, 1991) in the course of their discussion and thus the topic shift can be detected where the change in vocabulary items is most prominent. Hearst's TextTiling system calculates segment boundaries based on this principle, choosing a segment boundary when the bound exceeds a threshold. Figure 3.2 shows an example text from Hearst's original paper, which shows two large clusters of term occurrences (bottom center and top right), indicating at least three different topics (beginning, middle and end) that span the 95-sentence document.

3.2 Improving linear segmentation: SEGMENTER

The basic approach to linear segmentation, as put forth by TextTiling, uses shifts in the vocabulary item occurrences to detect topical transitions. As nouns frequently carry the topical content of a discourse, I focused on how the repetition of nouns could be further specialized to improve performance. Hearst's original algorithm

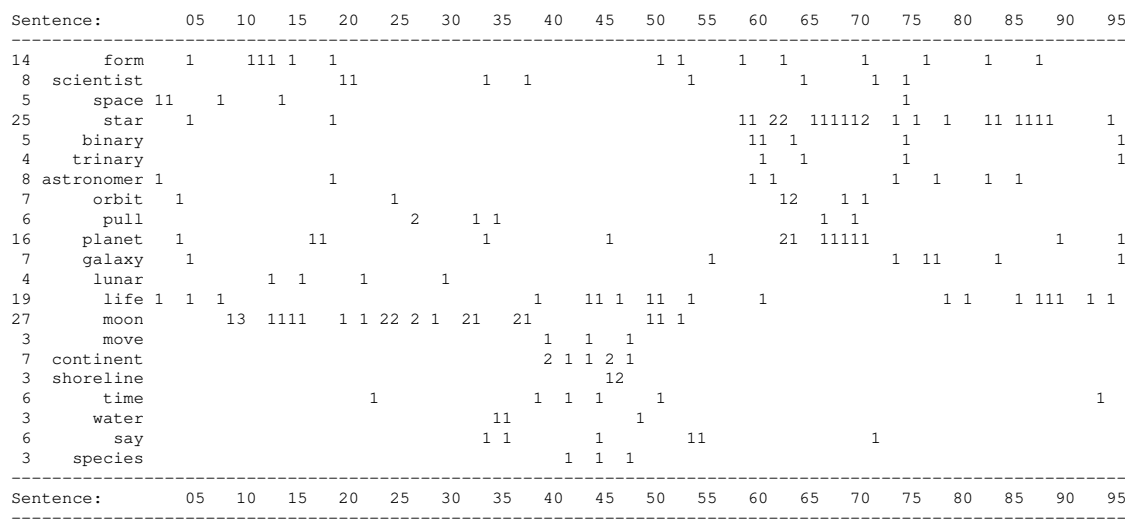


Figure 3.2: Distribution of selected terms from a text, as shown by TextTiling. Taken from Hearst (1994). Reproduced with permission.

does not discriminate among the word classes, so all words (aside from those on a stoplist) such as adjectives and verbs are also included as candidates for sources of lexical repetition. In my study, I subdivided noun phrases into three categories for study:

1. Proper noun phrases;

2. Common noun phrases;

3. Personal and possessive pronouns.

In this section, I present a topical segmentation program, called SEGMENTER that achieved a 10% increase in both accuracy and recall over the comparable previous work.

I evaluated the system on general-domain news articles. Generally, I found that longer articles, usually beyond a three-page limit, tended to have artifacts of logical structure, such as headers or bullets. These longer articles are probably

better segmented with techniques that capitalize on these spatial cues, an approach which I also investigated, discussed in Section 3.3.

In comparison, shorter news articles or expository documentation exhibit running prose more often, and topical segmentation of these articles requires methods that rely on lexical cues. I thus concentrated work on 20 shorter articles, roughly 800-1500 words in length: 15 from the Wall Street Journal in the Linguistic Data Consortium's 1988 collection, and 5 from the on-line The Economist from 1997. The segmentation methodology has three distinct phases (shown in 3.3), which are executed in a sequential fashion.

Figure 3.3: SEGMENTER architecture.

3.2.1 Extracting tokens

In order to find these three types of terms (proper noun phrases, common noun phrases and personal and possessive pronouns), the system first tags the text with part of speech (POS) information. I investigated two methods for assigning POS: a) run a specialized tagging program which takes a table lookup combined with contextual probabilities, or b) just use the table lookup. I use a standard table lookup method to assign POS tags which favors noun POS tags whenever possible, explained in detail in Appendix C. This is less accurate than state-of-the-art POS taggers but also much faster. In an internal evaluation comparison of assigned tags by lookup versus traditional POS tagging, precision of the overall program fell only slightly and the decline was not statistically significant. As segmentation boundaries rely on large amounts of lexical repetition rather than the correct tagging of a few occurrences, the method is robust to minor amounts of mistagging errors.

However, if a text processing system requires accurate POS tags after segmentation, then the cost of tagging is not an issue, and tagging should be used in place of lookup.

Once POS tags have been assigned, SEGMENTER retrieves occurrences of noun phrases by searching the document for this simple regular expression:

$$(\text{Adj} \mid \text{Noun})* \text{ Noun}$$

This expression captures simplex noun phrases (Wacholder, Ravin, and Choi, 1997), and is not meant to encompass more complex noun phrases such as "the amount of fat in the diet", seen in the text in Appendix B. I considered these possibilities and intentionally erred on the side of making the regular expression less powerful to capture as many noun phrases as possible, since the emphasis is on high noun-phrase recall. Thus, the example would be captured as three different phrases: "amount", "fat" and "diet".

After retrieving the terms, the system does some additional post-processing to combine related occurrences together; for possessive pronouns, the system merges each possessive with its appropriate personal pronoun ("his" or "him" with "he", etc.). Plural noun phrases are lemmatized and grouped with their singular forms, if both singular and plural forms occur. Noun phrases with common heads are conflated, if one is more general than the other. For example, if the noun phrases "blood cholesterol level" and "level" are found in a text, the system subsumes the occurrences of "blood cholesterol level" into the occurrences of "level", provided that other "level" headed phrases do not exist (e.g., "fitness level" or "stress level").

3.2.2 Weighting term occurrences

Once extracted, term occurrences are then linked together and scored, which results in a topical segmentation of the document.

Example sentences:

1. Usually the first line of defense involves changing one's living habits to avoid bringing on attacks of angina.
2. Controlling physical activity, adopting good eating habits, moderating alcohol consumption, and not smoking are some of the precautions that can help patients live more comfortably and with less angina.
3. For example, if angina comes on with strenuous exercise, exercise a little less strenuously, but do exercise.
4. If angina occurs after heavy meals, avoid large meals and rich foods that leave one feeling stuffed.
5. Controlling weight, reducing the amount of fat in the diet, and avoiding emotional upsets may also help.

`termer` output:

```
Frequency       Subsumption status      Term (location by sentence)
4               ok                      angina (1 2 3 4)
3               ok                      exercise (3 3 3)
2               head                    habits (2 1)
2               part                    meals (4 | 4)
1               ok                      help (5)
1               ok                      feeling (4)
1               ok                      amount (5)
1               ok                      large meals (4)
...
1               ok                      moderating alcohol consumption (2)
1               ok                      controlling weight (5)
```

Figure 3.4: Partial term list from the `termer` module of SEGMENTER from the example sentences shown above.

3.2.2.1 Linking distance

Given a single lemmatized term (noun phrase or a pronominal form) and the distribution of its occurrences, related occurrences are linked together. The system uses proximity as the metric for relatedness. If two occurrences of a term occur within a certain amount of sentences, the system links them together as a single unit. This process repeats until no larger units can be built. This idea is a simpler interpretation of the notion of lexical chains. Morris and Hirst (1991) first proposed this notion to chain semantically related words together, while I chose only repetition of the same lemmatized term. I also tried to semantically cluster terms by using (Miller et al., 1990) WordNet 1.5, traversing hypernyms and hyponyms of the lemma term for a single edge to add additional information determine relatedness.

However, results were mixed and the execution time was at least doubled, with the cost of lookup and subsequent conflation of links between terms.

However, for these three categories of terms, the linking distance differs depending on the type of term. Proper nouns exhibit the maximum distance and the pronominal forms exhibit the least. Proper noun terms generally refer to the same entity, almost exclusively of the intervening distance between occurrences. Common nouns often have a much shorter scope of reference, since different term occurrences can refer to different specific instances. Personal pronouns scope even more closely, since the referent to a personal pronoun could change over the course of a document. Thus, linking distance refers to the number of units (in SEGMENTER , the units are sentences) allowed to intervene between two occurrences of a term. Finally, the system drops any unlinked terms from further consideration. This means that the system implicitly filters noise from the results, discarding any pronouns or noun phrases that occur only once. This is supported in the literature by (Justeson and Katz, 1995), who concluded that only terms that occurred two or more times indicate topicality through their burstiness.

3.2.2.2 Assigning weights

After links are established, weighting is assigned. Individual links are first normalized to the paragraph level. Then for each term, SEGMENTER labels each paragraph with its positional relationship to the term's link(s). I describe these four categories for paragraph labeling:

- **No link** (`n`): any paragraph in which no links occur.
- **Front** (`f`): a paragraph in which a link begins.
- **Rear** (`r`): a paragraph in which a link no longer occurs. Equivalently, a succeeding paragraph after the paragraph in which a link ends.

- **During** (`d`): a paragraph in which a link occurs, but is neither a front nor a rear paragraph.

```
paras   1    2  3      4     5      6    7
sents   12345678901234567890123456789012345
angina :      1xx1                    1x21
type   :n    f  d      r     n      f    d
```

Figure 3.5: A term, "angina", and its occurrences and assigned term types (marked as `term`) per paragraph in a hypothetical document. Each column indicates a sentence, with the number of occurrences indicated by numbers, and linkages between occurrences indicated by '`x`'s.

Figure 3.5 shows the algorithm as developed thus far in the chapter, operating on the term "angina" in a hypothetical document (once in sentences 7, 10, 32 and 35; twice in sentence 34). The term appears a total of six times, as shown by the numbers in the central row. These occurrences have been grouped together into two term links, as joined by the "x"s: one occurring in paragraphs 2 and 3 (shown by "`1xx1`") and one occurring in paragraphs 6 and 7 (shown by "`1x21`"). The bottom "type" line labels each paragraph with one of the four paragraph relations. It is possible for a term to have multiple front and rear paragraphs, since a term might have more than a single link. For each category of paragraph and for each of the three types of terms, SEGMENTER assigns a different segmentation score, listed in Table 3.1, whose values were derived from training, which is discussed later in Section 3.2.2.4.

Term type	Paragraph type with respect to term				Link Length
	front	rear	during	non	
Proper NP	10	8	-3	*	8
Common NP	10	8	-3	*	4
Pronoun & Possessives	1	13	-1	*	0

Table 3.1: Overview of weighting and linking scheme used in SEGMENTER; starred scores to be established by zero sum procedure, explained later.

For noun phrases, the system assumes that the introduction of the term is a point at which a new topic may be opened; this is (Youmans, 1991)'s term *vocabulary introduction*. Similarly, when a term is no longer being used, as in *rear* paragraphs, the topic may be closed. This observation may not be as direct as vocabulary introduction, and thus presumably not as strong a marker of topic change as the former. Moreover, paragraphs in which the link persists throughout indicate that a topic continues; thus one can see a negative score assigned to *during* paragraphs. When one applies the same paragraph labeling to pronoun forms, the same rationale applies with some modifications. Since the majority of pronoun referents occur before the pronoun (i.e., anaphoric as opposed to cataphoric), the system does not weight the *front* boundary heavily, but instead place the emphasis on the *rear*.

3.2.2.3 Zero sum normalization

When this weighting process is iterated over each paragraph and term and then total up the scores assigned, SEGMENTER comes up with a numerical score for an indication of which paragraphs are more likely to be a topical boundary. The higher the numerical score, the higher the likelihood that the paragraph is a beginning of a new topical segment. However, segments are either present or not. Determining what the minimal score should be to consider a paragraph boundary also a segment boundary is a problem.

To solve this problem, SEGMENTER zero-sums the weights for each individual term. To do this, it first sums the total of all scores assigned to any *front*, *rear* and *during* paragraphs that the system have previously assigned a score to and then evenly distribute to the remaining no link paragraphs the negative of this sum. This ensures that the net sum of the weight assigned by the weighting of each term sums to zero, and thus the weighting of the entire article, also sums to zero. The

process of weighting, followed by zero-summing is shown by extending the "angina" example in Figure 3.6, as indicated by the `score` and `zero` lines.

```
paras   1    2  3      4     5      6    7
sents   12345678901234567890123456789012345
angina :      1xx1                    1x21
type   :n    f  d      r     n      f    d
score  :*    10 -3     8     *      10   -3

sum to balance in zero-sum weighting: +12
zero   :-6   10 -3     8     -6     10   -3
```

Figure 3.6: The term, "angina", from Figure 3.5 and its links and score assignment to paragraphs.

Paragraph scores are calculated per term, and the scores for each paragraph are summed to give a final score, which is either positive or negative. A positive score indicates a boundary, a beginning of a new topical segment, whereas a negative score indicates a continuation of a segment. This use of zero sum weighting makes the problem of finding a threshold trivial, since the data is normalized around the value zero.

3.2.2.4 Finding local maxima

Zero-sum weighting for long or medium length documents yields good results. For the documents that I have examined, namely documents of short length (800-1500 words), I have observed that multiple consecutive paragraphs, all with a positive summed score, usually only have a single true topic boundary. In these cases, SEGMENTER takes the maximal valued paragraph for each of these clusters of positive valued paragraphs as the sole segment boundary for these paragraphs. Again, this only makes sense for paragraphs of short length, where the distribution of words would smear the segmentation values across paragraphs. In longer length documents, this phenomenon will probably not occur, and thus this process can be skipped. After finding local maxima, the finalized topical segment boundaries are

established.

3.2.3 Algorithm training

To come up with the weights used in the segmentation algorithm in Table 3.1, I split the corpus of articles in four sets and performed 4-fold cross validation training, intentionally keeping the five Economist articles together in one set to check for domain specificity. The training phase consisted of running the algorithm with a range of different parameter settings to determine the optimal settings. I tried a total of 5 x 5 x 3 x 3 = 225 group settings for the four variables (front, rear, during weights and linking length settings) for each of the three (common nouns, proper nouns and pronoun forms) term types. The results of each run were compared against a standard of user segmentation judgments, further discussed in the next section. The results noted that a sizable group of settings (approximately 10%) produced close to optimal results. This group of settings was identical across all four cross validation training runs and thus indicates that the algorithm is fairly robust.

3.2.4 Evaluation

I implemented a web-based segmentation evaluation facility to gather segmentation judgments. Each of the twenty articles in the corpus was segmented by at least four human judges, and the majority opinion of segment boundaries was computed as the evaluation standard.

Human judges achieved on average only 62.4% agreement with the majority opinion, as seen in Table 3.2. (Passonneau and Litman, 1993) show that this surprisingly low agreement is often the result of evaluators being divided between those who regard segments as more localized and those who prefer to split only on large boundaries.

Approach	15 Wall Street Journal				5 Economist				**Total**			
	Precision		Recall		Precision		Recall		Precision		Recall	
	avg	S.D.	avg	S.D.	avg	S.D.	avg	S.D.	avg	S.D.	avg	S.D.
Monte Carlo	29.0%	9.2	33.3%	.02	32.8%	12.6	33.3%	.02	29.8%	9.9	33.3%	.02
Hypergeometric	30.6%	N/A	30.6%	N/A	32.9%	N/A	32.9%	N/A	32.0%	N/A	32.0%	N/A
TextTiling	28.2%	18.1	33.4%	25.9	18.3%	20.7	18.7%	18.5	25.8%	18.7	29.8%	27.8
Segmenter	47.0%	21.4	45.1%	24.4	28.6%	26.2	22.7%	25.2	42.6%	23.5	39.6%	25.9
Human Judges	67.0%	11.4	80.4%	8.9	55.8%	17.2	71.9%	4.6	62.4%	13.5	78.2%	87.6

Table 3.2: Evaluation results on precision and recall scales.

I then verified that the task was well defined by testing for a strong correlation between the markings of the human judges. I test for inter-judge reliability using (Cochran and Cox, 1992)'s Q-test, also discussed in (Passonneau and Litman, 1993). I found a very high correlation between judges that indicated that the task was indeed feasible to model; the results showed that there was less than a 0.15% chance on the average that the judges' segmentations agreed by chance.

Segmenter's results show a significant improvement over the initial algorithm of Hearst, both in precision and recall. I present two different baselines to compare the work against: the first being a Monte Carlo simulation that segments at paragraph breaks with a 33% probability. I executed this baseline 10,000 times on each article and averaged the scores. A second baseline is produced by applying the hypergeometric distribution, which calculates the probability of some number of successes by sampling without replacement. For example, this distribution gives the expected number of red balls drawn from a sample of n balls from an urn containing N total balls, where only r are red. If one allows the number of segments, r, to be given, one can apply this to segmentation to pick r segments from N paragraphs. By comparing the results in Table 3.2, one can see that the correct number of segments, r, is difficult to determine; TextTiling's performance falls below the hypergeometric baseline, but on the average, Segmenter outperforms it.

More recent work on text segmentation as exemplified by (Yaari, 1997; Beeferman, Berger, and Lafferty, 1997; Utiyama and Isahara, 2001) have since updated and improved upon the performance of the linear segmentation task that

SEGMENTER addresses.

In this section, I have shown how the basic approach to multi-paragraph text segmentation is improved by differentiating streams of information coming from proper and common noun phrases as well as pronominal forms. I have demonstrated a new algorithm that performs linear topical segmentation in an efficient manner that is based on linguistic principles (e.g., (Morris and Hirst, 1991; Hearst, 1993)). SEGMENTER achieves accuracy and recall levels in excess of ten percent over prior work.

3.2.5 Building the document topic tree from linear segments

The technique that I have described in the last section leads to a linear segmentation of a text into partitions. Building a hierarchical tree can be achieved by a hierarchical agglomerative clustering (HAC) approach (Yaari, 1999). In this algorithm, a partitioned text of n partitions is taken as input and the two most similar adjacent blocks are joined together in a superordinate unit. The process is repeated n-1 times until the text is joined as a whole. This process thus leads to a minimal tree of depth $log_2 n$. One weakness of HAC is that it results in only binary tree structures. Much expository discourse is marked by linear segments rather than a binary tree structure. To remedy this, a systematic flattening of the HAC tree can be performed by taking any n adjacent segments that immediately are joined in the consecutive HAC operations and present these as a flat n-child tree. Figure 3.7 illustrates this process.

In the context of related work in topic spotting and clustering (Liu, Zhao, and Yi, 2002) also use this agglomerative approach to build trees (also called dendrograms) because they can be truncated at different depths to yield clusters of varying granularity.

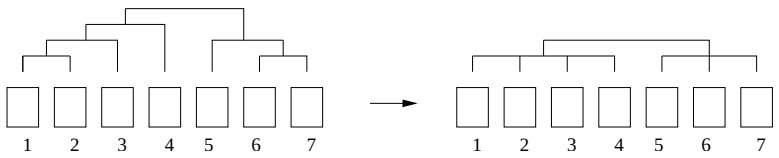


Figure 3.7: Flattening a binary tree derived from hierarchical agglomerative clustering.

Symbolic approaches can augment these statistical approaches. These include discourse parsing as previously mentioned in Section 3.1. This work captures the argumentative structure of the document but does not necessarily capture its topical structure, which complements topical structure.

As discussed throughout this chapter there are many methodologies that can be utilized to produce a segmentation of a document. For the remainder of this chapter, I will examine a machine learning framework that can utilize these disparate streams of information to improve segmentation accuracy.

3.3 Combining spatial and lexical features: CLASP

In many documents, both the lexical properties (discussed just previously) and the layout properties (discussed in Section 3.1.2) present two different streams of information that can be used to do topical segmentation. Having two different streams of information can either be viewed as a single, greater source of evidence, or as a source of redundancy, where one stream checks and validates the other. (Chen, Hu, and Sproat, 1999) view a similar problem of parsing email signature blocks using the former technique of combining both linguistic and spatial features in a weighted finite state transducer (WFST) framework. Co-training (Blum and Mitchell, 1998), a machine learning technique that uses two orthogonal streams to create two separate machine learners and which uses them to label each other's

training data, is also an exciting paradigm that can be utilized with respect to the document topic tree translation problem.

I believe that the combination of spatial layout recognition and lexical cohesion techniques allows for a wider range of texts to be dealt with. The two streams of information are independent of each other, and can be represented by orthogonal sets of features. My hypothesis is that in many texts these two streams would complement each other, reinforcing decisions on segmentation breaks. With certain data sources, one of these streams may not be present (i.e., no layout information in some emails), and the system could then rely on the other to provide information.

This section describes an alternate approach that I have taken to compute document structure using a framework that deals both with rich, semi-structured documents with layout features as well as impoverished, text stream-like documents. My system, the Combined Layout And Segmentation Preprocessor (CLASP), performs this task by computing visual layout and lexical cohesion features, and then combining them using supervised machine learning. In this framework, the output of the SEGMENTER algorithm and other linear segmentation algorithms, such as Hearst's TextTiling and Choi's C99 is folded into the machine learning framework as additional features to be used by the system to induce segmentation rules.

Thus, one of the design goals of CLASP is to provide a framework to combine and balance these data. Machine learning decides what types of information are salient. Machine learning allows the system to be customized automatically for a particular domain, and allows for training over a general collection to establish reasonable default rules.

Content-oriented and layout-oriented structures are relative ends of a single scale, discussed by Summers (1995). Since lexical cohesion works best on content-oriented structures (e.g., prose) and presumably not as well on layout-oriented structures (e.g., section headers or captions), CLASP provides lexical analysis only

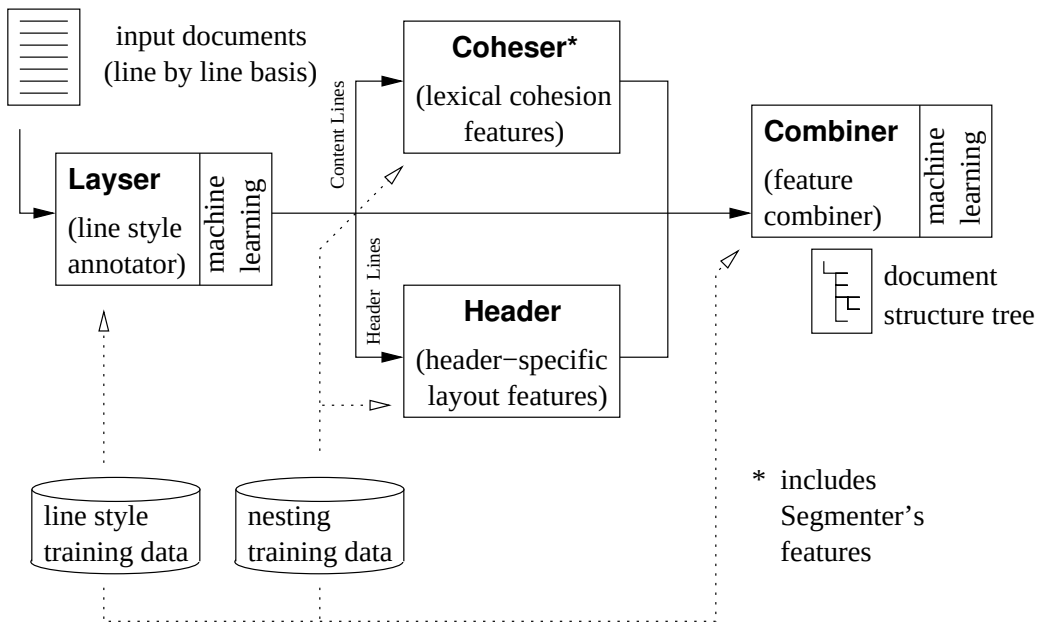


Figure 3.8: CLASP system architecture.

for the text in a document that is content oriented. In a nutshell, the layout preprocessing module (`Layser`) makes a first-pass decision on whether lines of the text are layout- or content-oriented. Content-oriented lines and layout-oriented lines are analyzed separately by the lexical cohesion module (`Coheser`) and by the header analysis module (`Header`), respectively. `Coheser`'s lexical cohesion features subsume the features used in SEGMENTER, and thus CLASP can be thought of as encompassing the work in SEGMENTER. The two streams of information are rejoined in the final machine learning module (`Combiner`), which weights the information streams appropriately and determines the document structure tree. This architecture is shown in Figure 3.8.

Another design goal in CLASP is wide applicability. For this reason, I use plain ASCII documents as input. Plain text is used as an original text form (e.g., in email) and as a lowest common denominator, and thus any type of textual data can be transformed into it. These transformations can be lossy (e.g., font sizes and weights are lost), and with richer original formats, CLASP's performance may

not be optimal. However, modules can be built to provide additional features that represent the enriched formatting or additional features (such as the segment output of SEGMENTER), as shown in Figure 3.9.

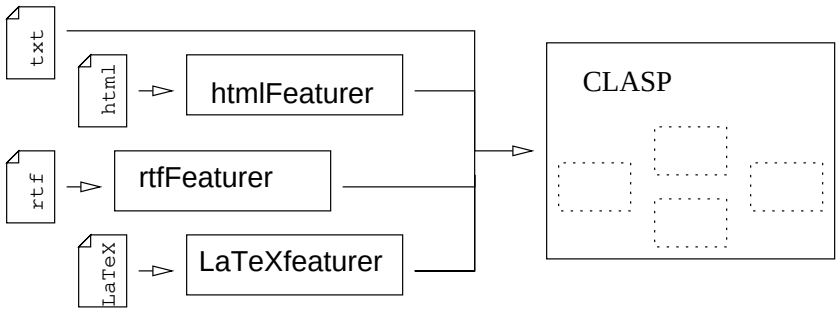


Figure 3.9: Extending CLASP system architecture to work with richer text formats.

Lines are used as the unit of granularity. I chose lines instead of sentences or paragraphs for simplicity, because it is easily accessible, robust, and not dependent on linguistic processing. I now describe CLASP's modules in order of execution.

3.3.1 Line style annotator module (`Layser`)

The development of information retrieval methods that query structured data (Loeffen, 1994) has increased the importance of both understanding the effects of layout as well as generating it. In CLASP, layout properties that are available in plain text are translated into features by the `Layser` module. `Layser` outputs an application-driven logical style (explained later) for each non-blank line.

Input Features. `Layser`'s machine learning features represent the visual cues available in plain text documents. We categorize the features into five groups similar to (Esposito, Malerba, and Semararo, 1994), all shown in Table 3.3 with example values:

1. **Spacing (4 features):** Both intra- and interline spacing are given as separate features. I also include alignment features such as the left and right

margins, which are normalized against the widest values found in the document.

2. **Marking (1 feature):** Orthographic case (e.g., upper or lower case) is the sole marking category feature. Font family, size and weight would be included here when CLASP is extended to handle more enriched text formats. It is calculated as the average value of the individual words' case.

3. **Punctuation (4 features):** These features capture different types of line-final punctuation (marking prose text) as well as line-initial ones (marking list items). To capture embedded headers (e.g., headers in-line with content), I also record the position of the first occurrence of a sentence-like punctuation.

4. **Word (1 feature):** Just the bag of words contained in the line. These can be used to find specific cue words that mark a particular logical style.

5. **Document (4 features):** These features address the overall characteristics of the document. I encode the approximate position of the line in the document, the document length, the document's average orthographic case, and the document's average number of interline blanks. This provides a method for passing exception information to the machine learner (e.g., when the entire document is in uppercase, it is essentially caseless and rules that use case should be dropped).

To model local context dependencies, `Layser` computes the first four feature categories for each line and for its neighboring previous and next non-blank lines. These $(4 + 1 + 4 + 1 * 3 =)$ 30 features are added to the 4 document features to make the final 34-dimensional feature vector used by `Layser`.

Output classes. `Layser` makes a first pass decision on the line's logical style by categorizing each line as one of 11 styles in Table 3.4. The line styles

␣Input␣Features.␣␣␣Layser's␣machine␣learning␣features␣represent␣the␣visual␣cues␣available␣in

Name	Description	Sample Value
Spacing Features		
intraSpace	average space between words (* 100)	$\frac{13}{12}$ * 100 = 108
interSpace	number of blank lines after	0
leftMargin	normalized to min left margin (widest: 0)	$\frac{1}{100}$ * 1000 = 10
rightMargin	normalized to min right margin (widest: 1000)	$\frac{90}{100}$ * 1000 = 900
Marking Features		
case	orthographic case (300 ≡ all words uppercase)	$\frac{(3*2)+(9*1)}{12}$ * 100 = 125
Punctuation Features		
initPunct	type of line initial punctuation	0
listNext	for lists; 1 if looks like next expected item	0
embedPunct	percentage position of first non-initial punctuation	$\frac{14}{89}$ * 1000 = 157
endPunct	type of line final punctuation	0
Word Features		
words	bag of words in the line	"input" "features" ...

Table 3.3: Sample line and calculated `Layser` feature values. Document category features not shown.

are motivated from the viewpoint of prospective applications: what types of logical styles might applications like to treat differently? For example, (Luc et al., 1999) showed that the syntax and style of enumerated list items have semantic meaning (i.e., bulleted items versus ordered items) and should be processed with the understanding of these semantics. Similarly, page information lines should be discarded in content processing but can help determine meta information about the text. Attribute-value pairs are often regarded as a specific kind of list item, but in CLASP have been given their own logical style because they can be directly inserted into a database. Since styles are motivated from the perspective of a generic application, CLASP excludes genre- and domain-specific logical styles, such as Bibliography or Salutation.

With the classification scheme, some lines could be classified as belonging to multiple classes: such as when List Item information is presented in Tables. To avoid this problem, annotators were asked to label each line's most salient logical style with respect to the end application. An exception was made in adding the Embedded Header tag, which marks a line as containing normal discourse text pref-

Line Style Class	Abbreviated Description
Table	part of the body of a table
Separator	rulelines, lines to separate sections
Attribute-value	left half of line has value in right half
List Item	(non-section like) list item
Embedded Header	headers on same line as content text
Header	headers, titles, subheaders
Caption	text attached to a picture, figure or table
Page Information	document openers, trailers, ToCs, page numbering
Auxiliary Text	secondary content (not main: e.g., bibliography, abstract)
Main Content Text	Content-oriented prose text
Unknown	*default tag, used for error checking*

Table 3.4: The inventory of logical line styles in the `Layser` and `Combiner` modules.

aced with an in-line header, since headers are needed to correctly infer segmentation structure.

At this point, `Layser` creates the features and then applies its machine learned model to produce a first pass classification of each line. Layout-oriented lines are passed to the `Header` module for further layout analysis, while content-oriented lines are passed to the `Coheser` module for lexical cohesion analysis.

3.3.2 Header priority module (`Header`)

This module receives Header and Embedded Header lines from `Layser`, and generates a feature set targeted at classifying the segment nesting depth of these lines, to directly produce a hierarchical segmentation. The initial feature set used in this task is similar to `Layser`: case, spacing, and punctuation features are all used. The system adds an additional feature, representing header scope, giving the percentage of the document that the header has immediate scope over (i.e., until the next line with a header line style).

Section headers may not manifest themselves in the same manner across documents; in one document the title may be centered, but in another it may be

Previous Header

␣␣␣␣Line␣style␣annotator␣module␣(Layser)

Current Header

␣Input␣Features.␣␣Layser's␣machine␣learning␣features␣represent␣the␣visual␣cues␣available␣in

Name	Description	Value (previous header - this header)
diffScope	lines header has immediate scope over	$\frac{2-28}{658}$ * 100 - = -4
diffCase	line's orthographic case	140 - 108 = 32
diffPrevBlank	blank lines before header	1 - 1 = 0
diffNextBlank	blank lines after header	1 - 0 = 1
diffPosition	position in text	228 - 242 = -14
diffEndPunct	end punctuation type	2 - 0 = 2
diffAvgSpace	average intraline space between words	100 - 108 = -8
diffEmbedPunct	intraline punctuation	914 - 157 = 757
diffLftMargin	left margin	40 - 10 = 30
diffRgtMargin	right margin	390 - 900 = -510

Table 3.5: Final `Header` features, calculated as the difference between initial `Layser` features.

in boldface type. Thus, the *relative differences* between these features in headers within the same document are likely to be correlated to their nesting depth, rather than their absolute values. `Header` thus computes its final feature set based on the differences in the values of these initial features in adjacent headers, shown in Table 3.5. This corresponds to learning whether one header dominates, is dominated by, or is on parity with an adjacent header. These pairwise features are `Header`'s output and are passed on to the `Combiner` final machine learning module.

3.3.3 Lexical cohesion module (`Coheser`)

Relationships between words of a document are known factors contributing to its structure. This factor has been used widely in discourse structure segmentation in the form of thesaural relations (Morris and Hirst, 1991; Kozima, 1993), cue phrases (Littman, 1996), word association (Pereira, Tishby, and Lee, 1993), as well as with different types of token repetition, as exemplified by SEGMENTER explained earlier in the chapter.

Lines containing prose (i.e., Main Content Text and Embedded Headers) by

`Layser` are fed into the `Coheser` module for lexical cohesion analysis. In documents without formatting, the entire document will be passed to `Coheser` as content text, and the system behaves like similar lexical cohesion based segmentation systems. Like the `Header` module, `Coheser` generates a set of features, but leaves the learning to the final module.

Input lines are first assigned part-of-speech tags from COMLEX (Grishman, Macleod, and Meyers, 1994) using the same method as in SEGMENTER (discussed in detail in Appendix C), and closed class words are discarded. I use the remaining content words per line as a bag of words, maintaining the same granularity as in `Layser`. CLASP computes a set of normalized similarity values between the source line's words and its neighboring non-blank lines (the target). To model the variance in cohesion strength over distance, the system computes the same set of features between the source line's words and varying sized neighboring targets (1, 2 and 3 adjacent lines, both before and after the source line).

The machine learning features computed are based on a battery of shallow indicators that have proved effective in detecting cohesion. These are categorized into four groups:

1. **Repetition (4 features):** These include term type and stem form repetition. The term type features model noun repetition but are subclassed as in SEGMENTER (e.g., pronouns, common nouns and proper nouns), as they have been shown in the SEGMENTER work to possess different cohesion strengths.

2. **Thesaural relations (8 features):** Similarity scores are calculated via w_{sim} (Resnik, 1995) for the two transitive, tree-structured relationships: noun and verb is-a (hypernym/hyponym) and noun part-of (holonym/meronym). Normalized counts provide separate features for the other WordNet relations.

3. **Word association (1 feature):** WordNet only captures word similarity

when the words participate in a structured relationship. To find other non-structural relationships, we use a word association feature which measures the Dice coefficient of correlation between grammatical subject noun heads in a large corpus (Schiffman and McKeown, 2000).

4. **Cue word (1 feature):** This feature finds cue words at the beginning of the source line. The feature has seven possible values, corresponding to the six categories of cue words listed in (Cohen, 1984): 1) parallel 2) detail, 3) inference, 4) summary, 5) reformulation, 6) contrast, plus a value for no cue word. This feature attempts to capture discourse cues used heavily by (Marcu, 1997b) to operationalize RST. Although the literature (Brown and Yule, 1983) finds cue words to be highly ambiguous with respect to which rhetorical relation they marked, for CLASP it is only important that they are noted as marking *some* discourse boundary (hopefully topical).

Table 3.6 gives an example source line with adjacent target window along with sample values.

3.3.4 Feature weighter (`Combiner`)

For each line, CLASP strings together all the vectors that have been computed. Null fields are inserted for lines that do not have `Header` and/or `Coheser` features, ensuring that all vectors have the same length. To build the document topic tree, the system runs the two tasks in series: first, the line annotation task is re-run with the combined features and then the hierarchical segmentation task is run.

An inductive rule learner, `ripper` (Cohen, 1995), is employed as the machine learner to generate the two classifiers. `ripper` outputs a human readable hypothesis file, which I used to validate whether certain common sense rules were learned by the system. Figure 3.10 shows some sample rules.

Source Line	`input features layser machine learning features represent visual cues available`
Target Window (2 lines before and after)	`translated features supervised learning layser module layser outputs application driven logical style explained later blank line` <*source_line*> `plain text documents categorize features five groups similar esposito shown table example values`

Name	Description	Sample Value
Repetition		
termPronoun	Pronoun repetition	0
termCommonN	Common noun repetition	$\frac{4}{10+29} * 1000 = 102$
termProperN	Proper noun repetition	$\frac{2}{10+29} * 1000 = 51$
lemmaRep	Simple token repetition (for all non-noun classes)	0
Thesaural		
wnSimIsA	Hyper/holonym similarity tree (using Resnik's W_{sim})	2821
wnSimPart	Holo/meronym similarity tree (again, via W_{sim})	155
wnAnt	Antonym (all classes) count	0
wnVerbCause	Verb "cause to" count	0
wnVerbEntail	Verb entailment count	0
wnNounAttr	Noun attribute count	0
wnPertain	Adjective pertaining to nouns count	0
Word Association		
wordAssoc	Word association (by Dice coefficient)	0
Cue Words		
cueWord	Type of cue word present at beginning of unit?	0 (no cue word)

Table 3.6: Lexical cohesion features and values calculated for sample source and target words used in the `Coheser` module.

3.4 Using CLASP to derive the document topic tree

The `Combiner` module produces the final output of the CLASP system. The results are easily transformed into a document topic tree, as shown in Figure 3.11. Lines that have been deemed headers by the second `Combiner` run are deemed topics. The structure of the topics constitute the tree structure for the topic tree. Siblings and parent/child relationships are calculated for each topic and encoded into the tree. The contents of the lines that are subsumed by each header (and thus topic) are analyzed for their contents. Word and paragraph counts are taken as are metrics of readability (in terms of Flesch-Kincaid scores (Flesch, 1946), although other readability metrics can be used (See (Harrison, 1980) for an overview). The

Sample logical style rules:
- if the next line is indented and looks like a list, then it's a *List Item*.
- if the average intraline spacing is high and it's not in the first 40% of the document, then it's a *Table*.
- if the line is preceded by blank lines and the line length is short and within the first 60% of the document, then it's a *Header*.

Sample segmentation rule for documents with headers:
- if the difference in the embedded punctuation feature is greater than 77, then the nesting level is *increased by one*.

Sample segmentation rule for documents without headers:
- if the previous line is blank and the next line's word association score on a window of five lines is small and the previous line's noun hypernym word similarity on a window of three lines is high, then this line is a *header of the same priority as the last one*.

Figure 3.10: Five translated rules from `Combiner`.

inventory of line styles subsumed by the topic are examined to derive statistics on the number and length of subordinate topics, tables, figures, and list items. These content features are encoded into the topic tree data structure as first described at the beginning of this chapter in Figure 3.1, in which individual topics contain statistics on what types of information they contain.

3.4.1 Style catalog

CLASP is a system that induces logical structure from its physical manifestation. In many text processing programs today, this division between logical structure and its mapping to a physical manifestation is respected (i.e., one could change all level 1 headers in a document to use a larger font by changing the style of the level 1 header, rather than changing each individually). The processes within CLASP can preserve the mapping of a document's logical styles to their physical manifestations,

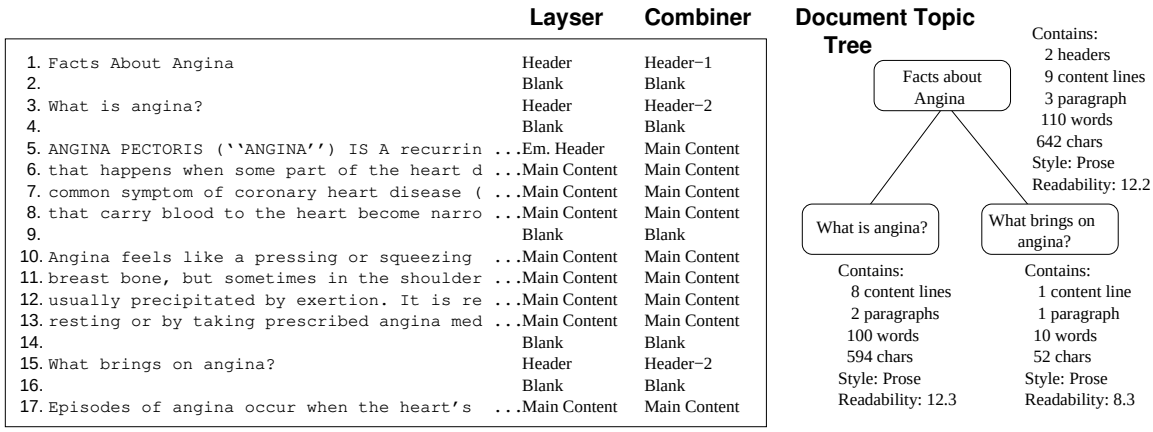


Figure 3.11: Excerpt of a CLASP run on the example angina document given in Appendix B, showing the processes the lead up to `Combiner` process and subsequent transformation into a document topic tree.

creating a *style catalog* (e.g., level 1 headers are centered and in all capitals).

A complementary process can then apply a style catalog to any document with logical structure to format the document according to the style catalog. This can be used to transform raw ASCII documents into a richer format or to canonicalize multiple documents with different physical manifestations. By placing these two components together, a feature with the same spirit as the `autoformat` feature in some word processing programs (e.g., Microsoft Word) is produced.

3.5 Evaluation of CLASP

I used seven documents, all from the medical domain (journal papers, health information book chapters and patient medical records), either converted from HTML or originally text, containing a total of 2515 non-blank lines. Eleven human subjects, all graduate students, given standardized instructions and no time limit, provided annotations for these documents. For logical line style, annotators categorized input text lines as one of the 11 logical styles. For hierarchical segmentation, subjects added nesting depth annotations to lines that they marked as Headers or Em-

bedded Headers. Also evaluated was CLASP's performance on the task of linear segmentation, judged at paragraph boundaries. To do this while reusing the human judgments, I removed all headers from the corpus to create a new headerless corpus, and reassociated the nesting annotations with the lines after headers as segment breaks.

The gold standard was established for all lines that had a majority of annotators agree on the line style or nesting depth. Average human precision, Kappa agreement, and baselines (all Main Content lines for the logical style task, all sections at the same level for the hierarchical segmentation task, no segment break for the linear segmentation task) were calculated for all gold standard lines. I then ran CLASP using 5-fold cross validation to find the system's performance. Table 3.7 gives figures for this glass box evaluation of the system.

Module	Task Type	Kappa	Baseline	$\overline{human}$ (σ_{xn})	5-fold C. V.
Layser	Line Style	.84	81.7%	90.1% (8.0%)	91.8% ± 0.6%
Combiner	Line Style				93.2% ± 0.6%
Combiner	Hierarchical Segmentation	.31	50.0%	53.4% (16.0%)	79.1% ± 6.6%
Combiner	Linear Seg. (at Ps only)	.90	70.0%	92.2% (6.4%)	74.3% ± 1.6%

Table 3.7: Glass-box, intrinsic evaluation of CLASP. Figures reflect precision versus group majority.

The results are very promising. With respect to the gold standard majority, CLASP performs above the level of the average human annotator, except for in the linear segmentation task. To further analyze CLASP's performance, I assessed the features used by `Ripper`, since it implicitly does feature selection when constructing its hypothesis. In the line style task, many of the lexical cohesion features replaced layout ones as more accurate indicators of logical style, making up 17% of the `Combiner` learned conditions, while shrinking the ruleset by 8% over the initial `Layser` one while increasing precision. Document features were not used much, as the corpus did not contain any exceptional documents (ones without certain features, such as a caseless email), but all other layout features targeted specific logical styles such that they proved useful.

In hierarchical segmentation, CLASP relies heavily on the difference features provided by `Header`, accounting for 10 of 24 conditions. Remaining conditions used hypernym and holonym word similarity as well as lemma repetition. The linear segmentation task is more difficult, since headers were removed. The rule base that resulted from `ripper` for this task was smaller (4 rules, 11 conditions), and word association and line position substitute for the unavailable `Header` features. As the system's performance is worse than in the task with headers, I assumed `Header` difference information is more valuable than lexical cohesion, which in turn is more useful than word association. Cue words and other count-based repetition features were pruned, due to sparse data.

The architecture of CLASP splits the construction of the structure of the document topic tree task into two different subtasks, that of line style and nesting. This distinction is real and is evidenced by the difference in Kappa. Also, annotation of hierarchical segmentation is more difficult and less certain on deeper levels: 91% agreement on level 1 headers, and 81% agreement on level 2, 3, 4 headers. For the hierarchical segmentation, Kappa indicates only weak agreement (.35), but when I conflated all segment breaks into a single category in linear segmentation, Kappa rises to a strong .90. This is a stronger level of agreement than other reported work, but in this work I have access to header information.

I also performed a task based evaluation of CLASP, using its resulting document topic trees to assist genre classification. The CLASP system was trained using the seven training documents in the intrinsic evaluation, and then applied to new documents from the Heart Information Network[1]. These documents are classified into one of the eight resource types: newsletters, questions, resources, articles, recipe listings, directories, educational and newsgroups. I took five documents from each category and ran CLASP to derive all (5 * 8) = 40 document structure trees.

[1]found at `http://www.heartinfo.org/reviews/` as of September 2002

I converted the document topic tree into features suitable for machine learning by taking the logical line styles and hierarchical segmentation and converting them into simple percentages, as shown in Figure 3.12.

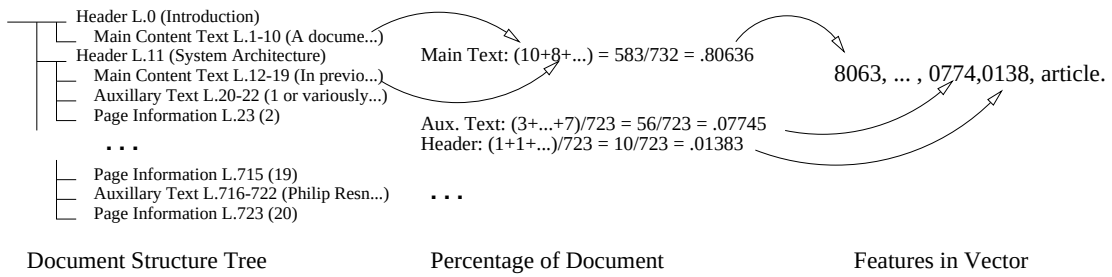


Figure 3.12: Converting a document structure tree to features.

The performance of these features were compared to both using a bag of words as a feature (BoW), and to Karlgren and Cutting's (1994) features (KC) to perform classification. Table 3.8 shows the results of the different learning methods using n-fold cross validation.

Baseline	Kappa	$\overline{human}$ (σ_{xn})	($n = 40$) fold C. V.		
			BoW	KC	CLASP
12.5%	.53	65.8% (2.7%)	20% ± 6.5%	17.5 ± 6.1%	22.5 ± 6.8%

Table 3.8: Black-box, extrinsic evaluation of CLASP via genre classification. Figures reflect precision versus group majority. BoW and KC figures are for both with and without the addition of CLASP features.

3.6 Conclusion

In this chapter, I have discussed how the discourse structure of documents is necessary to enable information retrieval at a more fine grained level. I have put forth that topic level is a suitable level of representation that is fine grained enough to localize concepts and relations between them, yet broad enough to be applicable to users' queries and alignable across documents. I showed how the construction

of the document topic tree, as proposed by others and supported by my work is a good representation of this topical structure, as it additionally supports scalable granularity.

I have also discussed how my work in linear segmentation using token repetition, embodied in the SEGMENTER algorithm, enables a more accurate topical segmentation to be calculated than was previously available. I have also elaborated on how the orthogonal stream of layout information can be added to this stream of lexical cohesion information, embodied by the implementation of CLASP, to extend the advances in linear segmentation to hierarchical tree representations, such as the target structure of the document topic tree.

Chapter 4

Creating composite topic trees

For large collections of documents that target the same specific domains and genres, there is a strong likelihood that many documents will discuss similar topics. Furthermore these documents often discuss these topics using a common discourse structure. This is also true of the documents returned from an online IR search engine in response to a user's query, since many subjects have multiple websites delivering a mixture of unique and redundant content relevant to the query. Examples of this phenomenon include consumer information on diseases, departmental websites, artist's biographies, company annual reports and travel brochures. For instance, a travel brochure may list the topics of cultural attractions, local customs and airfare information, in that particular order. Consumer health information on medical conditions outline the condition, its symptoms, diagnosis and treatment. In some cases, limitations on content and its structure may even be codified, e.g., journal submission guidelines or government mandated corporate filings.

In this chapter, I first outline the criteria that determine common structure in such documents, which I term *text type*. After defining text type, I describe a novel method for acquiring such a model of common structure for a text type. Basically, the approach merges topic structures in example documents to form a

composite model. This approach builds upon the last chapter, since a document's topic structure is equivalent to its document topic tree. This merging process thus combines several document topic trees into a composite one. This process results in a prototypical topic tree that holds a model representation in which the example documents can be viewed as instances. Figure 4.1 shows such a *composite topic tree* (CTT) for consumer healthcare documents on diseases, of which the example document on angina (given in full in Appendix B) is considered an instance.

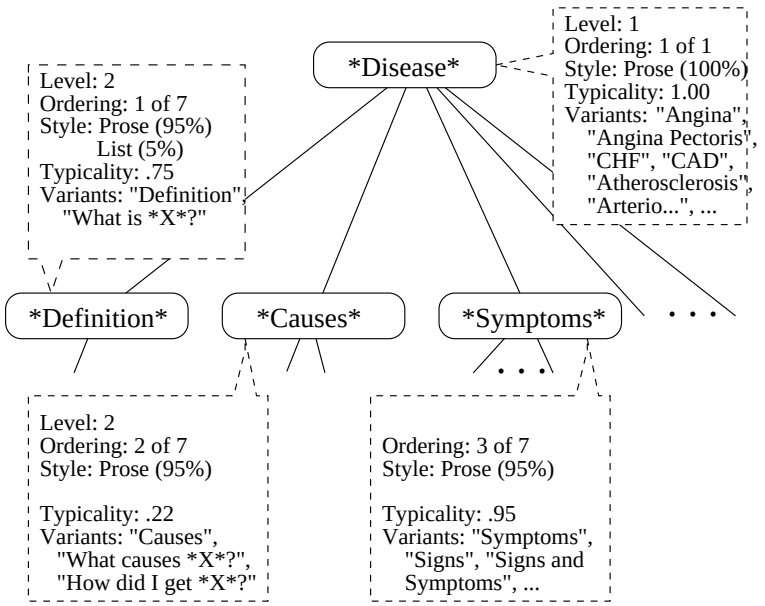


Figure 4.1: Excerpt of an automatically constructed topic tree for consumer healthcare documents on diseases.

Knowing the topic structure for a set of documents of particular genre and domain constitutes a domain model and has been shown to enhance the performance of systems in both analysis and synthesis. Indexing topical structures and content can enhance information retrieval applications in search (Hahn, 1990; Belkin et al., 1994). Topical knowledge can also serve as an organization for text summaries (DeJong, 1982; Lin, 1998), or can be used to select sentences for an abstract (Liddy, 1991). For specific domains and genres, this logic is often coded manually by

humans. When systems are extended to work in multiple genres and domains, it becomes desirable to have a system learn these topical conventions on its own.

Topic categories by themselves allow the computation of typical and rare topics, a key feature in computing similarities and differences in CENTRIFUSER. However, learning topic structure for domains and genres has additional benefits over learning keywords or flat topic categories. The imposition of tree structure from the topic tree further increases the accuracy of these topic frequency computations, as spatial location of topics in the documents are factored in. In addition, it enables a hierarchical organization and linear orderings of topics to be computed, which is a key feature that allows for intuitive spatial navigation of the information space, further detailed in next chapter.

In the next section, I review related work to the problem of constructing a normalized representation of documents. I then define *text type*, which characterizes the required relationship between input documents. I then discuss the representation for topics that make up a document's topical structure. I present the main contribution of this chapter, an algorithm for producing the composite topic tree, which encodes information on this common topical structure. The chapter concludes with an evaluation of the system on documents from two domains and a discussion of the possible applications for composite topic trees.

4.1 Related work

Creating a composite topic tree combines structural information as well as lexical information. Much of the related work in creating a topical representation of a document collection relaxes the structural constraints (thus the added information) used by CENTRIFUSER.

When structural information is neglected, the problem is simplified to creating (in text categorization) or detecting (topic spotting) topics from documents

that are represented as streams of words. A very wide field of research, this work has been a focus for both the text and speech community and a testbed application for many machine learning algorithms. Recent work has looked at incorporating linguistic features such as lexical chaining (Jing and Tzoukermann, 1999) with machine learning approaches such as semantic perceptron nets that integrate well with such approaches (Liu and Chua, 2001).

Although topic spotting and text categorization primarily deal with a set of topic clusters, a hierarchical tree of topics can also be obtained by applying agglomerative clustering techniques. This is the approach taken in generating the topic trees (different in definition than ones used by CENTRIFUSER) used by researchers in language modeling for speech recognition (Seymour and Rosenfeld, 1997). Furthermore, such topic trees are grown from clusters without any respect to intradocument topic ordering, which is crucial for the generation of an appropriate composite topic tree. Although such topic trees are similar in spirit to ones generated in this chapter in that they recursively specify more specific topic structure, they differ in their purpose (as a backoff method for speech recognition compared to an information norm about a common class of texts) and in their acquisition methodology.

4.2 Text type

CENTRIFUSER's composite topic tree construction module operates on documents that belong to the same text type. I define a *text type* to be a set of documents that share the same domain and genre.

- Genre restriction - Must be intended for the same purpose or communicative goal. Must also be expository in style.

- Domain restriction - Must be about the same subject area.

This broad view of text type is well-suited for our system, when one considers only expository genres. Biber describes the concept of genre as "the text categories readily distinguished by mature speakers of a language" (1989), which would be distinguished by their location (e.g., in a newspaper) and by their format. I differ from Biber's notion of genre, using purpose to define a genre rather than his features of location and format. Hoffman's work gives a clear inventory of features to use to identify text type (Hoffman, 1991). He focuses on a feature set that considers both the linguistic aspect (macrostructure, coherence, syntax, vocabulary and grammatical categories) as well as the communicative aspect (competence of writer/reader, intention/function, situation, subject matter). For my purposes, the intersection of Hoffman's intention (genre) and subject matter (domain) features define my notion of text type.

Documents which share the same text type often have a similar topical structure. Hoffman defines this as the macrostructure feature, part of the features that capture the linguistic aspects of documents. This phenomenon is also characterized as Domain Communicative Knowledge (DCK) (Kittredge, Korelsky, and Rambow, 1991). There are three aspects to this topical structure: 1) similar topical content, 2) similar ordering between these topics (e.g., symptom information before treatment, as one needs to know how to identify the medical condition before treating it), and 3) similar notion of importance among topics (e.g., information about complications of a disease occurs less often than symptom information).

In constructing the composite topic tree, my algorithm attempts to capture these three aspects. The CTT construction algorithm does this by modifying the document topic tree structure. It adds frequency and variant information to each of the data fields in the original document topic tree structure: Topic identity, level (i.e., granularity in the topic hierarchy) and ordering information (i.e., position among sibling of the same parent topic) are enriched with frequency information

that results from merging similar topics across documents.

Figure 4.2 shows the top-level portions of the document topic trees for several documents from the text type of consumer healthcare on diseases. A cursory review of these documents shows these certain regularities in the structure which are captured in the sample composite tree from Figure 4.1.

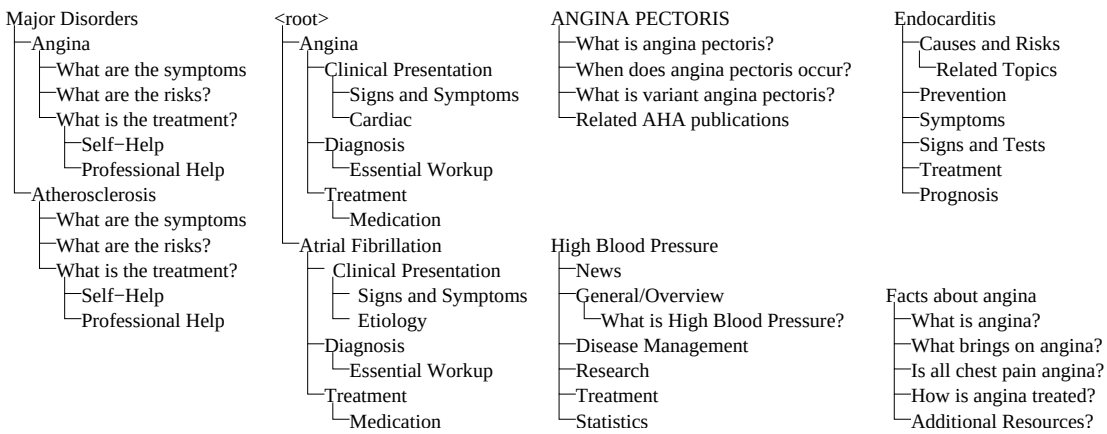


Figure 4.2: Excerpts of sample consumer health document topic trees (in outline form) containing disease information, used in part to construct the composite topic tree shown in Figure 4.1.

4.2.1 Document granularity

In Figure 4.2, there are several texts on different disease that are discussed using a common structure. In the respective document topic trees, the diseases and their common substructure are simply marked as topics. However, in the construction of the CTT, it is important to differentiate that the disease topic can vary (i.e., angina, high blood pressure, diabetes, etc.) but that the other topics in the common substructure are fixed. To do this, I use the term *instance* to refer to the varying parts of the topic tree structure, as each disease is an instance of the common topic structure. *Substructure* nodes make up this common structure. Thus, a text type may consist of documents that discuss infinitely many instances, but

their substructure composition is largely consistent across the instances. Table 4.1 illustrates this distinction in a number of text types.

Text type	Sample instances	Sample substructure nodes
Company Websites	IBM, Snapple, Republic Office Supply	Contact, Products, Address, Investor Relations
Legal Cases	Flippo vs. West Virginia, U.S. vs. Adams	Plantiff, Defendant Headnotes, Verdict
Travel Brochures	Laos, Alberta, Las Vegas, Hawai'i	Getting There, Accommodations, Dining, Nightlife, Costs
School Course Catalog	Introduction to Biology, Calculus III, Aerobics	Department, Location, Schedule, Prerequisites, Description

Table 4.1: Instances and sample substructure topic nodes for different text types.

Identifying and delimiting of instances from substructure is a key task in constructing the composite topic tree. This task is complicated by the fact that documents belonging to the same text type may differ in their granularity. Taking consumer healthcare information as an example, the documents in Figure 4.2 all discuss a single instance per document, in which the instance node is the root node in each document topic tree. However, this is not the only possible granularity. Documents giving disease information on consumer healthcare could also discuss diseases at a higher level, discussing multiple diseases, or at a lower level, discussing only specific facets of a disease. This breaks down into three different document granularities:

1. *Multi-instance documents* give information on several instances within the document. Instances are not given as the top-level node; a more general *genus* node describes the instances' relation to each other. This genus node is given as the root node of the document. These documents often have repeated substructure as part of its description of each instance. An example of this would be a document about cardiac diseases.

2. *Single instance documents* are most common type. Each document describes a single instance. The instance topic is sometimes prefaced or encompassed in a contextual or introductory topic. I have illustrated many documents of this sort, in Figure 4.2.

3. *Sub-instance documents* only discuss specific parts of the substructure. This can occur when a comprehensive resource or web site contains a sizable amount of information that is better organized as a group of distinct documents. Each document then examines a different part of an instance's entire substructure. As such, these sub-instance documents often do not have an explicit instance node. An example of this would be a document discussing treatment options for chronic angina. A discussion of the instance topic, angina, is not a part of the document.

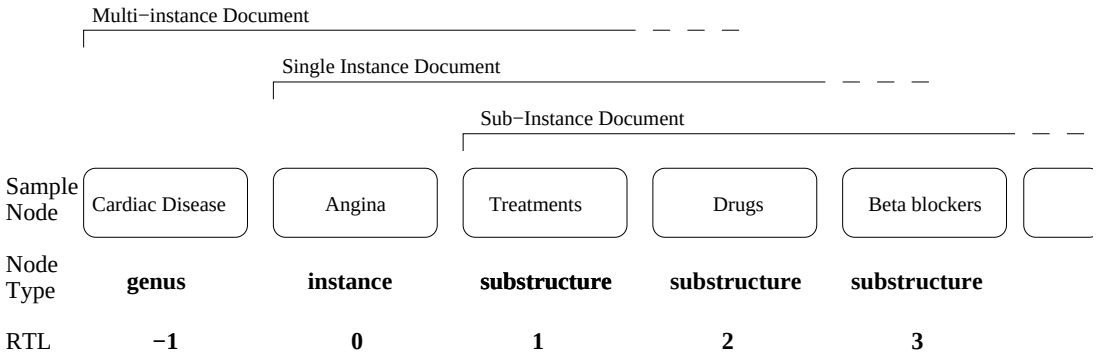


Figure 4.3: Granularity levels, from (l) most general to (r) most specific. Relative Topic Levels (RTL) will be discussed in Section 4.3.2.

Figure 4.3 relates these three granularity levels to their root node types, and examples of multi- and sub-instance documents can be seen Figure 4.4. To identify instances and correctly align the parts of the topic substructure it is necessary that a program be aware of these three different granularities. I will now explain the CTT construction algorithm and show how it capitalizes on these differences in

granularity to conserve computation time and improve accuracy in building the composite topic tree.

4.3 Creating the composite topic tree (CTT)

The input to the construction process is a set of document topic trees. Since the system does not know which documents are multi-instance, single instance or sub-instance beforehand, all nodes begin with a generic type. As the algorithm progresses and more information is learned, nodes are retyped to reflect whether they are genus, instance or substructure nodes and whether they have been merged with other headers internally in the document or across documents.

The composite topic tree algorithm has three main design features that increases its performance over a straightforward design. Instead of describing the algorithm in a linear fashion, I structure this part of the chapter to highlight these contributions.

4.3.1 Design feature 1: Structural similarity metric

Since the algorithm has access to a structured representation of topic nodes, it differs from other similarity metrics used to conflate words (e.g., lexical chaining) in that it capitalizes on the representation when calculating similarities. The similarity calculation uses a single metric which combines topic header, level, normalized level (RTL – discussed later in design feature 2), ordering and parent node information. This single value facilitates picking a best match when several nodes are somewhat similar to each other. The metric can be invoked with minimum thresholds for each feature to restrict topic matching to certain data fields in the topic node structure (e.g., "Only match if on the same level").

Similarity calculations for numeric fields, such as order[1] and level, are straightforward (Equations 4.1 & 4.2): similarity is the absolute difference between the node's level or order values.

Character fields, such as topic header, similarity is calculated by first removing stopwords and then computing the maximum percentage of word overlap between all variant forms for each node's header (Equations 4.3 & 4.4).

$$sim_{order}(x, y) = abs(avg(order_x) - avg(order_y)) \tag{4.1}$$

$$sim_{level}(x, y) = abs(argmax(level_x) - argmax(level_y)) \tag{4.2}$$

$$sim_{header}(x, y) = max(sim_{string}(x, y)) \tag{4.3}$$

$$sim_{string}(x, y) = \frac{overlap(x, y)}{max(|w_x|, |w_y|)} \tag{4.4}$$

$$overlap(x, y) = \sum_{i=1}^{|w_x|} \sum_{j=1}^{|w_y|} eq_w(x, i, y, j), \text{ where } \quad eq_w(x, i, y, j) = \begin{cases} 1 & \text{if } w_{xi} = w_{yj}, \\ 0 & \text{otherwise.} \end{cases} \tag{4.5}$$

$$sim_{parent} = \begin{cases} x & \text{if } parent_x = parent_y, \\ 0 & \text{otherwise.} \end{cases} \tag{4.6}$$

To merge two nodes, the system conducts a pairwise comparison between a source node and all possible candidate nodes. The target with the highest similarity must also select the source node from its possible merging candidates (in effect, a symmetric constraint on similarity).

When nodes are merged, the merged representation stores a composite profile consisting of all data for each of topic nodes that make up the composite topic: the topic's header, level, order, contents, parent and children fields. This is saved along with frequency information which indicates how frequent each value of a field is.

[1] In the document and composite topic trees, *order* is expressed as a fraction: A sequence of three topics having the same parent would receive *order* values of 0, .5 and 1, respectively

The number of nodes that have been merged to form a composite node is also stored. This raw measure can be used to represent the node's typicality (a node resulting from many merges is more typical than an unmerged node). Frequency also helps to provide accurate information about the merged node. For example, if the "Treatment" subtopic occurs five times after "Symptoms" and once before it, the module reports that it is more likely to appear after. Similarly, if the attractions subtopic is lexicalized sometimes as "Symptoms" or as "Signs", the module reports the more commonly used form. This is why average values are needed in the similarity formulas: if a two or more nodes are merged that have different values, the composite value should reflect the average between them. In sum, frequency information allows the use of the average or most frequent value of the data field when it makes sense. This is reflected in the similarity equations in their use of $argmax$ and avg.

4.3.2 Design feature 2: Using relative topic level (RTL)

The CTT construction module processes documents of all three document granularities. As such, these different levels need to be handled correctly when identifying and merging topics. An example of this is when topic nodes from one document reside on different tree levels in different documents. For example, a multi-instance document such as "Cardiac Diseases" could mention angina on level 2 of its document topic tree, whereas the example document on angina given in Appendix B has "Facts about angina" as its root node. It is important to normalize the levels such that subsequent similarity calculations can understand that these are conceptually on the same tree level. The CTT module performs the normalization by introducing a new field in the node's data structure, the *relative topic level* (RTL), which is the node's relative difference between a node's level in the topic tree compared to the document's instance level. This is shown on the RTL line in Figure 4.3, in

which genus nodes have negative RTL and substructure nodes have positive RTL. RTL enables the construction module to place topic nodes into a canonical location in the composite tree, although the actual level of the topic may differ in each document.

Using RTL gives a distinct advantage in handling documents of differing granularity. This is because RTLs can also be propagated from one document to another. Figure 4.4 illustrates this process in a simplified example. Document A, a multi-instance document, has repeated substructure nodes and identified instances. RTLs for each topic in Document A is calculated based on this knowledge (panel 1). During the identification and merging of similar topic nodes across documents, parts of Documents B and C are found to be equivalent to substructure nodes in Document A, and thus the RTLs for these substructure nodes are propagated to B and C (panel 2). RTL can then be calculated for both B and C and their document type identified (panel 3). One can see that RTL plays a particularly important role in the connection of sub-instance documents (such as in document C), when the instance itself is not present in the document. RTL allows these documents to be attached at the appropriate level, normalizing the differences in document granularity.

4.3.3 Design feature 3: Using a three tiered merging approach

A straightforward approach to constructing the topic tree is to compare the topic data structures across documents and merge them if they are similar. The merging process links nodes in the individual document topic trees together to form a single composite topic tree.

I adopt this approach, but divide the task into three discrete phases to better account for the three different document granularities. The initial phase

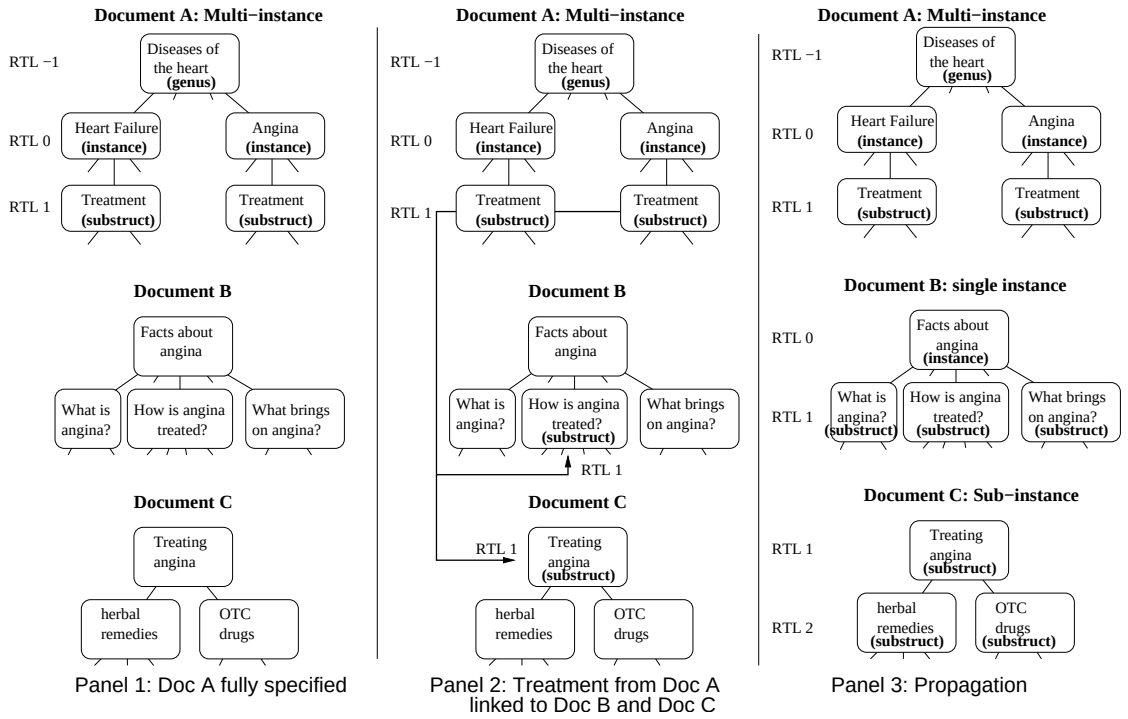


Figure 4.4: RTL in action. 1) original configuration, 2) after topic merging, 3) propagation and identification of document type. Nodes not essential to the RTL propagation process omitted due to space considerations.

only considers a narrow set of topics that are very likely to be similar. Subsequent phases expand the pool of topic nodes under consideration by lowering the similarity threshold. This modification improves the algorithm by using a tiered approach which considers high quality resources first.

4.3.3.1 Multi-instance document identification and merging

The topic merging process begins with multi-instance documents, outlined in Figure 4.5 in pseudo code form. In multi-instance documents, the wording of headers across topics is likely to be exactly the same: a document that describes several different disease will most likely repeat the child topic "Symptoms" for each of disease, but is unlikely to alternate it with variants such as "Signs". This assumption holds because the different instances share the same topic substructure, and can be

inferred from from Grice's maxim of manner: "Be clear, avoid obscurity" (Grice, 1975).

Recall that the composite topic tree construction process begins with all nodes untyped, as generic nodes. Repeated headers that are children of different parent nodes classify a document as a multi-instance. Furthermore, the nodes with repeated headers must be part of the substructure. Nodes having identical headers are merged to form a new node, and its constituents are retyped as locally merged substructure nodes.

Identifying repeated substructure topics in a multi-instance document also identifies its instance topics. This is done by identifying the smallest level (i.e., closest to the root) that contains a repeated substructure node. Topic nodes which are parent to this substructure node are thus instance nodes. For example, the repeated substructure node "Symptoms" might have three parent nodes that directly dominate it: angina, high blood pressure and heart failure, all of which are now retyped as instance nodes. Once the level where instance nodes is known, RTL is calculated for all nodes in the multi-instance document.

Occasionally, instance names appear in substructure headers, such as in "Symptoms of angina". If merged as is, the result will not be general to all instances (e.g., "Symptoms of angina" may be viewed as a constituent substructure for the instance "Diabetes"), which is incorrect. To correct for this phenomenon, each time a new instance is identified, the system replaces all occurrences of it with a generic <`instance`> token (e.g., "Symptoms of <instance>").

Repeated parts of a header can occur in nodes belonging to a common parent. An example of this is illustrated by the example document in Appendix B. Here, the single parent node "Facts about Angina" has many children that include the word "angina", including "What is angina?", "What brings on angina?", and "How is angina diagnosed?". The system infers that the common string "angina" is an

instance and that "Facts about <instance>" is a variant form of "angina".

```
foreach doc_k do
   foreach topic_t1,t2 in k where index_t1 ≠ index_t2
     if sim(topic_t1, topic_t2) == 1 then
       /* Have a multi-instance document */
       mergeNodes(topic_t1, topic_t2)
       typeDocument(doc_k, multiInstance)
       propagateRTL(doc_k)
     end if
   end foreach
end foreach
```

Figure 4.5: Pseudo code for multiple instance document merging.

4.3.3.2 Main cross document merging

At this point, multi-instance documents have most likely been identified, and the system has exhausted the advantages of processing them first. The second phase of the algorithm examines all documents to perform cross document merging. Related substructure and instance nodes across documents may not use the exact same wording nor be placed at the same level or order, so the merging uses a notion of similarity rather searching strictly for identical nodes as it does in the first phase.

This phase, outlined in Figure 4.6, iteratively merges the two most similar nodes across documents. All nodes are first placed into a pool and are then pairwise examined. If the two nodes belong to different documents and their similarity exceeds threshold, a new node representing their merge replaces both in the pool. Node type information (e.g., instance or substructure) is propagated and RTL is calculated in newly joined documents when needed. The process starts by merging pairs of nodes highest in similarity and continues until no pair of nodes have a similarity above the high threshold. The high threshold is necessary to keep improper merges to a minimum.

```
changed = true
while changed == true do
   changed = false
   foreach topic_{t1,t2} in doc_{j,k} where index_{t1} ≠ index_{t2} and j ≠ k
     if sim(topic_{t1}, topic_{t2}) ≥ threshold_{main} then
       /* Topics close enough in similarity, merge */
       mergeNodes(topic_{t1}, topic_{t2})
       if defined(nodeType_t1) == true or defined(nodeType_t2) == true then
         propagateNodeType(doc_j)
         propagateNodeType(doc_k)
       end if
       if defined(RTL_j) == true or defined(RTL_k) == true then
         propagateRTL(doc_j)
         propagateRTL(doc_k)
       end if
       /* continue loop if something gets merged */
       changed = true
     end if
   end foreach
end while
```

Figure 4.6: Pseudo code for main cross document merging.

4.3.3.3 Hierarchical merging

At this point, the system has merged similar instances and substructure across documents and has produced a pool of merged topics. In the last phase, the merging module sets the similarity threshold high to limit noise. However, there may still be remaining nodes that should be merged: these nodes may express variant headers with few or no words in common. The system can merge these nodes if their parent nodes are merged and identical, and if their intra-level ordering is similar. For example, even though "signs" and "symptoms" have no words in common, they can be merged together as their tree structures (as captured by parent, children and order fields) are identical. To do this, CENTRIFUSER needs to lower the similarity threshold to merge topics of this type, but limits errors by limiting the candidate nodes to be merged.

CENTRIFUSER employs top-down hierarchical merging in this last merging phase to limit the comparison. This step is given as pseudo-code in Figure 4.7. The purpose of this final phase is to locate substructure nodes that can be merged. This means that the system considers the topmost nodes that are most certain first,

and proceeds recursively to each of the children nodes in a breadth-first traversal, as in Figure 4.8. At each stage, the system limits the pool of candidate topics to be considered to those within n levels of the node's RTL. At this point in the algorithm, if a node still does not have a calculated RTL, its topmost (root to nth level) nodes are also considered in the merging, since it may be a sub-instance document, whose top level nodes can only now be merged.

I walk through a simplified version of this top-down merging shown in Figure 4.8. For clarity, I use $n = 1$ here to limit comparison to a single level and assume that the other two earlier phases yielded no merging results. The algorithm begins examining all three documents at their root level. Document D and E match at the top level, so their root level nodes are joined. The second iteration examines documents D and E at the second level, but keeps trying to find a match for document F at the root level (as it could not be joined). The second iteration merges F as sub-instance document, and in the third iteration the algorithm the recursive descent now examines children nodes in F as well.

After this depth-wise traversal of the document nodes, the system has completed all possible merging, and the algorithm is finished. The final result is the composite topic tree nodes, whose nodes give normalized level information in RTL and have pointers to their constituent document topic nodes.

4.4 Evaluation setup

The entire algorithm has been implemented and I have evaluated its performance and tested its robustness across domains. Since there are no corpora containing documents of specific text types nor accurate programs for detecting text types, I first needed to construct a text type corpus. I chose to test the system on patient health information on heart diseases and on another unrelated text type of consumer travel brochures. To gather the text type corpora, I first chose the ap-

```
/* construct pool of initial root HAC candidates */
level = minRTL()
pool = constructPool(level)            /* see subprocedure below */
/* start HAC process */
while numElements(pool) ≠ 0 do
  foreach topic_{t1,t2} in pool where index_{t1} ≠ index_{t2}
     if sim(topic_{t1}, topic_{t2}, similar) = threshold_{hac} then
       /* Topics close enough in similarity, merge */
       mergeNodes(topic_{t1}, topic_{t2})
       if defined(nodeType_t1) == true or defined(nodeType_t2) == true then
         propagateNodeType(doc_j)
         propagateNodeType(doc_k)
       end if
       if defined(RTL_j) == true or defined(RTL_k) == true then
         propagateRTL(doc_j)
         propagateRTL(doc_k)
       end if
       /* construct new HAC pool */
       level = level + 1
       pool = constructPool(level)
     end if
   end foreach
end while

subprocedure constructPool(rtlLevel)
foreach topic_{t1} in allTopics() do
   if (defined(RTL(getDoc(t1)) == false or RTL(t1) == rtlLevel) then
     addElement(pool,t1)
   end if
end foreach return pool
```

Figure 4.7: Pseudo code for the final HAC step.

propriate categories in three website directories (Yahoo!, About.com, and the Open Directory Project) and downloaded all documents on angina and heart attacks (for the text type of consumer health information on heart diseases) and on Laos and Morocco (for the text type of travel brochures). Since the directories categorize by subject, these documents fulfill the domain restriction but not necessarily the genre restriction (i.e., the list includes content pages, but also browsing pages, advertising, etc.). I asked 5 volunteers to pick out the documents that contained an expository discussion of the topic to manually simulate our genre restriction. The task was reported to be difficult but reached a moderate level of agreement (κ = .49, for travel and κ = .48 for patient information, p << 0.001 for both). I chose all documents that appeared on a majority of the subject's lists for our corpus (72 patient information ones and 120 travel documents), splitting it into equal halves

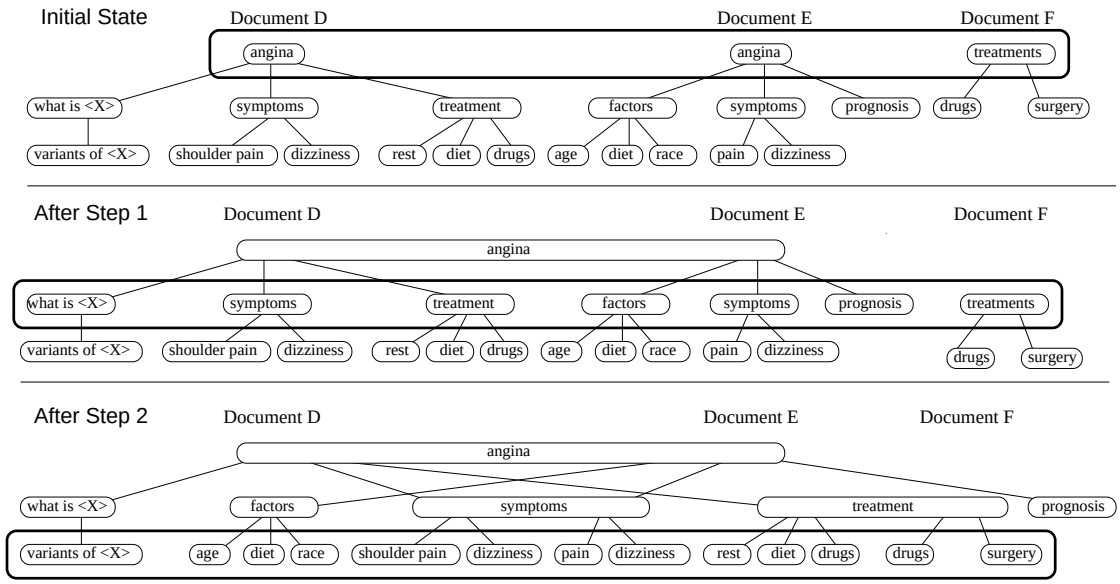


Figure 4.8: Three steps in the top-down hierarchical merging process. Topic nodes under consideration for merging indicated by the black ring.

for algorithm development and testing.

4.4.1 Results

I asked two reference librarians who specialize in health sciences to subjectively assess the quality of both the topic trees for travel (as non experts) and the patient information on heart disease (as experts). To do this, I converted the trees into an outline format where the level and order information are preserved and the relative importance (i.e., the frequency of the subtopic) was indicated by font color and size. The evaluation guidelines and the outlines are provided in Appendix D. The librarians were asked to evaluate the topic trees in the three areas of topical structure mentioned in Section 4.2: *content* (Should the items on the outline be there? Are any items missing?), *ordering* (Are there items that should be promoted or demoted a level in the outline?) and *typicality* (Are the larger font items ones that all documents of this type should have?).

Topics:

angina | angina pectoris | angina patient information | chest pain due to angina and other causes
heart attack myocardial infarction | heart attack | about heart attacks | heart attacks | what is a heart attack | ...

Outline:

1. <topic> disease
2. basic information
    2.1 description
3. signals of a <topic>
4. frequent signs and symptoms | signs & symptoms any of the following
5. the cardiac care unit ccu | care for a <topic>
6. symptoms
7. unknown
    7.1 atherectomy
    7.2 laser angioplasty | coronary angioplasty
8. coronary arteries in <topic> disease | coronary artery bypass | coronary bypass
9. atherosclerosis | what is atherosclerosis
    9.1 what is <topic>
    9.3 what symptoms can occur with <subject> |
        what are the symptoms of a <topic>

Figure 4.9: Excerpt of the evaluation outline for patient information on heart diseases. The entire evaluation outlines for both domains is replicated in Appendix D.

The results of the evaluations are summarized and reported in Table 4.2. The librarians reported satisfaction with the variants that the algorithm merged, but also reported (4 mistakes of a total 40) that semantically similar headers were not merged when they should have been, indicating that the system can relax the similarity threshold to allow additional merging. In the consumer travel text type, "cultural attractions" and "shopping" subtopics were conspicuously missing (2 of 8 content errors). Analysis of the corpus revealed that these problems were

Error Type	Consumer health information	Travel brochures
Content, total	9	3
- content	3	3
- grammaticality	2	0
- mischunking	4	0
- default variant	0	0
Structural, total	2	21
- wrong level	0	13
- wrong ordering	2	8
Typicality	2	3
All errors	13	27

Table 4.2: Errors reported in the CTT construction module as reported by two reference librarians.

artifacts of the testing documents: cultural attractions were directly named in the headers and shopping is not a primary attraction in either country. Minor problems with grammaticality (3 other content errors) (e.g., "What causes a <topic>" works when combined with "Heart Attack" but not "Angina") were the most prominent problems in this area. Introducing shallow parsing that would identify dependent articles and prepositions could help here.

Node ordering within the outline comprised the bulk of the problems. The librarians agreed that specific examples should be relegated to the end of the outline and that primary information should be moved to the front of the outline. Analysis revealed that sub-instance documents that were merged incorrectly as single topic documents sometimes caused this problem. Additional restrictions on the final hierarchical merging phase may help here.

Librarians were satisfied with typicality values as judged by the system (5 errors). They were also satisfied with the default header that the system chose.

Overall, the librarians both concurred that the system performed better on the patient information text type than on consumer travel. I believe this to be caused by the fact that the consumer travel corpus included both tourist travel as

well as business and investing travel. The outlines were comprised of 35 merged topics and subtopics for patient information, and 95 for the consumer travel, averaging one error every 2.7 headers in the former, 3.5 in the latter. Some headers produced two or more errors, thus the header to error ratio is higher. Based on this evaluation, I believe that the algorithm performs satisfactory but improvements could be made, especially with respect to structural considerations.

4.5 Conclusion

Composite topic trees can be used by a wide range of applications. The tree is a knowledge representation of a text type; a script that text generation can follow for structuring content. The topic tree encodes a descriptive relation between topics and their subtopic tree that can be used to augment a lexicon, complementing work done on the hyponym and meronym relationships (Hearst, 1992; Berland and Charniak, 1999). As discussed in related work, current text categorization approaches do not generally take document structure into consideration, and could combine such composite topic trees for richer source of data for classification. For the application of text summarization, documents can be compared against their text type's tree to classify its topics into ones that are generic and ones that contain unique information that should be reported. I detail how the interactions between the document topic trees and their composite one assists in generating useful task-oriented summaries in the next chapters.

In this chapter, I have shown a new method for generalizing this topic structures across related documents. I have defined the nature of the relatedness between documents that gives rise to similar topical structure – the notion of a text type, defined by the intersection of domain and genre.

I make three contributions to the state of the art in topic analysis in the architecture of our merging algorithm. First, I developed a similarity metric that

capitalizes on such structural information and allows fine grained control when calculating topic similarity. Second, I introduced relative topic level as a device for normalizing locations of topics across document of differing granularity. Finally, I used a three phased approach in the merging process that takes advantage of the different document granularities to reduce errors.

The composite topic tree is a norm which allows an automated system to make inferences about any document that is an instance of it. This includes the ability to determine whether a document contains any unique or typical information with regard to its topics, and to infer whether its topics are discussed using typical vocabulary, style and ordering. These are crucial abilities that allow CENTRIFUSER to identify salient and unique aspects of documents to use in building its summaries of documents, which I will elaborate on in the following chapter.

Chapter 5

Indicative and informative summaries for searching and browsing

In this chapter, I show how the document and composite topic trees developed in Chapters 3 and 4 are used to explicitly support the two basic information seeking modes of browsing and searching, as per Strategy 4 (Form a user model). Many IR interfaces support one of these two modes or support both in separate forms. But without knowing *a priori* which interaction type an information seeker may want or need, it makes sense to provide facilities for both within a unified interface.

This unified framework is the primary contribution discussed in this chapter. This chapter examines how the pre-processing tasks described in the earlier chapters come together to produce the CENTRIFUSER summary in the final online phase of processing. CENTRIFUSER has been designed specifically to produce summaries of different lengths, such that the results can be displayed on various computing platforms – large screen workstations as well as small screen personal digital assistants. This is a further adaptation of CENTRIFUSER to provide support

for different user models.

In a nutshell, the online integration component discussed in this chapter consists of three steps. First, the user's query is mapped to the closest matching topic in each document topic tree and composite tree. Next, differences and similarities are identified on a per-topic basis in the documents. Finally, each of the three components of CENTRIFUSER's interface – 1) the navigation bar, 2) a synopsis based on sentence extracts and the 3) generated (not extracted) indicative group summaries – uses these query topics to compute its relation to other topics in the topic trees and produces the component summary output. I explain the steps used by each of the three components and discuss how they work to fulfill the different needs of the browser and searcher.

5.1 Integration with offline processing

In a production information retrieval system, time efficiency is of great importance, second only to precision and recall. If queries are not answered in a timely manner, users will become frustrated and dissatisfied with the system. As such, it is important to precompute as many of the processes needed to answer queries and store their results for quick use.

Following this assumption, CENTRIFUSER's document and composite topic tree transformation algorithms are coded and executed separately from the critical, time-sensitive online portions of the system. Figure 5.1 illustrates this division of labor. In a digital library employing CENTRIFUSER as one of its interfaces, each document would have a precomputed document topic tree stored alongside its vector space representation. Documents would be categorized by their text type (as defined in Chapter 4), such that each document would be assigned to exactly one text type. A composite topic tree would be compiled for each text type, either automatically from a representative set of documents belonging to its text type, or

semi-automatically, involving interaction with human editors.

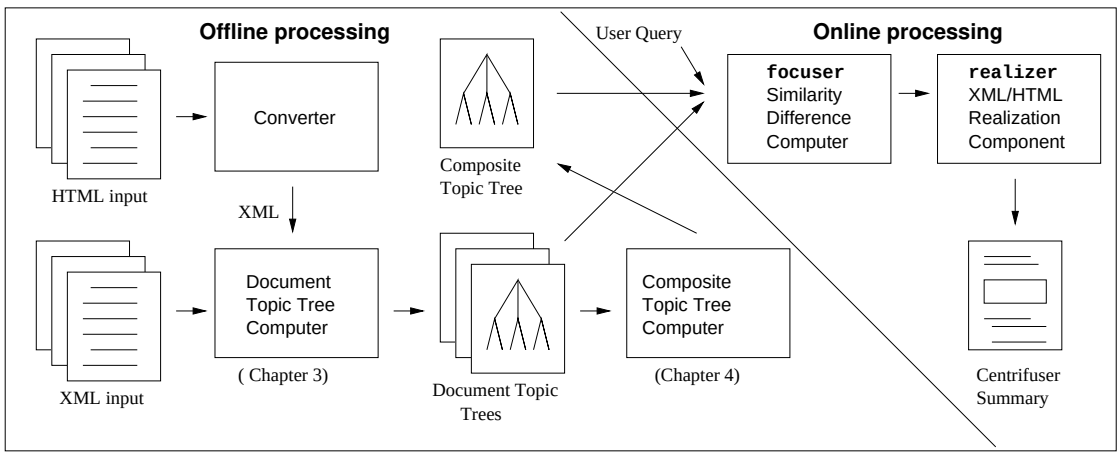


Figure 5.1: CENTRIFUSER's overall architecture.

As a user enters this scenario with a query, the standard IR framework uses the query as a document vector and retrieves the top n documents most similar to the query as its answers. When routed to the CENTRIFUSER user interface, the first online component, the similarity difference module – known as `focuser` – retrieves the corresponding document topic trees.

`focuser` first checks whether all documents belong to the same text type. CENTRIFUSER's current architecture only allows summarization of documents belonging to the same text type. If the documents belong different text types, they do not fit the input prerequisites of the system and are filtered out at this stage. If the document topic trees all correspond to a single text type, the corresponding composite topic tree is also retrieved and comparison processing begins with the process of query mapping.

5.2 Query mapping

Given document and composite topic trees, CENTRIFUSER links the user's query to the most similar topic in each document topic tree and to the composite topic

tree. This is done using the same similarity computation function as used in the construction of the composite topic tree, discussed earlier in Section 4.3.1. However, as the initial user query is a simple string, the metric cannot capitalize on tree structure similarity and reverts to simple word overlap with the topic's header string or with salient noun phrases contained within a topic's body. Currently, user queries are mapped to the best single topic node, and as such, future work that extends the framework to allow fractional, probabilistic query topic assignment is needed to handle more complex queries.

Equipped with these three pieces of information – document topic trees for each document in the result set, a composite topic tree for the text type, and the query mapped to a specific topic node in the document and composite trees – `focuser` is poised to produce the summary components. The components themselves are devised to explicitly support Strategy 4 (form a user model) and support the two main information seeking strategies of browsing and searching. I now examine how `focuser` handles each of these two user types in turn, by first covering how CENTRIFUSER supports browsing, and then searching.

5.3 Supporting browsing with navigation links and extracted similarities

Browsing is a methodology for exploring an information space. It can be utilized by an information seeker examining a new area or domain or as a method of navigation to find related topics and explore tangential areas. Browsing gives the information seeker explicit options on what to do next (e.g., click on a hyperlink or look at books on the same shelf), rather than place the responsibility of the information seeker to express his information need to the system. As such, it places less cognitive load on the user (Marchionini, 1992) and is an ideal methodology for navigating highly

frequented routes of information seeking.

Unlike the standard ranked list, the `focuser` module of CENTRIFUSER actively supports the browsing model, by giving the user access to two different components that divide his needs of into two areas:

- The navigation bar represents the corollary to a reference librarian's cooperative responses, as discussed earlier in Section 2.1.
- The synopsis gives an overview of the query topic, presenting general, high-level information about the topic from information culled across the result set.

Both components are shown below in Figure 5.2.

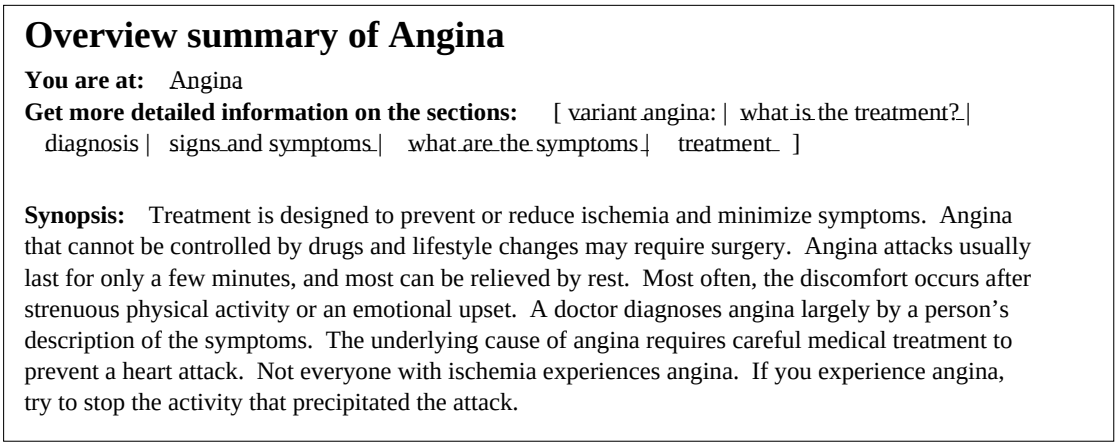


Figure 5.2: CENTRIFUSER's browsing components: navigational links (top); bottom; multidocument summary (bottom).

5.3.1 Navigation links

The first half of the browsing component is the navigation bar. The navigation bar allows the user to browse topics related to his original query by clicking on

its hyperlink. This causes an iteration: the query is sent to the IR framework to request documents on the new query which is then re-displayed by CENTRIFUSER.

After mapping the query to a topic in composite topic tree, its construction is straightforward. The query topic in the composite topic tree defines a browsing scope, a region of topics local to the query which are instantiated as browsing targets. Figure 5.3 illustrates this browsing scope with the dotted outline. In `focuser`, the browsing scope is defined as all the children of the parent of the query topic. These topics can be categorized with respect to their position to the query topic itself: a) the *parent* topic is the immediate ancestor of the query topic; b) the *sibling* topics are children of the parent topic, and represent closely related topics to the query; c) the *children* topics of the query topic represent sub-facets of the topic in question. Finally, d) *children* of *sibling* topics are ignored by the system.

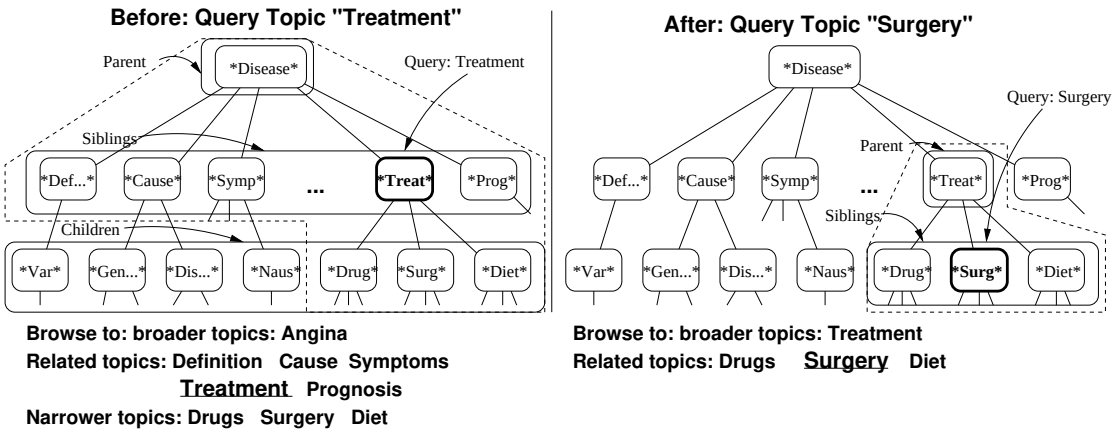


Figure 5.3: Navigation browsing scope, indicated by the dashed outline, as illustrated before and after browsing to the "Surgery" topic. Resulting navigation bar shown below the tree.

These first three types of topics represent logical browsing moves that are present in any ontology. A move "up" to a parent topic represents generalization (to a broader topic, BT, in terms of thesaural terminology); a move "down" to a child topic represents specialization (to a narrower topic, NT); a move "across" to

a sibling topic represents navigation to related topic of the same scope (RT). Each of these three types are mapped to separate text widgets placed in the navigation bar. When a user activates such widget (e.g., by clicking on a related topic), the activation is interpreted as a new query to the system and the textual information related to the topic is sent to the IR framework to retrieval documents and re-invoke CENTRIFUSER, with a new query topic representing the user's step to the new topic. In this fashion, it is possible to navigate to all topics within a composite topic tree through the navigation interface (which may retrieve different top documents to send to CENTRIFUSER).

The construction and display of the navigation component is simple enough when the composite topic tree is small and the screen is large enough to fit all of the topics. In practice and in supporting navigation on small displays, there are too many candidate topics to display, and the system must pick a limited number. The composite topic tree contains typicality information that reduces this problem to a simple, principled process – `focuser` chooses to display only the top n most typical topics. I modified this basic strategy to weight topics "closer" (i.e., sibling topics that are immediately previous or next and children topics that are direct descendants) heavier than more distant topics. In this manner, `focuser` imposes a ranking on the topics within the browsing scope that can be truncated as space allows. The final placement of the text widgets representing each topic is facilitated by its tree structure information in the CTT, such that any expansion in the available space for the navigation bar inserts topics in their appropriate position. Figure 5.4 shows this process in action.

5.3.2 Informative synopsis based on similarities

The user model of browsing also implies presentation of information at a high level of detail rather than a presentation of many fine-grained facts at a low level. To

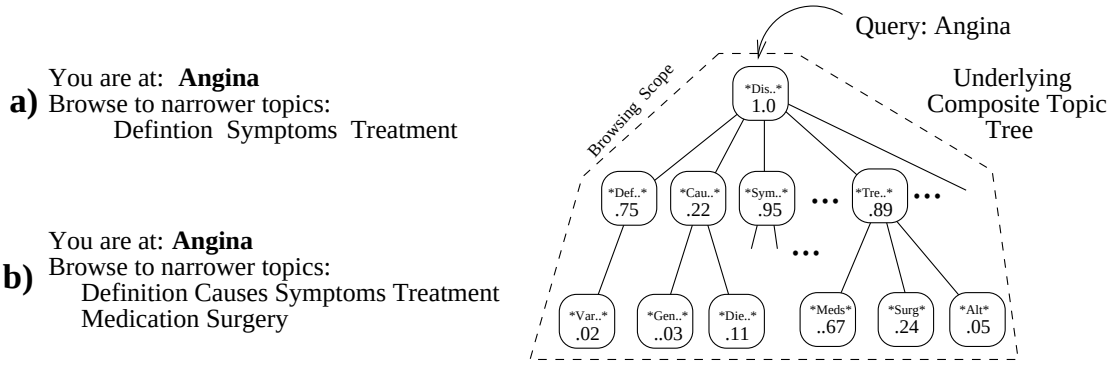


Figure 5.4: Navigation control construction under different space limitations: a) with less space, b) with more space.

address this need, `focuser` creates text that overviews the query topic, taking common information across documents. The idea here is that common information is of more importance and of a higher level of detail, similar to assumptions made in other multidocument summarization agents (Barzilay, McKeown, and Elhadad, 1999; Mani and Bloedorn, 1999; Monz, 2001).

The browser's synopsis is extracted from the text of the individual documents in the result set; its length in sentences is determined by a user-controlled parameter (or by constraints of the computing platform, see Section 5.6). Similar to the browsing scope used in the construction of the navigational links, the query node is used here to establish a scope of relevant topics in the composite topic tree. Child topics within a set depth k away from the query node form the scope of relevant topics. "Relevant" is interpreted as relevant to the production of a multi-document summary; non-relevant topics are either too *intricate* in detail for use in the synopsis (over k deep descendants from the query node) or just *irrelevant* (outside the subtree defined by the query node). These relationships partition a topic tree into these three (possibly empty) regions, as shown in Figure 5.5.

To build the extract, appropriate text must be chosen to represent each relevant topic. In our implementation, I use the method of sentence extraction (Paice,

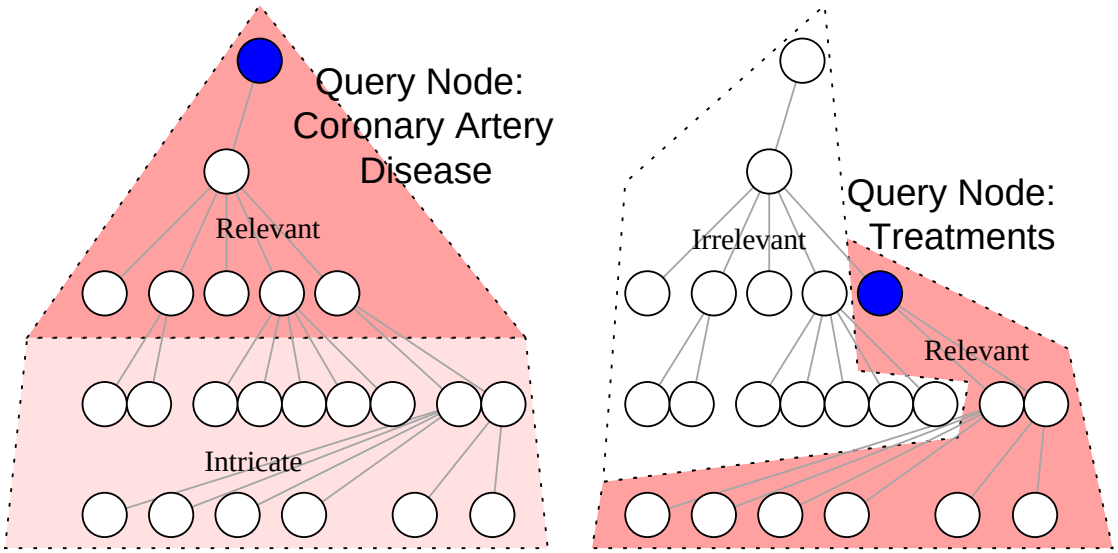


Figure 5.5: A pictorial representation of how *relevant*, *irrelevant*, *intricate* topic types are defined by the interaction of a document topic tree and two different queries, for k=2.

1990; Mani and Maybury, 1999), which is well-accepted since it is simple, fast and easy to evaluate. In this technique, sentences from the original documents are selected and put together to form a summary. The novel contribution of CENTRIFUSER in forming the multidocument summary is in how the algorithm capitalizes on the structural information given by the topic tree.

There is no actual text corresponding to the composite topics; rather, the text resides in the individual documents. To locate appropriate text, CENTRIFUSER maps the topics in the composite topic tree to the individual document trees, in the same method used to map queries to topic trees and in merging nodes in the construction of the composite topic tree. Figure 5.6 shows how the composite topic tree might be mapped to a specific document topic tree. However, it is important to note that the mapping process does not define a one-to-one mapping on all nodes in the composite and document trees. Two cases of failure can occur: 1) a composite topic does not have a corresponding instance in the document topic tree and 2)

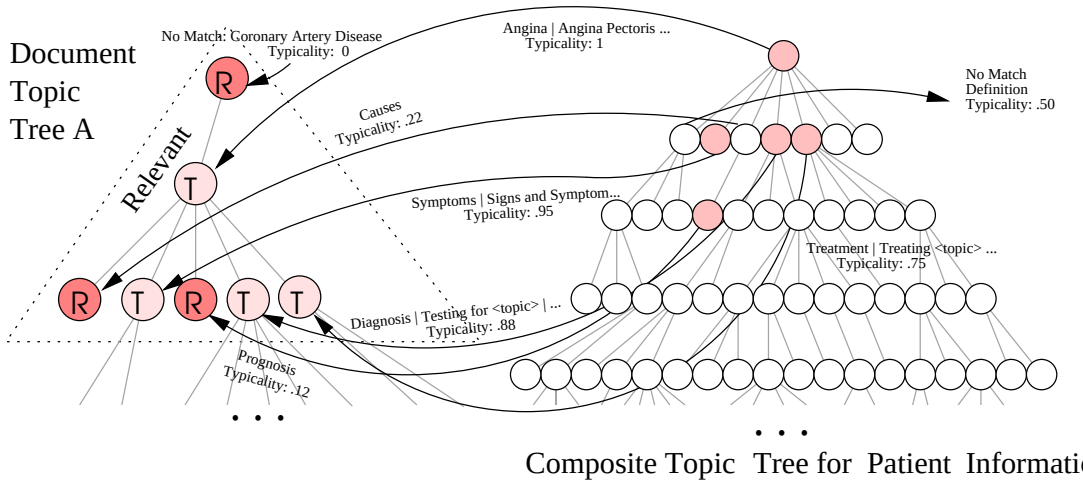


Figure 5.6: Aligning topics from the composite tree and individual topic tree. Typicality scores are then propagated across. "R"are and "T"ypical topic labels used by the indicative summarization process, discussed later in Section 5.4.3.

a topic in an individual document topic tree is not represented in the composite tree. On the right hand side of the figure, one can see that the composite topic "Definition" is not mapped to any topic in the document topic tree. This can occur if the document does not have information corresponding (aligned to) this particular topic (e.g., a document that discusses mostly treatment options may not have any information on the symptoms of the disease). On the left hand side of the figure, the document topic of "Coronary Artery Disease" is not mapped to a composite topic. This can occur if the composite topic tree was compiled without using this specific document or if the automatically compiled composite tree was edited.

If a composite topic does not map to any document topic tree, it is without any textual instance and cannot be summarized; this topic cannot be included in the synopsis since there is no text to represent it.

Once the mapping is complete, the browser's extract is created by the `realizer` module in three steps:

1. dividing the summary's allotment of sentences among the topics that are relevant, and instantiated by physical text;

2. selecting the sentences in the physical text;

3. ordering the selected sentences into an extract.

Similar to the space allocation problem in the navigational links, the space allocated to an extract is often too short to encompass all the possible relevant, instantiated topics. To fairly divide the summary's allotment of sentences the system imposes ordered ranking of the topics using their typicality rating, as done in the navigation bar construction process. In the composite topic tree in Figure 5.6, the topic "Causes" has lower typicality than "Symptoms" and "Treatment", and thus is more likely to be omitted over other more common topics when space is limited.

To assign an allotment of sentences to each topic, `focuser` follows a "card dealing" algorithm, with a sentence being analogous to a card, and topics players. Topics are sorted in order of descending typicality. Each topic is dealt a sentence until the sentence quota for the synopsis is exhausted. This approach ensures that the highest possible breadth in topics is covered within the sentence quota, and that the quota is used on the most typical topics first. For example, in Figure 5.6, six topics were aligned. Given a seven sentence synopsis, "Symptoms" would receive two sentences whereas "Prognosis" would receive only one.

Once a topic receives a sentence allotment, the system must choose the sentences to represent it. Since similar sentences may come from different documents and may contain redundant information, it is important to cluster them and chose a single representative sentence for the cluster to eliminate redundancy, similar to the goal of Maximal Marginal Relevance (Goldstein, 1999). To perform this task, I utilize a sentence clustering technique (McKeown et al., 1999; Hatzivassiloglou et

al., 2001) that takes as input a set of sentences and organizes them into clusters based on their sentential similarity. For each topic, the clustering program is run on the sentences of the topic instances, producing clusters of similar sentences as output, and a single sentence is chosen to represent each cluster.

The system re-ranks the clusters according to the number of different individual documents represented and resolves ties by size (number of sentences found to be similar). For example, if the texts for "Symptoms" result in two sentence clusters, one with sentences from three different documents, and the other with sentences only from a single document, it would select sentences from the cluster that represents information found in the three document cluster first. This heuristic reflects a bias to select information that is present across different sources.

The cluster's representative sentence is chosen based on a cascade of rules. The system prefers sentences from paragraph text, over list items or bullet points, over section headings. If sentences are of the same type, the sentence that occurs earlier in its instance text is chosen. If the sentences occur in the same location and are of the same type, the one that is closer to average in sentence length (in words) is chosen. These heuristics are used to select a sentence that is more likely to be fit for inclusion in a summary: sentential prose text (as opposed to solitary noun phrases common in headers) that do not have anaphors (which occur less at the beginnings of articles and paragraphs) and have ordinary length (too short or long sentences often have other language artifacts such as discourse cues and phrases that may not fit well in the summary (Schiffman, Nenkova, and McKeown, 2002)). In fact, as topics can be assigned a different number of sentences (i.e., clusters), they are independently scalable, a feature that is useful and discussed later in Section 5.6.

The final subtask is to order the selected sentences into a summary. CENTRIFUSER does this by first ordering the selected topics and then internally ordering

each topic's sentences. Topics are organized by their most prominent ordering found in the composite topic tree (e.g., "Symptoms" before "Diagnosis" before "Treatment"). Within a topic, sentences are ordered by their physical position. Sentences that come earlier in their instance text are positioned first.

This process results in an ordered extract, which is used as the browser's synopsis. The text represents an instantiation of the composite topic tree, made from component sentences found in the documents of result set. By choosing breadth over depth in allotting sentences, `focuser` creates an overview that surveys as many topics as possible. By clustering similar sentences and using only a single sentence per cluster, the system attempts to eliminate redundant information. When given more space, `focuser` attempts to cover more topics in the extract first before trying to include more detail about each topic. This makes the extract an informative summary, since it just covers the most important points, in order of typicality. Unlike an indicative summary, it gives a miniature version of the entire document, excerpting a main idea from each topic. The contribution of this part of the work is in providing a flexible framework that scales well for documents with many or few topics and for long or short summaries.

5.4 Supporting searching with generated indicative differences

Searching represents the other major information seeking paradigm. Searching is often facilitated by letting the user make his information need known to the system. In the IR interface, this is usually done by providing a text box, in which the user can type in their query. In describing the result set returned to the user, a system that explicitly supports search needs to highlight information about the documents pertinent to the query.

To support searching, CENTRIFUSER's display of the result set began as a simple listing of text extracts of each document. To better support the needs of the searchers, this display has since evolved to a list of document clusters summaries that use natural language generation rather than extraction, which differentiate themselves from each other by reporting salient and unique topics and metadata. These indicative summaries are a major focus of this thesis, and the algorithms to support this work are overviewed in this chapter. Detailed work concerning the distribution of metadata information in the summaries is given in Chapter 6 and work on generating these summaries based on corpora is the focus of Chapter 7.

In the next two subsections, I will show how I rationalized and developed the form of these summaries by examining how the single document summarization (i.e., sentence extraction) task was adapted to the online IR scenario to handle multiple documents and take interactive queries into account. I will then describe CENTRIFUSER's algorithm for generating these summaries at a high level in the remainder of this chapter, and will return to low-level details of the algorithm in the latter half of Chapter 6.

5.4.1 Generalizing to multiple documents

In the online information retrieval scenario, any system such as CENTRIFUSER has to handle multiple documents. Thus, I needed to extend my study to cover summaries over multiple documents. To do this I examined prescriptive guidelines for multidocument summarization.

The Open Directory Project's (an open source Yahoo!-like directory) editor's guidelines (2000) states that category pages that list many different websites should "make clear what makes a site different from the rest". "The rest" here can mean several things, such as "rest of the documents in the set to be summarized" or "the rest of the documents in the collection". This rule of thumb is familiar – it is a

rephrasing of Characteristic 2 (Differentiate items shown) – from Chapter 2, page 261. I recast this characteristic in terms of the text summarizer's goal as Rule A:

A. for a multidocument summary, an indicative text summarization agent should report differences in the document that deviate from the norm for the document's type.

This suggests that the summarizer has an idea of what values of a document feature are considered normal. Values that are significantly different from the norm could be evidence for a user to select or avoid the document; hence, they should be reported. For example, consider the document feature of length: if a document in the set to be summarized is of significantly short length, this fact should be brought to the user's attention. In fact, CENTRIFUSER has access to information about the norm from the composite topic tree. As the composite topic trees are calculated from only documents of identical text type (documents of the same domain and genre), it can model different value thresholds for different kinds of documents. In this way, the system can discriminate between "long" for consumer healthcare articles (over 10 pages) versus "long" for medical textbooks (over 400 pages).

5.4.2 Generalizing to interactive queries

Another facet that differs in online IR summarization compared to standard single document summarization is that the system needs to support interactivity and handle queries. As previously mentioned, the standard ranked list does this often by highlighting query terms and by providing the context around a query term. Generalizing this behavior, one arrives at Strategy 3 (Evaluate documents for initial relevance to the query), again recast here as rule of thumb B:

B. for a query-based summary, an indicative summarization agent should highlight differences that are relevant to the query.

This suggests that the query can be used to prioritize which differences are salient enough to report to the user. The query may be relevant only to a portion of a document; differences outside of that portion are not relevant. This is close in spirit to Grice's Maxim of Relevance (Grice, 1975). For example, in the consumer healthcare domain, a summary in response to a query on treatments of a particular disease may not want to highlight differences in the documents if they occur in the symptoms section. Keeping these two rules in mind as motivation for the structure of the indicative summary, I now examine the algorithm for generating the clustered, indicative summaries used to support searching in CENTRIFUSER.

5.4.3 Algorithm overview of CENTRIFUSER's indicative group summaries

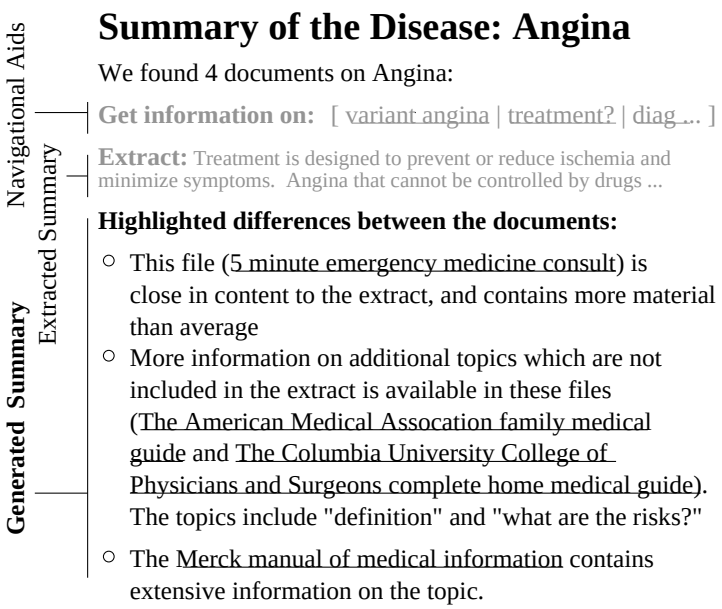


Figure 5.7: The sample summary from Figure 1.1, highlighting the generated indicative summary in the bottom half categorizes documents by their difference in topic distribution.

Figure 5.7 shows the resulting clustered document summaries formed by

CENTRIFUSER. They are similar to previous work in document clustering for IR (Nowell et al., 1996; Benford et al., 1995; Zamir and Etzioni, 1998; Chen and Dumais, 2000), but differ crucially in that they are formed using natural language generation technique (i.e., they are not formed using extracted text from the documents) and that their form takes into account how the query topic relates to the individual documents. Thus, the cluster descriptions are summaries that identify what topics are unique to the clusters and point out salient metadata characteristics about certain documents. Within the context of indicative multidocument summarization, it is important to show the differences between the documents (Rule A) and their relationship to the query (Rule B). One way to do so is to classify documents according to their topics' typicality and relevance to the query.

Thus, the text for the searcher's differences is created primarily using topic information from the document topic trees. The distribution of the topics within each document allows us to categorize each in a meaningful way for the searcher to pinpoint which document may be useful to retrieve. Some documents will be more relevant for specific searches and less relevant for others. For example, a document that specializes in treatments will be useful for the patient looking for the side effects of certain drugs, but may be useless for another person who is unsure of whether she has angina, and is interested in ways to diagnose the disease.

This type of query interaction can be modeled by examining the query nodes in relation to the individual document topic trees. In each document, the query node defines three regions, shown earlier in Figure 5.5: nodes that are relevant to the query, ones that are too *intricate*, and ones that are *irrelevant* to the query, as used in the construction the browser's synopsis. Each individual document's ratio of topics in these three regions helps the system assess the document's importance. In the earlier example, a document mostly on treatments would consist mostly of relevant topics to the treatment query but would consist mostly of irrelevant topics

to the diagnosis query.

Some documents will also be more interesting than others according to the type of relevant information they possess. For example, if "Prognosis" is a rare topic to find in an angina document, it may be worthwhile to report to the searcher in case they are looking specifically for this hard-to-find information. To model this aspect in CENTRIFUSER, typicality values from the text type's composite topic tree are utilized (recall that all of the documents belong to the same text type in CENTRIFUSER). The relevant topics in each document topic tree are each mapped to a composite node, if possible. The typicality score is inherited from the composite topic, or if no mapping was possible, it is considered unique (0 typicality). For convenience, the system has a threshold α, above which a relevant topic is *typical* and below which it is considered to be *rare*. Thus, a document that has many rare topics, such as "Prognosis", can be reported to the searcher as criteria for retrieval.

5.4.4 Refining topic types

To group documents along dimensions that fulfill Rules A and B, CENTRIFUSER uses information about each document's topics. Each document's ratio of topics in these three regions (relevant, intricate, and irrelevant) thus defines its relationship to the query: a document with mostly information on treatment would have a high ratio of topics in the relevant region to other topics if given a treatment query; but the same document given a query on symptoms would have a much lower ratio. Categorizing documents by the ratio of these three regions thus fulfills Rule B (highlight differences relevant to the query).

To fulfill Rule A, one needs to know whether a particular topic "deviates from the norm" or not. I interpret this as whether or not the topic normally occurs in similar documents – exactly the information encoded in the composite topic tree's

typicality score (see Chapter 4). As each topic in the document topic trees is an instance of a node in the composite topic tree, each topic can inherit its composite node's typicality score.

5.4.4.1 Document categories

The distribution of these four topic types – *rare*, *typical*, *irrelevant* and *intricate* – classify the document into a distinct document category. In developing the categories, I assumed that the most important topics are those that are most related to the query topic. Thus, the document categories consider the topic distribution in descending order of relevance to the query: first rare and typical topics, then intricate ones, and finally irrelevant topics. I explain the categories below and give specific details on the threshold set to detect them in Table 5.1. The examples in the list below pertain to a general query of "Angina".

Document Type	Topic Distribution	Description
Prototypical	*typical* >= 50% *typical* >= 50% of in-scope prototype topics	The typical document, which is well represented by the extract
Comprehensive	*typical* >= 50% of in-scope prototype topics	Contains more than just the typical topics
Specialized	*typical* >= 50%	Contains some of the typical topics
Atypical	*rare* >= 50%	Contains rare information
Deep	*intricate* >= 50%	Contains content that is too detailed for this query
Irrelevant	*irrelevant* >= 50%	Contains mostly information outside query focus
Generic	*n/a*	Contains a mix of topic types, no strong trends

Table 5.1: Conditions used to categorize documents into document types.

1. **Prototypical** - This kind of document has a topic distribution that matches the distribution of topics in the composite topic tree. This is interpreted as

two symmetric relationships. 1) Most of the typical topics in the composite topic tree are present as topics in the document. 2) Its relevant topics are mostly ones that are listed as typical in the composite topic tree. An example would be an average document about angina – *American Medical Association's Guide to Angina.*

2. **Comprehensive** - If only the first requirement of the prototypical document type is met, then the document has typical content but also contains other topics. The document thus covers more topics than usual, and is usually longer than other documents. An example of a comprehensive document could be a chapter of a medical text on angina.

3. **Specialized** - On the other hand, if only the second requirement of the prototypical document type is met (that its relevant topics are mostly typical), the document treats only a portion of the normal amount of typical topics relevant to the query. These documents specialize in its typical topics. A specialized example might be a drug therapy guide for angina.

4. **Atypical** - An atypical document (characterized by many a high rare-to-typical topic ratio) contains information that may relate the document's text type to other text types (interdisciplinary), or may contains information on special cases. If the topic "Prognosis" is rare, then a document about life expectancy of angina patients would be an example.

5. **Deep** - These documents are often barely connected with the query topic but have much underlying information about a particular subpart of the query. An example of this type is a whole document on "Treatments of Angina" when "Angina" is the query node and "Treatments" registers as a kth level topic (k again is the beam depth from the query node for which topics under k levels away are consider relevant).

6. **Irrelevant** - An irrelevant document contains a high irrelevant-to-relevant (= rare + typical) ratio of topics. The text contains information about the subject in question, but not in the particular area of interest. A document about all cardiovascular diseases that mentions angina briefly may be considered irrelevant.

7. **Generic** - These documents do not display tendencies toward a particular distribution of information.

Since the criteria for these categories are not mutually exclusive, the system apply the first applicable category to the topic. For example, if a particular document is comprised of 50% rare topics and 50% irrelevant topics, it would reported as an atypical topic.

Once each document has been categorized, CENTRIFUSER uses natural language generation in a realization module (`realizer`) to generate a short description of each document type category that contains at least one document. As with the extract portion of the summary, the length of this description text is controlled by the user. The generated descriptions vary in content according to the number of documents placed in the category. As the number of categories are limited (there are only seven) and since only a single description is generated per category (regardless of the number of documents belonging to it), it is possible to compress a query result set of many documents onto a single screen.

To decide exactly what information to generate in the textual description, I conducted a study of indicative online public access catalog summaries from the Library of Congress, described in further detail in Chapter 6. The main result is that topical information was most important, leading to design the description with obligatory topical information but having optional information about other document features if space allows (e.g., *Content Types* "does the document contain pictures or tables" or *Audience* "does it target medical students").

The generation of these indicative summaries has gone under two phases of development. In the initial development of CENTRIFUSER, a weighted context free grammar was used to generate the summaries. The exact algorithm will be explained in Chapter 6. The study of such human written summaries showed much variability in grammar and lexical choice, and thus further work on a more flexible generation system that would alleviate the repetitiveness of the text generated by a small grammar was needed. I developed a corpus-trainable system, which mimics the variability of the summaries. This augmented indicative summary generation module will be explained in detail in Chapter 7.

5.5 Time complexity of CENTRIFUSER

As stressed at the beginning of this chapter, CENTRIFUSER's offline processing allows much of the computationally expensive processing to be precomputed offline (e.g., computation of composite and individual topic trees). Thus, the time efficiency of the system is dependent only on the two online processing modules as shown earlier in Figure 5.1: in the similarity and difference computation of `focuser` and in the translation of these differences into natural language summaries for the differences and in extracting appropriate sentences for the similarities, done in `realizer`.

The topic similarity and difference computation done in `focuser` requires a mapping of the query to each document. This requires a similarity computation of each topic in each document. This operation has time complexity O(mn), where m is the number of topics per document and n is the number of documents. Once query mapping is completed, topic are assigned to one of the four types, which requires another inspection round of each topic – again O(mn)[1]. Census statistics

[1]In practice, the number of topics, m, is on average less than twenty, thus the time complexity is not generally quadratic.

necessary for categorizing each document into the seven document categories are also compiled during this second pass.

Taking the relevant topics and assembling a truncated list of topics for the construction of the navigation bar and extract is also linear in time complexity. However, sentence clustering (as done utilizing the SimFinder system (Hatzivassiloglou, Klavans, and Eskin, 1999)) is quadratic with respect to the number of sentences given, since pairwise comparison is needed. Rule based selection of the sentences after clustering is linear with respect to the number of sentences in the document.

For generating the indicative differences, identification of salient metadata and topics (to be discussed in Chapter 6) requires linear time, as their average statistics are stored in the composite topic tree. Computing salient aspects of a document category only requires a comparison between the document topic tree and the composite one. With a constant number of document categories and a closed (i.e., constant-sized) natural language grammar, the generation of the indicative group summaries requires constant time.

The computation efficiency of any process as it is scaled up is bounded by its most computationally expensive step. CENTRIFUSER's most expensive step in the online process is in its employment of the sentence clustering algorithm, which is quadratic in nature.

In practice, this turns out to be the case as well. Generating a five document summary with articles of average news document length (about 600 words each) takes about a minute; generation with ten documents takes about four minutes. These results are based on a 933 MHz Pentium 4 PC. Time profiling of the processes within CENTRIFUSER's online components confirmed that the majority of the execution time involves sentence clustering. Substituting the sentence clustering process and subsequent sentence selection process with a simpler process that

merely reports the n first sentences speeds up the CENTRIFUSER significantly. In this case, it takes a few seconds to produce the five-document summary and around ten seconds for the ten-document summary.

5.6 A note on support of level of detail

Throughout this chapter, I have stressed how each of the three components of CENTRIFUSER are independently scalable. This renders CENTRIFUSER's summaries applicable to a wide range of scenarios. If a user indicates that they are more interested in browsing, an application controlling the CENTRIFUSER interface can accordingly allocate more space to the browsing summary and less to the indicative results. Furthermore, as individual topics in the browser's extract and individual document categories in the searcher's differences are independent, they also can be independently scaled. This allows the creation of a fisheye user interface (Furnas, 1986); specific topics or document categories of interest can "grow" to accommodate a higher level of details and others not of interest can "shrink" to allow the space allotment for the entire summary to be conserved. This is the approach in independent user interface work being done to utilize CENTRIFUSER's results (Lok and Feiner, 2002).

5.7 Conclusion

In this chapter, I have shown how the document and composite topic trees can be used by a summarization system to better match the commonalities and navigational needs of the browser and differentiation needs of the searcher. Many search interfaces have document summaries that support one or the other, but relatively few have facilities that support both. For example, document summaries in the n top sentence style (e.g., AltaVista) present a high level overview of the document

suitable for browsers. A query word in context summary (e.g., Google) is more suitable for searchers. By supporting both major information seeking processes, CENTRIFUSER explicitly supports Strategy 4 (Form a user model). By placing both browsing and searching controls on a single interface, the system allows and encourages users to switch between these two information seeking modalities as needed.

CENTRIFUSER presents the user with a multi-document informative synopsis of the relevant documents as well as indicative qualities that differentiate them. The sentence extraction based synopsis provides users with high-level commonalities among the documents. Its aim is to provide a surrogate for retrieving an actual document for a browser's broad information needs. Navigation links also assist browsers in being able to select among closely related topics.

Topical indicative differences between the documents are also generated to differentiate the documents in terms of topical content as well as in terms of their metadata document features. The differences aim to assist the user in selecting the appropriate document. In the next chapter, I examine corpora of such indicative summaries and detail how the cluster summary organization and content are modeled after observation on these corpora.

Chapter 6

Indicative summarization

A primary focus of the CENTRIFUSER system is to provide searchers with indicative summaries that are well-suited to assist them in their searching task. In this chapter, I explore this goal in more detail, by first concretely defining what an indicative summary entails, then by examining existing indicative summaries, both in the form of prescriptive guidelines for creating summaries and in the form of summary corpora. In the second half of the chapter, I employ the knowledge gained from the first half of the chapter to detail the indicative summarization algorithm of CENTRIFUSER. I further show how the natural language generation components utilize the document categories presented in the last chapter to generate appropriate summaries.

6.1 Introduction

The distinction between indicative and informative summaries originates from the library science community (Cremmins, 1982), pp. 5-6. Cremmins focused on abstracts produced for experimental scientific articles. Thus, indicative abstracts summarize information about an experiment's purpose, scope and methodology,

but do not generally present results, conclusions or any recommendations. As such the indicative summary represents a biased summary, as it does not treat all parts of the document with equal importance. Rather, it highlights the aspects of a paper that would be of primary interest to a researcher looking for related work in their field. Parameters of an experiment (e.g., its scope and methods) with information on its findings can be used to determine relevance, but findings and conclusions without information on the experiment's scope and methods are useless, as one cannot figure out whether the findings are applicable to their particular situation. As such, an indicative summary that presents the parameters to an experiment may pique a scientist's curiosity to read the paper, but does not give away the author's findings.

In the literature, the indicative summary is often defined as a summary that "indicates what the document is about". I believe that this definition is too vague, and argue that the standard informative summaries also serve an indicative function. To make the distinction more clear, one can distill the indicative summaries to its core function. With respect to the IR context, the indicative summary is a summary that is *specifically tuned to help an information seeker perform a relevance judgment*. In contrast, an informative summary could be defined as a compression or summarization of a source text that treats all portions of the document with equal priority. This distinction implies that the information needed to make relevance judgments is not a balanced representation of the document.

The task of creating indicative summaries that help a searcher decide whether to read a particular document is a difficult task. This chapter examines the indicative summarization task from a generation perspective, by first analyzing its required content via published guidelines and corpus analysis. I show how these summaries can be factored into a set of metadata features, and how the implemented content planner in CENTRIFUSER uses the topicality metadata to create

indicative multidocument query-based summaries.

Automatic summarization techniques have mostly neglected the indicative summary (as I have defined it), in contrast to myriad techniques to build generic informative summaries. Indicative multidocument summaries are the most appropriate way of helping a searcher discriminate between the search results coming from an IR framework, and thus merit a closer study.

Traditional summarization systems are primarily based on text extraction techniques. For an indicative summary, which typically describes the topics and structural features of the summarized documents at a high level, these approaches can produce summaries that are too specific. I detail a natural language generation algorithm that creates indicative multidocument summaries for CENTRIFUSER. The module is based on metadata as per Characteristic 1 (Write descriptions to assist relevance judgment) in Chapter 2 (e.g., its distribution of topics and embedded media types).

Specifically, I focus on the problem of content planning in indicative multidocument summary generation. I address the problem of "what to say" in Section 6.2, by examining what document features are important for indicative summaries. By taking these guidelines into account, I will show how CENTRIFUSER uses the document categories based on topic distribution from the last chapter and natural language generation to produce its indicative summaries.

6.2 Metadata as potential summary content

Information about topics and document structure is often based on high level metadata. Such information typically does not occur in the document text *in situ*. My approach is to identify and extract the metadata that are relevant for indicative summaries. These features form the potential content for the generated summary and are represented at a semantic level in much the same way as input to a typical

language generator is represented. In this section, I discuss the analysis I did to identify metadata of individual and sets of documents that are relevant to indicative summaries and show how metadata selection is influenced by the user query.

Metadata can be divided into two simple categories: a) those which can be calculated from the document body (e.g., topical structure (Hearst, 1993) or readability using Flesch-Kincaid or SMOG (McLaughlin, 1969) scores), and b) features that may not be contained in the source article at all (e.g., author name, media format, or intended audience). To decide which of these features are important for indicative summarization, I examined the problem from two points of view. From a top-down perspective, I examined metadata standards and prescriptive guidelines for summarization and indexing. For the alternative bottom-up perspective, I analyzed a corpus of indicative summaries.

6.2.1 Cataloging metadata standards

Book catalogs index a number of different metadata in order to uniquely identify resources and provide enhanced search access. Structured metadata summarizes the identifying characteristics of a resource, and can be seen as a type of structured summary of a resource. There are several prescriptive guidelines for cataloging metadata that are used routinely in the library and information science community. The United States MARC format (2000), provides index codes for document-derived features, such as for a document's table of contents. Documentation on the MARC 21 format itself decomposes the approximately 200 "variable fields" into 14 broad categories. As MARC is a multipurpose format, the bulk of the variable fields assists the various library professionals in their tasks and are not meant to assist a non-specialist information seeker in determining relevance. For example, the format provides copious fields for unusual format, size, and special media, which follow from the inventory and identification aspect of bibliography.

In contrast, ANSI's standard on descriptions for book jackets and publisher's annual catalogs (1979) gives a more targeted 22-item list of bibliographic metadata to be used. It suggests best practices for the types of data to provide for different media (e.g., book jackets and catalogs) for different markets (e.g., institutional, educational, or mass consumer markets). An analysis of the guidelines show that the various markets and media are suggested to include mostly identical bibliographic data. These common fields identify the book by author, title, series and call number. A few fields vary among the different markets, for example educational markets need level indicators (e.g., "K-3") whereas institutional markets require Library of Congress call numbers (LCCNs) and Cataloging-in-Publication (CIP) information.

Metadata in the digital or electronic library can use the MARC format but a more simplified (and less specific) standard, the Dublin Core (Hillmann, 2001), has been adopted by many libraries and initiatives including the Open Archives Initiative (Lagoze and de Sompel, 2001). The Core offers 15 metadata fields to encode information about a resource. The fields are quite general (in comparison to the 200 fields of the MARC 21 standard) and reflect its generic purpose as a common denominator format, which includes uses in indexing worldwide resources in various electronic formats. There is work in interoperability between the standards, known as metadata "crosswalking" , such as projects to relate and translate between the Dublin Core and MARC standards (Guenther, 1997).

Table 6.1 summarizes and compares these various standards. It should be noted that this is not an exhaustive survey of metadata standards; in fact, there are many metadata standards that are used by various communities for various purposes (e.g., TEI, RDF, IAFA and GILS). Heery (1996) reviews these formats, and concludes granularity and linking (between different resources) are primary problems in metadata. Despite these problems in metadata exchange, this study then gives a general answer for the question:

General Label	MARC 21 categories (198 fields)	ANSI fields (22 fields)	Dublin Core fields (15 fields)
Unique identifier	0XX: control, number and code, classification and call number	10) ISBN, 19) CIP 20) LCCN	10) resource identifier
Author	1XX: main entry, 70X-75X: added entry	1) author	2) creator, 6) contributor
Title	20X-24X: title and title-related	2) title, 3) subtitle	1) title
Subject	6XX: subject access		3) subject
Edition	250-270: edition, imprint, etc.	8) Edition	
Series	4XX: series statements, 13) series added entry	13) series identification, 16) conference information	
Description	9) note	14) description, 17) pertinent information on authors	4) description, 8) type, genre, category, 11) source
Audience	521: target audience note	22) level indicator	
Coverage	522: geographic coverage		14) spatial or temporal coverage
Cross-reference	76X-78X: linking entry fields		13) relation
Holdings	841-88X: holdings, location, alternate graph, etc.		
Intellectual property		5) copyright date	15) rights management
Language	546: language note	14) translation information	12) language
Physical format	3XX: physical description fields	7) paging, volumes 12) special physical features, 18) size (if unusual)	9) physical or digital format
Price	3XX: physical description, etc.	4) list price	
Publication		9) publisher, 11) prepublication date	5) publisher, 7) date

Table 6.1: Metadata fields or categories in MARC, ANSI and Dublin Core standards.

Q1) Which types of information appear in catalogs?

Other questions are:

Q2) What information is included in textual indicative summaries?

and

Q3) What information is most helpful to searchers?

These three questions are related but their answers are different. Bates (1979) noted that the answer to Q1 was not the answer to Q3. She observed information seekers in their searchers and found that different cataloging metadata were not equally useful for information access. This is due to the different needs of the two primary directions of bibliography: access and identification, as detailed in Chapter 2.

6.2.2 Prescriptive summary guidelines

Answers to Q3 will vary depending on the task at hand and are hard to observe as the optimal information may not be provided or known to the searcher. I consider Q2 a better approximation to Q3 than Q1. That is, indicative summaries will better match the needs of information access than cataloging data in general. The contents of such indicative summaries and their ordering are variable and flexible to meet the needs of the intended user. Thankfully, the construction of such resources such as subject guides, online public access catalog summaries and annotated bibliographies is well-established and thus there is an established convention on the contents of the general-purpose bibliography.

Harmon (1989), pg. 107, gives a feel for some of these prescriptive guidelines. He lists criteria for compiling the annotated bibliography: "In the annotations, distinguish fact, opinion, conclusions judgment, inferences, and the like...Most subjects fields have style manuals and one likely exists for a field of interest. If not then there are many excellent style manuals of a general manual...Bibliography users expect an annotation to be an accurate and complete summary of the contents of an item. Direct value judgments such as 'a most important study' or 'a worthless commentary' should be avoided; these often indicate much more about the prejudices of the bibliographer than about the work being annotated...Each argument should be

given objectively with a minimum of judgmental interpolation. If you feel incapable of paraphrasing or summarizing an item accurately, rely on direct quotation to capture the thrust of an argument. Annotation need not be completely neutral, however. One can indicate the value of a publication with an overt judgment. Take advantage of the connotations of 'annotation verbs'."

I examined several descriptive guidelines that validate the above observations. Writing guides such as (Rees, 1970; Engle, Blumenthal, and Cosgrave, 1998; Lester, 2001; Anne Arundel Community College, 1998; Williams, 2002; Harmon, 1989) indicate specific types of information that should be included in annotated bibliographies; and are synopsized in Table 6.2.

	Ree70	EBC98	Les01	AACC98	Wil02	Harmon89
Accuracy/Currency		X		X		
Audience		X	X	X	X	X
Authority		X		X	X	
Cross-resource Comparison			X			
Contents			X		X	
Coverage	X			X		
Defects/Weakness			X		X	X
Navigation				X		
Purpose	X		X	X		X
Quality		X				
Relevance		X			X	
Subjective Assessment		X			X	X
Special Features			X	X		X

Table 6.2: Prescribed features of annotated bibliographies from several sources.

6.2.3 Descriptive summary corpora analysis

Inventorying the types of metadata recommended for summaries in guidelines approximates an answer to Q2. However, a more direct answer to Q2 is to statistically examine summary corpora themselves. To perform a detailed study of what information is normally present in such summaries, I examined two different corpora. In a preliminary study, I examined online public access catalog summaries, which I

discuss first. In a more comprehensive, follow-up study I examined a larger corpus of annotated bibliographic entries. In constructing the tagset, I initially used the tag set shown in Table 6.2. As I tagged the summaries it was apparent that the tagset was not comprehensive enough to catalog a variety of the data, and thus was further refined in the second study.

6.2.3.1 Online public access catalog summaries on consumer health information

Naturally occurring indicative summaries can also be found in library catalogs, since the goal is to help the user find what they need. I first extracted a corpus of single document summaries of publications in the consumer health literature text type from Columbia's local online catalog. The corpus contained 82 summaries, averaging a short 2.4 sentences per summary. Based on the previous metadata standards, I constructed a subset of fourteen metadata types that I used to characterize information in the summaries. Their percentage appearance in the corpus is presented in Table 6.3.

This first study reports results for a specific text type, but I felt that some general conclusions could be drawn. Document-derived features are most important (i.e., most frequently occurring) in these single document summaries, with direct assessment of the topics being the most salient. Other metadata features such as the intended audience, and the publication information (e.g., *Edition*) information are also often provided (91% of summaries had at least one metadata feature if they are assumed to be independently distributed).

6.2.3.2 Annotated bibliographic entries

To see whether the trend of the importance of different types of metadata would carry beyond the consumer medical information text types, I conducted another

Metadata	% appearance in corpus
Topicality (e.g., "Topics include symptoms, ...")	100%
Content Types (e.g., "figures and tables")	37%
Title	32%
Edition/Publication	28%
Author/Editor	21%
Readability (e.g., "in plain English")	18%
Purpose	18%
Audience	17%
Internal Structure (e.g., "is organized into three parts")	17%
Background	11%
Source	8%
Special Content (e.g., "Offers 12 credit hours")	7%
Media Type (e.g., "Spans 2 CDROMs")	5%
Conclusions (e.g., "based on a report")	3%

Table 6.3: Distribution of metadata features in library catalog summaries of consumer healthcare publications.

survey, this time with annotated bibliography entries. The collection of annotated bibliographic entries was drawn from the web and reflected a variety of domains. Two thousand entries were collected, the limiting factor being the time to manually delimit the entries. Of these, a random 5% (= 100) were analyzed and annotated for metadata. The details of the collection and annotation are detailed in Appendix E. I re-used the original 14 metadata features used in the library catalog summary work and further enriched the feature set to include 10 additional tags that better represented the range of information in the annotated bibliographic entries, giving a total of 24 tags. These tags were differentiated between topical metadata, detailed

in Table 6.4 and non-topical metadata, given in Table 6.5 and exhaustively covered the 100 entries, i.e., all text in the 100 entries had to be covered by a tag.

Metadata	# tag occurrences (tag frequency)	% entries possessing tag (document frequency)
Detail (Quotations, extracted sentences, parts of a chronology, conclusions)	139	47%
Overview (Generalized description of the entire resource, "This book is about Louisa Alcott's life.")	72	64%
Topic (High-level **list** of topics, e.g., "Topics include symptoms, ...")	34	28%

Table 6.4: Distribution of topical metadata in the 100-sentence annotated corpus.

6.2.3.3 Discussion

Different from the original library summary tagset, I divided the features into topically related and unrelated features. I distinguished between three different topically related features. *Overview* sentences usually begin the annotated bibliography entry and include a high level overview of the content of the resource. They appear in a majority of annotated bibliography entries and generally are limited to a single sentence. *Topic* features give a list of topics treated by the source, as an itemized or comma-delimited list. *Detail* sentences represent all other general item-specific sentences. In our observations across the 100 entries that were annotated, these sentences were the most variable in expressiveness, since they encoded domain-specific and resource-specific information. Short entries tended not to have any *Detail* tags, but as I examined entries of longer length, mostly *Detail* metadata fields were being added.

The data validates both prescriptive guidelines and the earlier work in showing that cataloger-provided metadata fields are important for summaries. *Audience* information, recommended by four of the five prescriptive guidelines, were shown to appear 12% of the time. Other metadata fields, such as *Purpose*, *Navigation*, *Subjective Assessment*, and *Readability* also play important roles.

Metadata	# tag occurrences (tag frequency)	% entries possessing tag (document frequency)
Media Type (e.g., "This book ...", "A weblet ...", "Spans 2 CDROMs")	55	48%
Author / Editor	43	27%
Content Types (e.g., "figures and tables")	41	29%
Subjective Assessment (e.g., "highly recommended")	36	24%
Authority	26	20%
Background (e.g., "based on a report")	21	16%
Navigation (e.g., "is organized into three parts")	16	11%
Collection Size	13	10%
Purpose	13	10%
Audience (e.g., "for adult readers")	12	12%
Contributor Name of the author of the annotated entry	12	12%
Cross-resource Comparison (e.g., "similar to the other articles"	10	9%
Size/Length	9	7%
Style (e.g., "in verse rhythm", "showcased in soft watercolors"	8	6%
Query Relevance (text relevant to the theme of the annotated bibliography collection)	4	3%
Readability	4	4%
Difficulty (e.g., "requires no matrix algebra"	4	4%
Edition / Publication	3	3%
Language	2	2%
Copyright	2	1%
Award	2	1%

Table 6.5: Distribution of non-topical metadata in the 100-sentence annotated corpus.

A noticeable difference between the earlier work on online public access catalog entries is that the *Title* field does not appear in any of the annotated bibliography entries. I surmise this is because its mention would be redundant, as the title is always given as text before the actual bibliographic entry. However, this is not true of *Author* information, as the metadata feature is often used to present the credentials of the author. In contrast, the online catalog entries did exhibit the *Title* field quite often. This may be because the catalog summaries were often book jacket or other related stand alone texts that may not have easy access to the bibliographic information.

6.3 Content planning in CENTRIFUSER's indicative group summaries

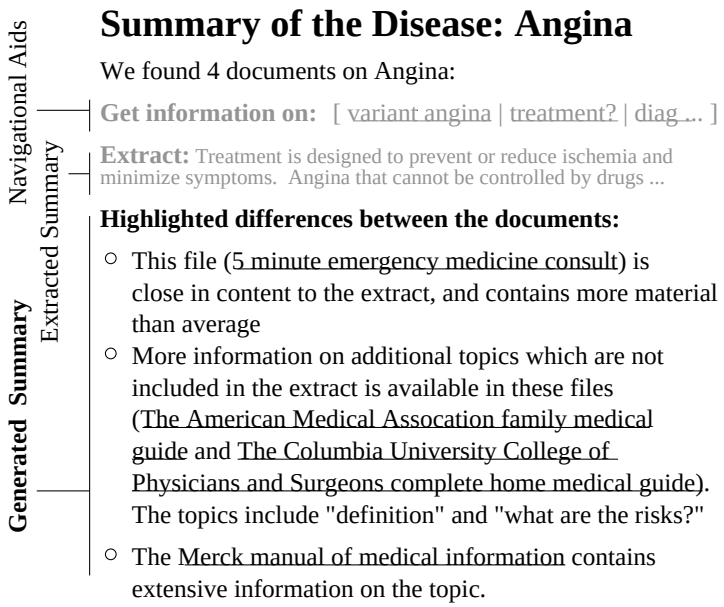


Figure 6.1: Sample CENTRIFUSER indicative group summaries, as shown earlier in Figure 5.7, repeated here for convenience.

I have described my study of indicative summaries and can now incorporate this knowledge into CENTRIFUSER. As described earlier, CENTRIFUSER uses natural language generation techniques to produce these indicative summaries rather than employing extractive methods. The summarizer's architecture follows the consensus NLG architecture (Reiter, 1994), including the stages of content determination and content planning. In this section, I follow the generation of a sample indicative multidocument query-based summary.

NLG systems traditionally have three components: content planning, sentence planning and linguistic realization. I will examine how the system generates such summaries shown earlier, recapped in the bottom part of Figure 6.1, by going over each of these three steps.

6.3.1 Content planning

The seven different document categories that were derived in the last chapter add a layer of abstraction over the topic types that allows the system to reason about documents. Figure 6.2 gives the different document categories used and the order in which information about a category is presented in the summary.

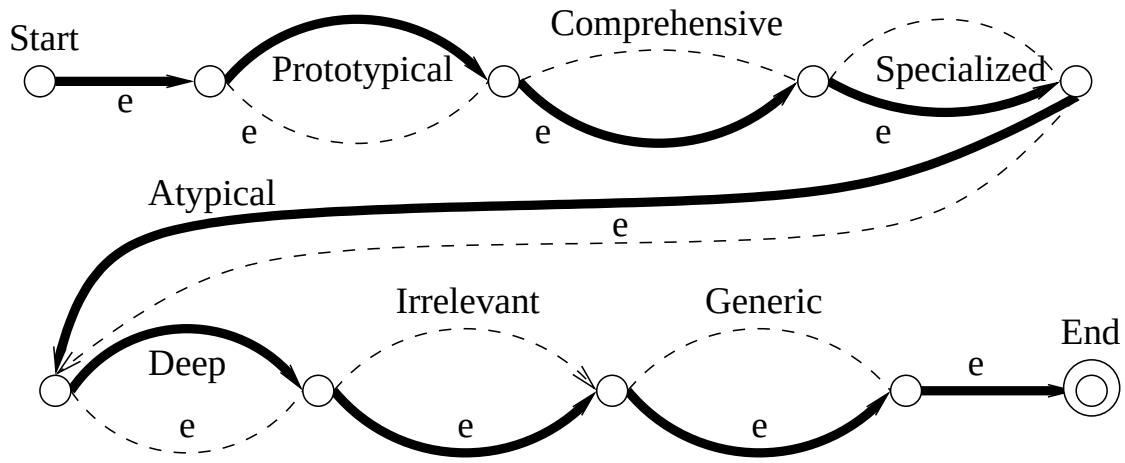


Figure 6.2: Indicative summary content plan, solid edges indicate moves in the sample summary in Figure 6.1.

In CENTRIFUSER, the text planning phase is implicitly performed by the classification of the summary document cluster into the document categories. If a document category has at least one document assigned to it, it has content to be conveyed. If the document category does not have any documents attributed to it, it is skipped using the ϵ arcs.

An instantiated document category conveys a couple of predicates. A description of the document type as well as the elements attributed to it constitutes the minimal amount of information to convey. Optional information such as details on the instances, sample topics or other unusual document features can be expressed as well.

The text planner must also order the selected predicates into a coherent plan

$$\left[\begin{array}{l} \textit{relation} : \text{setElements} \\ \textit{args} : \left[\begin{array}{l} \textit{docCat} : \text{atypical} \\ \textit{element} : \text{AMA guide} \\ \textit{element} : \text{CU guide} \end{array}\right] \end{array}\right] \qquad \left[\begin{array}{l} \textit{relation} : \text{description} \\ \textit{args} : [\ \textit{docCat} : \text{atypical}\ ] \end{array}\right]$$

$$\left[\begin{array}{l} \textit{relation} : \text{hasTopics} \\ \textit{args} : \left[\begin{array}{l} \textit{docCat} : \text{atypical} \\ \textit{topic} : \text{definition} \\ \textit{topic} : \text{what are the risks?} \end{array}\right] \end{array}\right]$$

Figure 6.3: Predicates instantiated for the *atypical* document category for the summary in Figure 6.1.

for subsequent realization. This is a problem on two levels: determining the ordering between the document category descriptions and determining the ordering of the individual predicates within the document category. In CENTRIFUSER's indicative summarization algorithm, the discourse plans for both of these levels are fixed. I first discuss the inter-category plan.

6.3.1.1 Inter-category plan

Document Category	Topic Distribution
1. Prototypical	$>$ 50+% *typical* and $>$ 50+% all possible *typical*
2. Comprehensive	$>$ 50+% all possible *typical*
3. Specialized	$>$ 50+% *typical*
4. Atypical	$>$ 50+% *rare*
5. Deep	$>$ 50+% *intricate*
6. Irrelevant	$>$ 50+% *irrelevant*
7. Generic	n/a (default)

Table 6.6: Document categories and their distribution of topic types.

CENTRIFUSER orders the document category descriptions based on the ordering given in Table 6.6, which recaps the document categories discussed in the last chapter. The rationale behind this ordering is reflected by the category's relevance to the user query. If one makes the simplifying assumption that information usually found in documents is information that is usually sought and searched for,

then documents with typical information are more likely to be useful to searchers. As such CENTRIFUSER orders the document categories by their relative closeness to the prototypical document, to explicitly organize the document categories to assist search. This organization of the documents thus fulfills Characteristic 3 (Organize listing to assist search).

Thus, document categories like *prototypical* whose salient feature is their high ratio of relevant topics, are considered more likely to be utilized by searchers than document categories that are defined by their ratio of intricate or irrelevant topics (e.g., *deep*). This precedence rule decides the ordering for the last few document types (deep $\prec$ irrelevant $\prec$ generic). For the remaining document types, defined by their high ratio of typical and rare topics, the system uses an additional constraint of ordering document types that are closer in content to the average document. This orders the beginning topics (prototypical $\prec$ comprehensive $\prec$ specialized $\prec$ atypical).

Figure 6.2 shows the resulting inter-category discourse plan. As stated in the text planning phase, if no documents are associated with a particular document category, it will be skipped, reflected in the figure by the ϵ moves. The sample summary contains prototypical (first bullet), atypical (second) and deep (third) document categories, and as such activates the solid edges in the figure.

6.3.1.2 Intra-category plan

Figure 6.4 shows the intracategory discourse plan. Ordering the predicates within a specific document category also follows a fixed ordering, enabling searchers to locate information rapidly as per Characteristic 9 (Use uniform descriptions). Obligatory information is expressed first, while optional information is expressed later if additional space is available for the group summary. The document category's identity in terms of its constituents and description always come first. As topical metadata

was found to be present in almost all of the summaries in both corpus studies, topic information is also considered obligatory and is realized second. Finally, other non-topical metadata can optionally be realized at the end of the category summary. This descriptive metadata section can be realized as many times as space allows and as long as there is salient metadata left to be communicated.

Salient metadata is determined by direct comparison to the composite topic tree. For numeric valued metadata (e.g., number of images), a salience score is determined based on the document set's average variance from the value in the composite topic tree. For set-valued attributes, the value of the metadata is considered salient if it differs from the value stored in the tree (e.g., if a document is written in "bulleted" form, when the composite tree says documents of its text type are usually written as "prose").

Each of the three sections of information corresponds to one or more predicates. The result is a partial ordering that is linearized later, as the order of the specific messages has not yet been fixed.

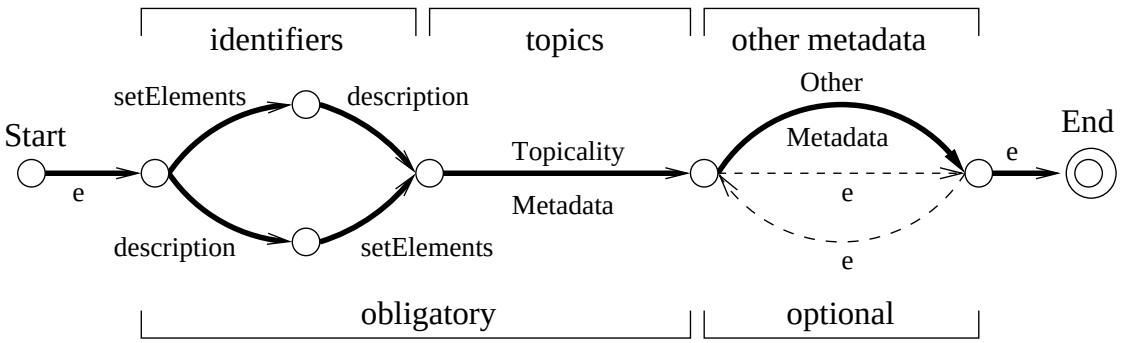


Figure 6.4: Intra-category discourse plan, solid edges indicate moves in the atypical document category. The final choice on which obligatory structure to use is decided later during realization.

6.3.2 Sentence planning and lexical choice

In the final step, the discourse plan is realized as text. First, the sentence planner groups predicates into sentences and generates referring expressions for entities. Lexical choice also happens at this stage. In this generation task, the grouping task is minimal; the separate categories are semantically distinct and need to be realized separately (e.g., in the sample, each category is a separate list item). Each of the three sections of a document category are realized as single sentences. In CENTRIFUSER's indicative summarization algorithm, no aggregation or combination operators were used to compact or increase the fluency of the resulting output.

6.3.2.1 Organizing metadata messages into sentences

Sentence planning within a document cluster summary is quite straightforward for the most part. Information on the document category is lumped into one sentence. However, as the metadata information (including topical information) can be repeated, it can contain multiple messages and the system needs to prioritize them and output them using a compact representation.

The problem here is that documents within the cluster can have different salient metadata. Figure 6.5 shows an example where a cluster has three documents with different salient metadata.

	Topics	Content Types	Length	Style
Doc 1	topics	normal	long	normal
Doc 2	topics	more images	normal	bulleted
Doc 3	topics	normal	normal	bulleted

Doc 1 has topics
Doc 2 has topics
Doc 3 has topics
Doc 2 has more images than usual
Doc 1 is longer than usual
Doc 2 is written in a bulleted style
Doc 3 is written in a bulleted style.

Figure 6.5: A document cluster consisting of three documents with salient metadata from a sentence planning viewpoint. Possible messages listed on the right.

If CENTRIFUSER is given two discourse units to discuss these metadata attributes, it is clear it cannot mention them all. However, it is possible to aggregate

related messages together to save space, in the style of Shaw (2002). The sentence planner within CENTRIFUSER implements a simple 1-distinct aggregation method that allows all messages pertaining to a single metadata feature or to a single document to be realized in one unit. Thus, CENTRIFUSER realizes a discourse unit to cover as many messages as possible, and does this iteratively given the space constraints the user or parent application. Figure 6.6 shows the same example, after selection.

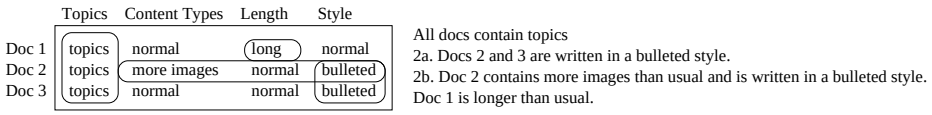


Figure 6.6: A document cluster with salient metadata with discourse units selected as ovals, as planned by the sentence planner. Possible aggregated messages on the right.

As CENTRIFUSER is scaled up to handle more metadata features, this will become a larger problem that will warrant more complicated aggregation routines (e.g., 2-distinct operations). This view of planning aggregation is akin to a well-known problem in hardware design, in which Karnaugh maps (Karnaugh, 1953) are used to visualize boolean functions. Karnaugh maps can thus be used to visualize 1-, 2-, 3- and higher n-distinct aggregation problems. This process is similar to the work done on difference generation in PEBA-II and ILEX systems (Milosavljevic, 1999).

6.3.2.2 Generating referring expressions

One concern for generating referring expressions is constraining the size of the sentence. This is an issue when constructing referring expressions to sets of documents matching a document type. For example, if a particular document category has more than four documents, listing the names of each individual document is not felicitous, as the document listing is too long. In these cases, an exemplar file is

picked and used to demonstrate the document type. This process is shown in Figure 6.7. If any realization of a document category needs to use this compaction operation to general its referring expression, the system must provide a means to recover the full listing of documents. In these cases, a link to expand all of the document categories is given at the head of the list of categories.

The *AMA family medical guide*, *The Columbia University home medical guide*, *The Angina Page*, *Information on Angina* and *Angina Pectoris Facts* have detailed information on particular subtopics of angina.

rewrite

There are 5 documents (such as the *AMA family medical guide*) that have detailed information on particular subtopics of angina.

Differences between the 10 documents:

rewrite

Differences between the 10 documents (show all the documents in the categories):

Figure 6.7: Compaction operation in referring expression generation.

Another concern in the generation of referring expressions is when the optional metadata information only applies to a subset of the documents in a category. In these cases, the generator may have to reorder the listing of the documents to make the subsequent referring expression more compact. Invoking this operation on list of two or three documents results in a reordering of the referring expression, as shown in Figure 6.8. In categories with more than four documents, it can cause a different exemplar to be chosen or the metadata clause to mention (and link to) a particular document by title if it is not the exemplar. This can occur when the summary length is long and many salient metadata attributes apply both to an exemplar document as well as other documents.

The first, third and fourth documents contain figures and tables as well.

rewrite

The first three documents contain figures and tables as well.

Documents *Angina Pectoris*, *Angina 1998*, *MedLinePlus - Angina*, and *Facts about Angina - NHBLI* are close in content to the above summary.

rewrite

Documents *Angina Pectoris*, *Facts about Angina - NHBLI*, *MedLinePlus - Angina* and *Angina 1998* are close in content to the above summary.

Figure 6.8: Reordering operation in referring expression generation in a document category consisting of four titles.

6.3.2.3 Lexical choice

Lexical choice in CENTRIFUSER is performed at the phrase level; entire phrases can be chosen all at once, akin to template based generation. In the initial realization algorithm, a path is randomly chosen to select a lexicalization. In the sample summary, the atypical document category's (i.e., the second bullet item) description of "more information on additional topics ..." was chosen as the description message among other phrasal alternatives. The sentence plan for this description is shown in Figure 6.9.

For certain document categories, a good description can involve information outside of the generated portion of the summary. For instance, Figure 1.1's prototypical document category could be described as being "a reference document about angina". But as a prototypical document shares common topics among other documents, it is well represented by the summary extract composed from the similarities across, detailed earlier in Section 5.3.2. As such, the system can use a phrasal description that directly references its results (e.g., in the actual description used for

```
(S1/description+setElements
  (V1 :value ‘‘be available’’)
  (NP1/atypical :value
     ‘‘more information on additional
       topics which are not included
       in the extract’’)
  (NP2/setElements :value
     ‘‘files (The AMA guide and
       CU Guide)’’))
(S2/hasTopics
  (V1 :value ‘‘include’’)
  (NP1/atypicalTopics :value ‘‘topics’’)
  (NP2/topicList :value
     ‘‘definition and
       what are the risks?’’))
```

Figure 6.9: Sentence plan for the atypical document category.

the prototypical document category in Figure 5.7).

6.3.2.4 Surface realization

Surface realization (or variously "sentence realization" or "linguistic realization") takes the sentence plan and produces actual text by solving the remaining morphology and syntactic problems. In the initial algorithm, CENTRIFUSER chooses a valid syntactic pattern at random, in the same manner as lexical choice. Morphological and other agreement constraints are minor concerns in this task and are handled by set rules. At this point, the sample summary has been fully realized.

6.4 Conclusion

At the outset of this chapter, I stated that indicative summaries are an important type of tool to help searchers perform relevance judgment and quickly locate the information needed. I have reviewed in brief three different approaches in catego-

rizing the information contained in indicative summaries: in metadata standards, in prescriptive guidelines for writing such summaries, and in analyzing the contents of indicative summary corpora.

In constructing CENTRIFUSER's group indicative summaries, summary content is based on document metadata that describes the document set's topic distribution with respect to the norm and to the query, rather than using extracted text from the documents. I use a simple text generation module that utilizes a text plan, derived from the sample corpora analysis, to guide the system in describing document topics as typical, rare, intricate, or irrelevant to the user query. Given this and other metadata about the documents, CENTRIFUSER is able to generate flexible indicative summaries for document clusters.

Chapter 7

Statistically informed generation for natural language

In Chapter 5, I introduced CENTRIFUSER's module for creating indicative group summaries. In the last chapter, I examined what types of data go into indicative summaries through corpus analysis. I also detailed the fixed, template-based algorithm that CENTRIFUSER uses to generate its summaries which was motivated by the corpus analysis. In this chapter, I examine how a new probabilistic, corpus-trained approach to generation produces single document indicative summaries and compare how this approach differs from the group summaries generated by CENTRIFUSER in both its content planning and surface realization aspects.

7.1 Introduction

Traditional natural language generation approaches rely heavily on human experts to code discourse, semantic, and lexical resources. These resources are used by systems to determine the structure of the text and word choice. This process is often resource intensive, requiring domain experts to expend time to distill proper

discourse plans and lexicons. The indicative summarization algorithm presented in the last chapter is an instance of this, as it uses a closed, fixed grammar and lexicon to realize the semantic predicates. The resources that CENTRIFUSER uses took approximately one person-month of time to assemble and correct.

Resource intensiveness aside, there are systemic shortcomings of employing a fixed, hand-coded content planner, grammar and lexicon. Subtle dependencies and preferences between different orderings of the predicates may be lost and the variations generated by the system are often limited and can result in stock repetition of phrases and hackneyed text. The lexicon also does not allow for flexibility in encoding other alternative phrasings that are found in natural indicative summaries that are missed by experts in their analysis. A system aiming to generate fluent summaries should attempt to model human authored summaries as close as possible.

In this chapter, I demonstrate a novel approach to automatically acquire such lexical and grammatical knowledge from an annotated training corpus and subsequently employ these resources to author new text. This method, implemented in the SIGNAL toolkit (for Statistically Informed Generation for NAtural Language), introduces a grammar and lexicon limited in variation only by the corpus it models. The system performs the generation of new expository texts by learning content planning and partial lexical resources from a training corpus. I employ SIGNAL to produce indicative summaries by using the corpus of annotated bibliography entries from the previous chapter. I will detail the steps that SIGNAL's algorithm uses to handle the application indicative summarization through examples.

In the first half of the chapter, I detail the resource acquisition process. I will first explain training corpus preparation and an experiment in automatic annotation of the training corpus, and then detail the two resource acquisition algorithms. The first module learns ordering and content planning constraints of elements in the text,

or *predicates*. It differs from related work in its handling of context. The second module learns partial surface realization patterns by separating predicates into two parts: *attributes*, which convey a predicates' semantics and *associated text* which conveys the information in the appropriate role.

In the second half of the chapter, I discuss how these raw resources are put to use in a regeneration phase to create new texts using a unification formalism. I first discuss background on generation using a unification approach and illustrate how the acquired resources are transformed into functional descriptions. The Functional Unification Formalism (FUF) of Elhadad (1993) gives the system the ability to express variations beyond the corpus through adjunction and coordination operations. I complete the chapter by showing how machine learning is employed to calculate the parameters of the fitness function employed to decide a final text. Figure 7.1 shows the architecture of SIGNAL as it is employed to create new texts.

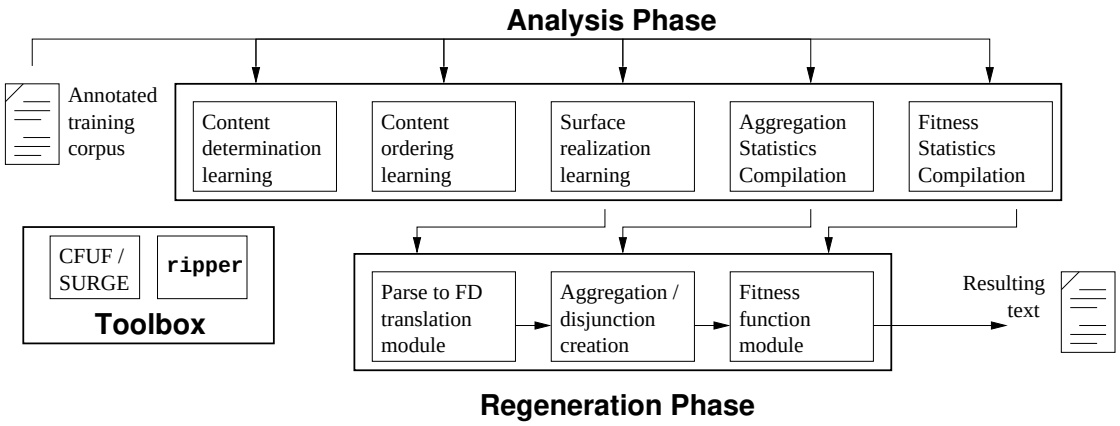


Figure 7.1: Overview of the SIGNAL architecture for text generation.

7.2 Training annotation of summary corpora

The SIGNAL algorithm produces new texts by mimicking the structure and style of a training corpus. To employ SIGNAL to generate indicative summaries, I used

the annotated bibliographic corpus from the indicative summary metadata study, analyzed for distributional properties in Section 6.2.3.2 and further detailed in Appendix E. Five percent of the corpus (100 entries) had already been exhaustively tagged, with a tag covering each word in every sentence. The 24 metadata elements are listed earlier in Tables 6.4 and 6.5. In the SIGNAL approach, these distinct metadata elements correspond to semantic predicates that are instantiated in each summary text. Annotation was done such that the entire predicate (attribute as well as associated text, when applicable) was tagged. Thus, in the sample sentence, "Intended for adult readers", the entire sentence was tagged as an *Audience* predicate, rather than just the attribute "adult readers".

7.2.1 Automatic annotation using machine learning

As with any corpus annotation, tagging these predicates is a time intensive task. Methods to automate tagging can save effort in annotation (Riloff and Lehnert, 1994) as well as help target specific predicates that need more examples to be more accurately tagged automatically (Lewis and Gale, 1994). Following the manual annotation procedure given in Appendix E, I annotated the 100 entry corpus with an average of 5.9 predicates per entry. This total broke down to 255 topical predicates and 337 non-topical predicates.

The 100-entry sample tagging allows the subsequent machine learning algorithms to learn to tag predicates automatically. To do this, the unannotated version of the hundred entries was parsed with Collins' dependency parser (Collins, 1996). The resulting parses were automatically aligned with the boundaries of the predicates in the annotated version. A large majority (over 95%) of the annotations matched constituents in the parse tree[1], and the annotations were assigned to the corresponding parse node. The failures that did occur in alignment mostly

[1]This is not to say that the 95% of all parse trees were correct, just that the constituent boundaries in the parse tree matched the annotation

happened because of missed or mis-recognition of sentence and clause boundaries on *Detail* and *Contributor* tags.

Using machine learning, automatic tagging assigns one of these 25 predicates (the 24 plus a default "no predicate") to each and every constituent in the parse trees. As most constituents don't match a predicate boundary, a majority class baseline would classify all constituents in the "no predicate" class. This gives a high precision and recall of 96.9% over all of the training instances (all 15208 parse nodes in the 100 tagged entries), but fails to recover any of the 24 semantic predicates (0% precision and recall). As our goal is to automatically tag these predicates, this method fails to help at all. To improve performance, I employed a machine learner, `ripper` (Cohen, 1995), to induce decision rules and used 5-fold cross validation to ensure results were stable across a variety of profiles of textual features, as shown in Table 7.1.

Precision/Recall	Feature Type		
	+ lexical	+ parse node & positional	+ contextual & genericity
24 metadata	77.7%/1.0%	76.8%/22.3%	74.9%/32.1%
3 topical	0.0%/0.0%	74.2%/52.8%	68.2%/59.0%
21 non-topical	77.7%/1.4%	83.0%/9.9%	84.4%/21.1%

Table 7.1: Summary semantic annotation accuracy over the 100 entries annotated corpus, using 5-fold cross-validation. Features are cumulative left to right.

The baseline scores were improved to 77.0% precision and 1.0% recall by using features that represent the set of words in the predicate. Recall was improved by including relative and absolute position in the summary as additional features. This resulted in a lower 76.8% precision but an increase to 22.3% recall.

7.2.1.1 A new feature: Genericity

The genericity feature is a new type of feature created in my study. It measures the extent of vocabulary variation in text, similar to the type/token ratio. The

lower the genericity score for a class of text, the more uniform the language use is in the text. I created this metric to measure the vocabulary variation in the texts associated with each semantic predicate. The idea is that the topical predicates that express domain-specific knowledge (e.g., *Detail*) would display a wider variety of vocabulary items across instances than non-topical predicates, such as *Audience*. These would display a more stable vocabulary profile (as any generic associated text would be the same across instances).

The genericity feature calculates the frequency of unigram, bigram and trigram open class word stems in the corpus. These frequencies are converted to a probability. For example, the bigram "new/JJ book/NN" occurs in the whole 2000-entry corpus seven times, out of all 523,459 total bigrams seen in the corpus, yielding a probability of $\frac{7}{523,459}$. A node in the parse tree is assigned a base genericity score based on the average probability of all available unigram, bigram and trigram information for it. The score captures whether the word sequences in the node appear often or not in the corpus.

As unigrams and n-grams in a language in general are not evenly distributed, a node may receive a high base genericity score because a) its words are frequent just within the annotated bibliography genre, or b) its words are frequent in the general language. As I wanted to specifically target the former factor, I introduced an extra factor to control for the latter, by penalizing words that are frequent across the English language in general. To do this, the unigram, bigram and trigram frequencies were calculated the Brown corpus, a corpus comprising a wide variety of English writing genres. These frequencies were converted into a penalty factor where the most frequent unigram, bigram and trigram are defined as having a unit penalty, and all other n-grams, a monotonically decreasing penalty ranging from 1 to 0. A modified genericity score is calculated, which adjusts for a word sequence's general English frequency, by multiplying the base genericity score by 1 minus the

penalty factor. In this way, a word sequence that is very common in general English will have a high penalty factor and its score will be scaled down significantly by the penalty factor, but a word sequence that never occurs in the generic Brown corpus receives its full base genericity score as its modified score.

These features were introduced along with features that modeled local context of the preceding and succeeding predicates. The results improved recall to 32.1% over all 24 predicates.

7.2.1.2 Discussion

An analysis of the machine learning performance reveals that certain predicates are recovered more than others. By examining the resulting rules, I draw some general conclusions about the task. Topicality predicates occur with less regularity and display more variability in their expression and are more difficult to categorize precisely. Non-topical metadata were easier to locate precisely due to regularity of their occurrence patterns. However, many of them seldom occurred and were not recovered due to data sparseness. Integration of such traditional machine learning with recent developments in active learning approaches (Ngai and Yarowsky, 2000; Hwa, 2001) will allow more precise targeting of problematic cases for training and for better sparse tag coverage.

7.3 Learning for the content planner

The full, semantically annotated corpus is the basis for learning the rule base for content planning. These rules determine what the discourse structure should look like, both in terms of a) content ("what to say") and b) its ordering ("where to say it"). I examine each of these two tasks in turn, and then illustrate how the modules perform both tasks.

7.3.1 Content determination

Summaries in the corpus discuss different resources and thus the same predicate (e.g., *Audience*) will have different attribute values in each (e.g., adults or children). In addition, some predicates are present in some summaries and not in others. For example, some summaries may have information about authors and others about editors. Sometimes the presence of the predicate is dependent on its attribute's value: for example, *Edition/Publication* information generally only occurs in summaries of books after their first edition.

In SIGNAL, the content determination resource is the list of the frequencies of the predicates in the training corpus. In planning new expository texts, predicates are included in texts with the same percentage occurrence as in the training corpus. This simple model works well, as the content plan for a particular instance is further refined. The initial, randomized choice of predicates based on frequency of occurrence plan is then limited to using only the actual available predicates for the instance text. A final influence comes from the user or parent application itself, which can force any predicate to be inserted into the text plan. This process is best explained by running through an example. In this chapter, I will use the *Columbia University Home and Health Guide* as a running example to illustrate how the SIGNAL approach works.

To generate a content plan for a resource, the inventory of available predicate attributes about a resource is given to the system. In the case of generating a summary of the *Guide*, its metadata is given as predicates, of which a user-defined number of predicates will be chosen by the system to be realized. In our example, *Edition/Publication* and *Navigation* are noted as salient. These requirements could be provided by CENTRIFUSER, which could compute that the six-part navigation and the third edition are values that merit mention in the summary.

In general, given the length of the summary in predicates, SIGNAL first

attempts to realize all marked predicates, prioritizing those that have higher probabilities in the training corpus. If space remains, the system selects among the remaining predicates in the input, again prioritizing those that have higher probability.

In the example of the *Guide*, I set the summarization system to generate average length summary (with respect to the training corpus) of five predicates. The *Guide*'s 14 available metadata predicates were given to the system, with *Navigation* and *Edition/Publication* selected as salient. The algorithm randomly weights the remaining predicates, using the probabilities in Tables 6.4 and 6.5, and selects *Content_Types*, *Topic*, and *Media_Types* to fill the three remaining predicates. This resulting situation is shown in Figure 7.2.

Columbia University's Home and Health Guide

Author/Editor: Columbia University, College of Physicians and Surgeons
Content_Types: (4) index, illustrations, charts, graphs
Detail: <*extracted sentences*>
Edition/Publication*: third edition
ISN: 0517596105
LCCN: 94013101
Length: 932 pages
Media_Types: book
Navigation*: six part
Overview: <*Extracted keyword*: >angina
Publisher: Crown Publishers, New York, c1995
Size: 29 cm
Topic: (2) "Using our health care system", "New approaches to wellness"

Figure 7.2: Statistical content determination example. Starred entries are metadata predicates that could automatically be determined to be salient.

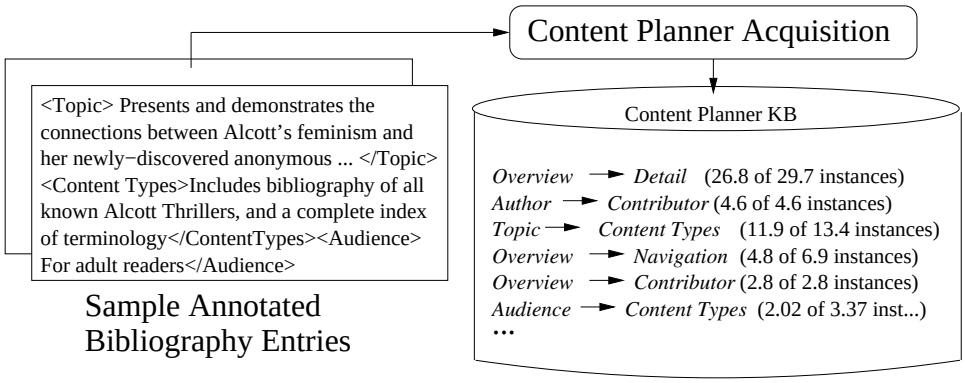


Figure 7.3: Examples of learned elements from the content planner. The content plan is a collection of probabilistic ordering constraints.

7.3.2 Content ordering

The presence or absence of a particular predicate depends greatly on the presence or absence of its peers. Thus, it is important to encode content structuring information, represented as local preferences rather than predefined schema.

Duboue and McKeown (2001) detail an approach for this problem which I initially tried. It uses techniques from computational biology, but is best suited for summaries with multiple instances of the same predicate. Their technique is also intended to model strong local constraints rather than enforce a global ordering. As per Characteristic 9 (Use uniform descriptions), I chose to use a method that would derive a global ordering, with the possibility of local variations. Thus, I calculated bigram statistics on pairs of adjacent predicates, recording which occurred before another. These statistics are used to find an ordering of the predicates that maximizes agreement with training observations. This approach was also utilized in work done on premodifier ordering (Shaw and Hatzivassiloglou, 1999), in which pairs of premodifiers were observed and used to find ordering constraints. The technique is also referred to as Majority Ordering in (Barzilay, Elhadad, and McKeown, 2001), in which bigram orderings were elicited from human subjects. This approach is suboptimal and both studies deal with this problem: with modifications to alle-

viate problems with sparse data in the former study, and as a replacement with an enhanced chronological ordering in the latter.

I chose to augment the basic approach by expanding the statistic to account for longer distance co-occurrences. This method better models the fading influence of context as one moves farther away from the decision point, as was also done in (Hatzivassiloglou, Duboue, and Rzhetsky, 2001). This is done by utilizing information provided in all previous n metadata predicates. I constructed two backoff schemes: one based on the harmonic series, the other based on the quadratic. In both, a precedence relationship of distance one (e.g., adjacent) is given a full strength score, but a distance n relationship is given $\frac{1}{n}$ unit score in the harmonic and $\frac{1}{2^n}$ in the quadratic.

I use these backoff schemes to calculate a global ordering that includes all possible predicates. Each particular observation of a pair of different predicates in the corpus contributes a weight according to the separating distance, in either a positive (i.e., predicate a before b or negative (i.e., predicate a after b) orientation. The weights are summed over all instances of the pair of predicates to compute a weighted preference for ordering.

Finding a maximally compliant ordering with such a set of binary weights has been shown to be an NP complete problem (Cohen, Schapire, and Singer, 1999). For small inventories of predicates, this ordering is feasible to calculate directly. However, an exact ordering is not necessary as small variations on weak preferences (either from sparse data or conflicting observations that display both orientations) are in many cases not harmful and even desirable. In SIGNAL, a hill-climbing algorithm is implemented and it finds good scoring orderings by starting with a random order and trying random permutations. Each ordering is evaluated with respect to the observations, receiving the positive weight of a predicate pair if the orientation matches the training data, penalized by the weight if the orientation is reversed, for

all predicate pairs in the data. Orderings that score better (i.e., that agree better with the training ordering) are kept for the next cycle of permutations, and lower scoring orderings discarded. This process continues until the permutations fail to find a new, higher scoring ordering within a threshold of attempts. The n highest score orderings are then generalized to allow for simple ordering variations, where n is given by the user. The generalization process simply freezes the most frequent bigrams in the ordering as fixed sequences of tags in the generalized pattern.

Background Overview Edition/Publication Collection_Size
Media_Types Language (*Author/Editor* | *Navigation* | *Topic*
Comparison Size Purpose | *Authority* | *Detail* | *Style Audience*
Query_Relevance | *Readability* | *Award* | *Content_Types Difficulty*
Copyright | *Subjective Contributor*)

Figure 7.4: Highest agreement full orderings of the indicative summary metadata predicates using harmonic penalties. Predicates are swappable where "|" occurs.

To calculate the ordering of the 24 metadata predicates in my summarization application, I used the hundred entry annotated corpus. As the annotated predicates can nest wholly within another, it is not immediately clear what is meant by a linear ordering of the predicates. As such, I defined a predicate's position as the parse node where the annotation starts. For example, if an *Overview* predicate covers an entire sentence that embeds a *Media_Types* predicate, the *Overview* predicate is defined to be ordered before *Media_Types*.

7.3.3 Evaluation of the global ordering

Running the harmonic ordering to produce a single ordering of the predicates rather than clustering the top n ordering is also possible and derives a more stringent ordering (as alternations are resolved). When the top twenty orderings are combined together using generalizations, a looser ordering is derived (as shown in Figure 7.4). As summaries with only a single predicate trivially obey the ordering, texts with a

single predicate should be ignored for evaluation. Table 7.2 shows the coverage of several orderings.

Ordering Description	Texts covered	Not covered
Random (averaged over 5 runs)	8	88
Best single ordering (no model)	32	64
Best single ordering (harmonic decay)	37	59
Best single ordering (quadratic decay)	36	60
Top 20 harmonic orderings generalized	46	50

Table 7.2: Coverage of different orderings.

In the annotated corpus of 100 entries, there were 4 single predicate entries, which were not included in the evaluation. Running the simple precedence model already improves performance dramatically over the random case. With the addition of harmonic and quadratic decay, the performance of the modeling improves an additional 10%. Since many of specific top orderings have equal or close fitness, the generalization process allows additional flexibility in handling other high scoring variants. When the top 20 orderings are generalized, it is apparent that order of the beginning tags are more fixed, and that the insertion of "don't cares" allows 20% more orderings to be captured with a small amount of variation ($\frac{10!}{24!} \approx \frac{1}{4.8 \times 10^{16}}$ of all possible variations of the 24 tags).

Using the harmonic backoff scheme, and generalizing the top 20 orderings (row 4 in Table 7.2), I derived a resulting global plan for the summary predicates, shown in Figure 7.4. This particular ordering was used to impose an ordering of the predicates chosen by the earlier content planning stage. If more than a single ordering of the chosen predicates results (from the valid alternations in the ordering), a random choice is made. The result of this process on the chosen *Guide* metadata is an ordered summary content plan, shown in Figure 7.5:

Title: *Columbia University's Home and Health Guide*

Edition/Publication $\prec$ *Media_Types* $\prec$ (*Topic* | *Navigation* | *Content_Types*)

Figure 7.5: Statistical content ordering of the *Guide* example.

7.3.4 Stochastic content plan comparison with rule-based content plan

Recall that CENTRIFUSER generates multi-document, interrelated summaries to assist searchers in distinguishing differences between documents. In contrast, the summaries modeled by SIGNAL are single document summaries (as the corpus consists only of such summaries).

Despite these differences, there are parallels between SIGNAL's and CENTRIFUSER's content plan[2]. CENTRIFUSER splits each category summary into three consecutive sections: 1) obligatory identifying information, 2) topical metadata and 3) non-topical metadata. In Figure 7.4, one sees that the high-level topical predicate *Overview* does indeed come in the beginning half of the content plan. However, the other topicality predicates *Detail* (more particular details; amendable to sentence extraction) and *Topic* (list of topics treated in the text) are intermixed with the other metadata. This shows that the fixed content ordering used in CENTRIFUSER could be further generalized – as the corpus shows more variation than a simple linear ordering of topical metadata first, and non-topical second.

7.4 Learning for surface realization

Once predicates for a text are chosen and ordered, the text representing the predicate must be produced. This is the task of surface realization in the traditional generation architecture: to convey the predicates as natural language. In tradi-

[2]previously explained in Section 6.3.1.2

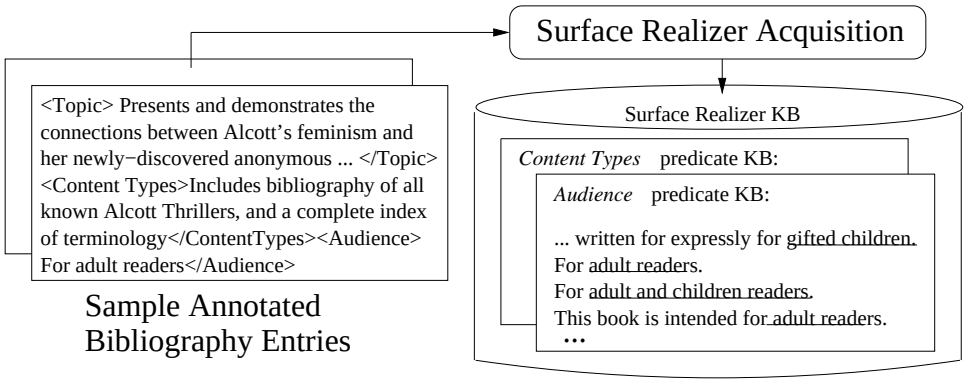


Figure 7.6: Examples of learned elements from the surface realizer for the *Audience* predicate. The surface realizer consists of attribute values (underlined), and associated text that convey the predicate's semantics.

tional NLG, surface realization is often broken down into three separate tasks: (1) sentence planning, which takes individual messages or propositions and assigns them to specific sentences and determines the sentences' basic syntactic structure; (2) lexical choice, which determines the words used, and (3) syntactic realization, which uses a grammar to produce the sentence. In this section, I examine the creation of lexical resources from the corpora that can be used in the production of new texts that assist in tasks 1 and 2.

The SIGNAL framework divides a predicate into two components, shown on the right hand side of Figure 7.6: the predicate's *attribute value* itself (e.g., for an *Audience* predicate: "adult readers") and the *associated text* that casts this information in the appropriate semantic role dictated by the predicate ("This book is meant for <attribute_value>" for the *Audience* predicate). This notion of "predicate" is identical to the notion of "slots" or "tags" as outlined by Varges and Mellish (2001); similarly, "attribute values" are equivalent to "fillers". In a dependency framework, the attribute value is the child and the associated text the head of a dependency relationship (e.g., "This book is meant for"$_{head}$ → "adult readers"$_{child}$). In this framework, surface realization begins with the process of choosing the most

appropriate associated text among alternative verbalizations found in the training text given the attribute values. The associated text and attribute values are then realized as sentences, phrases or words, which are combined to form a new text.

Topical predicates, which are highly specific to the resource being summarized, are best handled by existing techniques of sentence extraction (Paice, 1990; Mani and Maybury, 1999) or domain- and genre-specific text grammars (Liddy, 1991; Rama and Srinivasan, 1993). For example, extraction of salient keywords or topic signature generation (Lin and Hovy, 2000) can fill *Overview* attribute values. Metadata attributes for the *Topic* predicates are well-handled by the CENTRIFUSER system's topic tree node titles and variants. *Detail* predicates are the most specific and focused of the three types of topical predicates that I classified, and are best extracted entirely by sentence extraction. Identifying and extracting these domain specific predicates is not the focus of the research in this chapter. I will assume that appropriate sentence or keywords can be extracted to fill the metadata attribute values and associated text.

In the remaining discussion, I will focus on the other, non-topical predicates that are independent of the content of the summarized documents (e.g., *Content_Types* and *Audience*), as these types of predicates are not covered by sentence extraction techniques.

The division of the semantic predicate into a core, attribute value and the associated text that makes the predicate palatable in running text, SIGNAL is able to capture variations of a predicate in a natural manner. It allows SIGNAL to capture when and with what constraints an associated text should be used to communicate a given predicate attribute value. I compare the variation captured by SIGNAL with the fixed grammar used by CENTRIFUSER at the end of the chapter.

7.4.1 Delimiting attribute values from associated text

In annotating the training corpus, each word was assigned to at least one predicate. This means that associated texts are annotated with their embedded attribute values. Some attribute values will appear in the text without associated text, standing alone. Both cases can occur depending on the syntactic structure of the text and on the predicate, as shown in Figure 7.7:

<Readability>This book is well-written</Readability>,<Topic> and discusses ...

<Topic>The <Readability>well-written</Readability> book discusses ...

Figure 7.7: Instances of the *Readability* predicate, with associated text (top) and standing alone (bottom).

As the annotations cover both cases of stand-alone attribute values as well as ones with associated texts, the first task is to differentiate and delimit the attribute values from the associated text in the training corpus. An analysis of these texts in the indicative summaries indicated that attribute values are highly flexible in location within the summaries and in their grammatical structure. In order to capture this flexibility, I capitalize on the stemmed, lexical dependency framework used in parsing the entries. The training corpus is tagged using Ratnaparkhi's maximum entropy tagger (Ratnaparkhi, 2000), then assigned a parse using Collins' parser (Collins, 1996) and the resulting parse tree stored with stemmed information using the Porter stemmer (Porter, 1980). This framework conflates phrases such as "index included", "includes an index" and "inclusion of indices" (found in instances of the *Content Types* predicate) together into a single stemmed lexical dependency pair of "include"$_{head} \rightarrow$ "index"$_{child}$.

The differentiation process starts by collecting all annotations of a specific predicate (e.g., a collection of *Audience* instances), which can be full sentences, phrases or words. For each collection of predicate instances, the first strategy is to

identify highly frequent (threshold = $\bar{x} + 2\sigma_x$) stemmed lexical dependency pairs. Frequent child lexical items in the dependency pair are potential attribute values in the sentence ("index"$_{child}$ as an attribute value for *Content Types*, as seen in Figure 7.8, #3). From this set, a second strategy prunes frequent dependency pairs that occur with other predicates; this prevents frequent, corpus-wide dependencies such as "book"$_{head}$ → "this"$_{child}$ from appearing as potential attribute values, as they are not exclusively frequent within a single predicate (so #1 and #2 are not attribute values for any of the 24 semantic predicates).

#1, *Topicality*, book$_{head}$ → this$_{child}$:
(e.g., "This book discusses Alcott's works ...", "this book covers the theories")
#2, *Content Types*, book$_{head}$ → this$_{child}$:
(e.g., "This book also comes with a biography", "is discussed in this book")
#3, *Content Types*, include$_{head}$ → index$_{child}$:
(e.g.,"Indices are included", "includes an index", "The book includes an index")
...
[below threshold]
#30, *Content Types*, include$_{head}$ → figure$_{child}$:
(e.g., "Includes figures", "figures and tables are included")
...

Figure 7.8: A portion of the list of stemmed lexical dependencies for various predicates, sorted by order.

A third strategy utilizes the observation that heads of frequent dependency pairs also serve as heads in dependencies with other less frequent child attribute values (dependency head "include" supports the frequent attribute value "index" in #3, but also less frequent ones such as "portrait" or "figure" in #30). Thus, in the simplified example in Figure 7.8, "index" and "figure" are attribute values for the *Content Types* predicate. A fourth strategy calculates the majority dependency type and prunes pairs whose dependency type does not conform to the majority. For

indicative summary generation, the application of the first two strategies gave good (94.8%) precision, but poor recall, due to the high threshold. Applying the third strategy recovered these less frequent attribute values (95% additional attributes were recovered) with a minimal increase in error (92.2% precision). Table 7.4 summarizes the precision achieved by applying these four strategies incrementally, by listing the predicates and their associated levels of precision in labeling attribute value/associated text pairs.

In entries where an attribute value was identified, the remaining text is tentatively labeled as *associated text*. From my perspective, these lexical dependencies can be said to embody lexical choice and resulting syntactic choices internal to the predicate: the alternative forms of the associated texts help to convey the same semantic information (the attribute value) but with different words and syntactic structures. The associated text is also similar to work done in acquiring paraphrases (Barzilay and McKeown, 2001; Lapata, 1999; Jacquemin, Klavans, and Tzoukermann, 1997), in which similar contexts are used to identify candidates. The work here differs from previous work in that the associated texts can be said to be paraphrases only when they have been lexicalized (their slots filled in). Paraphrase identification has aimed at idiomatic, phrasal expressions and morphological variants, and not at full sentences or phrases with slots to instantiate.

These candidate associated texts are automatically screened to filter easily identifiable incorrect matches. If a candidate contains any proper nouns, the candidate is discarded. Additionally, if a candidate text has a coordination in which one branch contains the attribute value, the other branches are discarded. Thus a sentence such as "This <*Media_Types*> contains <*Content_Types*> and discusses *Topic*" will be broken down into two separate associated texts: a) "This <*Media_Types*> contains *Content_Types*>", and b) "... discusses *Topic*".

These attribute values and their associated texts are stored in the lexicon for

use in the regeneration process. Both the associated text and attribute values are syntactically typed, complete with parse information, which permits the subsequent regeneration phase to produce phrasal variations. As will be seen in the following sections, the statistical properties of the attribute values in the training corpus are used to choose appropriate associated texts for new texts. Table 7.3 shows sample output values of associated texts recovered by the algorithm.

[...] includes <*Content_Types*>	This <*Media_Types*> discusses <*Overview*>
Also includes <*Content_Types*>	The <*Author/Editor*> then provides <*Content_Types*>
[...] for <*Audience*>	A <*Media_Types*> contains <*Content_Types*>
... discuss <*Overview*>	Section <*Navigation*> also contains <*Topic*>
Also covers <*Topic*>	A <*Media_Types*> <*Topic*>
The <*Media_Types*> uses <*Difficulty*>	This <*Media_Types*> still contains <*Difficulty*>
The <*Media_Types*> discusses <*Topic*>	*Demonstrates these dangers using <*Content_Types*>
Section <*Navigation*> contains <*Topic* >	This <*Media_Types*> is <*Readability*> read
[...] contains <*Difficulty*>	The <*Author/Editor*> discuss <*Overview*>
Contains no <*Difficulty*>	A <*Subjective*> <*Readability*> <*Media_Types*>
Describes <*Topic*>	*The <*Media_Types*> then provides <*Navigation*>
[...] covers <*Topic*>	The <*Media_Types*> then provides <*Topic*>
[...] provides <*Topic*>	This is a <*Readability*> <*Media_Types*> that contains <*Difficulty*>
It then reviews <*Topic*>	<*Author/Editor*> also includes <*Content_Types*>
*The Integrator uses <*Detail*>	... containing <*Difficulty*>
*Contains <*Audience*>	This <*Media_Types*> is meant to be an introduction to <*Topic*>
This <Media_Types> assumes <*Audience*>	This is a <*Readability*> <*Media_Types*>
*It then covers <*Nav*>	*The <*Media_Types*> contains <*Audience*>
It includes <*Content_Types*>	An <*Subjective*> <*Media_Types*> that provides <*Content_Types*>
*<*Author/Editor*> uses <*Content_Types*> to present facts for which *Author/Editor* cites <*Content_Types*>	
*This <*Edition/Publication*> of the roll provides <*Content_Types*>	
*During the meetings the personal notepads of the participants are linked together using <*Detail*>	
This is a <*Subjective*> <*Readability*> <*Media_Types*> containing no <*Difficulty*>	

Table 7.3: Random sample of the recovered associated texts, simplified from their parse trees. Those marked with a star were judged incorrect.

7.4.1.1 Discussion

A quick perusal of the results shows that this technique works very well in recovering certain predicates but does not recover some predicates at all. A deeper analysis of the process shows two reasons for these results.

First, only predicates that manifest themselves in the attribute value/ associated text paradigm can be recovered by this method, as the dependencies used in the calculation are local to words within the tag. This means that short, stand-alone attributes (such as *Media_Type* and *Author/Editor*) do not fall in this

Predicate (Frequency in 100 entry corpus)	Strategy 1 (Frequent dependencies)	Strategy 2 (Prune non-unique)	Strategy 3 (Same heads)	Strategy 4 (Prune non-majority)
Totals	31/51 (60%)	19/27 (70%)	128/199 (64%)	96/117 (82%)
Media Types* (55)				
Author/Editor* (44)				
Content Types (41)	4/8 (50%)	4/4 (100%)	45/78 (57%)	36/39 (92%)
Subjective (36)	3/4 (75%)	0/1 (0%)	10/16 (62%)	8/9 (88%)
Authority (26)	1/1 (100%)	0/0 (0%)	0/0 (0%)	0/0 (0%)
Background (21)	0/0 (0%)	0/0 (0%)	0/0 (0%)	0/0 (0%)
Navigation (16)	6/8 (75%)	2/3 (66%)	15/18 (83%)	12/14 (85%)
Collection Size (14)	3/3 (100%)	2/2 (100%)	5/5 (100%)	5/5 (100%)
Purpose (13)	0/2 (0%)	0/1 (0%)	0/4 (0%)	0/4 (0%)
Audience (12)	3/5 (60%)	3/3 (100%)	11/19 (57%)	7/9 (77%)
Contributor* (12)				
Comparison (10)	0/3 (0%)	0/1 (0%)	0/6 (0%)	0/4 (0%)
Size (8)	5/6 (83%)	2/3 (66%)	2/3 (66%)	1/1 (100%)
Style (8)	1/1 (100%)	1/1 (100%)	1/1 (100%)	1/1 (100%)
Query Relevance (4)	1/5 (20%)	1/4 (25%)	1/4 (25%)	1/1 (100%)
Difficulty (4)	2/2 (100%)	2/2 (100%)	20/21 (95%)	13/13 (100%)
Readability (4)	2/2 (100%)	2/2 (100%)	18/21 (85%)	12/14 (85%)
Award (2)	0/1 (0%)	0/0 (0%)	0/0 (0%)	0/0 (0%)
Language* (2)				
Copyright* (2)				
Edition/Publication (1)	0/0 (0%)	0/0 (0%)	0/0 (0%)	0/0 (0%)

Table 7.4: Precision (by type) of predicate attribute value/associated text labeling, categorized by predicate frequency. Starred predicates (without values) are not amenable to the attribute value/associated text paradigm.

category and are trivially captured by using the entire tagged, attribute value. These predicates are listed in the table without values and are starred.

Second, other predicates that fared poorly (e.g., *Authority* and *Background*) that are amendable to the attribute value/associated text paradigm were not captured by the approach because of the variety of patterns that were manifest in the corpus, or because they were statistically sparse in the corpus. Further high-quality tagging of these predicates would probably enable better detection, a hypothesis that has held true for other stochastic generation systems (Bangalore, Chen, and Rambow, 2001).

7.4.2 Comparison versus annotation of only attribute values

In the annotation guidelines for the training corpus, I specified that the annotations must exhaustively cover all words of each text. This requirement causes both standalone attribute values as well as those with associated texts to be conflated together in a single category. An alternative approach is to require that annotations only cover attribute values, such that associated text would not be marked. The task of learning the bounds of the associated texts would be a corresponding problem. One possible method is to grow the attribute values from the core attribute value in the parse tree until the process a) runs into conflict with another attribute value; or b) envelops an entire sentence or clause. A first pass at resolving conflicts could employ the hierarchy found in the parse tree to decide which attribute value should own the common ancestor constituents. However, this particular method can result in ties if the attribute values are in equivalent levels in the parse tree. Additionally, differences in semantic distinctions do not necessarily correspond to differences grammatical structure. As a result of these factors, I believe that an approach of propagating the associated texts from known attribute values would be more error-prone.

The exhaustive annotation requirement that is a part of the current approach has some additional benefits. First, I was able to refine the definitions of existing predicates and second, I discovered when a new predicate was necessary. If an exhaustive covering was not required, many predicate instances and certain predicates types may have been missed entirely.

A final method is to have the annotators mark up both the attribute value and the associated texts of a predicate directly with two different tags. This markup would bypass this learning step and have experts enter this information into the lexicon. This method is akin to the traditional method of lexicon development,

which is more resource intensive but accurate.

It should be noted that these methods are not exclusive of each other. Patterns that are recovered by the automatic method presented in this section can be checked, corrected and augmented by human experts. As stated in Table 7.4, in the summarization application about 82% of the attributes judged were correct, which is good enough to demonstrate a limited variety of summaries without expending further human resources. Thus, the automatic method can act to establish of a base lexicon to be employed in rapid prototyping.

7.5 Regeneration using resources

Thus far in this chapter, I have discussed only the analysis process. In the remainder of this chapter, I address the generation process, in which the SIGNAL toolkit uses the attribute value / associated text lexicon to produce new expository texts. At this point, the associated texts are slotted parse trees that can be filled with appropriate attribute values to produce a new text.

In the regeneration process, SIGNAL fills these syntactically typed slots in the associated texts with input attribute values (e.g., metadata fields for a new, previously unseen resource) to create lexicalized predicates. SIGNAL then uses aggregation to combine these predicates in ways that obey the content ordering derived earlier in Section 7.3.2, to generate a larger variety of text structures to choose from. A final stylistics module chooses the actual text to serve as the summary from all of the possibilities.

Related Work. The architecture of SIGNAL's regeneration pipeline is an over-generation process followed by a selection among alternatives that express same semantic predicates that obey the content ordering. As such it is similar in architecture to other approaches to statistical natural language generation. This approach

is most similar to work done by (Walker, Rambow, and Rogati, 2001), in which a trainable sentence planner (SPoT) overgenerates possible sentence plans which are selected from in a ranking stage. SIGNAL differs from SPoT in that SIGNAL has a larger variety of text structures (associated texts) to choose from initially, whereas SPoT's aggregation component handles more varieties of aggregation than SIGNAL. This overgeneration and pruning approach has also been used in the surface (sentence) realization task, in producing statistically based sentence realizers, embodied in the FERGUS, Nitrogen, HALogen, and IGEN systems (Bangalore and Rambow, 2000; Langkilde, 2000; Langkilde-Geary, 2002; Varges, 2002).

Using CFUF/SURGE. To validate any new variations that are created, the system requires knowledge of a broad coverage grammar for the language in question. Fortunately, the SURGE grammar for English (Elhadad and Robin, 1996) has been developed and has been extensively employed in the generation community. I developed the SIGNAL tools to employ the SURGE grammar in the functional unification framework (FUF) (Elhadad, 1993), which has also been widely used with SURGE and with other, domain-specific grammars in a variety of text generation applications such as `PLANDoc` (McKeown, Kukich, and Shaw, 1994), Streak (Robin, 1994) and `FlowDoc` (Passonneau et al., 1996). The SIGNAL tools use the more time-efficient interpreter for FUF implemented in the CFUF package (Kharitonov, 1999).

The FUF generation system employs functional descriptions (FDs) as its data structure. A functional description is a set of attribute names and their values pairs, and can be recursively nested (i.e., an attribute's value can be a set of attribute names and values as well). Tailored to the purpose of language generation, an FD's attributes can hold information about the agent, patient or predicate information, as shown in Figure 7.9. In the generation process, the single operation of unification is used to generate the desired output text. Unification is

a process in which information from two objects are unified to create a new object which combines information from both objects. The unification operation is closely related to the set union operation, except that the unification operation requires that attributes of the same name in both FDs must have the same value or the unification operation will fail.

Input Specification (I_1):

$$\begin{bmatrix} cat & clause \\ process & \begin{bmatrix} type & composite \\ relation & possessive \\ lex & \text{“}hand\text{”} \end{bmatrix} \\ partic & \begin{bmatrix} agent & \begin{bmatrix} cat & pers_pro \\ gender & feminine \end{bmatrix} \\ affected & \boxed{1}\begin{bmatrix} cat & np \\ lex & \text{“}editor\text{”} \end{bmatrix} \\ possessor & \boxed{1} \\ possessed & \begin{bmatrix} cat & np \\ lex & \text{“}draft\text{”} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

Output Sentence (S_1): “She hands the draft to the editor”

Figure 7.9: An example functional description used in SURGE. Reproduced with permission from Elhadad and Robin (1996).

Input to generation consists of the FD specifying the semantic representation of the sentence to be generated, and an FD representing the grammar. The unification process enriches the semantic representation with syntactic information from the grammar that is used to produce a resulting linearized and lexicalized sentence.

To enforce syntactic agreement, it is necessary to adapt the basic FD structure to include mechanisms for structure sharing. This is done with *paths* in FUF, and is demonstrated in the example FD in the Figure 7.9, in which the *possessor* and the *affected* participants are expressed as the same entity: “the editor”.

In the remainder of the chapter, I will use the LISP notation to display functional descriptions, as shown in Figure 7.10. In the LISP representation, an

FD is a list of attribute/value pairs, each pair being a list of two items. The value may be a list of attribute/values itself. Paths are written with curly braces and the caret (^) character indicating a level upward from the origin of the path itself.

```
((cat clause)
 (process ((type composite)
           (relation possessive)
           (lex "hand")))
 (partic ((agent ((cat personal-pronoun)
              (gender feminine)))
          (affected ((cat np)
                     (lex "editor")))
          (possessor {^ affected})
          (possessed ((cat np)
                      (lex "draft"))))))
```

Figure 7.10: Same input as Figure 7.9, in LISP form.

7.5.1 Parse tree transformation to functional description

The CFUF generation system and SURGE grammar are used together to generate new variants of the associated texts and attribute values for use in the regeneration process. The analysis modules of SIGNAL identify these in parse tree form, but they need to be converted to to a functional description form for input to generation. The conversion process is not trivial, as parse trees lack much of the semantic information needed for generation; this is further complicated by the fact that parse trees often correspond to more than one semantic representation. This can happen when the variation of some attribute values in a semantic representation may not result in a difference in the produced English sentence. For example, knowing that an agent is animate may not make a difference in the lexicalization of the verb in a main clause:

The person stopped in time. `(animate yes)` The process stopped in time. `(animate no)`

Figure 7.11: Sentences in which the `animate` attribute value does not change the surface form.

But the difference will become apparent if the clause is relativized, as the choice of the wh-preposition is based on animacy:

The person who stopped in time. `(animate yes)` The process which stopped in time. `(animate no)`

Figure 7.12: Minimal pairs in which the `animate` attribute value changes the surface form.

There are many examples in which the English surface form of a sentence can be interpreted in more than a single way, and are sometimes the basis for jokes:

Customer: I'd like a rear view mirror for my Yugo. Car Dealer: Sounds like a fair trade to me.

Figure 7.13: A joke relying on different circumstantial interpretations.

Here, the prepositional phrase can be interpreted as a circumstantial adjunct, giving information on whose `behalf` the mirror is for (by the customer for his Yugo car), or as an `substitution` circumstantial (by the car dealer, in exchange for the mirror). SURGE handles these variants and many more. This shows that there sometimes can be multiple correct, semantic interpretations of a single parse tree. As such the translation of the parse tree to a suitable FD is problematic: which semantic representation is the best one for a specific parse tree? This problem has to be addressed if the power of a traditional generation system is to be harnessed to produce paraphrases of the predicate components.

Rather than solve this problem directly, I solve an expanded definition of

the problem. I expand the problem from translating a parse tree to a single, correct FD to translating to a set of appropriate FDs. An appropriate FD is an FD that generates the closest lexicalized string (with respect to a tree edit distance function) to the parse tree equivalent. As many FD variations may generate the same lexicalized string, this translation process often produces a set of FDs. The translation program non-deterministically chooses the attribute values of features in the construction of the FD for the associated text, such as including the `animate` feature with a choice point [`yes`|`no`]. These choice points are expanded into a set of FDs by a separate process that unwinds all the choice points to generate the FD sets. This process makes the FD translation simpler, as inferring correct values for properties can often be left to the non-deterministic process.

This places the burden on finding the appropriate FDs on the generation system, as each choice point (e.g., `(animate` [`yes`|`no`]`)` at least doubles the number of generated FDs. This grows the resulting FD set exponentially. To minimize this problem, the SIGNAL parse tree translator only encodes alternatives that may result in a lexical change in the resulting text when combined or divided in producing alternative phrasings. Recall that the SIGNAL system employs CFUF/SURGE specifically to create new (possibly valid) variations on the associated texts, so it is only necessary to handle the variations in the semantics which may change the output lexical string. Other parts of SURGE that do not change the surface form when aggregated are simplified. For example, the module handling the translation of circumstantials simplifies the SURGE system to use position, rather than the default, more complex, corpus-based analysis of circumstantials done by Robin (1994). This is an approach that is also taken in the literature on regeneration (Barzilay, McKeown, and Elhadad, 1999). Figure 7.14 shows two associated text Collins' parse trees and their non-deterministic FD equivalents.

Input parse tree P_1: [TOP/includ/Includes|CONTENTTYPES [SG/includ/Includes [VP/includ/Includes VBZ/includ/Includes [NP-A/___/bibliography]]]]

Output non-deterministic functional description set SFD_1:

```
(define fd-CONTENTTYPES-1
  '((cat clause)
    (proc ((type lexical)
        (lex "include") (tense present)
        (subcat ((1 {^3 lex-roles agent})
            (2 {^3 lex-roles patient})))))
    (lex-roles ((agent ((cat common)
            (definite [yes|no]) (animate [yes|no]) [(determiner none)|]
            (lex "T") (gap yes)))
        (patient ((cat common)
            (definite [yes|no]) (animate [yes|no]) [(determiner none)|]
            (signalPredicate CONTENTTYPES)
            (signalHead "bibliography")))))))
```

Input parse tree P_2: [NP/paper/paper|MEDIATYPES [NPB/paper/paper DT/a/a JJ/___/good|SUBJECTIVE PUNC,/,/, JJ/___/clearly-written|READABILITY NN/___/paper]]

Output non-deterministic functional description set SFD_2:

```
(define fd-MEDIATYPES-SUBJECTIVE-READABILITY-1
  '((cat common)
    (definite no) (number singular)
    (describer ((cat list)
                (distinct ~(((cat adj)
                             (signalPredicate SUBJECTIVE) (signalHead "good"))
                            ((cat adj)
                             (signalPredicate READABILITY)
                             (signalHead "clearly-written"))))))
    (definite [yes|no]) (animate [yes|no]) [(determiner none)]
    (signalPredicate CONTENTTYPES)
    (signalHead "bibliography")))
```

Figure 7.14: Translation of two associated texts (in parse tree form) by the translator to non-deterministic functional descriptions.

7.5.1.1 Evaluation

An evaluation of the translation module was carried out in conjunction with its implementation. A training set of 961 sentences from the summary annotation training corpus was split into two halves, one half for implementation analysis and one for testing.

The translation module was tested by transforming parse trees to an FD set and using CFUF/SURGE to generate the sentence. If any of the resulting FDs generated the string that the parse represented, it was considered correct. Table

7.5 gives the result of the translation.

	Training (sents. 1-480)	Testing (sents. 481-961)	Total
Number of parses correctly regenerated	252 (52%)	198 (41%)	450 (46%)
Number of parses partially covered	107 (22%)	177 (36%)	284 (29%)
Number of parses wrong	121 (25%)	106 (22%)	227 (23%)
Number of parses, total	480	481	961
Number of words correctly regenerated	1463 (40%)	1829 (34%)	3292 (37%)
Number of words, total	3627	5302	8929

Table 7.5: Parse tree to functional description translation accuracy.

In the development of the translation module, I noted that a large portion of the errors were due to errors in the input and in the capacity of the generation system to handle certain real text phenomenon. To track this problem, I analyzed the errors that caused the parse to FD translation to entirely fail in 20 examples in the training corpus. A total of 30% of the errors (6 of 20) were due to problems with tagging and parsing errors. This is a higher rate than in reported literature, but the indicative summary text type contains many gapped subjects (as the book or resource is almost always the focus), which makes it challenging to construct a proper parse. The remaining errors made by the parse to FD translation system ranged from simple (does not handle left and right bracket punctuations, accounting for 2 errors) to complex (problems with coordination and embedded clause lexical role assignment, accounting for 11 errors).

In general it should be noted that the translation of the parse tree to FD task is very difficult. The process is basically responsible for enriching the parse tree with proper semantics that allow for generation. Additional work in handling cases of embedded conjunctions, and relative clauses constructions would improve performance to the 60-70% level. As the process fuses domain independent semantics with the parse tree, it is somewhat akin to semantic parsing of which the best reported results are in the 70% range (Thompson, Califf, and Mooney, 1999).

7.5.2 Unifying associated texts and attributes

The translation of an associated text into a parse tree leaves slots where the attribute values go. To fill in the slots, the attribute values from an input text are needed. As stated earlier and shown in Figure 7.2, these values are inputs to the SIGNAL system. Attribute values themselves may be provided by different people or systems and it is likely that their grammatical categories may vary. For example, the *Navigation* predicate could be an adjectival phrase: "six part" or a clause "Organized into six parts". The system should be able to handle these variations to ensure that the greatest possible variations of sentence structure can be generated.

For this reason, the input attribute values are processed in the same manner as the training corpus. Attribute values with more than a single value (such as the *Content_Types* predicate in the *Guide* example) in addition to being translated individually, are expanded into a coordination using a simple grammar (e.g., "index, illustrations, charts and graphs"). They are first part-of-speech tagged and then parsed to obtain a parse tree. The parse tree is then converted to an FD with the translation module from the last section.

The process of joining attribute values to associated texts is thus a simple unification operation that can be performed by FUF. During the FD translation process, the associated texts and input attributes are enriched with an additional attribute (which is ignored by the SURGE grammar but propagated by FUF) in the FD specifying the predicate to be filled in associated text FDs or provided in input attribute value FDs. This is done to enforce semantic type checking (in addition to syntactic category checking), a condition necessary to prevent *Readability* slots from being filled with *Navigation* attribute values. Figure 7.15 illustrates this process for an associated text for the *Topic* predicate with an additional slot for *Media_Types*.

Both the FDs for the associated texts that have been unified with attribute values as well as the FDs representing the attribute values themselves are passed

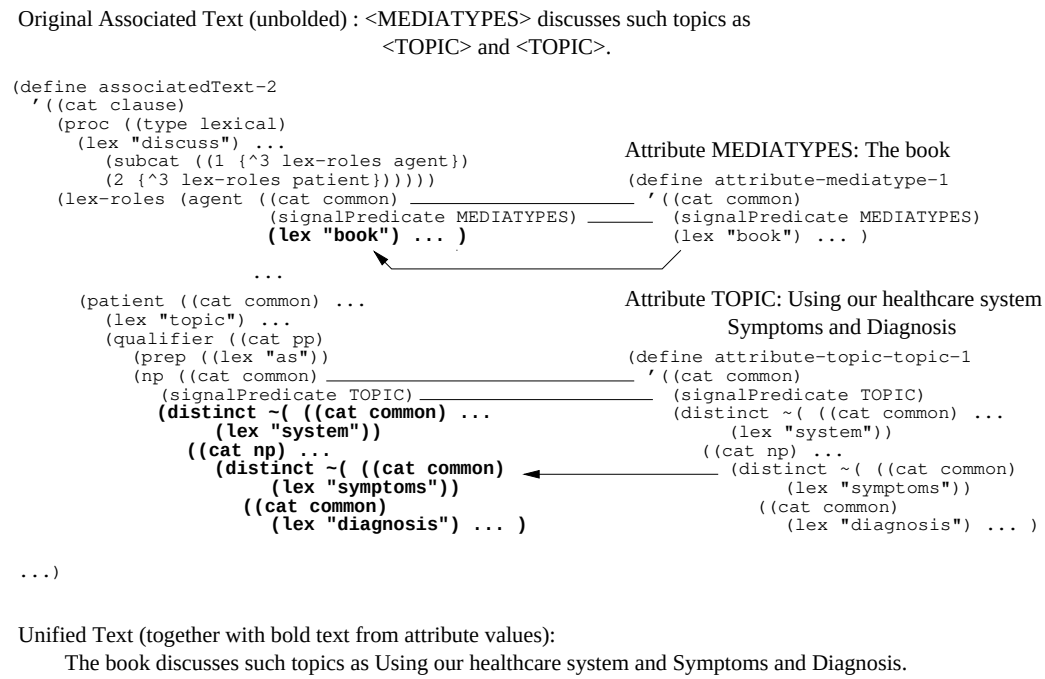


Figure 7.15: Portions of filled associated text (l), showing `signalPredicate` and `cat` fields unified from source attribute values (r).

on to the aggregation phase. Stand-alone attribute values without accompanying associated texts are still good valid candidates for text generation as entire sentences or clauses can be input as predicates (e.g., *Overview* or *Detail* predicates in the summarization application). Also, the requirements of generating a shorter text often demands that information be pared down to a minimum. In a sense, associated text acts to improve fluency of the resulting text and but can be sacrificed for conciseness when necessary.

7.5.3 Aggregation module

A final module performs simple aggregation of the resulting inventory of the lexicalized (i.e., possibly unified with associated text) predicates. The module takes two predicates in FD form and performs syntactic aggregation (e.g., interpretative and referential aggregation is not handled) to join them into a single FD. This takes the

form of coordination of predicates of the same syntactic category (e.g., two clauses being combined together) or adjunction when the syntactic categories differ (e.g., a preposition phrase being attached to a clause). This is an iterative process and is akin to the work on revision based generation done by (Robin, 1994), but highly simplified. For example, coordination of shared constituents is handled, but other complex types of coordination, such as non-constituent coordination, gapping and right node raising (which are handled in other systems like MAGIC (Shaw, 2002)) are not.

The aggregation phase works over the constituents in the entire predicate text, not just the associated text. For example, clausal attribute values (such as the extracted sentences that make up the *Detail* predicate in the summary application) may be modified by adjoining smaller units. Resulting FDs generated by the aggregation system are placed back into the input pool of FDs for aggregation, such that multiple rounds of aggregation can be performed on a nuclear predicate.

Source FD_3 (`FD-Topic-1` as `(cat s)`):
"$<$ *trace* $>$ Focuses on using our health care system"

Aggregation round 1:
Adjoining FD (`FD-Media_Types-1` as `(cat common)`):
"This book"
Satisfies threshold and ordering as `MEDIATYPES` $\prec$ `TOPIC`
Output FD_4 (`FD-Topic-Media_Types-1`) as `(cat common)`:
"This book which $<$ *trace* $>$ focuses on using our health care system"

Aggregation round 2:
Adjoining FD (`FD-Overview-1`) as `(cat vp)`:
"Discusses angina"
Satisfies threshold and ordering as `OVERVIEW` $\prec$ `MEDIATYPES` $\prec$ `TOPIC`
Output FD_5 (`FD-Overview-Media_Types-Topic-1`) as `(cat clause)`:
"This book, which focuses on using our health care system, discusses angina"

Figure 7.16: An aggregation example for an FD in the *Guide* example, through two iterations.

Aggregation is a complex phenomenon that has been studied in depth by many researchers (Shaw, 2002; Dalianis, 1996) that is subject to many constraints and circumstances. The aggregation module as described thus far does not use constraints reported in the literature. Thus, if the aggregation module is run unconstrained, it produces many ungrammatical modifications and is computationally expensive. For this reason, the training corpus is used to constrain the aggregation process. Since sentences in the training corpus containing two or more predicates are examples of aggregation, SIGNAL examines the sentences and compiles statistics on their predicates' aggregation. The sentences' parse trees are examined to see whether a predicate is adjoined to another or coordinated. Observed instances of the aggregation of predicate pairs are tallied, along with data on their 1) syntactic categories and 2) stemmed lexical heads. Predicate pairs that coordinate or adjoin above a threshold and additionally conform to the linear ordering specified in the content plan are allowed. Examples of deriving these statistics are shown in Figure 7.17.

Parse$_6$: <Difficulty>The paper contains no matrix algebra, and <Audience>is appropriate for an advanced undergraduate student</Audience></Difficulty>.
Analysis$_6$: 1. [Coordination] (*Difficulty*, `s`, contain) $\prec$ (*Audience*, `s`, be)

Parse$_7$: <Overview>Biographies of dive-bomber specialists from the major powers, <Detail>including one from Sweden</Detail></Overview>.
Analysis$_7$: 1. [Adjunction] (*Overview*, `common`, biographi) $\prec$ (*Detail*, `pp`, includ)

Figure 7.17: The compilation of aggregation statistics of two multi-predicate sentences from the 100-entry corpus. Semantically aligned parse trees are used to generate the analysis, but the annotated text is shown here for clarity.

Executing the aggregator using a low threshold and allowing backoff by discarding head and syntactic category information produces more variants but will result in more errors. Using a higher threshold and mandating that potential aggregations must also satisfy both auxiliary data attribute results in less variation. In the sample application, aggregation of multiple predicates into a single top level

constituent occurs in 77 of the 100 entries, occurring in 116 different types of predicate pairs; but when a higher threshold is used only 40 (as opposed to 116) different aggregations are allowed.

When all allowable aggregations are created, the system places all of the aggregated FDs along with the attribute values (both unified with associated texts and stand-alone ones) in a pool. Each FD in the pool satisfies one or more predicate slots in the content ordered plan. A candidate output text is then a linear sequence of FDs such that each predicate in the content plan is satisfied. Figure 7.18 illustrates a simplified version of this process for the example summary of the *Guide* to be generated with the *Topic* $\prec$ *Edition/Publication* $\prec$ *Media_Types* $\prec$ (*Nav* | *Content_Types*) ordering derived in Section 7.3.2.

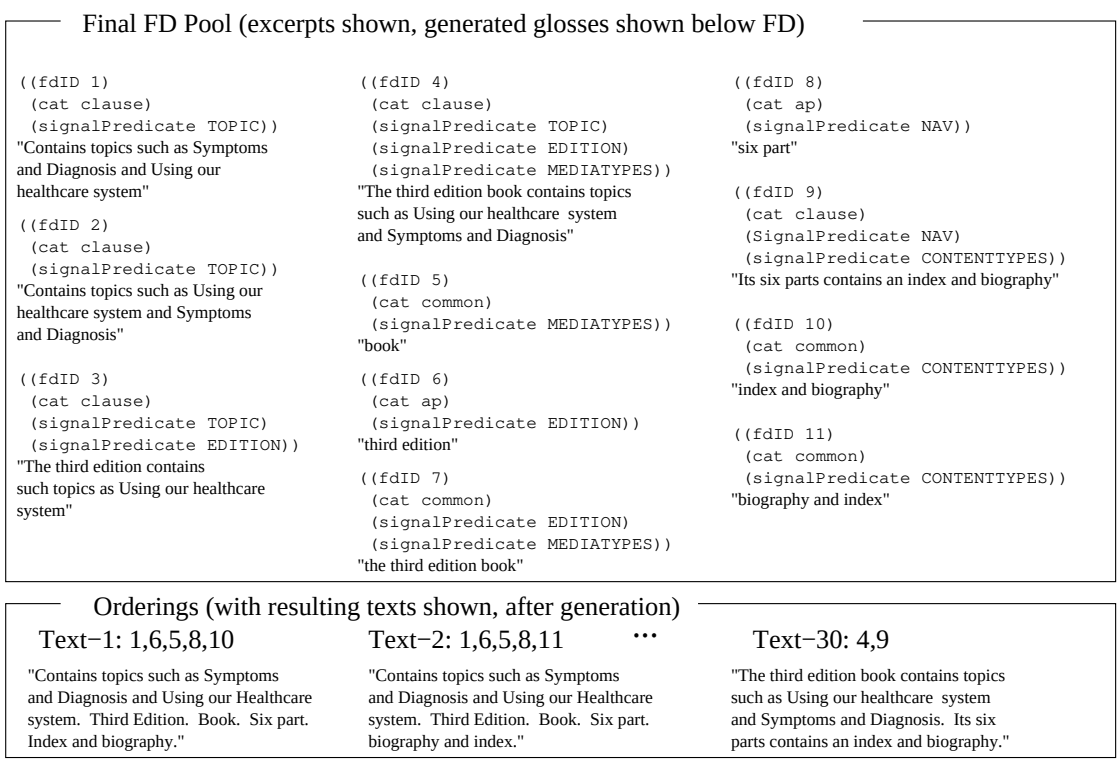


Figure 7.18: Portions of FDs showing conveyed predicates and their use in valid FD sequences for summaries of the *Guide*.

An approximate word count for each FD is generated by counting the words

in the `lex` features of the FD. This information is propagated to the FD sequences so that an approximate word count for each resulting text is computed. FD sequences that have an approximate word count within a user desired range (provided on the command line of the tool) are then passed on to the stylistic module.

7.5.4 Stylistic evaluation module

The stylistics module receives the valid FD sequences from the aggregation module and scores them according to how well each mirrors the linguistic features of training texts that are approximately the same size, similar to work in SPoT (Walker, Rambow, and Rogati, 2001) and IGEN (Varges, 2002). This module is needed to select the final text that will be realized by CFUF/SURGE and displayed as the system's output to the user. The fitness module's goal is to implicitly capture language constraints without needing to state them explicitly. Such constraints might be to use stand-alone attribute values when space is limited (and use associated texts when extra space is present), as well to vary the sentence structure (e.g., don't use an adjunct in the same position if the same construction was used in the last FD). For the module to make these decisions thus requires that the system learn stylistic constraints and apply them in the generation process. This can be broken down into three subtasks:

1. **Identify and encode linguistic features.** Define features that capture stylistics and calculate their values per FD sequence.

2. **Use machine learning to predict each feature.** Train machine learners to predicate each of the features for each FD in each sequence.

3. **Score candidate text and select best.** Calculate how close each FD sequence comes to the ideal predicted feature values. Pick the best as the output text.

I now discuss each of the three steps in more detail.

7.5.4.1 Identify and encode linguistic features

What exactly constitutes stylistics is an open question, and the topic of much in-depth research. As the system has access only to the sequence of FDs, SIGNAL equates stylistics to an inventory of linguistic features, rather than features that directly embody pragmatics and discourse style. I examined a body of features for their potential to predict which FD would be most felicitous given its context. Related work on descriptive appositive language reuse (Radev, 1998), and genre identification (Biber, 1989; Karlgren and Cutting, 1994; Kessler, Nunberg, and Schütze, 1997) defines a large set of basic features to use such as character and word-level features, and positional and contextual features. (Walker, Rambow, and Rogati, 2001) also gives a list of 8 domain-independent features that made a significant impact in evaluating sentence plans. I implemented a total of 25 of these features in SIGNAL to test their efficacy in identifying appropriate constraints.

The problem here is related to these previous studies, but differs in some key respects. Choosing descriptions often involves choosing between descriptions that convey different semantic information ("Clinton" as "senator" versus "president"), whereas my associated texts generally convey the same information but realize it differently, similar to paraphrasing. Genre identification differs from my problem mostly in scale; whereas whole texts are input to the genre categorization process, in this problem the system has access to a sequence of FDs, representing a short sequence of single sentences or clauses. Genre identification work also focuses primarily on surface level features rather than assuming a deeper analysis of the text. As one might expect, some of the features (e.g., number of "me"s) from genre detection did not play a significant role in the stylistic determination since the specific domain of investigation (e.g., expository indicative summaries) did not manifest a

wide range of values for these features.

Origin	Description	Value in FD_{1-1}*	Value in FD_5
2 features from Radev (1998)	1. Maximum nesting in FD	6	12
	2. # of Words	2	12
18 features from Karlgren and Cutting (1994)	1. # of nouns (= base NPs)	1	3
	2. # of adverbs	0	0
	3. # of second person pronouns	0	0
	4. # of first person pronouns	0	0
	5. # of present verbs	1	2
	6. # of prepositions	0	0
	7. # of long words (> 6 characters)	2	5
	8. Type to token ratio	1	1
	9. Average # of characters per word	10	5
	10. # of sentences (=FDs)	1	1
	11. # of characters	18	61
	12. # of present participles	0	0
	13-18. # of "which"s, "that"s, "I"s, "it"s, "therefore"s and "me"s	0,0,0,0,0	1,0,0,0,0
5 character level features from Kessler, Nunberg and Schütze (1997)	1. # of Question marks	0	0
	2. # of Capitalized words	1	1
	3. # of Hypenated words	0	0
	4. # of Exclamation marks	0	0
	5. # of Acronyms	0	0
1 feature from SPoT (2001)	1. # of relative clauses	0	1
10 new features	1. # of predicates in FD	1	3
	2. # of conjunctions	0	0
	3. # of prepositional phrases	0	0
	4. % of `signalHead` matching actual value	100%	33%
	5. Average # of adjuncts present	0	0
	6. Average # of qualifiers present	0	0
	7. Average # of describers present	0	1
	8. Gapped sentence	yes	no
	9. Top node type	vp**	clause
	10. Passive construction	no	no

* - the FD formed from SFD_1 by taking the first choice at all six choice points, and filling the *CONTENT_TYPES* slot with "bibliography"
** - analyzed as a VP due to the gapped subject

Table 7.6: List of features used in the stylistic evaluation module, with sample values for two example FDs from Figures 7.14 and 7.16.

Table 7.6 lists the inventory of features used, and gives sample values of the features as derived from the two example FDs translated in Figures 7.14 and 7.16. As is apparent from the values in the table, many of the baseline features taken from related work do not distinguish between very different FDs. As such, I developed several additional features which capitalize on the detailed FD representation that SIGNAL had access to. I introduce an additional 10 features that I believe have a stylistic impact, such as ones that look for different types of adjunct constructions

and passive constructions.

7.5.4.2 Predict features using machine learning

To measure the effectiveness of each feature, I again employed the inductive rule learner `ripper` (Cohen, 1995) to determine which features play a role in predicting the characteristics of the candidate text.

First, the 100 texts in annotated indicative summary corpus were translated entirely into FD format using the non-deterministic FD translation setup. If a particular sentence or clause in translation resulted in a set of equivalent FDs (which is often the case), a random selection was made to select a single FD per parsed "TOP" node.

The training corpus is again utilized to derive training tuples for predicting the value of each feature. The length of each text was calculated and the range partitioned into 5 equally distributed segments of roughly 20 texts each. Feature values calculated for a particular text would contribute only to the length range segment that the text belongs to.

In a second step, 90 texts were randomly selected as a training portion and the remaining 10 were set aside for testing. Values for all 36 (26 existing + 10 new) features for four different contexts surrounding each FD in each of the 90 training texts were calculated: values derived from the (1) previous and (2) next FDs, when applicable, (3) average values from the entire FD sequence, and (4) composite values that measure the difference between previous and next FD features. This total of 4 x 36 = 144 features were computed for each of the FDs that were part of the 90 training texts. The actual values for each of the 36 features in the FDs were then combined with the 144 context features to create training tuples. By running `ripper` on these tuples, a set of 36 machine learners were created to independently predict one of the features. This architecture is shown graphically in Figure 7.19.

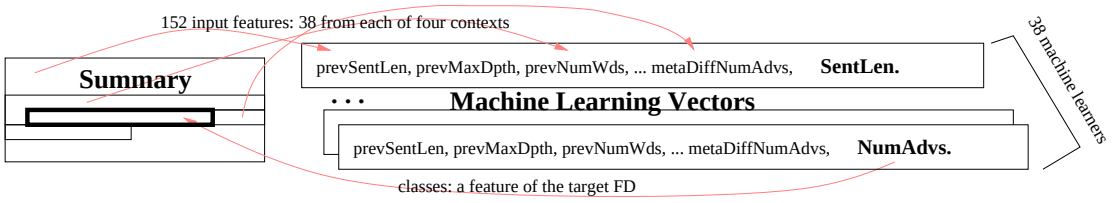


Figure 7.19: Machine learning architecture for the features of the target predicate, with sample tuples shown.

As previously mentioned, several of the 36 features from previous work do not display much variation in the summary corpus, and thus the resulting induced rule set for these features was a simple default. For example, the rule set for number of question marks and exclamation marks simply had a default rule: `default: 0`. However, a few of them consisted of a few rules each, exemplified in Figure 7.20.

1. Generate 1 or 2 relative clauses if the FD sequence number <= 2 and if the previous FD has 0 relative clauses and if the next FD has <= 0 relative clauses.
2. Otherwise, generate 1 or 2 relative clauses if number of nouns in the previous FD >= 6 and the maximum nesting of the previous sentence <= 11.
3. Otherwise, generate 0 relative clauses.

Figure 7.20: Translated machine learned rule set (from native `ripper` format) for the relative clause feature.

7.5.4.3 Score candidate texts

The final step is to select a candidate text based on how well each FD in the candidate text matches its predicted features. Each FD in each FD sequence is scored based on how well its actual features match with the predicted values from the array of machine learners.

As the choice is limited to whole FDs and not single words or constituents as in other stochastic approaches (Langkilde, 2000; Varges and Mellish, 2001), this search process is constrained and does not present an efficiency problem. For numerical features (e.g., number of words), I use a normalized difference between

the predicted value and the FD's actual value to calculate its goodness of fit. For set valued features (e.g., parse node type: NP versus PP), the feature either matches or does not (1 or 0). The algorithm weighs all features equally, a score per FD in the sequence is generated by summing up the matching score for all of the 38 features. The FD sequence with the highest average FD score is picked as the final resulting text. The result of this process is shown in Figure 7.21 for the running example of the *Guide*.

The third edition book contains topics such as using our healthcare system and symptoms and diagnosis. Its six parts contain an index and bibliography.
(rank 1: 24 words, length range 4 of 5 [longer than average], rated good)

A third edition book describes symptoms and diagnosis. Six parts. Contains an index.
(rank 2: 13 words, length range 2 of 5 [shorter than average], rated good)

A book. Includes a third edition index and graphs and covers topics such as using our health system and symptoms and diagnosis. Six parts.
(rank 6: 24 words, length range 4 of 5 [longer than average], rated poor, as ``third edition'' shouldn't modify ``index'')

Figure 7.21: Summaries of varying quality for the *Guide* at different lengths.

7.5.5 Evaluation

I evaluated the stylistic module based on the ten testing texts, of which the *Guide* was one. The input metadata for each of the ten were manually constructed by the author and fed to the system to unify with associated texts and the results to the aggregation module. The result of this pipeline is a set of alternative FD sequences to be ranked by the stylistics module.

I evaluated the quality of each alternative FD sequence text, as well as keep-

ing a separate record of the top ranked FD sequence text chosen by the stylistics module. The stylistics module was run with length parameters to generate a summary of approximately the same length as the test text (i.e., in the same length range as the test text, out of the total five segments). *Good* texts were ones that read well, and conveyed the same information as the original. Texts that were *Okay* exhibited some disfluency, but the aggregation and ordering yielded texts that were understandable and factually accurate. *Poor* texts either were incoherent (often due to bad pairings between associated texts and attribute values) or had factual mistakes (due to underconstrained aggregation). Table 7.7 gives the results of the evaluation.

Test text ID	# metadata predicates	# of alternatives	Top choice quality	Quality of top 10 ranked choices		
				Good	Okay	Poor
The *Guide*	5	27	good	7	2	1
157	7	93	good	5	4	1
614	4	8	good	6	0	2
712	7	73	poor	0	1	9
934	4	11	good	3	6	1
1203	5	28	okay	2	2	6
1442	4	18	poor	1	2	7
1762	3	8	good	5	0	3
1884	2	6	okay	1	3	2
1953	3	26	okay	4	2	4

Table 7.7: Evaluation measures for the stylistics module.

7.6 Relation to CENTRIFUSER

In this chapter, I have used the SIGNAL tool set in an application in the genre specific task of indicative summarization. The SIGNAL approach was used to induce typical structure of summaries in both for content planning as well as for phrasal surface realization. In the construction of a new summary text for a new resource, that resource's metadata must be made available to the system. The CENTRIFUSER system itself has the capability to distill some parts of this metadata.

Topics and *Overview* metadata are covered by CENTRIFUSER's document topic trees. The document topic trees also note the *Style* and *Content_Types* associated with particular topics that serve as the respective metadata.

In the digital library, metadata is generally created by human catalogers, as evidenced by the use and variety of metadata standards overviewed in Chapter 6. These enormous stores of metadata can be put to use in creating first-pass indicative summaries of resources using the SIGNAL approach. These metadata attribute values from the resource's cataloging record can interact with surface realization constraints to produce a set of sentences or phrases that correspond to each predicate.

The indicative summaries produced using the SIGNAL approach differ from ones produced by the CENTRIFUSER system in two crucial ways: 1) it produces single-document summaries rather than multidocument ones, and 2) it produces summaries that are standalone and do not interrelate, unlike the ones formed by CENTRIFUSER. Due to these differences, the two systems are for different purposes and not easily substituted for each other or easily compared. The SIGNAL approach described in this chapter improves upon the rule-based approach to CENTRIFUSER's original natural language indicative summary text generation, by adding flexibility and variability in the text generation process that is a hallmark of human-produced natural language. This is an advantage as well as a handicap, as the SIGNAL approach thus requires a corpus, and as such, has only been applied to single document summarization. CENTRIFUSER's module for indicative summarization focuses on summaries of multiple documents, rather than on single documents. Additionally, the SIGNAL corpus focuses on document summaries that stand individually, without relation to other summaries. In CENTRIFUSER, the system produces a set of indicative summaries that are related to each other, in that highlighted topics pull out salient differences (as per Characteristic 2) among the clustered articles. Until

a corpus of comparative summaries exist, the SIGNAL approach cannot be directly applied to the same scenario as CENTRIFUSER.

7.7 Conclusion

In this chapter, I have described a new architecture for NLG that takes advantage of annotated corpora. The method takes a new approach to NLG by using machine learning to capture semantic and stylistic constraints that are traditionally hand coded by human experts. The induced constraints are reused in a traditional text generation pipeline to realize new instances of similar texts given new attribute values.

Domain communicative knowledge (DCK) is needed to properly perform the tasks of content planning and surface realization in NLG systems (Kittredge, Korelsky, and Rambow, 1991). This is done in most NLG systems using an cyclical approach of development by consulting domain experts and fine-tuning the target NLG system (Reiter and Dale, 2000), Chapter 4. In the development of the KNIGHT system, Lester and Porter discuss the difficulty of developing an explanation generator for biological knowledge. In a cyclical developmental process, a domain expert was consulted, which resulted in the finished automatic system "after many passes through the critiquing and revision phases" (Lester and Porter, 1997). Similar experiences in domain modeling and message definition have been shown by Kukich and colleagues in their work on the PLANDoc system (Kukich et al., 1994). In these efforts, many person-months are expended in the development process.

SIGNAL harnesses training data to bootstrap the process of creating fluent and concise texts for specific text types. Table 7.8 shows all the processes in SIGNAL that reference the training corpus. Furthermore, as the outputs of the system are generated using a traditional generation engine, the outputs can be edited as

SIGNAL module	Description of training corpus use
Content planing	Inference of predicate important based on statistical frequency
Content ordering	Inference of predicate precedence based on context decay model
Partial surface realization	Inference of variable (associated text) and core (attribute value) portions of predicates
Aggregation	Limits on allowable aggregation (coordination and adjunction) of two or more predicates
Stylistics	General, domain-independent syntactical language model for feature-faceted machine learning array

Table 7.8: SIGNAL modules that utilize the training corpus.

the system develops into a mature NLG system. SIGNAL requires a training corpus consisting of texts of the same text type (as defined and discussed in Chapter 4), and thus fulfills the DCK requirement for generation[3].

The SIGNAL toolkit represents a holistic approach to take advantage of stochastic approaches at many stages of the text generation process. To the best of my knowledge, the SIGNAL approach is the first to use lexical dependencies in combination with language constructs at different levels of granularity (word, phrase, sentence) to allow for flexibility in lexical and syntactic choice. In summary, SIGNAL knits together such acquired resources for high-level content determination and low-level surface realization using acquired resources in sentence planning to generate new texts.

[3]Texts outside the expository genre (e.g., augmentative or narrative text or dialog) have a significant rhetorical element and would not be handled well by SIGNAL as currently implemented.

Chapter 8

Evaluation

Throughout this thesis, I have introduced and illustrated how the various system components of CENTRIFUSER and SIGNAL work. I have also done component-wise evaluations of each of these subsystems:

- Evaluations of the segmentation algorithms in Chapter 3: of both the SEGMENTER linear segmentation algorithm, and of the CLASP combined approach;
- An evaluation of the composite topic tree merging algorithm in Chapter 4;
- An evaluation of components within the SIGNAL statistical generation framework in Chapter 7, including evaluation of the automatic semantic annotation, evaluation of inferred global content plan orderings, and an evaluation of the stylistic evaluation functions.

However, up to this point in the thesis, CENTRIFUSER as a whole has not been evaluated. This chapter addresses this issue, and describes the comparative usability evaluation of CENTRIFUSER. In a nutshell, I examined CENTRIFUSER task-based user satisfaction in providing medical health information to lay consumers. The

domain of medicine and more specifically heart disease and genre of patient and consumer information have been discussed throughout this thesis. The evaluation is comparative in the sense that CENTRIFUSER was assessed against three other IR systems that also indexed and presented health information, namely with Google, Yahoo! and About.com. The evaluation was set up to assess task-based usability, in which qualified subjects examined each system's answers to relevant to their questions about specific health conditions.

Along with a team of evaluation specialists, I conducted an evaluation that assessed the CENTRIFUSER system in comparison to currently available Web-based search engines. An approach based on usability testing and cognitive analysis was used, which employed subjects consisting of relatives and friends of patients in a real health care settings (a waiting room outside an operating room at the Columbia Presbyterian Medical Center). To conduct the evaluation, I employed full video recording of user interactions and audio recording of subjects thinking aloud while using CENTRIFUSER and the other search engines. The objective of this work included assessing CENTRIFUSER's capability in addressing users' information needs along the dimensions of content and user interface.

Results of this evaluation showed that subjects found CENTRIFUSER's summarization capability useful and easy to understand. In comparing CENTRIFUSER to the three other search engines, subjects ratings varied; however, specific interface features were deemed useful across interfaces.

8.1 Methods

The experimental design followed a similar design for an earlier healthcare evaluation of the MAGIC system (McKeown et al., 2000). The study was approved by the University's Internal Review Board as exempt from board review under exemptions 14057b, X10064.

I chose the specific IR user interfaces in the evaluation to sample the wide range of features and paradigms used in the current information retrieval environment (overviewed in Appendix A). As there were no domain specific search engines for the text type of medical consumer health information, I sampled the range of domain-independent search engines. The three that I chose to evaluate CENTRIFUSER against exhibit IR retrieval with different levels of human intervention – from fully automatic to fully manual.

1. Google was chosen to represent the standard ranked list interface. The user model of search is explicitly supported with such interfaces, as discussed in Chapter 2. Entirely automated, Google's interface and query word in context (QWIC) descriptions are constructed solely by automatic means.

2. Yahoo! represents an interface that puts more emphasis on browsing and which harnesses human knowledge to construct a high quality hierarchy of web sites. Human editors employed by Yahoo! have to approve submitted descriptions (or "snippets" as referred to by Amitay (2000)) before the content is added to Yahoo!'s site. Thus, all content in Yahoo! is edited by employees in the company for quality assurance and consistency.

3. About.com represents an interface that relies solely on humans to mediate the information seeking task. Volunteer expert guides construct web pages within the About.com site on their specialty. They are responsible for creating, editing and reviewing content to construct their specialist site as well as to link to others. Another possible choice to represent this category was WebMD.

CENTRIFUSER in its default configuration (and as tested) represents a system that constructs its IR user interface in a fully automated manner. By comparing CENTRIFUSER to systems with different levels of human involvement, one can assess

the usability gains and losses of CENTRIFUSER normalized against the upper-bound of entirely human-mediated approaches.

In this evaluation, it was difficult to control for independent variables outside of ones that directly affect the usability of the user interface. For example, each of the search engines draws on a different body of indexed documents. Thus, there is no way to control the search engines to ensure that the documents returned for a search are identical. Furthermore, CENTRIFUSER itself is a post-processor that works with the result set returned by a search engine, and thus has no native body of indexed documents. To establish parity with at least one system, I decided to test CENTRIFUSER as a post-processing module for the documents returned by Google for its first page. CENTRIFUSER's length settings were set to give a single screen of information as output (10 sentence extract synopsis length and 10 message length group indicative summaries) as determined by a 800 by 600 pixel resolution screen and 12 point font.

Other independent variables such as ordering and brand recognition were controlled for, as will be discussed in the **Procedure**.

Queries: Medical professionals were consulted to select three widely applicable cardiac medical conditions that were used in evaluating the interfaces. Diabetes, hypertension (high blood pressure) and angina (chest pain) were chosen.

Subjects: Thirteen subjects participated in this study. All subjects were recruited from the waiting room at the cardiac and surgical intensive care and were either friends or relatives of patients undergoing treatment at the hospital for one of the three conditions described above. Most subjects were individuals that went through the study alone; however two of the subjects were actually groups of two and three family members that decided to perform the evaluation together due to time constraints or comfort levels. Agreement between these composite subjects did

not turn out to be a problem as these groups readily reached a consensus opinion.

Procedure: Subjects were asked to select one of the three conditions that they wanted further information about "tell me about angina", "tell me about diabetes", or "tell me about hypertension". Subjects were then sequentially presented with their selected query results as displayed by the four systems (CENTRIFUSER, Google, Yahoo and About.com) in random order.

Subjects were allowed to examine the initial screen of results that were returned by each search engine. Because of the controlled nature of the experiment, I executed the searches on all four search engines in preparation for the experiment and stored their results for the experiment to display. This was necessary to remove variables that did not directly associate with the usability of the interface, such as time to execute the searches and with problems with network connectivity.

In order to control for any subject's bias associated with the commercial search engines, I constructed and executed a procedure to anonymize each search engine's content. The procedure included changing of identifying colors and standardizing font sizes, removing graphical images, replacing identifying text with anonymized text (e.g., "Google search results" with "System C search results"), and replacing hyperlinks with blue underlines such that subjects that inadvertently tried to click off of the initial page would not be able to do so. The evaluation framework was programmed to present the interfaces in randomized order, and furthermore, to balance the randomization such that an equal number of subjects would view any one ordering. As there were only 13 subjects and $4! = 24$ orderings, each ordering was examined only by one or zero subjects. Since the evaluation framework needed to present the anonymized systems in linear order, the first system was given the name "System A" and the last "System D", regardless of the actual system that was assigned to the position in the ordering.

Before starting an interview, subjects were informed that they could stop

the experiment at any time and that they were allowed to skip or refuse to answer any questions, as per the university's Internal Review Board protocol. Once subjects assented to be interviewed, they were asked to verbalize their thoughts or "think aloud" as they examined each of the interfaces. Additionally, subjects were probed about their thoughts regarding certain aspects of the interface, i.e., its ability to fulfill their information needs, its ability to allow for navigation, and its presentation. After viewing all four interfaces, subjects were then asked to complete seven-point Likert scales (Barnett, 1991) addressing the following areas: a) usefulness of content, b) types of information available, c) ease of deciding next step, d) ease of locating information, e) layout and f) overall satisfaction.

Subject interviews were slated to be 30 minutes in length, with five minutes for viewing and thinking aloud as they saw each of the four interfaces and the remaining ten minutes for the cross comparison rating assessment. As I interviewed all thirteen subjects, I found that many subjects found the task to be quite engaging once they started, and took anywhere between 30 to 75 minutes to complete an interview. Before beginning or after finishing the evaluation, subjects were asked to fill out a demographic survey. The survey was also optional; a couple of the participants decided not to participate in the survey.

Data analysis: All of the numerical ratings from the six Likert scales on the interfaces were first tabulated. Demographic data was likewise compiled. The audio portion of the subjects' "thinking aloud" and response to probes was transcribed verbatim and third-party colleagues were asked to verify that the probes and lines of questioning were unbiased toward eliciting positive or negative comments on a particular search engine.

A coding scheme was adapted from related work on health information systems to tag comments on various aspects of the usability of the interfaces (Kushniruk, Patel, and Cimino, 1997). The scheme included categories for subject com-

ments on: understanding of information, usefulness of information, content of information, linkages to other sites, organization of information, interface consistency, and understanding labels and instructions. Both the audio and video transcripts were enriched with these tags by applying an analysis of video data of human-computer interaction previously developed in previous literature, by (Kushniruk, Patel, and Cimino, 1997) annotating the verbal transcripts with the codes.

Oh this is good, because it gives you a lot of, the angina, you can find the definition, the cause, symptoms and
treatments, I think it covers everything, it gives definition, gives symptoms cause and treatment
COMMENT: RANGE OF CONTENT AND COVERAGE OF MATERIAL
This sums up everything we want in a nutshell, I find the synopsis very useful
COMMENT: USEFULNESS OF CONTENT - SYNOPSIS
This is good here because it tells you the different kinds, tells you there are four articles
COMMENT: USEFULNESS OF CONTENT - ARTICLES
This "differences between documents", I'm assuming this will show me the different types of angina with heart attacks,
this is not clear
POTENTIAL PROBLEM: UNDERSTANDING OF LABEL ("DIFFERENCES BETWEEN ARTICLES")

Figure 8.1: Excerpt of a coded transcript of a subject while examining the CENTRIFUSER system.

8.2 Results

Each hour of video data took about two to three hours to code and analyze. The coding was reviewed by a second research assistant, with minor disagreement being resolved during subsequent discussion. An excerpt from the coded transcript of a subject "thinking aloud" while interacting with CENTRIFUSER is given in Figure 8.1 (coded comment categories are in bold and numbers indicate the corresponding time offset).

8.2.1 Qualitative usability analysis

Initial reviews of the collected "think aloud" protocols indicated that the different systems tested (CENTRIFUSER and the three search engines) all had aspects or features which some subjects preferred and other aspects which were considered by subjects as being problematic, or in need of improvement. Indeed, no one interface was found to be clearly superior from the users' perspectives, but rather certain features of the different interfaces and systems were identified as being useful or as being problematic by the majority of subjects. In order to obtain preliminary data on which features of the systems were identified as being useful or problematic, the coded comments (again, see Figure 8.1) were classified as being either positive or negative with respect to the particular system feature. For example, comments where the subject indicated there was a problem with regard to a specific feature of the system, such as poor navigation capability, would be classified as a negative comment regarding that feature. Table 8.1 summarizes the data that was coded this way to indicate frequency of positive and negative comments made by subjects.

Examination of the frequencies of positive and negative comments indicate certain patterns regarding what system features were found desirable or in need of improvement. For example, while subjects liked About.com for its clarity of labeling and its range of linkages to broad resources (see category "Range of Information Available"), they were critical of the relevance of links that Google provided (several subjects commented that they felt Google did not filter their information request very well, providing links to many irrelevant sites), in the context of their specific health care question. For several subjects, the amount of information that was provided by Yahoo was seen as problematic. The majority of positive comments regarding the content of information provided (in terms of it usefulness and understandability) were made about CENTRIFUSER. Specific comments were made by several of the subjects regarded the perceived usefulness of having a synopsis

and differences made available to them in response to their queries (as illustrated in the excerpt in Figure 8.1).

Features:	Search Engine							
	CENTRIFUSER		Yahoo		Google		About.com	
	+	-	+	-	+	-	+	-
Content - Usefulness	10	1	2	1	1	4	3	1
Content - Overall understanding	9	3	1	1	1	1	4	1
Organization of information	2			5	1		5	4
Understanding labels			5		2		3	7
Navigational ability	3		1	1		1	1	2
Effort to find information	1	1		1		5	1	
Relevance of Links		1			1		9	
Amount of information		2		6		2	1	4
Number of links available	2	1	1			4	3	
Range of information available	4		1		1		10	
Format/layout of information						1	2	1
Search capability		1					1	

Table 8.1: Frequency of positive (+) and negative (-) coded verbal comments made by subjects regarding the usability and content of the four systems tested.

In-depth analysis of the transcripts was conducted to pinpoint the nature of the users' comments for each of the coded and time-stamped issues that they raised. For example, for the "Understanding labels" category in Table 8.1, five subjects who commented on this feature said it was unclear what information would be contained in the indicative grouped summaries section of CENTRIFUSER's interface, as it was confusingly labeled as "Differences between documents". Comments related to content of the information provided were likewise considered in light of the verbatim transcripts, one subject indicated that the synopsis generated by the

system should always begin with a definition of the medical condition the text dealt with, while analysis of interaction with another subject indicated the reading level of the synopsis in terms of medical content might need to be adjusted to take into account users with less education. This area is addressed in subsequent work that extends CENTRIFUSER, in which definition mining from lay documents (Klavans and Muresan, 2000) is employed to expand undefined medical terms in documents. Finally, CENTRIFUSER was set to produce only a single page of results (although it could have been set to provide a higher level of detail), whereas Google and About.com often gave verbose results (up to five pages). As such, CENTRIFUSER was able to achieve its satisfaction level with subjects with much less information, but was penalized by two subjects for having less information compared to the other interfaces.

8.2.2 Comparative ratings of all interfaces

As mentioned, after examining the interfaces sequentially, subjects assigned numerical ratings to compare and rank the four interfaces. As the focus of the evaluation as a whole emphasized capturing qualitative feedback that was quite time intensive, statistical significance was not reached and results in this quantitative section are not definitive. Figure 8.2 shows the average score across the 13 subjects for each question and system combination, while Table 8.2 (found at the end of the chapter) gives the raw results of the quantitative survey.

About.com's human expert site consistently outperformed all of the other system interfaces, with an emphasis on high quality content and range of different ways to access that information (questions 1 and 2: content and information types). Yahoo's human-created hierarchy performed next best, consistently outscoring or equaling the remaining two systems. Yahoo performed least well, comparatively, in providing different information access mechanisms (again, question 2: information

types).

CENTRIFUSER and Google form the lower tier. They both used the same underlying documents (since CENTRIFUSER post-processes Google output), but had their strengths and weaknesses. According to subjects, CENTRIFUSER's layout provided an easy way to locate relevant information, whereas Google's consistent placement of links in a long list may have been the reason that subjects found it easy to decide what action to take next, but disliked the cognitive effort of searching the individual descriptions of documents to assess relevance. CENTRIFUSER's ability to produce clustered indicative summaries alleviated these same concerns, but the comparative novelty of the types of information provided by CENTRIFUSER may have contributed to the confusion subjects found in deciding what to do next.

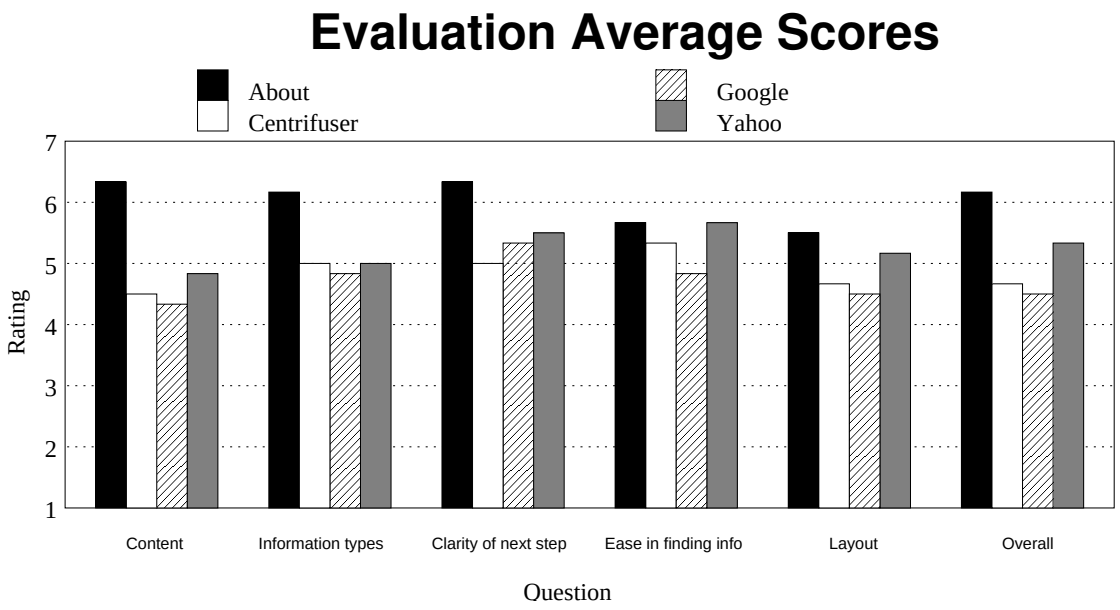


Figure 8.2: Quantitative evaluation based on raw data in Table 8.2.

8.2.3 Demographics

In even such a small sample of data such as the one presented here, there were still a few strong trends that were evident. Tables 8.3 and 8.4 show the summed

results compiled over the survey. The survey indicated that practically all of the subjects had a quite a bit of computer experience and familiarity, both at work and at home. As the interviewer and the recruiter of patients, I noticed that several potential subjects opted not to help with the evaluation when the word "computer" was mentioned. I was only able to persuade one subject (subject #14) with the help of our collaborating surgeon to take the survey after telling the subject that she would only have to look at the screen of the computer while I showed the interfaces.

Another aspect that came to light in the survey was that the majority of the subjects came from an older population and were somewhat affluent. The former can be explained by the fact that younger people were less likely to have friends or relatives that were affected by these cardiac diseases. The latter can be explained by the fact that exposure to computers is correlated to a level of wealth, and that being treated at a high-quality cardiac intensive care unit is similarly correlated.

8.3 Discussion

In this chapter, I have employed a usability engineering approach to evaluate the comparative performance of CENTRIFUSER. By collecting and analyzing both video and audio data on users interactions with the system, I have been able to characterize those aspects of web interfaces that are useful to health information seekers. Additionally, by coding for categories of user comments, I have located areas where healthcare information systems can be improved. In general, this approach to data collection, analysis and reprogramming has lead to systems that are more acceptable in areas such as healthcare (Coble et al., 1997). With the widespread use of Web-based information resources by patients and their families, this type of user-centered evaluation is increasingly important (Pearrow, 2000).

With this system-wide evaluation, CENTRIFUSER has been validated as a

useful paradigm in information retrieval, addressing the problem of selection and access to information in the information seeking process. Although the evaluation found that no single system contained features or capabilities that completely met the needs of all subjects, general trends were found where one system was rated slightly higher than another on particular criteria, analysis of the "think aloud" data indicated that there are features of each of the interfaces tested which subjects preferred (e.g., the capability of providing users with a focused summary of multiple sources of information, as CENTRIFUSER does). As simplicity in search systems tends to do better than overly feature-rich systems, it is likely that future search technology will have to employ user modeling to find out which features are most well-suited for each population group.

Subjects in the evaluation gave strong marks to CENTRIFUSER in providing high quality content in a succinct manner by utilizing automatic text summarization. This was shown in both the analysis of the think aloud transcripts as well as by the quantitative rankings of the systems given by the survey data. As stated, CENTRIFUSER either outperforms or performs equally well as Google on most criteria, despite errors generated in the production of the summaries. By having subjects compare CENTRIFUSER with three search engines of different levels of automation, my initial hypothesis was confirmed: automatic text summarization can add to the level of usability and satisfaction in the standard ranked list interface.

Disease	hbp	ang	hbp	dia	ang	hbp	dia	dia	ang	ang	ang	ang	hyp
Ordering	cyag	cgay	agcy	gcya	cgya	cagy	cayg	gcay	ycga	acgy	acgy	yagc	agyc
Subject #	2	3	4	5	6	7	8	9	10	11	12	13	14
Question & System*													
1a	7	7	7	7	7	7	4	6	6	7	7	4	5
1c	6	3	5	7	2	6	6	5	4	4	6	3	1
1g	4	4	3	7	2	7	2	7	2	5	6	4	4
1y	4	2	4	7	3	6	6	5	7	3	6	5	6
2a	7	7	7	7	6	6	5	6	7	5	6	4	7
2c	5	3	5	7	6	6	5	6	5	6	5	5	1
2g	3	4	5	7	5	6	3	6	7	3	7	4	4
2y	3	3	6	7	5	6	6	5	7	3	5	5	4
3a	7	7	7	7	5	7	5	6	7	7	5	4	7
3c	3	4	3	7	6	7	6	6	7	5	6	5	1
3g	6	4	6	7	6	6	2	6	6	6	5	4	5
3y	3	4	7	7	6	7	6	6	7	2	6	5	4
4a	7	n/a	7	7	4	7	5	5	7	3	7	4	n/a
4c	6	n/a	3	7	6	5	5	5	7	2	7	4	n/a
4g	4	n/a	4	7	3	6	2	6	6	6	6	4	n/a
4y	4	n/a	7	7	5	7	6	5	7	4	7	4	n/a
5a	7	7	7	7	1	7	6	6	7	3	5	4	6
5c	6	4	6	7	2	7	5	4	4	6	6	4	1
5g	3	5	6	7	3	7	2	6	3	5	5	4	4
5y	3	3	5	7	3	7	6	5	7	5	7	4	6
6a	7	7	7	7	4	7	6	6	7	6	7	4	5
6c	6	3	5	7	2	6	5	5	5	5	6	4	1
6g	3	5	5	7	3	6	3	6	2	4	7	4	4
6y	3	3	6	7	4	7	6	6	7	3	5	4	5

* Question: 1) Content, 2) Information types, 3) Clarity of next step 4) Ease in finding information, 5) Layout, and 6) Overall satisfaction
System: A)bout, C)entrifuser, G)oogle, and Y)ahoo!

Table 8.2: Evaluation raw ranks.

Age (average)	**44.8 years old**
Less than 20 years old (20)	1
20-35 (27.5)	1
35-45 (40)	4
45-60 (52.5)	5
60-75 (67.5)	1
Sex (mode)	**Female**
Male	4
Female	8
Highest Education Background Completed (average)	**2.9**
Elementary School (1)	1
High School (2)	2
Technical School (2)	3
College Degree (Arts) (3)	1
College Degree (Science) (3)	1
Master of Arts (4)	3
Master of Science (4)	1
Ph.D. (5)	1
Salary (average)	**$54.7K**
Student	1
$30K - $45K (37.5)	2
$45K - $75K (60)	4
$75K - $100K (87.5)	1
Over $100K (100)	2
No response	1
Student? (mode)	**No**
Yes	1
No	11

Table 8.3: Evaluation summed general demographics. Weights used for averages shown in parentheses. Only non-zero summed lines shown.

Computer Use - Times per Day (average)	**Once a week (7)**
Daily (1)	6
Weekly (3)	3
Once a week (7)	1
Monthly or less (31)	2
Computer Use - Years Experience (average)	**4.1 years**
Less than 1 year (1)	1
1-3 years (2)	1
3-5 years (4)	4
Over 5 years (5)	6
Computer at Work? (mode)	**Yes**
Yes	10
No	1
Computer at Home? (mode)	**Yes**
Yes	10
No	2
Years using Internet (average)	**3.7 years**
Never (0)	1
Less than 6 months (.5)	1
1-3 years (2)	1
3-5 years (4)	3
Over 5 years (5)	6
Search Engines Used (average - 11 respondents)	**1.7**
AltaVista (1)	4
Google (1)	2
Hotmail (1)	2
Mamma.com (1)	1
Yahoo! (1)	10
Computer Systems Used (average - 11 respondents)	**1.2 types**
IBM Compatible	10
Macintosh	2
Mainframe	1
Operating Systems Used (average - 11 respondents)	**1.2 types**
DOS	1
Macintosh	1
Windows	11
Programs Used (average - 11 respondents	**3.6 program types**
Database (1)	3
Email (1)	10
Games (1)	10
Spreadsheet (1)	6
Word Processing (1)	11
WWW Use - Times per Day (average)	**Weekly (2.0)**
Never (0)	3
Monthly or less (1)	1
Weekly (2)	3
Daily (4)	4
Health via WWW? (mode)	**Yes**
Yes	8
No	0
Medical Sites Known (average - 11 respondents)	**1 site (1.0)**
Medicine Online (1)	1
NIH.gov (1)	2
Oncolink (1)	2
WebMD (1)	3
Other (1)	3

Table 8.4: Evaluation summed computer demographics. Weights used for averages shown in parentheses. Only non-zero summed lines shown.

Chapter 9

Conclusion

The research presented in this thesis identifies weaknesses in the current method of presenting search results as it occurs in the digital library and beyond, and offers both theoretical and implemented solutions to address these weaknesses. In the course of producing this thesis, I have touched upon many related disciplines of information and library science, computational linguistics and natural language generation in order to develop a complete picture of the information search and presentation process and implement practical and usable solutions to the problems identified.

In this chapter, I summarize the main research contributions of this work. I then overview the state of the current implementation, list the main limitations of this approach, discuss future research directions and conclude the thesis with implications of this work on the field of information retrieval as a whole.

9.1 Main contributions

This research makes four main contributions to the state of the art in research in natural language processing and the study of information retrieval:

- assessing the fit between user needs and the standard interface,
- hierarchical topic segmentation,
- multidocument text summarization, and
- stochastic natural language text generation.

I summarize these four individually in the following text.

9.1.1 Assessing the fit between user needs and the standard ranked list

In the survey of related work in Chapter 2, I examined the field of information and library science to review techniques used by information professionals in assisting others with the information seeking process. I divided their techniques into online approaches, where a reference librarian can interact in a dialog with the information seeker and offline approaches, in which a cataloger makes commonly used information available in a easy to utilize, simple format. Unique to my work, I characterize these techniques found in the physical library into nine different strategies, and showed how the standard ranked list user interface commonly used in digital libraries and other online searching fails to meet many of these characteristics.

This work is a contribution bridging the areas between library and information science and computer science, in that it enumerates specific professional strategies in information seeking and relates them to new facilities that can supplement and improve upon the standard IR interface. This aspect of the thesis allows CENTRIFUSER's design and implementation to be based fully on the needs distilled from the library science literature. This work alleviates concerns in some computer science based user interface work, in which systems are built without reference to user needs. Likewise, solutions proposed by human factor studies can

be computationally complex and difficult to scale up to real world environments. CENTRIFUSER reflects a balance between both necessities.

9.1.2 Document and composite topic trees

One of the goals of this thesis was to demonstrate the value of revealing document substructure to the user. In expository texts, this is particularly true: the text discussion is structured into points and subpoints, possibly concerning different topics. Information retrieval on subparts of documents is currently a topic gaining popularity, that can be thought of encompassing the areas of passage retrieval and question answering. This thesis contributes to the understanding and development of a framework that extends information retrieval from the document level to multiple levels of granularity. I have operationalized this notion in the development of topic trees, which have resulted in improvements in topical segmentation algorithms, both as the problem of partitioning documents into topical segments (the linear segmentation problem) as well as structuring these segments into a hierarchy (the hierarchical segmentation problem).

An innovation in this thesis is the development of the composite topic tree (CTT), which was discussed in detail in Chapter 4 - a topical representation of information on related items and subtopics. In the development of the CTT, I formalized the notion of a text type as a group of documents of the same domain and genre and showed how these documents exhibit regular information presentation structure and ordering. I used this observation to develop a new automatic method to compute such a composite topic tree that represents a text type's expected information content.

9.1.3 Topically targeted summarization and metadata integration

The development of CENTRIFUSER also contributes to the field of automatic text summarization. CENTRIFUSER's summarization protocols capitalize on the availability of the hierarchical document and composite topic trees which allow targeted summarization. It enhances the state of the art by allowing summarization at specific levels of granularity. This allows the system to summarize and rank just the relevant sections, rather than have the user search documents to find relevant material.

A key contribution of this work has been the investigation of how different types of metadata can contribute to better quality summaries. This is a field of work in which very little previous work exists. I have reviewed the literature on prescriptive guidelines for metadata use in summaries as well as performed two corpus studies on summary corpora to analyze to what extent guidelines are followed. I utilize the results of corpus analysis to formulate the structure and content of the document cluster indicative summaries, first described in Chapter 5.

9.1.4 Stochastic natural language generation

This thesis also contributes to work in natural language generation by demonstrating a new statistical method for modeling and generating texts. I demonstrate an end-to-end system that uses semantically annotated texts to analyze texts and employs the resulting resources for generation of new texts for different resources. The SIGNAL toolkit makes this possible by supplying tools for both analysis and generation. Guided by a traditional generation engine, SIGNAL is able to produce valid natural variations on phrases in the training corpus, beyond those of the corpus and possibly missed by human experts.

The analysis tools allows the induction of several resources for generation, including content planning constraints, a lexicon of partial surface realization for semantic predicates, and occurrence statistics for aggregation and stylistic phenomenon. The generation tools use these resources, coupled with new semantic input, to generate new texts that meet the content ordering and stylistics constraints found in the training corpus.

9.2 Deliverables

Aside from the theoretical contributions that this thesis makes, this thesis contributes to the state of the art by making tools available to the community. CENTRIFUSER is a fully implemented system, and its components for producing summaries, building topic trees, evaluation, integration with other projects, demonstration and data collection and annotation are available to the public and have been released and used by other researchers. The SIGNAL toolkit is also fully implemented and is available for further use. Aside from the code used to produce CENTRIFUSER and SIGNAL, I enumerate some of the deliverables that this thesis has generated.

- `Topics.pm` - A perl module use by the core of CENTRIFUSER's executables, the `Topics` module incorporates many of the basic functions for instantiating topics in both document and composite topic trees, calculating topic similarity, and facilitates the calculation of composite trees. It also contains many debugging features that enable a developer to monitor the growth and analysis of topic trees.

- `centrifuser.cgi` - To present and report this work to the community, CENTRIFUSER needed a web interface. The development of the web interface as a CGI (Common Gateway Interface) script enabled high level debugging and

display, and incorporates both a simple and advanced mode, complete with online, context-sensitive help.

- SEGMENTER and CLASP - The SEGMENTER and CLASP text segmentation modules are themselves projects that have been distributed and utilized in the community. SEGMENTER has been distributed and licensed worldwide and has been used by over 15 researchers to support their projects. A segmentation evaluation special interest group was started to examine the feasibility of compiling reference segmentation corpora that could be used to comparatively assess different segmentation programs.

- Annotation modules - As part of the empirical, corpus-based study of both online public access catalog summaries and annotated bibliographies, I compiled a corpus of summaries (all MARC records with summary information in the MARC 520 field) from the U.S. Library of Congress as well as from sources on the web. These have been made available to the public. Further corpus studies are necessary and, to this extent, I have worked with the Library of Congress to select a corpus of over 400,000 summaries of cataloged material in their catalog for future study. Publishers' description of books are yet another form of summary and are extremely rich in content and variety. The ONIX consortium[1] has made such a corpus of over 30,000 records available for public use. I have collected the ONIX corpus and enhanced it with simple utilities to make analysis and intrarecord field retrieval easier for researchers.

  In the course of my work on the indicative summary SIGNAL and on hierarchical topical segmentation, annotation of training and testing was necessary. I have developed and used a number of web-based, Tk/Tcl-based standalone

[1] `http://www.editeur.org/onix.html`

tools for easy annotation.

- SOAP integration - Although not a focus within this thesis, CENTRIFUSER is an integral part of a larger digital library framework (PERSIVAL[2]), which aims to summarize medical multimedia documents for both experts and lay consumers (McKeown et al., 2001). CENTRIFUSER's role is in producing summaries of health information, such as disease and drug information for laypeople. As part of this larger project, I developed wrapper modules for CENTRIFUSER's integration in the project using XML and SOAP (Box et al., 2000) protocols.

9.3 Limitations and future work

The development of CENTRIFUSER and SIGNAL go some distance in fulfilling user needs in the information seeking process. However, CENTRIFUSER has several limitations that currently make it impractical to run on a large scale. I examine these obstacles and make recommendations for future research that can address these issues.

- CENTRIFUSER **currently produces summaries of documents belonging to the same text type.** Many of CENTRIFUSER's advantages derive from its ability to capitalize on the composite topic tree, which encodes expected information and ordering for specific text types. However, composite trees can only be derived for specific text types, thus presenting the limitation that CENTRIFUSER's techniques in its standard form can only be applied to documents of the same text types. CENTRIFUSER currently assumes that the input texts are of the same text type. Most queries retrieve documents over a range of domains and genres, and therefore over many text types. To address

[2] `http://persival.cs.columbia.edu/`

this limitation in a large-scale IR system, CENTRIFUSER needs to coupled with domain and genre classifiers that can provide accurate classification of documents with respect to domain and genre. There is much related work on genre classification (Biber, 1989; Kessler, Nunberg, and Schütze, 1997) and on domain classification (Joachims, 1998; Yang, 1999) that hold the promise of fulfilling this need. Integration of these two dimensions in future work is necessary to identify text type as it is needed for CENTRIFUSER and other applications that are sensitive to specific domains or genres.

- **Further assessment of the effects of the composite topic tree in CENTRIFUSER.** In Chapter 5, I showed how CENTRIFUSER utilizes the CTT to directly construct the navigation controls and contribute ordering and frequency information needed for constructing the information extract for browsers and the indicative summaries for searchers. It has been noted that this information could be approximated by weaker data structures (e.g., a bag of topics rather than a tree structure). The evaluation of the CTT by reference librarians carried out in Chapter 4 will be further extended to quantitatively assess the impact of utilizing tree structure in both the intrinsic task of topic alignment and the extrinsic task of user satisfaction with tree structured navigation.

- **Controlled evaluation of CENTRIFUSER versus other IR frameworks.** In the extrinsic evaluation of CENTRIFUSER given in Chapter 8, I evaluated CENTRIFUSER versus Google, Yahoo! and About.com to assess the relative merits of these interfaces versus CENTRIFUSER. However, in employing both Yahoo! and About.com in such a comparative evaluation, I could not control for the source documents that were used in the evaluation. Although the evaluation controlled for these source documents in its comparison with Google (as CENTRIFUSER used the same first ten documents from Google), future testing of CENTRIFUSER in comparison with About.com, Yahoo and

other health sites such as WebMD will be carried out with documents downloaded from those sources. This will allow an authoritative assessment of the benefits of CENTRIFUSER.

- **Link SIGNAL with CENTRIFUSER.** In Chapter 7, I discussed why it was not possible to directly transfer the work on the statistically modeled natural language generation to build proper indicative summaries for CENTRIFUSER: a) the corpus used in SIGNAL is a corpus of independent, single document summaries, in contrast to the multidocument, interrelated summaries generated by CENTRIFUSER; b) a proper corpus of multidocument, interrelated summaries does not currently exist. These facts mean that I cannot run the SIGNAL process to generate the proper output for CENTRIFUSER's scenario. However, if I relax my definition and conflate the two summary types together, it is possible to bring the content planning and partial surface realization knowledge gleaned by SIGNAL from the single document indicative summary corpus to build a more sophisticated natural language generation engine for indicative summaries within CENTRIFUSER.

Aside from the limitations of CENTRIFUSER and SIGNAL, there are some natural extensions of this thesis in the direction of future research. The use of metadata is a primary focus of this dissertation and is an area I believe of importance in future research in information retrieval, that joins analytical user studies done by information science with the algorithmic and statistical techniques of computer science. I look briefly at three routes of future research:

- **Inferring metadata automatically** - Although a sizable and nontrivial task, cataloger and specialist annotated metadata provided in the physical library is being transferred into digital form. As I have demonstrated throughout this thesis, this metadata – data beyond the raw contents of the document

– provides a summary of the information in the document at a high level and with particular perspectives. Annotating metadata is an extremely time consuming task that catalogers and subjects specialists perform in the library. Having tools that can automatically infer metadata values or aid a specialist creating the metadata has a large potential in altering and improving the quality and quantity of metadata annotation. General, domain independent metadata inference is likely to be very difficult and infeasible, thus inferring metadata for small, limited domains seems to be a good approach to test automated techniques. This is the approach used by a Syracuse team (Sutton, Liddy, and Kendall, 2002) in their research to automatically infer metadata values for the limited domain of lessons plans used in state, national and international standards.

- **Examining metadata needs among different domains and context** - Another reason to concentrate on the inference of metadata in limited domains rather than general ones is that metadata has different levels of usefulness among different domains and for different types of media. My study in Chapter 6 on the salience of metadata attributes covers all varieties of textual documents; however, within specific areas the distribution of importance will differ. For example, for children's books, the readability and audience metadata may play a larger role. In scientific articles, the authority and credential of the authors may play a similarly important role. For different types of media, such as maps (images) or CD-ROMs (computer software), different types of metadata are going to be critical. Developing tools to automatically discover the important metadata fields within a set domain will be an important tool to verify expert assertions about metadata in larger domains and to facilitate discovery of metadata priorities in smaller, unexamined domains.

- **Metadata values in digital context** - In this thesis, I have consider documents to stand alone and function independently of each other. The coupling between different books in the physical library exists only through bibliographic citations and cataloging records. However, in the World Wide Web and in other digital media, the distance between resources are lowered, even transparent. This interlinking causes problems in generating the right metrics to generate metadata for summarizing web pages. Developing a framework where consumer information on angina is indexed with parity regardless of whether it is provided on a single web page or scattered on interlinked pages will be of the utmost importance in the interconnected digital library.

- **Diversifying capabilities in** CENTRIFUSER - To bring CENTRIFUSER into maturity as a real-world IR system, additional capabilities must be added to the current prototype. Modifications needed include a probabilistic framework for query mapping and the implementation of restricted search and resorting. Complex queries often touch upon many topics, and the current algorithm which maps a user's query to a single topic will not suffice for many scenarios. A probabilistic framework where topics can be fractionally mapped to the query is a potential solution to this problem. Additionally, the standard ranked list interface (as defined in Appendix A) includes the ability to do search limits. Adding this in addition to supporting Characteristic 4 (support alternative search methods) would further empower users in the information seeking process.

9.4 Revisiting the information seeking process in the digital library

In 1945, Vannevar Bush foresaw that advanced technology would be able to meet the storage demands of growth in scholarship across all fields (Bush, 1945). The bottleneck, he forecasted, lay in the problem of access and communication of the vast amount of data – which he termed the problem of selection. As a testament to his vision, this problem is exactly the bottleneck in information access, and has been called by other names such as "information overload".

Selection, as Vannevar defines it, is the still the main problem in the information seeking process. Studies have shown that both the physical and digital libraries often do contain the materials and answers needed by a scholar but that mechanical barriers of indexing and conceptual barriers of using a search interface often prevent a scholar from finding that information (Borgman, 1986). I have posited that **automatic text summarization can augment the standard interface in tailoring results to users needs.** Through the development of CENTRIFUSER in this thesis, I have shown how this hypothesis is verified and makes a step in addressing the conceptual barriers in using online information retrieval systems.

References

Amitay, Einat. 2000. Trends, fashions, patterns, norms, conventions... and hypertext too. Technical Report 66, CSIRO.

Amitay, Einat. 2001. Trends, fashions, patterns, norms, conventions ... and hypertext too. *Journal of the American Society for Information Science and Technology*, 52(1):36–43.

Anklesaria, F., M. McCahill, P. Lindner, D. Johnson, D. Torrey, and B. Alberti. 1993. The internet gopher protocol. Technical report, Network Working Group, University of Minnesota, March.

Anne Arundel Community College. 1998. Writing an annotated bibliography. http://www.aacc.cc.md.us/library/annobib.htm, Last accessed March 2002.

ANSI. 1979. American national standard for describing books in advertisements, catalogs, promotional materials and book jackets. New York, USA. ANSI Z39.13-1979.

Balay, Robert, editor. 1996. *Guide to Reference Books.* American Library Association, Chicago, USA, 11th edition. R028 G944.

Bangalore, Srinivas, John Chen, and Owen Rambow. 2001. Impact of quality and quantity of corpora on stochastic generation. In *Proc. of the 2001 Conf. on Empirical Methods in Natural Language Processing (EMNLP01)*, Pittsburgh, Pennsylvania.

Bangalore, Srinivas and Owen Rambow. 2000. Corpus-based lexical choice in natural language generation. In *Proc. of the 38th Meeting of the Association for Computational Linguistics (ACL'00)*, Hong Kong, China.

Barnett, Vic. 1991. *Sample Survey principles and methods.* Oxford University Press, New York, New York, USA.

Barzilay, Regina, Noemie Elhadad, and Kathleen McKeown. 2001. Sentence ordering in multidocument summarization. In *Proc. of Human Language Technology '01*, San Diego, CA, USA.

Barzilay, Regina and Kathleen R. McKeown. 2001. Extracting paraphrases from a parallel corpus. In *Proc. of ACL/EACL 01.*

Barzilay, Regina, Kathleen R. McKeown, and Michael Elhadad. 1999. Information fusion in the context of multi-document summarization. In *Proc. of the 37th Annual Meeting of the Association of Computational Linguistics (ACL '99).*

Bates, Marcia J. 1979. Information search tactics. *Journal of the American Society for Information Science*, 30(7):205–214.

Beeferman, Doug, Adam Berger, and John Lafferty. 1997. Text segmentation using exponential models. In *Proc. of the 2nd Conf. on Empirical Methods in Natural Language Processing*, pages 35–46.

Belkin, Nicolas J., Colleen Cool, Adelheit Stein, and Ulrich Thiel. 1994. Cases, scripts, and information-seeking strategies: On the design of interactive information retrieval systems.

Belkin, Nicolas J., R. D. Hennings, and T. Segger. 1984. Simulation of a distributed expert-based information provision mechanism. *Information Technology*, 3(3):122–141.

Benford, Steve, Dave Snowdon, Chris Greenhalgh, Rob Ingram, Ian Knox, and Chris Brown. 1995. VR-VIBE: A virtual environment for cooperative infor-

mation retrieval. In *Proc. of Eurographics '95*, pages 349–360, Maastricht, The Netherlands, August.

Berland, Matthew and Eugene Charniak. 1999. Finding parts in very large corpora. In *Proc. of the 37th Annual Meeting of the Association of Computational Linguistics (ACL '99)*.

Biber, Douglas. 1989. A typology of English texts. *Linguistics*, 27:3–43.

Blum, Avrim and Tom Mitchell. 1998. Combining labeled and unlabeled data with co-training. In *COLT: Proc. of the Workshop on Computational Learning Theory, Morgan Kaufmann Publishers*.

Borgman, Christine. 1986. Why are online catalogs hard to use? lessons learned from information-retrieval studies. *Journal of the American Society for Information Science*, 37(6):387–400.

Box, Don, David Ehnebuske, Gopal Kakivaya, Andrew Layman, Noah Mendelsohn, Henrik Frystyk Nielsen, Satish Thatte, and Dave Winer. 2000. Simple object access protocol (soap) 1.1. Technical Report 08 May 2000, W3C, May.

Brin, Sergey and Lawrence Page. 1998. The anatomy of a large-scale hypertextual Web search engine. In *Proc. of the 7th Int'l World Wide Web Conf. (WWW7)*, pages 107–117, Brisbane, Australia, April.

Brown, G. and G. Yule. 1983. *Discourse Analysis*. Cambridge Textbooks in Linguistics. Cambridge University Press.

Bush, Vannevar. 1945. As we may think. *The Atlantic Monthly*, 176(1):101–108, July.

Callan, James P., Zhihong Lu, and W. Bruce Croft. 1995. Searching Distributed Collections with Inference Networks. In E. A. Fox, P. Ingwersen, and R. Fi-

del, editors, *Proc. of the 18th Annual Int'l ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 21–28, Seattle, Washington. ACM Press.

Chamlers, M and P Chitson. 1992. BEAD: Exploration in information visualization. In *Proc. of 15th SIGIR '92.*

Chen, Hao and Susan Dumais. 2000. Bringing order to the web: Automatically categorizing search results. In *Proc. of the ACM SIGCHI Conf. on Human Factors in Computing Systems (CHI '00)*, pages 145–152.

Chen, Hao, Jianying Hu, and Richard W. Sproat. 1999. Integrating geometrical and linguistic analysis for e-mail signature block parsing. *ACM Transactions on Information Systems (TOIS)*, October.

Clapp, Verner W. 1950. The role of bibliographic organization in contemporary civilization. In Jesse H. Shera and Margaret E. Egan, editors, *Bibliographic Organization: papers presented before the Fifteenth Annual Conf. of the Graduate Library School.* University of Chicago Press, Chicago, pages 3–23. Z 1001.U54 1950g.

Coble, JM, J. Karat, M. J. Orland, and M. G. Kahn. 1997. Iterative usability testing: Ensuring a usable clinical workstation. In *Proc. of the 1997 AMIA Annual Fall Symposium*, pages 744–748.

Cochran, William Gemmell and Getrude G. Cox. 1992. *Experimental Designs.* Wiley-Interscience, 2nd edition.

Cohen, Robin. 1984. A computational theory of the function of clue words in argument understanding. In *Proc. of the 1984 Int'l Computational Linguistics Conf. (COLING 84)*, pages 251–255, California, USA.

Cohen, William, Robert Schapire, and Yoram Singer. 1999. Learning to order things. *Journal of Artificial Intelligence*, 10:243–270.

Cohen, William W. 1995. Fast effective rule induction. In *Proc. 12th Intl. Conf. on Machine Learning*, pages 115–123. Morgan Kaufmann.

Collins, Michael John. 1996. A new statistical parser based on bigram lexical dependencies. In *Proc. of the 34th ACL*, Santa Cruz.

Cremmins, Edward T. 1982. *Art of Abstracting*. ISI Press.

Cutting, Douglas R. 1997. Real life information retrieval: Commercial search engines. Part of a panel discussion at SIGIR 1997: Proc. of the 20th Annual ACM SIGIR Conf. on Research and Development on Information Retrieval.

Dalianis, Hercules. 1996. *Concise Natural Language Generation from Formal Specifications*. Ph.D. thesis, Royal Institute of Technology, April.

DeJong, Gerald F. 1982. An overview of the FRUMP system. In Wendy G. Lehnert and Martin H. Ringle, editors, *Strategies for Natural Language Processing*. Erlbaum, Hillsdale, pages 149–176.

Dieckmann, Sylvia and Urs Hölzle. 1997. The space overhead of customization. Technical Report TRCS 97-21, University of California, Santa Barbera, December.

Duboue, Pablo A. and Kathleen R. McKeown. 2001. Empirically estimating order constraints for content planning in generation. In *Proc. of the ACL-EACL 2001*, Toulouse, France.

Egan, Dennis E., Joel R. Remde, Louis M. Gomez, Thomas K. Landauer, Jennifer Eberhardt, and Carol C. Lochbum. 1989. Formative design-evaluation of superbook. *ACM Transactions on Information Systems*, 7(1):30–57.

Elhadad, Michael. 1993. *Using Argumentation to Control Lexical Choice: A Functional Unification Implementation.* Ph.D. thesis, Columbia University, New York, New York, USA.

Elhadad, Michael and Jacques Robin. 1996. An overview of SURGE: a reusable comprehensive syntactic realization component. Technical Report 96-03, Ben Gurion University, Beer Sheva, Israel.

Engle, Michael, Amy Blumenthal, and Tony Cosgrave. 1998. What is an annotated bibliography. http://www.crk.umn.edu/library/links/annotate.htm, Last accessed March 2002.

Esposito, Floriana, Donato Malerba, and Giovanni Semararo. 1994. Multistrategy learning for document recognition. *Applied Artificial Intelligence*, 8:33–84.

Fields, Robert E. and Nicholas A. Merriam. 1997. Modelling in action. Report from the DSVIS '97 working group.

Flesch, Rudolf Franz. 1946. *The art of plain talk.* Harper and Brothers, New York, 1st edition. 829.69 F631.

Ford, Nigel. 1999. Information retrieval and creativity: towards support for the original thinker. *Journal of Documentation*, 55(5):528–542.

French, James C. and Allison L. Powell. 2000. Metrics for evaluating database selection techniques. In *Proc. of World Wide Web.*

Furnas, George W. 1986. Generalized fisheye views. In *Proc. of the ACM SIGCHI Conf. on Human Factors in Computing Systems (CHI '86)*, pages 16–23, New York, USA. ACM Press.

Goldberg, David, David Nichols, Brian Oki, and Douglas Terry. 1992. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35(12):61–70, December.

Goldstein, Jade. 1999. Automatic text summarization of multiple documents. Thesis Proposal, Carnegie Mellon University.

Gravano, Luis, Chen-Chuan K. Chang, Héctor García-Molina, and Andreas Paepcke. 1997. STARTS: Stanford proposal for Internet meta-searching. In *1997 ACM SIGMOD Int'l Conf. On Management of Data*, pages 207–218.

Gravano, Luis and Héctor García-Molina. 1995. Generalizing GlOSS to vector-space databases and broker hierarchies. In *Int'l Conf. on Very Large Databases, VLDB*, pages 78–89.

Grice, H. P. 1975. Logic and conversation. In P. Cole and J. Morgan, editors, *Syntax and Semantics III: Speech Acts*. Academic Press, pages 41–58.

Grishman, Ralph, Catherine Macleod, and Adam Meyers. 1994. Comlex syntax: Building a computational lexicon. In *Proc. 15th Int'l Conf. Computational Linguistics (COLING 94)*, pages 268–272, Kyoto, Japan, August.

Guenther, Rebecca. 1997. Dublin core to marc crosswalk. Technical report, United States Library of Congress. http://www.loc.gov/marc/dccross.html.

Hahn, Udo. 1990. Topic parsing: Accounting for text macro structures in full-text analysis. *Information Processing & Management*, 26(1):135–170.

Harmon, Robert B. 1989. *Elements of Bibliography: a simplified approach*. Scarecrow Press, Metuchen, NJ, USA, revised edition. Z 1001 .H29 1989.

Harrison, Colin. 1980. *Readability in the Classroom*. Cambridge University Press, Cambridge, United Kingdom. ISBN 0 521 22712 7.

Hatzivassiloglou, Vasileios, Pablo A. Duboue, and Andrey Rzhetsky. 2001. Disambiguating proteins, genes, and rna in text: A machine learning approach. In *Proc. of the 9th Int'l Conf. on Intelligent Systems for Molecular Biology*, Tivoli Gardens, Denmark, July.

Hatzivassiloglou, Vasileios, Judith L. Klavans, and Eleazar Eskin. 1999. Detecting text similarity over short passages: Exploring linguistic feature combinations via machine learning. In *Proc. of EMNLP '99*.

Hatzivassiloglou, Vasileios, Judith L. Klavans, Melissa L. Holcombe, Regina Barzilay, Min-Yen Kan, and Kathleen R. McKeown. 2001. Simfinder: A flexible clustering tool for summarization. In *Proc. of the Workshop on Automatic Summarization, NAACL 2001*.

Hearst, Marti A. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proc. of COLING-92*, pages 539–545, Nantes, France, August.

Hearst, Marti A. 1993. Text tiling: A quantitative approach to discourse segmentation. Technical report, University of California, Berkeley, Sequoia.

Hearst, Marti A. 1994. Multi-paragraph segmentation of expository text. In *Proc. of the 32nd Annual Meeting of the Association for Computational Linguistics*, pages 9–16, New Mexico State University, Las Cruces, New Mexico.

Hearst, Marti A. 1995. TileBars: Visualization of term distribution information in full text information access. In *Proc. of the ACM SIGCHI Conf. on Human Factors in Computing Systems (CHI '95)*.

Hearst, Marti A. and Chandu Karadi. 1997. Cat-a-Cone: an interactive interface for specifying searches and viewing retrieval results using a large category hierarchy. In *Proc. of SIGIR-97, 20th ACM Int'l Conf. on Research and Development in Information Retrieval*, pages 246–255, Philadelphia, US.

Heery, Rachel. 1996. Review of metadata formats. *Program*, 30(4):345–373, October. http://www.ukoln.ac.uk/metadata/review.html.

Henkle, Herman H. 1950. The natural sciences: Characteristics of the literature, problems of use, and bibliographic organization in the field. In Jesse H. Shera and Margaret E. Egan, editors, *Bibliographic Organization: papers presented before the Fifteenth Annual Conf. of the Graduate Library School*. University of Chicago Press, Chicago, pages 140–60. Z 1001.U54 1950g.

Higgins, Richard. 1997. The encoded archival description: using sgml to create permanent electronic handlists. *Business Archives Principles and Practice*, 73, May.

Hillmann, Diane. 2001. Using dublin core. Technical report, Dublin Core Metadata Initiative. http://dublincore.org/documents/2001/04/12/usageguide/.

Hodges, Gail. 2000. Systems of knowledge organiziation for digital libraries. Technical report, The Digital Library Federation.

Hoffman, Lothar. 1991. Texts and text types in LSP. In Hartmut Schröder, editor, *Subject-oriented Texts: Languages for Special Purposes and Text Theory*, Research in Text Theory. Walter de Gruyter, Berlin, pages 158–66.

Hu, Jianying, Ramanujan Kashi, and Gordon Wilfong. 1999. Document image layout comparison and classification. In *Proc. of the Intl. Conf. on Document Analysis and Recognition (ICDAR)*.

Hwa, Rebecca. 2001. On minimizing training corpus for parser acquisition. In *Proc. of the 5th Computational Natural Language Learning Workshop*, July.

ISO. 1958. Bibliographical references: Essential elements. Technical Report R 77, Int'l Organization for Standardization, Switzerland. Z 1001.I52.

Ivory, Melody Y. and Marti A. Hearst. 2002. Statistical profiles of highly rated web sites. In *Proc. of the Conf. on Human Factors in Computing Systems*, Minneapolis, Minnesota, USA, April.

Jacquemin, Christian, Judith Klavans, and Evelyne Tzoukermann. 1997. Expansion of multi-word terms for indexing and retrieval using morphology and syntax. In *Proc. of the ACL/EACL 97*, pages 24–30, Madrid, Spain. ACL.

Jennerich, Elaine Zaremba and Edward J. Jennerich. 1987. *The Reference Interview as a Creative Art*. Libraries Unlimited, Littleton, Colorado.

Jing, Hongyan and Evelyne Tzoukermann. 1999. Information retrieval based on context distance and morphology. In *Proc. of 22nd Annual Int'l Conf. on Research and Development in Information Retrieval (SIGIR 99)*, pages 90–96.

Joachims, Thorsten. 1998. Text categorization with support vector machines: learning with many relevant features. In Claire Nédellec and Céline Rouveirol, editors, *Proc. of ECML-98, 10th European Conf. on Machine Learning*, number 1398, pages 137–142, Chemnitz, DE. Springer Verlag, Heidelberg, DE.

Justeson, John S. and Slava M. Katz. 1995. Technical terminology: some linguistic properties and an algorithm for identification in text. *Natural Language Engineering*, 1(1):9–27.

Karlgren, Jussi. 1999. Non-topical factors in information access. Invited talk at *WebNet '99*, Honolulu, Hawai'i, USA, October.

Karlgren, Jussi and Douglass Cutting. 1994. Recognizing text genres with simple metrics using discriminant analysis. In *Proc. of COLING '94*, Kyoto, Japan.

Karnaugh, M. 1953. A map method for synthesis of combinational logic circuits. *Transactions of the AIEE, Communciations and Electronics*, 73(1):593–599, November.

Katz, William A., editor. 1987. *Introduction to Reference Work.* McGraw-Hill, New York, USA, 5th edition. Z 711.K32 1987.

Kessler, Brett, Geoffrey Nunberg, and Hinrich Schütze. 1997. Automatic detection of text genre. In *Proc. of the 35th Association of Computational Linguistics (ACL '97)*, pages 32–38, Madrid, Spain.

Kharitonov, Mark. 1999. Cfuf: A fast interpreter for the functional unification formalism. Master's thesis, Ben-Gurion University, Beer Sheva, Israel, April.

Kittredge, Richard, Tanya Korelsky, and Owen Rambow. 1991. On the need for domain communication knowledge. *Computational Intelligence*, 7(4).

Klavans, J. L. and S. Muresan. 2000. Definder: Rule-based methods for the extraction of medical terminology and their associated definitions from on-line text. In *Proc. of AMIA Symposium 2000*, page 1096.

Konomi, Shin'ichi, Yusuke Yokota, Kazuhiro Sakata, and Yahiko Kambayashi. 1997. Cooperative view mechanisms in distributed multiuser hypermedia environments. In *Proc. of 2nd IFCIS Int. Conf. on Cooperative Information Systems (CoopIS'97)*, pages 15–24.

Konstan, Joseph A., Brad N. Miller, David Maltz, Jon L. Herlocker, Lee R. Gordon, and John Riedl. 1997. Grouplens: Applying collaborative filtering to usenet news. *Communications of the ACM*, 40(3):77–87.

Kozima, Hideki. 1993. Text segmentation based on similarity between words.

In *Proc. of the 31th Annual Meeting of the Association for Computational Linguistics*, pages 286–288, Columbus, OH, USA.

Kraeling, Carl H. 1950. The humanities: Characteristics of the literature, problems of use, and bibliographic organization in the field. In Jesse H. Shera and Margaret E. Egan, editors, *Bibliographic Organization: papers presented before the Fifteenth Annual Conf. of the Graduate Library School*. University of Chicago Press, Chicago, pages 109–26. Z 1001.U54 1950g.

Kukich, Karen, Kathleen McKeown, James Shaw, Jacques Robin, J. Lim, N. Morgan, and J. Phillips. 1994. User-needs analysis and design methodology for an automated document generator. In Antonio Zampolli, Nicoletta Calzolari, and Martha Palmer, editors, *Current Issues in Computational Linguistics: Essays in Honour of Don Walker*. Giardini Editori e Stampatori and Kluwer Academic Publishers, Pisa and Dordrecht, pages 109–115.

Kushniruk, Andre W., Vimla L. Patel, and Jim J. Cimino. 1997. Usability testing in medical informatics: Cognitive approaches to the evaluation of information systems and user interfaces. In *Proc. of the 1997 AMIA Annual Fall Symposium*, pages 218–222.

Lagoze, Carl and Herbert Van de Sompel. 2001. The open archives initiative: Building a low-barrier interoperability framework. In *Proc. of the ACM/IEEE Joint Conf. on Digital Libraries*, pages 54–52, Roanoke, Virginia, USA, June.

Langkilde, Irene. 2000. Forest-based statistical sentence generation. In *6th Applied Natural Language Processing Conf. (ANLP'2000)*, pages 170–177, Seattle, Washington, USA.

Langkilde-Geary, Irene. 2002. An empirical verification of coverage and correctness for a general-purpose sentence generator. In *Proc. of 2nd Int'l Natural Language Generation Conf.*, pages 17–24, Harriman, New York, USA, July.

Lapata, Maria. 1999. Acquiring lexical generalizations from corpora: A case study for diathesis alternations. In *Proc. of ACL'99*, pages 397–404, College Park, Maryland, USA.

Layne, Sara Shatford. 1998. Modeling relevance in art history. In Tschera Harkness Connell and Robert L. Maxwell, editors, *The Future of Cataloguing*. American Library Association, Chicago, pages 33–41. Z 694.A15 F86 2000.

Lester, James C. and Bruce W. Porter. 1997. Developing and empirically evaluating robust explanation generators: The KNIGHT experiments. *Computational Linguistics*, 23(1):65–101.

Lester, James D. 2001. *Writing Research Papers : A Complete Guide*. Longman, 10th edition.

Lewis, David D. and William A. Gale. 1994. A sequential algorithm for training text classifiers. In *Proc. of the 17th Annual Int'l ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 3–12.

Library of Congress. 2000. Marc 21 format for classification data : including guidelines for content designation. Washington, D.C., USA. ISN 0660179903.

Liddy, Elizabeth. 1991. The discourse-level structure of empirical abstracts: An exploratory study. *Information Processing and Management*, 27(1):55–81.

Lin, Chin-Yew. 1998. Assembly of topic extraction modules in summarist. In *Proc. of the AAAI Spring Symposium on Intelligent Text Summarization*.

Lin, Chin-Yew and Eduard Hovy. 2000. The automated acquisition of topic signatures for text summarization. In *Proc. of the COLING Conf.*, Strasbourg, France, August.

Littman, Diane J. 1996. Cue phrase classification using machine learning. *Journal of Artificial Intelligence Research*, 5:53–94.

Liu, Bing, Kaidi Zhao, and Lan Yi. 2002. Visualize web site comparisons. In *Proc. of World Wide Web Conf. 2002*, Honolulu, Hawaii, USA, May.

Liu, Jimin and Tat-Seng Chua. 2001. Building semantic perceptron net for topic spotting. In *Proc. of the ACL-EACL 2001*, pages 370–377, Toulouse, France.

Loeffen, A. 1994. Text databases; a survey of text models and systems. *SIGMOD Record*, 23(1):97–106.

Lok, Simon and Steven K. Feiner. 2002. The AIL automated interface layout system. In *Proc. of the Int'l Conf. on Intelligent User Interfaces (IUI 2002)*, San Francisco, USA.

Lubetzky, Seymour. 1953. *Cataloging rules and principles; a critique of the A.L.A. rules for entry and a proposed design for their revision.* Processing Dept., Library of Congress, Washington, D.C., USA.

Luc, Ch., M. Mojahid, J. Virbel, Cl. Garcia-Debanc, and M.-P. Pery-Woodley. 1999. A linguistic approach to some parameters of layout: A study of enumerations. In *Using Layout for the Generation, Understanding or Retrieval of Documents*, AAAI Fall Symposium, pages 20–29, North Falmouth, Massachusetts, November. AAAI.

Mani, Inderjeet and Eric Bloedorn. 1999. Summarizing similarities and differences among related documents. *Information Retrieval*, 1(1-2):35–67.

Mani, Inderjeet and Mark Maybury, editors. 1999. *Advances in Automatic Text Summarization*. MIT Press.

Mann, William C. and Sandra A. Thompson. 1988. Rhetorical structure theory: Toward a function theory of text organization. *Text*, 8(3):243–281.

Marchionini, Gary. 1992. Interfaces for end-user information seeking. *Journal of the American Society for Information Science*, 43(2):156–163, March.

Marcu, Daniel. 1997a. From discourse structures to text summaries. In *Proc. of the ACL'97/EACL'97 Workshop on Intelligent Scalable Text Summarization*, pages 82–88, Madrid, Spain, July.

Marcu, Daniel. 1997b. The rhetorical parsing of natural language texts. In *Proc. of 35th Annual Meeting of the Association of Computational Linguistics and 8th Annual Meeting of the European Association for Computational Lingustics*, pages 96–103, Madrid, Spain.

McKeown, Kathleen R., Desmond Jordan, Steven K. Feiner, James Shaw, Elizabeth Chen, Shabina Ahmad, Andre Kushniruk, and Vimla Patel. 2000. A study of communcation in the cardiac surgery intensive care unit and its implications for automated briefing. In *Proc. of the American Medical Informatics Association 2000 Symposium.*

McKeown, Kathleen R, Judith Klavans, Vasileios Hatzivassiloglou, Regina Barzilay, and Eleazar Eskin. 1999. Towards multidocument summarization by reformulation: Progress and prospects. In *"Proc. of the 17th National Conf. on Artificial Intelligence (AAAI-99)"*. ACL, July.

McKeown, Kathleen R., Karen Kukich, and James Shaw. 1994. Practical issues in automatic documentation generation. In *Proc. of the 4th ACL Conf. on Applied Natural Language Processing (13-15 October 1994, Stuttgart).*

McKeown, Kathy, Shih-Fu Chang, Jim Cimino, Steven Feiner, Carol Friedman, Luis Gravano, Vassileios Hatzivassiloglou, Stephen Johnson, Desmond Jordan, Judith Klavans, Andre Kushniruk, Vimla Patel, and Simone Teufel. 2001. PERSIVAL, a system for personalized search and summarization over multimedia healtcare information. In *Proc. of the 1st JCDL*, Roanoke, USA.

McLaughlin, Harry. 1969. SMOG grading: A new readability formula. *Journal of Reading*, 12(8):639–646.

Miller, George, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine Miller. 1990. Five papers on WordNet. CSL Report 43, Cognitive Science Laboratory, Princeton University, July.

Miller, Nancy E., Pak Chung Wong, Mary Brewster, and Harlan Foote. 1998. Topic islands^TM - a wavelet-based text visualization system. In *Proc. of IEEE Information Visualization 98*. IEEE CS Press.

Milosavljevic, Maria. 1999. *The Automatic Generation of Comparison in Description of Entities*. Ph.D. thesis, Macquarie University, Syndey, Australia, February.

Monge, Alvaro E. and Charles Elkan. 1997. An efficient domain-independent algorithm for detecting approximately duplicate database records. In *Research Issues on Data Mining and Knowledge Discovery*.

Monz, Chrisof. 2001. Document fusion for comprehensive event description.

Morris, Jane and Graeme Hirst. 1991. Lexical cohesion computed by thesaural relations as an indicator of the structure of text. *Computational Linguistics*, 17(1):21–48.

Morse, Emile, Michael Lewis, and Kai A. Olsen. 2002. Testing visual information retrieval methodologies case study: Comparative analysis of textual, icon, graphical and "spring" displays. *Journal of the American Society for Information Science and Technology*, 53(1):28–40.

Ngai, Grace and David Yarowsky. 2000. Rule writing or annotation: Cost-efficient resource usage for base noun phrase chunking. In *Proc. of the 38th Annual Meeting of the ACL*, pages 117–125, Hong Kong, China, October.

Niyogi, Debashish. 1994. *A Knowledge-Based Approach to Deriving Logical Structure From Document Images*. Ph.D. thesis, State University of New York at Buffalo, Buffalo, New York, USA, August.

Nowell, Lucy Terry, Robert K. France, Deborah Hix, Lenwood S. Heath, and Edward A. Fox. 1996. Visualizing search results: Some alternatives to query-document similarity. In *Proc. of the Int'l ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 67–75, Zürich, Switzerland, August.

ODP. 2000. Open Directory Project guidelines. http://dmoz.org/guidelines.html, Last accessed November.

Paice, C. D. 1990. Constructing literature abstracts by computer: techniques and prospects. *Information Processing and Management*, 26(1):171–186.

Passonneau, Rebecca, Karen Kukich, Vasileois Hatzivassiloglou, Larry Lefkowitz, and Hongyan Jing. 1996. Generating summaries of work flow diagrams. In *Proc. of the Int'l Conf. on Natural Language Processing and Industrial Applications*, pages 204–210, New Brunswick, Canada, June.

Passonneau, Rebecca J. and Diane J. Litman. 1993. Intention-based segmenta-

tion: human reliability and correlation with linguistic cues. In *31st Annual Meeting of the ACL*, pages 148–155, June.

Pearrow, M. 2000. *Web site usability.* Charles River Media, Rockland, Massachusetts.

Pereira, Fernando, Natalie Tishby, and Lillian Lee. 1993. Distributional clustering of english words. In *Proc. of the 31st Annual Meeting of the Association of Computational Linguistics*, pages 183–190, Columbus, Ohio, USA. ACL.

Pikrakis, Aggelos, Tilemahos Bitsikas, Stelios Sfakianakis, Mike Hatzopoulos, David De Roure, Wendy Hall, Siegfried Reich, Gary J. Hill, and Mark A. Stairmand. 1998. MEMOIR: Software agents for finding similar users by trails. In Hyacinth S. Nwana and Divine T. Ndumu, editors, *Proc. of the 3rd Int'l Conf. on the Practical Applications of Agents and Multi-Agent Systems (PAAM-98)*, pages 453–466, London, UK.

Pirolli, Peter, Patricia Schank, Marti Hearst, and Christine Diehl. 1996. Scatter/gather: Browsing communicates the topic structure of a very large text collection. In *Proc. of the ACM SIGCHI Conf. on Human Factors in Computing Systems (CHI '96)*, pages 213–220, April.

Porter, Martin. 1980. An algorithm for suffix stripping. *Program*, 14.

Radev, Dragomir R. 1998. Learning correlations between linguistic indicators and semantic constraints: Reuse of context-dependent descriptions of entities. In *Proc. of COLING/ACL 98*, Montreal, Canada.

Rama, D. V. and Padmini Srinivasan. 1993. An investigaton of content representation using text grammars. *ACM Transactions on Information Systems*, 11(1):51–75, January.

Ratnaparkhi, Adwait. 2000. Trainable methods for surface natural language generation. In *Proc. of the 6th Applied Natural Language Processing Conf. (ANLP-NAACL 2000)*, pages 194–201, Seattle, Washington, USA.

Rees, Herbert. 1970. *Rules of Printed English.* Darton, Longman and Todd, London.

Reiter, Ehud. 1994. Has a consensus nl generation architecture appeared, and is it psycholinguistically plausible? In *Proc of the 7th Int'l Workshop on Natural Language Generation (INLGW-1994)*, pages 163–170, Kennebunkport, Maine, USA.

Reiter, Ehud and Robert Dale. 2000. *Building Natural Language Generation Systems.* Cmabridge University Press, Cambridge, United Kingdom.

Resnick, Paul, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. 1994. GroupLens: An Open Architecture for Collaborative Filtering of Netnews. In *Proc. of ACM 1994 Conf. on Computer Supported Cooperative Work*, pages 175–186, Chapel Hill, North Carolina. ACM.

Resnik, Philip. 1995. Using information content to evaluate semantic similarity in a taxonomy. In *Proc. of the 14th Int'l Joint Conf. on Artificial Intelligence (IJCAI).*

Riloff, Ellen and Wendy Lehnert. 1994. Information extraction as a basis for high-precision text classification. *ACM Transactions on Information Systems*, 12(3):296–333.

Robin, Jacques. 1994. *Revision-Based Generation of Natural Language Summaries Providing Historical Background: Corpus-Based Analysis, Design, Implementation and Evaluation.* Ph.D. thesis, Columbia University, New York, New York, USA.

Rushall, David A. and Marc R. Ilgen. 1996. Depict: Documents evaluated as pictures. In *Proc. of IEEE Information Visualization 96*, pages 100–107, Los Alamitos, CA, USA, October. IEEE CS Press.

Salton, Gerald and Chris Buckley. 1988. Term-weighting approaches in automatic text retrieval. *Information Processing and Management*, 24(5):513–523.

Schiffman, Barry and Kathleen R. McKeown. 2000. Experiments in automated lexicon building for text searching. To appear in the Proc. of the Int'l Conf. on Computational Linguistics (Coling 2000).

Schiffman, Barry, Ani Nenkova, and Kathleen R. McKeown. 2002. Experiments in multidocument summarization. In *Proc. of Human Language Technology Conf.*

Seymour, Kristie and Ronald Rosenfeld. 1997. Using story topics for language model adaption. In *Proc. of Eurospeech '97*, pages 1987–1990, Rhodes, Greece, September.

Shaw, James and Vasileios Hatzivassiloglou. 1999. Ordering among premodifiers. In *Proc. of the 37th Association for Computational Linguistics (ACL '99)*, pages 135–143, College Park, Maryland, USA, June.

Shaw, James C.-K. 2002. *Clause Aggregation: An Approach to Generating Concise Text.* Ph.D. thesis, Columbia University, New York, NY, USA.

Shippey, Gordon, Ashwin Ram, Florian Albrecht, Janis Roberts, Mark Guzdial, Richard Catrambone, Michael Byrne, and John Stasko. 1996. Exploring interface options and multimedia educational environments. Technical Report 2, Georgia Institute of Technology, Atlanta, Georgia, USA.

Simon, Herbert Alexander. 1996. *Sciences of the Artificial.* The MIT Press, Cambridge, Massachusetts, 3rd edition.

Stokes, Roy Bishop. 1969. *The Function of Bibliography.* Andre Deutsch, London. Z 1001.S84 .c.2.

Stokes, Roy Bishop. 1981. *Esdaile's Manual of Bibliography.* Scarecrow Press, USA, 5th edition. Z 1001.E75 1981.

Summers, Kristen M. 1995. Toward a taxonomy of logical document structures. Technical report, Cornell University, Ithaca, New York.

Summers, Kristen M. 1998. *Automatic discovery of logical document structure.* Ph.D. thesis, Cornell University, August.

Sutton, Ellen D. and Leslie Edmonds Holt, 1995. *The Reference Interview*, chapter 2, pages 36–53. Libraries Unlimited, Inc., Englewood, Colorado, USA, 2nd edition.

Sutton, Stuart A., Elizabeth D. Liddy, and John Kendall. 2002. Standardconnection: Correlating educational resources in digital libraries to content standards. In *Proc. of the 2nd ACM/IEEE-CS Joint Conf. on Digital Libraries*, page 408, Portland, Oregon, July.

Taeuber, Irene B. 1950. The social sciences: Characteristics of the literature, problems of use, and bibliographic organization in the field. In Jesse H. Shera and Margaret E. Egan, editors, *Bibliographic Organization: papers presented before the Fifteenth Annual Conf. of the Graduate Library School.* University of Chicago Press, Chicago, pages 127–39. Z 1001.U54 1950g.

Thompson, Cynthia A., Mary Elaine Califf, and Raymond J. Mooney. 1999. Active learning for natural language parsing and information extraction. In *Proc.*

16th Int'l Conf. on Machine Learning, pages 406–414. Morgan Kaufmann, San Francisco, CA.

Toms, Elaine G. 2000. Serendipitous information retrieval. In *Proc. of the 1st DELOS Network of Excellence Workshop on Information Seeking, Searching and Querying in Digital Libraries*, pages 11–12. DELOS, December.

Utiyama, Masao and Hitoshi Isahara. 2001. A statistical model for domain-independent text segmentation. In *Proc. of the 2nd Annual Meeting of the North American Association for Computational Linguistics*, Pittsburgh, Pennsylvania, USA.

Varges, Sebastian. 2002. Fluency and completeness in instance-based natural language generation. In *In Proc. of the 19th Intl. Conf. on Computational Linguistics (COLING-2002)*, Taipei, Taiwan.

Varges, Sebastian and Chris Mellish. 2001. Instance-based natural language generation. In *Proc. of NAACL 01*.

Wacholder, Nina, Yael Ravin, and Misook Choi. 1997. Disambiguation of proper names in text. In *Proc. of the 5th Conf. on Applied Natural Language Processing*, pages 202–208.

Wagner, Gulten. 2002. Virtual reference services down-under: A cautionary tale. In *Implementing Digital Reference Services: Setting Standards and Making It Real*. Neal-Schuman.

Walker, Marilyn, Owen Rambow, and Monica Rogati. 2001. SPoT: A trainable sentence planner. In *Proc. of the North American Meeting of the Association for Computational Linguistics*.

Wang, Rui. 2002. A defined path: Information-seeking behavior of users and digital reference services. In *Implementing Digital Reference Services: Setting Standards and Making It Real*. Neal-Schuman.

Watanabe, Toyohide. 1999. Document analysis and recognition. *IEICE Transactions On Information and Systems*, 82-D(3):601–610. Invited Survey Paper.

Williams, Owen. 2002. Writing an annotated bibliography. http://www.crk.umn.edu/library/links/annotate.htm, Last accessed March 2002.

Witcombe, Chris. 2002. Art history resources on the web. http://witcombe.sbc.edu/ARTHLinks.html, Last accessed January.

Yaari, Yaakov. 1997. Segmentation of expository texts by hierarchical agglomerative clustering. In *Recent Advances in Natural Language Processing (RANLP'97)*.

Yaari, Yaakov. 1999. *The Texplorer*. Ph.D. thesis, Bar Ilan University, Israel, April.

Yan, Tak W. and Héctor García-Molina. 1995. Duplicate removal in information dissemination. In *Proc. of the 21st Int'l Conf. on Very Large Data Bases (VLDB '95)*, pages 66–77, San Francisco, CA, USA, September.

Yang, Yiming. 1999. An evaluation of statistical approaches to text categorization. *Information Retrieval*, 1(1):69–90.

Youmans, Gilbert. 1991. A new tool for discourse analysis: The vocabulary-management profile. *Language*, 67(4):763–789.

Zamir, Oren and Oren Etzioni. 1998. Web document clustering: A feasibility demonstration. In *Proc. of the Int'l ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 46–54.

Zizi, Mountaz and Michel Beaudouin-Lafon. 1995. Hypermedia exploration with interactive dynamic maps. *Int'l Journal on Human Computer Interaction Studies*, 43:441–464.

Appendix A

Survey of information retrieval displays

Information retrieval (IR), in its most simple form, is the process of gathering information on a particular subject. In its most basic terms, it is the process of matching a need to available knowledge. Studies on information retrieval have approached IR from two major angles: from a rational approach (Simon, 1996) which views IR as a mathematical model, as well as from a cognitive approach (Belkin, Hennings, and Segger, 1984) which views IR as an analysis of the process of information gathering done by people. In this sense, information retrieval systems include not only search engines (a term we will refrain from using in this thesis), but also human constructed hierarchies, annotated bibliographies, and other specialized methods of presenting materials.

A definition of IR For the purposes of this thesis, "IR" unqualified is taken to mean information retrieval as it applies in a rationalist approach. To formalize this, within the scope of this thesis, a *standard* (=rationalist)*information retrieval system* is a system that matches a structured user *query* to a set of *documents*, yielding

a retrieval of a subset of those documents – the *results*. The subset is ordered along some dimension, typically a calculated notion of relevance. Performance in a rational system of IR is equated with precision and recall metrics, comparing a system's retrieval performance with a gold standard compiled manually by humans.

Design of IR systems in general have benefited immensely from the rationalist perspective in developing useful and scalable paradigms for handling vast amounts of information using a concise and expressive calculus. As strong as the rationalist approach is for these purposes, it is not well-suited to provide useful mechanisms for access to the information when humans appear in the loop; that is, when a human user interacts with an IR system. Here a cognitive engineering approach may be more fitting, but unfortunately, the user interface of a rational IR system is sometimes an afterthought, since performance is measured intrinsically and not extrinsically by user satisfaction or by task-based evaluation.

Distinction between the user interface and framework The good news is there is much room for improvement. To this end, I make a distinction between the underlying framework of an IR system versus its user interface. The framework itself is responsible for providing the set of documents that are relevant to a query (i.e., calculating the *result* subset), the output user interface (UI) is responsible for displaying information about the returned subset of documents.

This division between two parts of IR does not put a precedence on one or the other. In the rational approach, it is often the case that the framework and its capabilities drive the structure of the user interface; in the cognitive approach, the user needs of the interface drive the underlying framework.

Dimensions in IR framework and user interface Let me paraphrase and narrow the thesis statement with respect to what has been said so far: user interfaces to standard (=rationalist) IR systems can be improved using automatic text

summarization. However, I have not yet defined what is meant by the standard IR user interface. The purpose of the remainder of this Appendix is to define the standard IR interface, by surveying several different IR systems. The commonalities between the IR systems determine the standard.

Defining what is the standard IR user interface itself has many dimensions. One can ask questions of the user interface itself: is it textual or does it use graphics? What type of navigation is allowed in the results? Do the results incorporate any domain- or genre-specific information? We can also examine dimensions that categorize aspects of the IR system's framework (as per the discussion above, this can influence the UI): is the system completely automated by computer, or does it require human intervention? What domain- or genre-specific information does it index? What information is stored in an entry? Is an entry text, or does it have multimedia elements?

A.1 Survey methods

In the remainder of this chapter, I examine several online IR systems, which I selected because they use diverse methodologies in their retrieval framework. As a result, their user interfaces also vary. My purpose here is to survey a wide variety of IR systems to establish the features of what I called the standard ranked list user interface. I examine ones used both in the confines of specialist domains and for general materials; ones used both in digital libraries as well as ones used on the World Wide Web.

A.1.1 Gopher

Gopher was a popular protocol for information retrieval in the late 1980's to early 1990's, before the ubiquity of the web. It is an example of information retrieval

design from the rational perspective. Gopher's designers wanted to make documents on remote computers accessible to users sitting in front of their own computers. As such, a natural method to structure the remote documents in the same way documents were represented on local computers: as a hierarchical filesystem. The Gopher standard (Anklesaria et al., 1993) points out a few different reason why this user interface was a good model to follow: a) it is familiar to many users, b) it is expressed in a simple syntax, making it easy to set up a new information service, c) and is easily extensible, allowing new providers to seamlessly integrate services with older ones.

Gopher servers make their services and documents available to a remote computer by requiring an administrator to manually configure which resources are available and to organize both the resources' descriptions (to be given to the client user) as well as their hierarchical organization. As such, developing Gopher services that are optimally organized require lots of manual labor, as proper logical organization and description of the resources is left up to the developer.

Gopher clients connect to a Gopher server and typically ask for a listing of documents and services that the server provides. These can be displayed however the client chooses; typically the list is formatted as-is, with titles of resources truncated to fit the screen and long lists transformed into a multiple page list with scrolling ability.

Gopher was originally built around the distribution of text documents structured in hierarchical directories, but also has the capability to serve information coming from different services, by encoding the format or service type provided by each particular directory entry. In this way, Gopher provides meta-information about the directory entry, indicating whether the link points to a text document, image, or a service that provides search.

A majority of gopher clients mirrored the simplicity in the protocol, by

doing little more than format the information given by a Gopher request to fit the screen. Popular Gopher clients (such as TurboGopher, Curses-based gopher, and WSGopher) all share the same characteristics and do not try to process the gopher source data beyond simple formatting. Since the clients only have two pieces of information that are supposed to be viewed by the user (the type of resource in question, and a string description of the resource), these are displayed in a straightforward fashion, usually with an graphical icon next to the string description, which a user can either click on or select using cursor keys.

A.1.1.1 Archie, Veronica and Jughead

Software such as Archie, Veronica, Jughead were later developed to index the multitude of FTP and Gopher servers on the network such that a single search could match items from all over the network. These early protocols for indexing provided very little control of the search. Archie, which indexes items available by anonymous FTP, only indexes the physical filenames of the items, and both Veronica and Jughead, which index Gopher accessible items, only index the string descriptions of items available in Gopher menus. Jughead, a variant of Veronica, allows users to search a subset of gopher hosts, rather than the entire Veronica index (whose goal is to list all Gopher sites), to speed up the querying process. In all three cases, no additional information was indexed to allow more sophistication in search or display; such as full-text search or search restricted to particular types of information (text documents, or images).

A.1.2 WAIS

Parallel to Gopher was the development of the Wide Area Information System (WAIS), which performed indexing at a more fine grained level than Archie or Veronica. The WAIS protocol indexes at the word level, allowing more compre-

hensive retrievals to take place. Unlike Gopher, WAIS was developed primarily to search databases consisting of records of structured fields, rather than free text. As such the system supports operators that allow specific fields' values to be specified in a number of manners. Operations on common types of data available in databases are also supported (e.g., search for dates less than 1/1/90, or currency amount greater than 99.99). For text fields, exact match, incomplete match, proximity, and sounds-like (soundex) queries are all handled within WAIS' framework. The notion of query terms having different levels of significance or weight is also introduced, on the basis of Salton (see (Salton and Buckley, 1988)).

The display of the search results are also a bit more complex in the more advanced WAIS clients (e.g., SFgate). By default, a list of documents matching the query is returned to the user, in which the user can select a single document by clicking on the title of the document. A more advanced user can activate "fine tuning" options to allow checkboxes to appear next to the document links. By checking multiple boxes and pressing a retrieve button, you can retrieve all of the selected matches together on a single page. A user can also elect to have all the matching records retrieved using a switch that essentially checks all of the boxes.

These finer grained controls are useful in WAIS because the typical results returned are single rows of a database, possibly with very little information, and as such, reflect the type of information that is being indexed by WAIS.

A.1.3 World Wide Web

With the introduction of the World Wide Web in 1990, a new tradition of networked information retrieval started. Gopher and WAIS variants were all based on text documents; in contrast, web documents followed the initial Hypertext Markup Language (HTML) specification that allowed for multimedia documents having both images and text together. Crucial to the success of the Web was the fact that

linking to documents on remote systems was possible and easy to do, through the use of hypertext links. These links made connections between documents across a multitude of hosts, whose documents in turn would reach other hosts. This single development allows anyone to build a search engine for the web by starting with a few seed documents that would branch out to eventually reach the majority of documents on the network.

By building the HTML specifications off of the syntax of SGML, which enriches raw text document with tags but leaves the plain text readable, subsequent HTML tags have been introduced that allow more fine-grained control over text fonts, embedding of multimedia within documents and placement of controls for user querying and feedback through forms.

Most information retrieval systems available through the web protocol today use controls in the HTML forms library to get information from the user and report the results of a search on a query as an HTML document. The basic protocol that transports web documents, the Hypertext Transport Protocol (HTTP) coupled with the documents written in HTML make up the majority of information that is presented to users on the web.

However, the combination of HTTP/HTML has many limitations that affect the possibilities of user interaction and for display (e.g., not being able to track the mouse, or accept other input modalities such as speech), but many of these constraints have workarounds to still function as applications embedded in web documents.

The specification of how to detect whether a particular WWW client supports which tags is particularly arduous, but the majority of clients, either graphical or text based, do support HTML 3.0 which introduces the fill-out form controls which enable a wide variety of information retrieval servers to function with WWW client software.

These web-based information retrieval systems come in many different varieties, so I will examine a range of these systems, starting with systems that do not require any human intervention for indexing (similar to Veronica or Archie), to information retrieval systems that require full human intervention to answer queries. These user interfaces to such IR engines are not static, they are revised as new retrieval technology is developed. As such, I only examined the user interfaces sampled here at a single point in time (late 2001) and do not necessarily reflect the current state of the systems.

A.1.3.1 AltaVista

AltaVista is a web information retrieval system most famous for its large scale indexing and relative coverage of the web. From a user interface standpoint, AltaVista has a simple interface and has the same functionality of many earlier clients, including WAIS and Veronica. The total number of results are listed, titles of the first ten documents are listed by default. Navigation is similar all clients discussed previously; there are controls that allow one to go to the previous and next ten matches that the framework has retrieved, or skip to a particular page of matches.

Like other web IR systems, AltaVista continues Gopher's tradition of differentiating different types of sources by providing dedicated searches for different types of media matching the query (audio, images, video). AltaVista also displays similar queries which other users have issued – a form of collaborative filtering, and suggests queries that are similar in spelling or pronunciation (similar to the WAIS soundex type queries) to help fix queries that have been mistyped.

Examining the format of results matching a query, each entry has a title, as in WAIS, but is expanded with a short description as well other pieces of information. The short description in AltaVista is just the first couple of lines of the matching document. With web sites that host many matching documents, it only shows

the top pages with the highest computed relevance, but permits the additional truncated pages to be viewed by a "More from this site" link.

AltaVista's framework also handles relevance feedback; cases where a document is good and the user wants to see more results similar to the marked document. Often the document itself is used as a query to retrieve other documents that are similar. In the user interface, a hyperlink entitled "Related Pages" is included for each entry for relevance feedback. Unique to AltaVista is it inclusion of a link to translate the pages into other languages.

A.1.3.2 Google

Google is well-known for being the first information retrieval framework that uses ideas coming from bibliometrics to measure the relative importance of web pages. Much in the same way that citations in one paper indicate that the cited paper is an authority, Google uses metrics based on hyperlinks to a document to determine its relative importance (Brin and Page, 1998).

Within an entry, Google displays several pieces of information similar to AltaVista, such as the title and the URL of the page. However, there are notable differences between the two IR systems: for example, Google's short description field shows query words in their local context (QWIC), instead of the first couple of lines shown by AltaVista. Also, when a description is given by the document (through the use of HTML <META> tags), Google will use this description instead. Google also shows the date when the document was indexed. For other types of text not in HTML format (and thus not necessarily accessible to all WWW clients), it provides a text-only copy of these documents. Google also keeps a cached copy of standard HTML documents. For both of these documents it highlights search terms used in the query when the cached or text-only copy is retrieved.

Google also embeds listings from the Open Directory Project (ODP), which

categorizes pages similar to Yahoo! (see A.1.3.3 below). If a document in the result list is also listed in the ODP, Google will report the category as a hyperlink to the category page.

A.1.3.3 Yahoo!

In contrast to the other web based IR systems mentioned thus far, Yahoo! is organized as a searchable and browsable web hierarchy. Links to documents in Yahoo! have been sorted through by human editors by hand and a description of the document either provided by the editor or edited from a description provided by the document's author. These descriptions are even shorter than ones used in AltaVista or Google; and usually consist of a list of topics which are discussed by the respective document.

Yahoo! has two main views of information on a subject, depending on whether you are searching using a specific query, or browsing using one of the categories in the Yahoo! hierarchy. In the search mode, only documents within the Yahoo! catalog are searched for. In this mode, documents are listed as per other search engines; by calculated relevance. Title, the human-edited short description and URL are listed with each entry as well as links to the category pages that the entry belongs to. Search mode pages are additionally prefixed with a listing of hyperlinks to relevant categories and advertising links.

In the browsing mode, a particular category in the hierarchy is displayed. In this mode, links to each of the direct ancestors of the category is provided at the top of the page. Direct descendant and related categories are listed in alphabetical order as the first part of the page. The categories are listed by name, and the size of the direct descendant categories (in documents) are given in parentheses. Of course, categories group together documents, and the list of documents are listed in alphabetical order at the end of the page, rather than in relevance order. In

browsing mode, the description of the matching documents are shortened: only the title and short edited description are shown. Entries that are particular interesting to the editor or have been newly added are marked with icons.

A.1.3.4 About.com

About.com is similar to Yahoo! in that is also utilizes humans to provide and edit content for the information seeker. However, it does so in a very different way than Yahoo! Volunteer human editors (called guides), who are experts on particular subjects, spend time to select among websites and other web-accessible content to solve the problem of selection for common subject areas.

About.com's interface divides into five main parts, arranged top to bottom in its web page. These five parts can be somewhat lengthy, making the result of a search extend to many pages. The first area gives related queries to the user's search query. This is a form of collaborative filtering that harnesses previous searches made by others similar to the user's query, available for execution. The remaining four areas consist of sponsored links, About.com guide sites, general web sites, and links to affiliated search partners. Each of these four areas have the same format: a hyperlink to the resource with the resources title, a short 10- to 50-word description, and the web URL to the resource.

A.1.4 Domain specific search engines

Domain specific search engines tailor their interfaces to tune to the specific needs of their clientele. I examine a few large-domain IR engines, again concentrating on the features of the output user interface used to assist the information seeker in pinpointing relevant documents and in performing follow-up searches. As these search engines may not be as familiar, I have provided screen shots inline with their descriptions.

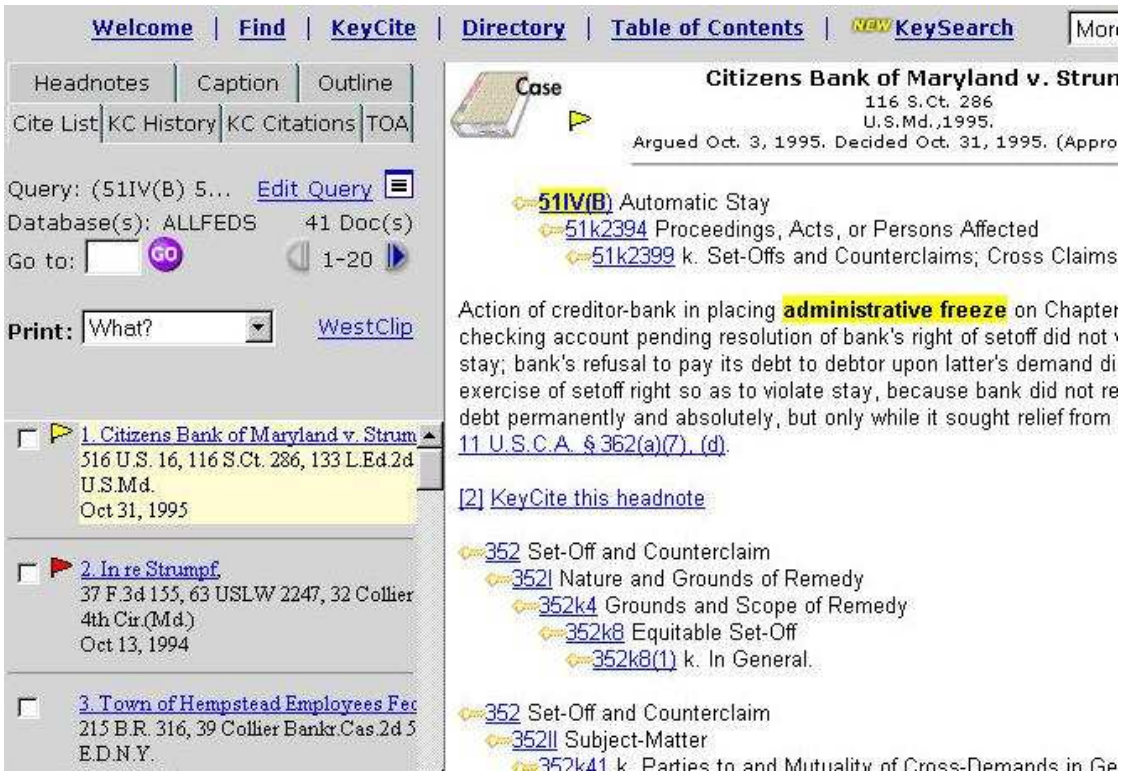
A.1.4.1 Northern Light

Northern Light is search engine for commercial publications, in the domains of business and general interest. It incorporates document clustering technology. Returned results are first categorized into by patented technology into "custom search folders" in a hierarchical tree given on the left half of the search screen. Clusters are built dynamically and partition information according to "subject, type, source, and language". A cluster is represented by a short one to five word phrase, along with a folder icon. A cluster can be further expanded by clicking on it and subclusters are shown, nested within the parent cluster in the hierarchy.

The initial search can be thought of as a single large cluster that has expanded to show its subclusters. On the right hand side of the interface is the document listing – in which the top, most relevant documents to the cluster are shown ranked by relevance. As subclusters are a subset of the documents its parent cluster, a document shown in the document list may also appear in its parent's document list. Northern light gives each document's rank, the title of the collection that houses it, title, percent relevance, short excerpt, date of publication, source

and availability. The short excerpt is provided in QWIC format.

A.1.4.2 Westlaw



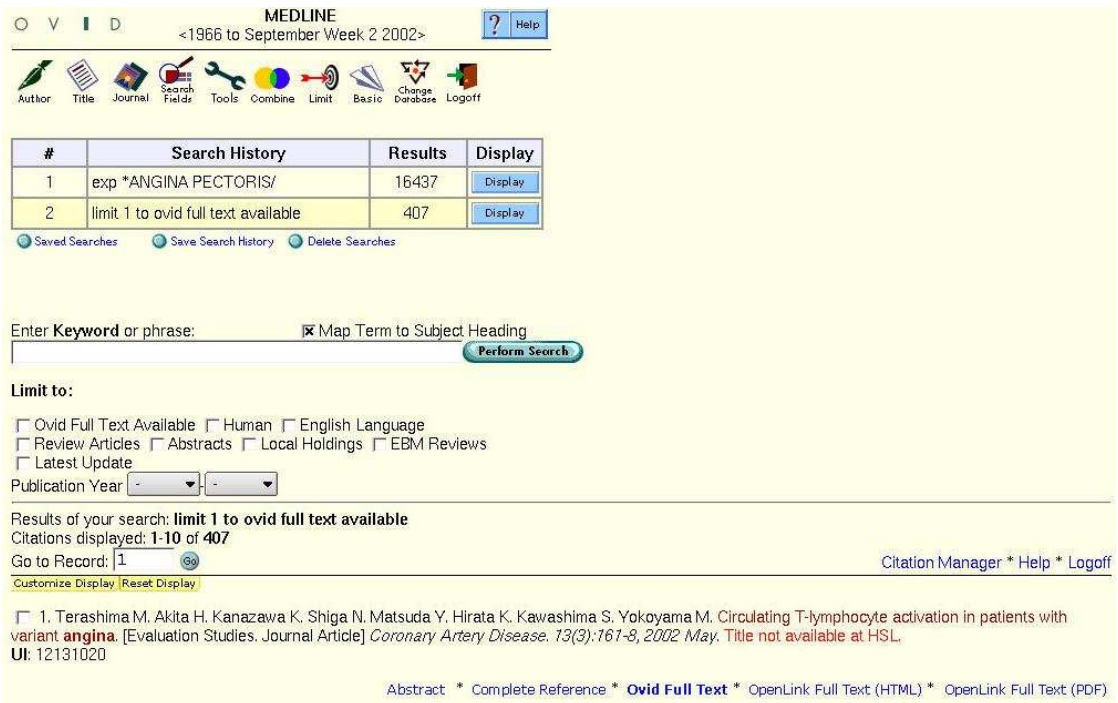
Westlaw is a search engine for legal materials. Customization allows a specific user model to be set to specify important global settings (e.g., the legal jurisdiction of the lawyer: New York). Customization also allows the user to change the default search criteria displayed to the user – more experienced users can specify to see more criteria, where novice uses may opt to see less to lessen control clutter and cognitive load.

Search within Westlaw allows a high level of control of over the scope, database, metadata fields and keyword content of the individual documents. Search formulation can be difficult for novices or for experts searching in areas in Westlaw beyond their expertise. Westlaw provides several facilities to ease this problem, by structuring commonly sought information into a browsable hierarchy. The hierar-

chy's leafs add appropriate search terms to the command line search interface that can be further modified by the user before execution.

Results of a search are shown with a split interface showing a small ranked list of documents on the left and a document view on the right. Either side can be zoomed to take up the entire screen. In the ranked list, documents are sorted in reverse chronological order. The document descriptions in the rank display gives specific case citation information: including the parties' name, address, date, and court. In the document view, specific citation information is fixed at the top of the view while the body of the record is scrollable and highlighted query terms. The document view consists of five parts: citation information, case synopsis, applicable controlled vocabulary, headnote summaries (that summarize each point of law in a case) and court opinions.

A.1.4.3 OVID



OVID Technologies provides a user interface that unifies searching across

many subfields and databases of medical information. In addition to keyword searching, the OVID input interface allows the use of controlled vocabulary and authority, and search limit for various features, such as full text availability and publication date.

The user interface displays the resulting list of documents in citation format, which includes the document's title, authors and publication source. This can be further modified using customization. After the document's citation, links to the different available formats for the document are given. There is no default sort order for the document listing; OVID allows you to specify a primary and secondary sort key to resort the listing.

A key part of OVID's search interface is the search history interface which is shown at the top of the results display. New searches and refinements to ongoing searches are added to the search history. The search history allows set operations such as union and intersection on results of different queries.

A.1.4.4 EDGAR

EDGAR is the United States Securities and Exchange Commission clearinghouse of publicly available information on company information and filings. These filings are indexed by company name. Input of an ambiguous company name shows an abbreviated list of companies with identifier information (e.g., state of registry) to help pinpoint the company. The main user interface to display results sorts doc-

uments by reverse chronological order and gives each filing's form name, available formats, form description (not a summary of the contents of the form), filing date and text size. Identifying company information is listed at the top of the form (e.g., address, unique EDGAR company identifier). To facilitate narrowing a result set, EDGAR provides search limits on the results page, such that a user can retrieve only filings made after a certain date or of a certain form type.

A.2 Survey results

Gopher*	WAIS	Alta-Vista	Google	Yahoo!	About .com	Northern Light	West law	OVID	EDGAR
Description:									
has title									
Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
is an human authored summary									
				Yes	Yes				
is first n sentences or bytes									
		Yes							
has query words in context									
			Yes	Yes		Yes	Yes		
can give related pages via relevance feedback									
		Yes	Yes						
has pointers to other records									
							Yes		
lists available formats									
			Yes				Yes	Yes	Yes
gives indication of file type									
Yes			Yes			Yes	Yes	Yes	Yes
contains other, domain-specific metadata									
						Yes	Yes	Yes	Yes
gives location									
		Yes	Yes		Yes	Yes			
has date of publication, indexing									
		Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes

* - Gopher includes variants Archie, Veronica and Jughead.

Table A.1: Features present in document descriptions across surveyed IR systems.

Tables A.1 and A.2 show the results of the survey. From the survey, it is clear that the majority of interfaces share several common attributes. Mostly, they list subsequent, more focused or related topics as items with short descriptions. The items are ordered in a list, either by alphabetical order or by computed relevance. They often include some textual snippet description of the item, either a high-level indicative description of the text, or extracted portions from the item itself. These descriptions refer to the item itself and usually do not make any reference to the surrounding context of other items on the page. If the number of returned documents is large, the result set is broken up into multiple pages, accessible with navigation links or buttons. Advance search in terms of searching limits (restricted search) is usually allowed; the controls are hidden from casual users to prevent over crowding of controls. More specific search engines, operating in specific domains

Gopher*	WAIS	Alta-Vista	Google	Yahoo!	About .com	Northern Light	West law	OVID	EDGAR
Has Document Clustering: clusters into fixed categories									
				Yes		Yes			
clusters into dynamic categories, calculated at run-time									
					Yes				
Has Controls for Follow-up search: has search textbox									
Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
has advanced search hidden									
		Yes	Yes		Yes	Yes	Yes	Yes	
can add limits to search results									
		Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
has access to search history									
								Yes	
shows related searches									
				Yes	Yes				
gives authority term matches									
								Yes	Yes
Navigation of result set: next and previous page									
Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
random access page links									
		Yes	Yes						
Screen display: split, ranked list + document display									
							Yes		
only ranked list(s)									
Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes

* - Gopher includes variants Archie, Veronica and Jughead.

Table A.2: Features present in output interfaces aside from the document descriptions, in the surveyed IR systems.

allow more expansive controls on searching. These expansive controls could overwhelm even the experienced user. Fortunately, many searchers (even within these specific domains) only use a subset of the features, thus customization allows them to set defaults and do away with infrequent controls, again to lower the cognitive load on the user.

To conclude, the standard user interface has:

- a list of documents in order of relevance;
- document descriptions give title, a short description in QWIC format, with some other metadata elements;

- navigation controls to see previous or next results;
- and facilities to modify the original search using limit (restricted search) that may placed on a separate page (accessible by link)

These set of common user interface criteria define the *standard ranked list user interface*, or ("standard ranked list" for short). Information systems that provide information using this set of standard interfaces clearly benefit from obeying *de facto* norms and can devote less screen area for instructions. The standard ranked list possesses good features. It provides an ordering of the items. It also provides simple to use navigation between items and gives some level of details on each specific item.

Appendix B

Sample document on Angina

This Appendix reproduces an original file on the web, downloaded on the 14th of June 2002 from the URL:

http://www.nhlbi.nih.gov/health/public/heart/angina.htm.

As of the 11th of September 2002, it could be found at a slightly different URL:

http://www.nhlbi.nih.gov/health/public/heart/other/angina.htm.

This file is used as an example in several of the chapters in the thesis, including Chapters 3, 4 and 5.

Facts About Angina

What is angina?

ANGINA PECTORIS ("ANGINA") IS A recurring pain or discomfort in the chest that happens when some part of the heart does not receive enough blood. It is a common symptom of coronary heart disease (CHD), which occurs when vessels that carry blood to the heart become narrowed and blocked due to atherosclerosis

Angina feels like a pressing or squeezing pain, usually in the chest under the breast bone, but sometimes in the shoulders, arms, neck, jaws, or back. Angina is usually precipitated by exertion. It is usually relieved within a few minutes by resting or by taking prescribed angina medicine.

What brings on angina?

Episodes of angina occur when the heart's need for oxygen increases beyond the oxygen available from the blood nourishing the heart. Physical exertion is the most common trigger for angina. Other triggers can be emotional stress, extreme cold or heat, heavy meals, alcohol, and cigarette smoking.

Does angina mean a heart attack is about to happen?

An episode of angina is not a heart attack. Angina pain means that some of the heart muscle in not getting enough blood temporarily–for example, during exercise, when the heart has to work harder. The pain does NOT mean that the heart muscle is suffering irreversible, permanent damage. Episodes of angina

seldom cause permanent damage to heart muscle.

In contrast, a heart attack occurs when the blood flow to a part of the heart is suddenly and permanently cut off. This causes permanent damage to the heart muscle. Typically, the chest pain is more severe, lasts longer, and does not go away with rest or with medicine that was previously effective. It may be accompanied by indigestion, nausea, weakness, and sweating. However, the symptoms of a heart attack are varied and may be considerably milder.

When someone has a repeating but stable pattern of angina, an episode of angina does not mean that a heart attack is about to happen. Angina means that there is underlying coronary heart disease. Patients with angina are at an increased risk of heart attack compared with those who have no symptoms of cardiovascular disease, but the episode of angina is not a signal that a heart attack is about to happen. In contrast, when the pattern of angina changes–if episodes become more frequent, last longer, or occur without exercise–the risk of heart attack in subsequent days or weeks is much higher.

A person who has angina should learn the pattern of his or her angina–what cause an angina attack, what it feels like, how long episodes usually last, and whether medication relieves the attack. If the pattern changes sharply or if the symptoms are those of a heart attack, one should get medical help immediately, perhaps best done by seeking an evaluation at a nearby hospital emergency room.

Is all chest pain "angina?"

No, not at all. Not all chest pain is from the heart, and not all pain from the

heart is angina. For example, if the pain lasts for less that 30 seconds or if it goes away during a deep breath, after drinking a glass of water, or by changing position, it almost certainly is NOT angina and should not cause concern. But prolonged pain, unrelieved by rest and accompanied by other symptoms may signal a heart attack.

How is angina diagnosed?

Usually the doctor can diagnose angina by noting the symptoms and how they arise. However one or more diagnostic tests may be needed to exclude angina or to establish the severity of the underlying coronary disease. These include the electrocardiogram (ECG) at rest, the stress test, and x- rays of the coronary arteries (coronary "arteriogram" or "angiogram").

The ECG records electrical impulses of the heart. These may indicate that the heart muscle is not getting as much oxygen as it needs ("ischemia"); they may also indicate abnormalities in heart rhythm or some of the other possible abnormal features of the heart. To record the ECG, a technician positions a number of small contacts on the patient's arms, legs, and across the chest to connect them to an ECG machine.

For many patients with angina, the ECG at rest is normal. This is not surprising because the symptoms of angina occur during stress. Therefore, the functioning of the heart may be tested under stress, typically exercise. In the simplest stress test, the ECG is taken before, during, and after exercise to look for stress related abnormalities. Blood pressure is also measured during the stress test and symptoms are noted.

A more complex stress test involves picturing the blood flow pattern in the heart muscle during peak exercise and after rest. A tiny amount of a radioisotope, usually thallium, is injected into a vein at peak exercise and is taken up by normal heart muscle. A radioactivity detector and computer record the pattern of radioactivity distribution to various parts of the heart muscle. Regional differences in radioisotope concentration and in the rates at which the radioisotopes disappear are measures of unequal blood flow due to coronary artery narrowing, or due to failure of uptake in scarred heart muscle.

The most accurate way to assess the presence and severity of coronary disease is a coronary angiogram, an x-ray of the coronary artery. A long thin flexible tube (a "catheter") is threaded into an artery in the groin or forearm and advanced through the arterial system into one of the two major coronary arteries. A fluid that blocks x-rays (a "contrast medium" or "dye") is injected. X-rays of its distribution show the coronary arteries and their narrowing.

How is angina treated?

The underlying coronary artery disease that causes angina should be attacked by controlling existing "risk factors." These include high blood pressure, cigarette smoking, high blood cholesterol levels, and excess weight. If the doctor has prescribed a drug to lower blood pressure, it should be taken as directed. Advice is available on how to eat to control weight, blood cholesterol levels, and blood pressure. A physician can also help patients to stop smoking. Taking these steps reduces the likelihood that coronary artery disease will lead to a heart attack.

Most people with angina learn to adjust their lives to minimize episodes of

angina, by taking sensible precautions and using medications if necessary.

Usually the first line of defense involves changing one's living habits to avoid bringing on attacks of angina. Controlling physical activity, adopting good eating habits, moderating alcohol consumption, and not smoking are some of the precautions that can help patients live more comfortably and with less angina. For example, if angina comes on with strenuous exercise, exercise a little less strenuously, but do exercise. If angina occurs after heavy meals, avoid large meals and rich foods that leave one feeling stuffed. Controlling weight, reducing the amount of fat in the diet, and avoiding emotional upsets may also help.

Angina is often controlled by drugs. The most commonly prescribed drug for angina is nitroglycerin, which relieves pain by widening blood vessels. This allows more blood to flow to the heart muscle and also decreases the work load of the heart. Nitroglycerin is taken when discomfort occurs or is expected. Doctors frequently prescribe other drugs, to be taken regularly, that reduce the heart's workload. Beta blockers slow the heart rate and lessen the force of the heart muscle contraction. Calcium channel blockers are also effective in reducing the frequency and severity of angina attacks.

What if medication fails to control angina?

Doctors may recommend surgery or angioplasty if drugs fail to ease angina or if the risk of heart attack is high. Coronary artery bypass surgery is an operation in which a blood vessel is grafted onto the blocked artery to bypass the blocked or diseased section so that blood can get to the heart muscle. An artery from inside the chest (an "internal mammary" graft) or long vein from the leg (a "saphenous

vein" graft) may be used.

Balloon angioplasty involves inserting a catheter with a tiny balloon at the end into a forearm or groin artery. The balloon is inflated briefly to open the vessel in places where the artery is narrowed. Other catheter techniques are also being developed for opening narrowed coronary arteries, including laser and mechanical devices applied by means of catheters.

Can a person with angina exercise?

Yes. It is important to work with the doctor to develop an exercise plan. Exercise may increase the level of pain-free activity, relieve stress, improve the heart's blood supply, and help control weight. A person with angina should start an exercise program only with the doctor's advice. Many doctors tell angina patients to gradually build up their fitness level–for example, start with a 5-minute walk and increase over weeks or months to 30 minutes or 1 hour. The idea is to gradually increase stamina by working at a steady pace, but avoiding sudden bursts of effort.

What is the difference between "stable" and "unstable" angina?

It is important to distinguish between the typical stable pattern of angina and "unstable" angina.

Angina pectoris often recurs in a regular or characteristic pattern. Commonly a person recognizes that he or she is having angina only after several episodes have occurred, and a pattern has evolved. The level of activity or stress that provokes

the angina is somewhat predictable, and the pattern changes only slowly. This is "stable" angina, the most common variety.

Instead of appearing gradually, angina may first appear as a very severe episode or as frequently recurring bouts of angina. Or, an established stable pattern of angina may change sharply; it may by provoked by far less exercise than in the past, or it may appear at rest. Angina in these forms is referred to as "unstable angina" and needs prompt medical attention.

The term "unstable angina" is also used when symptoms suggest a heart attack but hospital tests do not support that diagnosis. For example, a patient may have typical but prolonged chest pain and poor response to rest and medication, but there is no evidence of heart muscle damage either on the electrocardiogram or in blood enzyme tests.

Are there other types of angina?

There are two other forms of angina pectoris. One, long recognized but quite rare, is called Prinzmetal's or variant angina. This type is caused by vasospasm, a spasm that narrows the coronary artery and lessens the flow of blood to the heart. The other is a recently discovered type of angina called microvascular angina. Patients with this condition experience chest pain but have no apparent coronary artery blockages. Doctors have found that the pain results from poor function of tiny blood vessels nourishing the heart as well as the arms and legs. Microvascular angina can be treated with some of the same medications used for angina pectoris.

Additional Resources:

Facts About Blood Cholesterol (revised 1994), NIH Publication No. 94-2696

Fact About Coronary Heart Disease (reprinted 1993), NIH Publication No. 93-2265

Facts About Heart Failure (reprinted 1995) NIH Publication No. 95-923

Facts About Heart Disease and Women: So You Have Heart Disease, NIH Publication No. 95-2645

High Blood Pressure and What You Can Do About It, No. 55-222A

So You Have High Blood Cholesterol (revised 1993), NIH Publication No. 93-2922

Step by Step: Eating to Lower Your High Blood Cholesterol (revised 1994) NIH Publication No. 94-2920

For Further Information

Call or Write:

National Heart, Lung, and Blood Institute
Information Office
P.O. Box 30105
Bethesda, MD 20892-0105
Telephone: (301) 592-8573

U.S. DEPARTMENT OF HEALTH AND
HUMAN SERVICES
Public Health Service
National Institutes of Health
National Heart, Lung, and Blood Institute
NIH Publication No. 95-2890
Reprinted September 1995

Printed Copies

Order online

Order by telephone, fax, or mail

Price Information

Single copy	Free
Each additional copy	$1.25
25 copies	$15.00
100 copies	$50.00

NHLBI Home Page

Back to Other Cardiovascular Information for the General Public

Please send us your feedback, comments, and questions
by using the appropriate link on the page, Contact the NHLBI.

Note to users of screen readers and other assistive technologies: please report your problems here.

Appendix C

Building a "One POS per word" dictionary from COMLEX Syntax

To recap the task, I derive a one part of speech tag per lexical term mapping for the English language for the purpose of aiding the term finding portion of SEGMENTER program. All standard dictionaries assign each lexical term with all possible parts of speech that it may be used in, mostly because the lexical pattern matches two or more different semantic terms, perhaps in various inflected forms.

If I had used a dictionary in which frequency of use information was provided, I could just list the most common semantic term's part of speech. To gather such a corpus would be relatively simple, a matter of collecting frequency of occurrence information on the part of speech assigned to every lexical item of a large corpus.

However, I chose to work with a standard resource, COMLEX Syntax dictionary (Grishman, Macleod, and Meyers, 1994), and from its listings, derive a single part of speech from each lexical term. I do this in two steps: (1) a simple set of precedence relations and a (2)stoplist of common words to demote. These two steps will be explained in more detail below.

As mentioned in the main text, the specificity of the part of speech tagging

necessary for noun phrase detection is quite coarse – a tagging of the text which denotes only 7 different types of words: punctuation, discourse markers, proper nouns, common nouns, adjectives, pronouns, and any other category, is enough. Many automated taggers as well as dictionaries list more finer-grained distinctions, and the task is to categorize the particular tags of your resource to the super categories above.

C.1 Step 1 - Maximizing coverage

Working with COMLEX, I derived these 6 gross parts of speech for every word mentioned in the COMLEX dictionary via the chart below. Each row in the table is tested against the lexical term's possible part of speech and if a valid part of speech exists, that part of speech is assigned to the term. The goal of these precedence relations were to capture as many possible noun phrases from the article being processed.

I took the COMLEX syntax dictionary, lowercased all lexical entries, and applied the rule set above to derive an initial part of speech tag for each.

C.2 Step 2 - Exception list

However, with just the above scheme to maximize possible noun phrases, one inevitably get undesirable candidates, resulting from several sources. Several words, whose part of speech was derived from several possible ones, are much better to be placed in an exception list. Other nouns are rightfully nouns, but are hardly ever used in a semantically meaningful way. Thus, the following words below were manually corrected in the dictionary.

These words were demoted to the *Other* category as they were too general, based on a manual inspection of the salient errors generated by the term finding

Original COMLEX POS Name If the any of the lexical item entries fit this category...	Gross POS (ordered by priority) ...then the derived gross part of speech is...
some WORDs	Punctuation (1)
SCONJ CCONJ PREP SCOPE	Discourse Markers (2)
NOUN	Common Noun (3)
ADJECTIVE ORDINAL CARDINAL	Adjective (4)
PRONOUN	Pronoun (5)
WORD AUX ADVPART ADVERB VERB DET	Other (6)
Not listed and has capitals	Proper Noun (7)

Table C.1: Conflation of POS in COMLEX to term finding gross POS.

algorithm.

time way case type kind object idea piece part point example thing use lot number week month year day work post go can are

Figure C.1: Demoted nouns.

In addition, with respect to discourse marker type words, these entries were also modified in the dictionary and promoted to the level of cue discourse markers.

Finally, some additional punctuations, abbreviations, and several words were inserted into the dictionary. The final dictionary was then converted into a standard DBM file pair, for use with the termer utility.

accordingly again also alright alternately although altogether and anyway anyways as_well_as as_a_result because but by consequently conversely equally finally fine first for_example for_instance further furthermore hence hopefully however incidentally indeed in_fact in_particular instead in_turn in_general last like likewise listen meanwhile moreover namely next nevertheless nonetheless nor now oh ok okay only or otherwise overall rather say second see similarly still so far so then therefore though thus too unless well where whereas why yet

Figure C.2: Promoted discourse markers.

C.3 Exceptions

Notably verb past forms (speech related) from WordNet 1.6 (found by executing `wn say -hypov`, except sense 8) were switched from possible adjectives to verbs. As such they were reclassified as noun phrase delimiters.

said added exclaimed introduced mused restated presented stated misstated answered replied responded prefaced premised announced declared articulated enunciated vocalized declared noted observed mentioned remarked appended supplied explained gave pleaded alleged told ordered enjoined directed instructed commanded required compelled called

Figure C.3: Demoted *Adjectives*, reclassified as *Other*.

Appendix D

Composite topic tree evaluation materials

This Appendix reproduces certain materials used in the evaluation of the composite topic tree construction module, discussed in detail in Chapter 4. The evaluation was run in November of 2000.

D.1 Guidelines for evaluation

This section reproduces the directions for the composite topic tree evaluation. Two volunteers from the health science library participated in the evaluation: Susan Klimney and Tracy Allen. They were shown the instructions and the author clarified the materials before they saw the results. The author then interviewed them for general impressions. The errors with the outline were tabulated afterward, independent of the post-evaluation interview.

The terminology for topic types was changed for the evaluation. Instance topic nodes (i.e., countries or diseases) were known as "subjects" in the evaluation, genus nodes as "metasubjects", and substructure topic nodes were simply "topics".

Hello:

First, thank you for volunteering to help me in my research. The research is part of a Digital Library Initiative here at Columbia. My project is about learning the topical structure of certain subjects that have an expository purpose.

For example, given example web pages from library websites, it tries to generalize and output a prototypical structure of a library website in an outline form.

Now I need your help. I need an information specialist like yourself to critique the output of the system, the topic outlines. Basically I would like you help me decide which parts of the outlines are reasonable and which are wrong. If possible, try to characterize the types of mistakes the system is making. I've included a sample outline and questions below to think about.

When you are ready, please let me know. The session consists of evaluating two outlines from the system's run on patient information articles and on travel brochures. The whole evaluation should not take more than 30 to 45 minutes.

Thanks!

Min

Figure D.1: Guidelines for the CTT evaluation process, part 1.

Here is some example of output:

N.B. See notes below about how to critique this tree.

Output is divided into Themes and Topics. A theme is described by multiple topics, and the topics may be nested.

```
Theme: Columbia Library

Topic: 1 About|About the library|Patron Information
        (default: About)
        [Moderately important]

        Topic: 1.1 Directions to the library|Directions|How to find us
        (default: Directions)
        [Not important]
```

Understanding the Topic description:
Nesting and ordering: The "1" and "1.1" indicate the topic's position in the outline. More nested topics are both indented and have additional levels indicated by "."s in their ordering information.

Variant forms: "—" separate variant forms of topics considered the same by the system. The system's best guess and what a generic header for this topic is given in the "(default: ") form.

Importance: is given in the "[]". The three levels are "Important", "Moderately important", and "Not important".

```
Subject: Columbia library|Columbia University Library|Columbia
Libraries|Columbia Milstein Library
Subject: Rice library
Subject: New York Public Library
Subject: Barnard library

Subtopic: 1 About|About the library|Patron Information
        (default: About)
        [Moderately important]

        Subtopic: 1.1 Directions to the library|Directions|How to find us
        (default: Directions)
        [Not important]

        Subtopic: 1.2 Hours|Hours of Operation|Opening Hours
        (default: Hours)
        [Moderately important]

Subtopic: 2 Catalogs|Catalog|Card Catalog|Books
        (default: Catalog)
        [Important]

        Subtopic: 2.1 About the on line catalog|About the Catalog|This
                    option temporarily unavailable
                (default: This option temporarily unavailable)
                [Not important]

        Subtopic: 2.2 Go to online catalog|Online Catalog
                (default: Online Catalog)
                [Important]

Subtopic: 3 References|Reference Desk
        (default: References)
        [Important]

        Subtopic: 3.1 Ask a reference question
        Subtopic: 3.2 Reference Hours
```

Figure D.2: Sample output used in instructions.

Critique:

1) Please comment on the outline's content.

Answers might touch upon these aspects: (Themes) Are the themes that were detected describable by the outline? Are the themes that have been merged together as a single theme correctly merged? (Topics) Do the topics that are listed completely describe the type of theme? If not, can you suggest other topics. Do the topics that are merged really one topic? Are they incorrectly merged or split across different topics? Is the selection of the default topic header a good one? If not, what might be a better choice? Do some topics appear in more than one place? If so, where do you think the best place for this combined topic might be?

[Topic 2.1 seems to have merged together something wrong. "This option ..." doesn't make any sense here. Also 5 "Services" should not be lumped together with Interactive services. ...]

2) Please comment on the outline's sense of ordering and nesting for topics, given by the outline label (e.g., "1.2.4"). Note that this is not about the priority or importance of the topic (see next question):

Answers to this question might touch upon these aspects: (Ordering) How do you feel about the ordering of topics for all topics that are labeled "Important"? Are there themes or topics in which the order of its children topics are especially sensitive to ordering? Or some which whose children can be listed in any order? Should of the topics be switched around? (Nesting) Should any of the more nested topics be promoted or vice versa? Are the differences between different nesting levels sensible? Should some intermediate nesting levels be introduced between levels in the ordering? Are some of the topics on the same level incompatible?

[Topic 4 "Reserves" might be better nested down under services or a missing, maybe under topic 3 "References"...]

3) Please comment on the outline's sense of priority, as reported in the topics and theme's priority given (in brackets: "[]"):

Answers to this question might touch upon these aspects: Are the important topics really important? Would you consider any of the topics to be mislabeled in terms of priority? Are the 3 levels of priority enough to differentiate different levels of detail that people might want to know about? Are some topics switched in priority.

[I think "Directions" is pretty important...]

General Comments:

Do you feel the distinction between importance and nesting is clear? Do you feel the outline captures the internal structure of the type of theme well? Do you feel the outline captures a good default view of its themes, or is there a better way to present the information? Do you feel that the outline is sensitive to a particular medium of distribution (e.g., print, website)?

[This outline of library information emphasize online resources, whereas many libraries don't have this information ...]

Figure D.3: Critiquing guidelines in the CTT evaluation process.

D.2 Evaluation composite topic trees

In this section, I reproduce the actual outline formats of the composite topic trees that were used in the evaluation. Table D.1 gives a summary of the topic node

types as inferred by the algorithm. The original evaluation materials were also in color, where the typicality (raw frequency) of the field was translated into a color. Red meant that the topic was important (or typical) and blue meant the topic was least important (or rare). I have translated this to the grayscale equivalent, using black to indicate typical topics, and varying degrees of greys to indicate rarer topics.

	Consumer health information	Travel brochures
Genus nodes	5	35
Instance nodes	3	15
Substructure nodes	32	80

Table D.1: Node types produced by the algorithm for the two text types in the CTT evaluation.

D.2.1 Consumer healthcare documents on diseases

Subject: **angina** | angina pectoris | angina pectoris information | chest pain due to angina and other causes
Subject: first aid for choking | awake adult
Subject: heart attack | about heart attacks | heart attacks | heart attack myocardial infarction | recovery | overview | heart attack guide part two receover rehabilitation medications exercise lifestyle modifcations depression | what is a heart attack | heart | chest pain / heart attack symptoms

Metasubject: disorders
Metasubject: non traumatic emergencies
Metasubject: ruling out a <subject>
Metasubject: expect to be there awhile
Metasubject: in the er | medications often given in the er | procedures

Outline:
1. <subject> disease
2. basic information
    2.1 description
3. signals of a <subject>
4. frequent signs and symptoms | signs & symptoms any of the following
5. the cardiac care ccu | care for a <subject>
6. symptoms
7. <unknown>
    7.1 atherectomy
    7.2 laser angioplasty | coronary angioplasty
8. coronary arteries in <subject> disease | coronary artery bypass | coronary bypass
9. atherosclerosis | what is atherosclerosis
10. what is <subject>
    10.1 what is <subject>
    10.2 what symptoms can occur with <subject> | **what are the symptoms of a <subject>** | symptoms
    10.3 what brings on <subject>
    10.4 **what causes a <subject>** | what causes <subject> | causes | what are other causes of chest pain
    10.5 how does a patient recover from a <subject> | does <subject> mean a <subject> is about to happen | how long does it last
    10.6 what are the symptoms of <subject>
    10.7 cause
    10.8 other types of <subject> | are there other types of <subject>
    10.9 variant <subject> or prinzmetal s <subject> | what s new in the evaluation of <subject>
        10.9.1 microvascular <subject>
    10.10 risk factors | **what are the risk factors for <subject>**
    10.11 diagnosis | why is it important to establish the diagnosis of <subject> | making a diagnosis
    10.12 <subject> and <subject> risk
    10.13 when are you likely to have an attack of <subject> | is an <subject> attack the same as a <subject>
11. general measures | **preventative measures**
    11.1 outcome | expected outcome
    11.2 the danger of chest pain | chest pain what to expect in the er | is all chest pain <subject>
12. <subject> an ji nuh or an juh nuh pek tuh ris
    12.1 see also
    12.2 what is a <subject>

D.2.2 Consumer travel brochures

Subject: moroccan highlight
Subject: point of arrival
Subject: getting there
Subject: luang prabang | prabang luang attractions
Subject: highlight | muonglao com
Subject: laos | laos travel information
Subject: tourist | usefull addresses
Subject: address | contact info
Subject: outdoor & adventure online | the
Subject: economy
Subject: culture
Subject: <subject> | calculators
Subject: travel around | morooco travel tips
Subject: food
Subject: morocco

Metasubject: salam walaykoom
Metasubject: introducing
Metasubject: first time
Metasubject: independent
Metasubject: international bicycle fund
Metasubject: travel guide to <subject> by bike
Metasubject: <subject> virtual tours of <subject> maghreb
Metasubject: sections
Metasubject: admin
Metasubject: map
Metasubject: tips
Metasubject: postcards
Metasubject: essaouria tips
Metasubject: fez tips
Metasubject: shopping
Metasubject: <subject> other essentials | <subject> other <subject>
Metasubject: <subject> blue gate of fez
Metasubject: our official guide rashid and palace gates
Metasubject: beautiful farsi metal work
Metasubject: one of many mosques in <subject> fez medina
Metasubject: <subject> fez medina
Metasubject: medina gate
Metasubject: chefchaouen stoned and smiling
Metasubject: casablanca time did go by
Metasubject: erdouz valley
Metasubject: take a good map
Metasubject: camiones
Metasubject: essaouira port
Metasubject: camel on <subject> beach
Metasubject: <subject> fish stalls at <subject> port will cook up your choice cheap
Metasubject: essaouira medina
Metasubject: portuguese gate
Metasubject: castles in <subject> sand
Metasubject: riding
Metasubject: peter m geiser s internet travel guide
Metasubject: adventures of <subject>
Metasubject: asian connection

Outline:
1. clothing
2. soussan group sites
    2.1 enteleky com
    2.2 soussangroup com

- 2.3 alhoceima com | convergenceway com
- 2.4 pcrif net | s earch net

3. city of alhoceima
   - 3.1 w h a t i s n e w | n e w s
   - 3.2 special resources
   - 3.3 art gallery
   - 3.4 banks
   - 3.5 bazaar
   - 3.6 biz reporter

4. business network |doing business
   - 4.1 calendar local |calendar personal
   - 4.2 chat room
   - 4.3 city directory |city walls
   - 4.4 short facts |city facts
   - 4.5 city info

5. city news |news
   - 5.1 computer training

6. currency |currency exchange
   - 6.1 education
   - 6.2 hotels
   - 6.3 internet cafe
   - 6.4 investing
   - 6.5 investor guide
   - 6.6 local forum
   - 6.7 mailing list
   - 6.8 market news
   - 6.9 <subject> index |<subject> 3000
   - 6.10 creativity & invention
   - 6.11 phone info
   - 6.12 satellite pix

7. search |search <subject> |search site
   - 7.1 slides
   - 7.2 travel agents
   - 7.3 weather
   - 7.4 web design

8. <subject> people |people
   - 8.1 money
   - 8.2 air
   - 8.3 environment
   - 8.4 road
   - 8.5 drink
   - 8.6 staying healthy |healthy |staying
   - 8.7 rail
   - 8.8 drugs

9. post and communication
   - 9.1 guides
   - 9.2 <subject> anti atlas and sarhro |ifrane d anti atlas
   - 9.3 L bargaining
   - 9.4 <subject> sirwa
   - 9.5 transport
   - 9.6 <subject> rif

10. in numbers |mountain tourism in <subject>
    - 10.1 <subject> ski stations

11. moroccan activities |sports and leisure activities |activities

12. accommodation
    - 12.1 skiing
      - 12.1.1 porterage
      - 12.1.2 organisation
    - 12.2 golf
      - 12.2.1 information
    - 12.3 <subject> protection of <subject> mountains |<subject> valley of <subject> ziz |<subject> valley of <subject> draa |<subject> valley of <subject> dades

12.4 useful information
12.4.1 situation
12.4.2 climate
12.4.3 language
13. time
14. getting to <subject> | **getting around**
14.1 getting to <subject> mountains
14.2 imperial cities |<subject> imperial cities |a tour of <subject> imperial cities
15. rabat
16. meknes
17. fe z
18. marrakesh
19. <subject> souks
20. practical information

Appendix E

On the collection of annotated bibliographic entries

Broadly speaking, my research has focused on how automatic text summarization techniques can be applied to understanding search engine results. My goal is not to analyze what makes one summary better than another, but to learn how to generate a suitable summary of a resource based on machine learning over a compiled corpus. A "suitable" annotation can span many different dimensions, but in my case mainly concerns space/length limitations.

In the comprehensive study, I examined a class of summary texts, the annotated bibliographic entry. Annotated bibliographies are created mostly by abstractive methods and include both indicative and informative forms. An annotated bibliographic entry is a summary of a book or other resource that annotates a resource with a description of the text, as shown in Figure E.1.

From empirical observations of both annotated bibliography entries, snippets (Amitay, 2000), and online public access catalog entries, bibliography entries have some unique features that make them attractive and challenging to process. Bibliography entries often:

Maxwell, S. E., Delaney, H. D., & O'Callaghan, M. F. (1993). Analysis of covariance. In L. K. Edwards (Ed.), Applied analysis of ...

This paper gives a brief history of ANCOVA, and then discusses ANCOVA in the context of the general linear model. The authors then provide a numerical example, and discuss the assumptions of ANCOVA. Then four advanced topics are covered: ... This paper is quite theoretical and complex, but contains no matrix algebra.

Figure E.1: Sample excerpt from an annotated bibliography entry.

- are lengthier than both online catalog summaries and snippets. They often exhibit more variation of sentence structure and lexical choice. This makes the subsequent analyses rich and allows (re)generation based on these analyses to construct more varied and interesting text.

- are organized around a theme, making them ideal standard for "query-based" summaries. Bibliography entries also have more explicit comparison of one resource versus another, which can help a user determine whether which document to choose for a particular purpose.

- have prefacing text that overviews the documents in the bibliography. This preface text is a good model for summarizing a set of related items (e.g., different books on arms and armor or different earthquakes reports in 1992). This is in contrast to multidocument summaries that summarize articles with mostly overlapping information (news reports on a single event and updates to the event).

- are rich in meta-information document features — they often mention edition, title, author and purpose. These document features are not always present

in or inferable from the body text of a source document. The preliminary study of online catalog entries (detailed in Section 6.2.3.1) showed that these document features are well represented (and thus important).

E.1 Annotated bibliography language resource

I designed the language resource of annotated bibliography entries to ease the collection of the corpus as well as to make many features available for subsequent analysis for summarization and related natural language applications.

E.1.1 Collection methodology

The collection of the bibliography entries was done by spidering search result pages from two search engines (AltaVista and Google) for the keywords "annotated bibliography". The collection was compiled in September 2001 and software filters were written to parse and retrieve the contained URLs from each site (200 from AltaVista and an additional 1000 from Google). By my estimates, roughly 60% of the pages that were gathered had errors in retrieval (e.g., were stale URLs), were duplicate entries, or did not contain bibliographic entries. This leaves an approximate 500 pages with actual bibliographic entries to draw from.

An examination of the materials in these remaining documents revealed that most pages organized around a specific purpose, and varied greatly in collection size. Most common were large collections of 20 to 100 entries and introductory pages to even larger collections (over 1000 entries). Pages that only annotated a few items were much less common; we suspect that this is due to the inherent bias of the search engine ranking metric to rank sites that are more prominent (which we believe is highly correlated with larger collections). The smaller collections were often a part of a larger website or were the last section of a larger webpage

on the topic of interest. With this structure in mind, I decided to take at most 50 entries from each source document to ensure that we covered a breadth of annotated bibliography entry sources in collecting the final corpus. I examined the documents in order of their appearance on the AltaVista hitlist, and as a result, only a total of 64 documents from the AltaVista spidered collection were used to create the 2000-entry corpus. If all of the bibliographic entries were extracted from the documents, it would easily exceed 20,000 entries in size (as many of the collections had many more than 50 entries).

E.1.2 Encoding the XML bibliographic entry corpus

Bibliography entries from the 64 spidered pages were then manually cut-and-pasted into the corpus collection web interface. This was both to ensure that the entries were being correctly delimited, and to add fields to each entry that may assist in future analysis and serve as a gold standard for future machine learning tasks. The corpus is encoded in XML and includes the following fields in addition to the bibliographic entry itself.

- Subject: the subject or theme of the annotated bibliography page.
- Domain: annotated to aid analysis of differentiation of features that are domain-independent from ones that are domain-dependent. I encoded the domain rather coarsely (e.g., all of `medicine` as a single domain) and in an ad-hoc manner without the assistance of an ontology. Finer granularity is provided by the above `subject` field.
- Micro Collection (optional): the internal division in the bibliography page that the entry is a part of (e.g., "reference books" section of a bibliography on the colonial times in Jamestown).

- Macro Collection (optional): the division that the physical bibliography page represents in the set of related bibliography pages (e.g., "all colonies in colonial times in the U.S." with respect to the last example). The macro collection field is used when the bibliography physical page relates itself to other physical pages. In our observations, only very large collections exhibit both micro and macro collection attributes. Figure E.2 illustrates the relation of these two attributes.

- Offset: the position of the entry on the page.

- Before Context: text before the body of the annotated entry itself. This often contains cataloging and bibliographic information, such as the title, author, and call number[1].

- After Context (optional): text that is distinctly marked off as coming after the body of the annotated entry. Used sometimes to mark publisher information, web URLs and pointers to other resources. Information that typically is contained in this field in one document may simply be appended to the end of the bibliographic entry in other documents; this distinction may be more of a stylistic one.

- URL: the web location of the source document where the entry was drawn from.

To facilitate the local analysis of the corpus, all of the bibliographic entries have also been parsed with a probabilistic dependency parser (Collins, 1996). These parsed entries are also included in the XML corpus, as a separate XML field attached to each entry (the `parsedEntry` field). Figure E.3 shows a sample entry after it has been parsed into our XML format.

[1] Currently, this is saved as an unstructured text field. It would be best to parse these entries

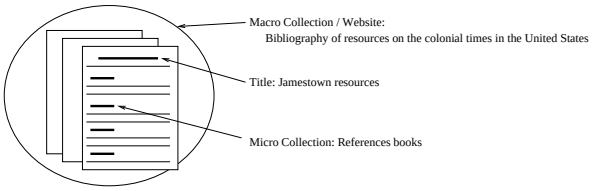


Figure E.2: Relation of micro and macro collection attributes
b.

```
<bibEntry id="id26" title="Analysis of covariance"
  url="http://www.math.yorku.ca/SCS/biblio.html"
  type="paper" domain="statistics"
  microCollection="Analysis of Covariance"
  offset="4">
  <beforeContext>
    Maxwell, S. E., Delaney, H. D., & O'Callaghan,
    M. F. (1993). Analysis of ...
  </beforeContext>
  <entry><OVERVIEW>This <MEDIATYPES>paper</MEDIATYPES>
    gives a brief history of ANCOVA, and then discusses
    ANCOVA in ... contains no matrix
    algebra.</DIFFICULTY>
  </entry>
  <parsedEntry>
    PROB 14659 -112.252 0 TOP -112.252 S -105.049 NP-A
    -8.12201 NPB -7.82967 DT 0 This NN 0 paper ...
  </parsedEntry>
</bibEntry>
```

Figure E.3: Portion of the annotated bibliographic entry from Figure E.1, represented as structured fields in our XML corpus.

E.1.3 Semantic annotation of metadata

Once the bibliography entries are collected and stored, one needs to annotated the entry in order for the machine to learn from it. I tag *metadata predicates* (examples listed below) so that SIGNAL can learn how these elements appear in the summary: for example, as sentences, noun phrases, or as adjectives.

I use a greedy tagging methodology that is used to exhaustively cover all parts of an entries. The reason is that this will enable SIGNAL to learn all variations of the the patterns exhibited by the corpus. I give an example, where the bold text

into structured fields but our focus is on the text and content of the entries themselves, and not these auxiliary fields.

below all is tagged with the *Readability* tag:

1. A short, **well-written** book that goes on to demonstrate ...
2. The book is perhaps a bit short but **is written well**, and thus is ...
3. **The book is well-written.** A bit short but ...

The three tags are of three different granularities and are equally valid ways of expressing the information's contents. I disregard problems if intervening function words do not strictly belong to the semantic category being tagged.

E.1.4 Listing of semantic metadata predicates

Below is a list of semantic types of materials I have come across in the analysis of bibliographic entries:

- *Audience.* What is the target audience of the document? Who is it useful for? What is the target age group?
- *Author/Editor.* Who is the author(s) or editor(s) of the resource? Information on the authors, their credentials should be included in this tag.
- *Authority.* Information on the why this document is important or considered an authority. Information on detail, comprehensiveness, if not seemingly subjective and given by the annotator should be tagged with this tag.
- *Award.* What accolades has this document received? Has it won any specific awards? Are there any short quotations from other authoritative or respected sources that review or give information on the document?
- *Background.* Introductory or prefatory information that is needed to explain the topic of the document. May be hard to distinguish from the material

in the book. Historical material needed to contextualize the contents of the document is included in this tag.

- *Collection Size.* In a collection document (that gathers together items – essays, articles, chapters), any reference to the number of items in the collection

- *Cross-resource Comparison.* Any information that compares this particular resource versus others explicitly. Don't get this confused with *Authority*, or *Subjective* which may use superlatives or relatives without naming specific resources.

- *Content Types.* Different types of media besides raw document text that distinguishes this resource. This includes multimedia as well as text features such as appendices, table of contents and indices.

- *Contributor.* Name of person or company who wrote this particular annotation. Credits for the annotation.

- *Copyright.* Any copyright statement assigned to the annotation.

- *Difficulty.* What information does the reader need to know before being able to read this resource. Related to the *Audience* tag. If the document is introductory or prefatory and requires no background, this tag can apply to these modifiers.

- *Detail.* Any information that is detailed information on the subject. This type of information is very fine-grained and detail-oriented, and should not be confused with more generalized comments on the topics of the document (see *Topic*). I theorize that information of this sort might be sentence extracted from the document.

- *Edition/Publication.* Who is the publisher? When was it published? What edition is the resource? Any information on the publisher, either original or current. Any information on the publication. Note well that rare or old documents have an extensive material on this axis.

- *Language.* What language is the resource written in?

- *Media Type.* What is type of resource is it? Is it a book, software, website, article, journal? Classification of the resource.

- *Navigation.* How is the document structured? Does it have chapters? Structuring information for the entire document as well as for subparts of the document are included.

- *Overview.* What is the document about (in a succeed, brief clause or sentence). Can be a plot summary.

- *Purpose.* What is the purpose of the document? What does it set out to do? Not what the document is about (*Topic*) or who it is intended for (*Audience*).

- *Query Relevance.* Annotation of the document with respect to the axis of the bibliography or query at hand. Not to be confused with *Detail* or *Comparison.*

- *Readability.* Description of the writing style of the document. Is the document particularly dry, well-written, confusing, or well-organized?

- *Topic.* Information at a medium or high granularity concerning the topics or content of the document. See also *Detail.*

- *Size/Length.* Physical size or length of the document.

- *Style.* General manner or description for how the book is written. For general descriptive features outside of those covered by other facets. An example is "written on actual parchment" or "using a bulleted style".

- *Subjective.* An annotator's personal view of the resource.

E.2 Corpus attributes

Table 6.3 also lists distribution of the tagged document metadata in the 100 annotated entries. The first column shows the number of times that the annotated feature was used to mark information in the entries. The second column gives the percentage of documents that have an instance of the feature in question. Features were marked at the sentence level or on smaller units. The columns are highly correlated, and show that multiple occurrences of the same tag within an entry happen quite frequently.

Table E.1 shows how the distribution of the 24 document features varies with length and indicates where the metadata predicates occur within the summary. The numbers between 0 and 1 in parentheses indicates how close the average instance of the document feature is to the beginning (0) of the summary entry or to the end (1). Middle range numbers (e.g., .50) often indicate that the field occurred widely across different positions in the entries, especially when the feature frequency is high. Entries tended to include 2 to 6 document features, and long bibliography entries were fairly rare (entries with 13 or more document feature instances represent only 6% of the annotated corpus). Normal entries containing 2 to 6 document features correspond to 2 to 4 sentence- or phrase-length entries.

Examining the ordering data, it is quite apparent that some of the fields naturally occur before or after others. *Overview* predicates generally comes very early in the bibliography entry, and information on who wrote the entry (the contributor) usually comes very late. *Subjective* assessment or critique of a resource usually comes after an explanation of the resource, thus comes later in the summary. Ordering among the features is quite variable, but it is obvious that many of

features either tend to occur earlier (e.g., bibliographic information) or later (e.g., subjective assessment or complicated types of metadata) with topical information filling in the space between.

Feature	Number of tags in entry							
Entry Length	1	2	3	4	5	6	7	8
# of Entries of Indicated Length	(4)	(10)	(14)	(16)	(16)	(9)	(5)	(7)
Detail			8 (.56)	14 (.69)	21 (.64)	18 (.66)	9 (.50)	13 (.62)
Overview	1 (N/A)	4 (0)	10 (.20)	10 (.13)	10 (.10)	8 (.05)	6 (.31)	8 (.05)
Media Type		1 (1)	6 (.58)	8 (.38)	8 (.83)	4 (.35)	3 (.33)	7 (.41)
Author / Editor		2 (1)	3 (.67)	2 (.67)	4 (.62)		3 (.61)	6 (.50)
Content Types		1 (1)	3 (.67)	4 (.83)	8 (.47)	1 (1)	1 (1)	3 (.76)
Subjective Assessment	1 (N/A)	2 (1)	2 (.50)	2 (.67)	6 (.71)	4 (.65)	3 (.67)	2 (1)
Topic		4 (.50)	2 (1)	2 (.67)	8 (.28)	2 (.30)	1 (.67)	4 (.57)
Authority		2 (.50)		1 (.33)	4 (.94)	3 (.47)	3 (.50)	4 (.64)
Background / Source			2 (0)	4 (.33)	2 (.38)	1 (.20)		2 (.21)
Navigation					1 (.75)		2 (.50)	
Collection Size		1 (0)	1 (0)	2 (.83)	2 (.38)		1 (.17)	1 (.57)
Purpose			3 (.83)	2 (.33)	1 (.50)		1 (.50)	1 (.29)
Audience		1 (0)		3 (.33)	3 (.42)			2 (.79)
Contributor				3 (1)	2 (1)	2 (1)		1 (1)
Cross-resource Comparison	2 (N/A)		1 (1)	2 (.33)		3 (.60)		
Size/Length				1 (0)		2 (.20)	1 (.67)	
Style						1 (.40)	1 (.83)	2 (.36)
Query Relevance								
Readability						3 (.53)		
Difficulty				3 (.67)				
Edition / Publication		1 (0)		1 (0)				
Language		1 (1)	1 (.50)					
Copyright								
Award						2 (.70)		

Feature	Number of tags in entry							
Entry Length	9	10	11	13	14	15	18	20
# of Entries of Indicated Length	(3)	(5)	(5)	(1)	(2)	(1)	(1)	(1)
Detail	4 (.50)	7 (.52)	12 (.58)		6 (.63)	6 (.48)	16 (.56)	5 (.53)
Overview	3 (0)	3 (.15)	5 (.22)	1 (.33)	2 (.12)	1 (0)	1 (.06)	
Media Type	2 (.19)	4 (.28)	8 (.28)	1 (.50)		2 (.36)		1 (.16)
Author / Editor	4 (.50)		4 (.68)	1 (.75)	7 (.34)	3 (.83)		4 (.53)
Content Types	2 (.50)	8 (.54)	7 (.70)	1 (.83)				2 (.45)
Subjective Assessment	3 (.62)	6 (.78)	2 (.65)		2 (.27)			
Topic		5 (.36)	3 (.27)		3 (.44)			
Authority	3 (.62)	1 (.67)		1 (0)		1 (.07)	1 (0)	2 (.47)
Background / Source	1 (.38)	1 (0)	3 (.13)	2 (.12)	2 (.88)			1 (.68)
Navigation	1 (.88)	5 (.56)	2 (.55)	2 (.33)		1 (.50)		
Collection Size		1 (.22)	2 (.60)					2 (.24)
Purpose			1 (.60)	1 (1)	3 (.36)			
Audience	1 (.62)			1 (.92)	1 (1)			
Contributor	1 (1)		1 (1)		1 (1)			1 (1)
Cross-resource Comparison			3 (.50)					
Size/Length		3 (.22)		2 (.62)				
Style		2 (.39)	2 (.85)					
Query Relevance	2 (.75)							
Readability					1 (.92)			
Difficulty		1 (1)						
Edition / Publication		1 (1)						
Language								
Copyright		2 (.94)						
Award								

Table E.1: Feature distribution across entries of different document lengths. Frequency of document feature given as entry, average relative position of feature given in parentheses (0 indicates the beginning of the entry, 1, the end of the entry). Document features listed in order of descending frequency in the annotated corpus.

Appendix F

CENTRIFUSER evaluation materials

This Appendix reproduces certain materials used in the holistic, system-wide evaluation of CENTRIFUSER, discussed in detail in Chapter 8. The evaluation ran from September to November of 2001.

F.1 Guidelines for evaluation

This section reproduces the guidelines and the interviewer's script for the think aloud interview. It was developed by the author and edited with input from the evaluation team, especially Dr. Andre Kushniruk. The evaluation was piloted on 15 August by two volunteers from the electrical engineering department: Javier Gomez-Castellanos and Hari Sundaram.

Pilot Evaluation Experiment README
Start date: Wednesday 15 August 2001

Part 1 - About 5-10 minutes

Ask questions about a healthcare scenario, elicit questions that the user would ask about.

- Pretend that you experience very sharp chest pains during running. You know that the medical terminology for this condition is called "angina". What questions would you have about this condition?
* How would you go about looking up information about this?
* Would you use a computer?
* Would you use the internet / WWW?
* What types of information might you look for?
* What types of internet resources might you try to look at?
* Describe the types of information you are expecting to find in your search.
* Give examples.

Suppose you've gone to a search engine as a first step for looking for information. It provides a simple text box for your query.
* What type of query might you initially try?
- Would you use keywords or sentences?
- Would you add alternate phrases or words in a first query or not?
- Would you rather choose from some predefined queries?

Figure F.1: Guidelines for the evaluation process, part 1.

Part 2 - About 20-25 minutes
Individual search engines in randomized order
(Put search engine shell on screen)

Let's go through a particular scenario. We have issued a query related to your search (<interviewer: read query here>) to four different types of search engines. You're going to look at the initial screen of each one of them sequentially. We'd like you to think aloud while you are doing this and we'd like your comments about the following:

* usefulness of information presented
* organization and structuring of information
* understanding of the information

Any problems understanding the information presented?
Any suggestions for improving the information or its presentation?

Probes:
- After this initial screen of results, do you think you have an answer to your question or information need?
- If not, does the information on the screen help you decide what to do next?
- Do you think you would start your search over instead of following up with the information that you've been presented with on this screen?
- Describe exactly what you'd do next

Figure F.2: Guidelines for the evaluation process, part 2.

Part 3 - About 5-10 minutes
Cross comparison of engines

(subject can refer to screen, go back to search engines forgotten about; mic should still be on. N.B.: They may have already done some cross-comparison during the individual analysis, get them to repeat it for easier analysis)

Now that you've seen all four search engines, we're going to ask you to compare them against each other.

Please rank the four engines – A, B, C and D – in terms of their *INITIAL* result screen's usefulness of the information content provided (not what types of information is present), on a scale of 1 (worst) to 5 (best) in answering your question. Please justify why.

On a scale of 1 to 7, (7 = most clear what to do next) the engines in terms of their ability to help you decide what to do next (either click on a link or revise your search).

* Please justify why / explain?

Again, please rank on scale of 1 (worst) to 7 (best) the usefulness of the TYPES OF INFORMATION (independent of the content) provided by the search engine (e.g., navigation buttons, URLs included, summaries). Was it obvious what each clickable link was supposed to do?

* Please justify why / explain?

On a scale of 1 to 7 (7 = very quickly), how quickly do you think you can locate the information you need using each search engine?

* Please justify why / explain?

On a scale of 1 to 7 (7 = best), please rank the engines in terms of their layout of information that it provided you. Can you suggest a better layout of any of the content that would help fellow users?

* Please justify why / explain?

Please also provide general comments about what you liked and disliked about each of the four systems.

Probes:
- In any of the search engines, are there any follow up links or features that you wish that the interface had?
- Can you think of other pertinent information that isn't expressed in any of the search engines that you would have liked to know about?

Figure F.3: Guidelines for the evaluation process, part 3.

F.2 Screenshot of the evaluation interface

Figure F.4: Screenshot of the evaluation interface.

Figure F.4 gives a screen capture of the evaluation interface as seen by subjects. Subjects were interacting with the author (as interviewer) using the evaluation interface on a laptop computer in the intensive care unit of the hospital. Evaluation questions and probes for the individual search engines were asked by the interviewer during the course of the first part of the interview.

F.3 Ranking survey

In Figures F.5 and F.6 show a reproduction of the comparative ranking survey used during the holistic evaluation. The surveys were given on paper; answers were circled by the subjects.

First, please enter your email address (this way we can contact you, we won't use or sell your email address for any reason):

[pop up main window]

1. Please rank the systems in terms of their screen's CONTENT provided (not what types of information are present), on a scale of 1 (1 = worst) to 7 (7 = best) in answering your question.

[pop up main window]

System A: **worst** 1 2 3 4 5 6 7 **best**
System B: **worst** 1 2 3 4 5 6 7 **best**
System C: **worst** 1 2 3 4 5 6 7 **best**
System D: **worst** 1 2 3 4 5 6 7 **best**

Please justify why you ranked the systems as you did.

2. Now please rank on scale of 1 (1 = worst) to 7 (7 = best) the range of information sources (independent of the content) provided by each of the systems (for example, sources such as navigation buttons, locations of documents, summaries).

[pop up main window]

System A: **worst** 1 2 3 4 5 6 7 **best**
System B: **worst** 1 2 3 4 5 6 7 **best**
System C: **worst** 1 2 3 4 5 6 7 **best**
System D: **worst** 1 2 3 4 5 6 7 **best**

Please discuss how you would justify your rankings.

3. On a scale of 1 (1 = most confusing of what to do next) to 7 (7 = most clear what to do next), please rank the systems in their ability to help you decide what your next step would be (for example, either click on a link or revise your search).

[pop up main window]

System A: **most confusing** 1 2 3 4 5 6 7 **most clear**
System B: **most confusing** 1 2 3 4 5 6 7 **most clear**
System C: **most confusing** 1 2 3 4 5 6 7 **most clear**
System D: **most confusing** 1 2 3 4 5 6 7 **most clear**

Please justify why you ranked the systems as you did below.

4. On a scale of 1 (1 = very slowly) to 7 (7 = very quickly), how quickly do you think you can locate the information you need using each system? If you found the information you were looking for, was it difficult (1 = difficult) or easy (7 = easy) to find?

[pop up main window]

Figure F.5: First page of the comparative ranking survey.

Why do you think so? Again, please discuss this question and your rankings below:

5. On a scale of 1 (1 = worst) to 7 (7 = best), please rank the systems in terms of their pres

[pop up m

System A: **worst** 1 2 3 4 5 6 7 **best**
System B: **worst** 1 2 3 4 5 6 7 **best**
System C: **worst** 1 2 3 4 5 6 7 **best**
System D: **worst** 1 2 3 4 5 6 7 **best**

Again, please justify your answer.

6. Overall, how happy are you with the results from each system?

[pop up m

System A: **unhappy** 1 2 3 4 5 6 7 **happy**
System B: **unhappy** 1 2 3 4 5 6 7 **happy**
System C: **unhappy** 1 2 3 4 5 6 7 **happy**
System D: **unhappy** 1 2 3 4 5 6 7 **happy**

Please discuss your reasons for your rankings:

Figure F.6: Second page of the comparative ranking survey.

F.4 Demographic survey

In Figures F.7 and F.8 show a reproduction of the demographic survey used during the holistic evaluation. The surveys were also given on paper; answers were circled by the subjects.

Pre-evaluation Survey

Please fill out the following questionnaire before proceeding to the main experiment. The details of the survey are for demographic purposes; they won't be used to identify you. You don't have to answer any question you don't feel like answering, but the answers will greatly assist the evaluators.

Thank you.

I. Demographic Questions

1. Age:

< 20 years old	35-45	60-75
20-35	45-60	>75 years old

2. Your sex:

Male Female

3. Educational background (check off the following that best describes the highest level of education you have achieved:

Elementary School	High school	Technical School	College Degree (Arts)
College Degree (Science)	Master of Arts	Master of Science	Ph.D.
Medical Doctor	Other		

4. Economic status (based on yearly average income)

Employed (student)	Unemployed	$15-30,000	$45-75,000
<$15,000	$30-45,000	$75-100,000	>$100,000

5. Are you currently a full-time student?

Yes No

II. Computer Background Questions

1. How often do you use a computer?

Several times every day	Once a week	Several times a week
Once a month or less often		

2. How many years have you used a computer?

Figure F.7: First page of the demographic survey.

Yes No

4. Is there a computer at your home?
   Yes No
5. How long have you used the internet for?
   6 months - 1 year 3-5 years < 6 months
   1-3 years > 5 years
6. Please check the search engines you've used, if any:
   yahoo.com dogpile.com mamma.com other
   hotbot.com altavista.com webcrawler.com never used a search engine
7. Please check all the computer systems which you currently use, if any:
   IBM compatible Macintosh Mainframe computer
   Other
8. Please check all the computer operating systems that you currently use, if any:
   Windows DOS Macintosh or Apple OS
   UNIX Other
9. Please check all those types of computer programs that you use or have used, if any:
   Word Processing e-mail
   File or Database Management Systems Spreadsheets
   Computer Games Programming Languages
10. Please check how often you use the World Wide Web:
    Several times every day Once a day Several times a week
    Once a month or less often
11. If you've used the World Wide Web, have you ever used it to get information about hea
    Yes No
12. Please check the general medical on-line sites you have used, if any:
    WebMD.com medicineonline.com other
    Dr Koop.com medicinenet.com never used an on-line site

Figure F.8: Second page of the demographic survey.

Index

About.com, 251
  as used in comparative evaluation, 190–191, 212
AlgoNet2, 33
AltaVista, 36, 119, 248, 249
  as source for corpus, 286
angina
  sample file, 260–270
annotated bibliography entries, 22
  comparison to OPAC entries, 131, 284
  corpus of, 128–132, 284–294
    features of, 293
  standards, 127–128
Archie, 245
ask-a service, 21
associated text, 146, 158–166
ATS, *see* automatic text summarization
attribute value, 146, 158–166
automatic corpus annotation, 147
  genericity feature, 148
automatic text summarization, 3
  aspects of, 35
  generic, 36
  indicative, 36, 121, 144
    as neglected by IR community, 123
    definition of, 122
  informative, 36, 121
  multidocument, 8, 36, 103, 109
  query-based, 36, 110

bibliography
  as inventorying, 27
  definition of, 22
browsing scope, 101

C99, 57
Cat-a-cone, 33
cataloging
  as branch of bibliography, 23
Centrifuser, 4–7
  architecture of, 98
  as compared to standard ranked list, 16–31

as indicative and informative summarizer, 37
components of, 97
holistic evaluation of, 189–201
settings used in evaluation, 192
time complexity of, 117
CFUF, 167–179
cgi, *see* Common Gateway Interface
co-training, 57
cognitive engineering, 12
collaborative filtering, 2, 19
Collins parser
use in corpus analysis, 147, 288
COMLEX, 260, 270
as POS lookup dictionary for Segmenter, 271–274
Common Gateway Interface, 209
content planning, 134, 157
hand-coded, 145

DEPICT, 33
discourse markers, 273
document category, 114–117, 134–136
definition of, 114
inventory of, 114
document clustering, 252
document granularity, 80, 95
document type
multi-instance, 80, 85
single-instance, 80
sub-instance, 80, 88
Domain Communicative Knowledge, 187

EDGAR, 255
evaluation
anonymization used in, 193
of CLASP, 68
of CTT merging, 91–94, 276–283
of Segmenter, 53
of tagging versus lookup for part-of-speech, 47
survey
demographic, 194, 199, 200
ranking, 198
use of hypergeometric distribution, 54
use of Kappa, 69
use of Q-test, 54
exception list, 272

Fergus, 166
File transfer protocol, 245
FTP
seeFile transfer protocol, 245
FUF, *see* Functional Unification Formalism
Functional Unification Formalism, 146

Google, 36, 120
as source for corpus, 286
as used in comparative evaluation, 190–191, 212
Gopher, 243–245, 248

HAC, *see* hierarchical agglomerative clustering
HALogen, 166
hierarchical agglomerative clustering, 55
hierarchical merging
for CTT construction, 88

IGEN, 166
indexing
as an access method, 29
multiple, 29
indicative group summaries, 5, 135, 143
corpus of, 7
information overload, *see* problem of selection
information retrieval
cognitive approach, 241
definition of, 242
framework, 2, 17, 30, 242
interface, *see* standard ranked list, 12, 31, 242
graphical, 32
textual, 34
rational approach, 241
informative extract, 5
Interactive Document Map, 33

Jughead, 245

Knight system, 187

layout- versus content-oriented structure, 58
level of detail, 119
in differences, 116
in extract, 106
in navigation bar, 102
lexical chain, 49
lexical chaining, 57, *see* lexical cohesion
lexical choice, 141
lexicon
automatic acquisition of, 157–164
hand-coded, 145
librarian
field specialist, 25
librarians
interaction with, 9
techniques used by, 13–31, 109–111
adapting to online context, 20

Library of Congress, 116, 210
Likert scale, 194
linking distance
    as approximation to lexical chain, 49
logical line styles, 59–62
logical structure tree, *see* topic tree

machine learning
    use of Ripper, 65, 147, 182
machine learning features
    in Coheser, 64–65
    in Header, 62
    in Layser, 59–60
MAGIC, 175
mapping, *see* topic mapping
Maxim of Relevance, 111
metadata
    annotation of, 214
    as summary content, 123
    automatic annotation of, 147–150
    automatic inference of, 26, 213
    corpus inventory of, 129–131, 290–293
    crosswalking of, 125
    in different domains, 214
    standards, 124–125, 143
        Dublin Core, 125
        MARC, 124, 210
    use in relevance judgments, 25
    use of, 213
metasearching, 17

name authority, 256
natural language generation, 206
    use in Centrifuser, 116–117
navigation links, 4
Nitrogen, 166
node type
    genus, 79, 84, 276
    instance, 79, 88, 276
    relation to RTL, 84
    substructure, 79, 88, 276
nonverbal communication, 15
normalization
    zero-sum
        used by Segmenter, 52
Northern Light, 252
noun phrase
    head of, 47
    regular expression used by Segmenter, 47
    simplex, 47
noun phrases
    use in linear segmentation, 45

OCR, *see* optical character recognition

ODP, *see* Open Directory Project
ONIX, 210
online public access catalog, 1, 12, 116, 210
OPAC, *see* online public access catalog
Open Archives Initiative, 125
Open Directory Project, 109, 249
optical character recognition, 43
OVID, 254

paragraph types
  in Segmenter, 49
PERSIVAL, 211
PLANDoc, 187
problem of selection, 216

query word in context, 25, 36, 191, 249
QWIC, *see* query word in context

readability metrics
  Flesch-Kincaid, 67, 124
  others, 67
  SMOG, 124
reference interview, 13
reference queries
  difficulty in formulating, 14
  distribution of, 13
  underspecification of, 14, 17
relative topic level, 83–84
rhetorical structure theory, 41
RST, *see* rhetorical structure theory
RTL, *see* relative topic level

Scatter-Gather, 34
search plan
  formulation of, 15, 17
semantic predicate, 134
  topical versus non-topical, 159
sentence clustering, 107
  time complexity of, 118
  use in removing redundant information, 108
sentence extraction, 123
SIGNAL, 189
  linking with Centrifuser, 213
SIGNAL comparison with Centrifuser, 157, 185–187
snippet, 191, 285
SOAP
  use of, 211
SPoT, 166
standard ranked list, 3, 16–31, 216
  definition of, 257
  ranking by relevance, 28
structural similarity metric, 81–83, 209
subject guide, 12

subject guides, 22, 29
SuperBook, 34
surface realization, 142
SURGE, 167–179
syntactic aggregation, 175

text summarization
  multidocument, 206
text type, 73, 129, 211
  consumer health, 129
  manual simulation of, 91
TextTiling, 44
think aloud protocol, 195–197
TileBars, 33
Topic Islands, 33
topic mapping
  failure of, 104
  of DTT topics to CTT, 104
  of query topic, 98
topic segmentation
  hierarchical, 206
topic tree
  as container for metadata, 39
  as having multiple levels of granularity, 39
  composite, 5, 97–98, 110, 137, 207
    as model of expected information, 10
    as norm, 95
    use in navigation bar, 101
  document, 6, 34, 39–72, 97–98
topic type, 113, 143
  definition of, 103
  relation to browsing scope, 103
  use in computing document category, 112–116
  use in extract, 104
topical versus non-topical metadata
  distribution in corpus, 293

user customization, 31, 253, 255
user model, 19, 97, 120, 253
  browsing, 20, 99–108
  reference librarian constructing a, 21
  searching, 20, 108–113

Veronica, 245
virtual reference desk, 21
VSComp, 34

WAIS, 245–246, 248
WestLaw, 253
WordNet, 49

XML, 211
  as corpus encoding method, 287
  style sheets for, 42

Yahoo, 250
    as used in comparative evaluation,
        190–191, 212