UNIVERSITY of CALIFORNIA

Santa Barbara

Pharos: A Scalable Distributed Architecture for Locating Heterogeneous Information Sources

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Computer Science

by

Ron A. Dolin

Committee in charge:

Professor Divyakant Agrawal, Co-Chair
Professor Amr El Abbadi, Co-Chair
Professor James Frew
Doctor Linda L. Hill
Professor Terence R. Smith

June 1998

The dissertation of Ron A. Dolin is approved:

[illegible]

[illegible]

[illegible]

Divyakant Agrawal

Co-Chair

[illegible]

Co-Chair

June 1998

Pharos: A Scalable Distributed Architecture for Locating Heterogeneous Information Sources

Copyright © 1998

by

Ron A. Dolin

All rights reserved.

To Pepa

"Imagination is more important than knowledge."

Albert Einstein

"The world will little note, nor long remember,
what we say here..."

Abraham Lincoln, Gettysburg Address, 1863

"Why settle for excellence when perfection is attainable?"

Mr. Mathesson, High School English Teacher

"There are two kinds of dissertations: those that are perfect and
those that are finished."

Unknown

Acknowledgements

In martial arts, the attainment of a black belt is often considered to be the point at which one may begin to learn. The attainment of a Ph.D. might be thought of similarly. In one sense this implies that this dissertation is a work in progress. And yet, I hope that I have reached the level of emotional and intellectual maturity necessary to begin to learn about Computer Science and its increasingly numerous and diverse applications. "No man is an island," and I simply could never have accomplished any worthwhile results without a lot of support, in a thousand different ways, from many people around me. I would like to thank at least a few of them here.

Santa Barbara is a wonderful place to live, and it presents demanding alternatives to studying – SCUBA and kayaking not the least among them. I would like to thank the staff of UCSB Arts & Lectures for providing so many wonderful distractions. I would like to thank the dolphins. And the whales, sea lions, sea gulls, hawks, and pelicans. Also in Santa Barbara is Salvatore Espresso Systems, Illy beans, and all the staff at Internet Café!

This work was supported by NSF/DARPA/NASA under grant IRI94-11330. Their financial support has been matched by the useful comments, suggestions, questions, and contacts at the DLI meetings. Among all the contacts, I would especially like to thank Eugene Miya at NASA and Maria Zemankova at NSF.

Amy Friedlander, editor of D-Lib Magazine, provided me with helpful editorial comments, visibility, and friendship from an unexpected place.

I would like to thank all the staff of the Computer Science Department at UCSB, especially Ben Humphrey and Louise Brooks-Sciutto. They have made graduate school more enjoyable and have helped build a sense of community in the department.

Justin Pearlman spent the summer of 1997 working with me as an undergraduate intern. He helped implement a key component of the prototype. He was very enjoyable to work with and I'm sure he'll go far.

I received help from many staff members at UCSB. In particular, Trish Kreiss made available to me the MARC records, and Kevin Schmidt the newsgroups; without these collections much of this work would not have been possible. Grace Thomas, Principal Cataloger at the UCSB Davidson Library, spent a lot of time hand classifying several news articles for me.

The staff at the Map and Imagery Laboratory have given me help and support for years, and have provided me with a second community outside the Computer Science Department that I call home. Ann Hefferman always included me in the parties. Larry Carver and Mary Larsgaard have been incredibly supportive of my work and have never held back helping me whenever I've asked for it. Beyond professionalism, Greg Hajic and Jason Simpson have become long-term friends. I could never say in public all the fun things we've done together.

The faculty of the Computer Science Department have provided an excellent learning environment. They have also successfully provided a graduate community that encourages friendship, responsibility, and intellectual development. I would like to thank Laurie Dillon for her help in advising me before leaving for

Michigan. I would particularly like to thank Alan Konheim for being a valuable mentor and friend, and for providing such candid advice as only he could.

The Alexandria Digital Library (ADL) Project has provided the framework within which I was able to pursue my research. ADL has involved a wonderful interaction of interdisciplinary engineers and scholars. Mary-Ana Rae (may she be Dr. Rae soon!) has always been particularly supportive and encouraging.

Friends too numerous to mention, with experiences too valuable to describe, have provided far more than I could have hoped for: fun, feedback, introspection, support, and on and on. Dave Flynt, Murat Karaorman, Dante De Lucia, Liz Keate, Robert Danen, Dan Tauber, everyone in the Distributed Systems Lab back to the time of Phivos Aristides (especially the ADL gang – Daniel Wu, K.V. "Ravi" Ravi Kanth, Sunil Prabhakar) – what a collection of friends to have.

My committee has been wonderful. Terry Smith has often provided me with useful insights, even if it took me a few days at times to get the meaning. Terry's valuable high-level views were always complemented well with the nuts-and-bolts perspective of Jim Frew. Jim's sense of humor is invaluable – but I'm not sure the jokes are appropriate for this venue... Linda Hill has been exceptionally available and supportive. Her background in library science and information retrieval, her willingness to explain and answer questions, and her thorough and valuable comments and suggestions about my research have made me forever indebted to her.

My co-chairs Amr El Abbadi and Divy Agrawal have patiently guided me through the process of becoming a Ph.D. After switching topics and advisors, their willingness to take me on and work with me kept me on track during a crucial period. They helped me formulate my topic and continued to critique my work constructively. Amr and Divy have excellent writing skills; their editorial comments have always been very helpful in clarifying my points, achieving conciseness, and structuring the high-level perspectives. They have consistently demonstrated the highest level of integrity and dedication to research and teaching, while maintaining a healthy balance between their work and personal lives. I have enjoyed working with them and hope to continue to do so in the future.

When his brother, his last remaining living relative, died in 1997, Kurt Vonnegut wrote, “I was the baby of the family. Now I don’t have anybody to show off for anymore.” I am the baby of my family too, and, fortunately, they let me show off to them all the time. My parents came to my Ph.D. defense. They are all attending my graduation! They are all proud of me! I couldn’t ask for a more supportive family.

Tambien querría dar las gracias a mi familia de España, que siempre han sido tan cariñosos y me han apoyado tanto. Yo les robé a su niña, y ellos me han tratado como familia; por ello me considero afortunado.

Saving the best for last, I would like to thank a friend who, among other things, once made a wise investment in a professional espresso machine as a gift when I advanced to candidacy. It has already almost payed for itself financially, not to mention how many meals and papers it has enhanced. Whether discussing multidimensional vector functions, getting feedback on my latest paper, covering for me when I was busy with exams or deadlines, or helping with job applications – I continually have had a wonderful partner with whom to share the ups and downs and day-to-day struggles of a graduate *career.* After many years together, life still gets better all the time. I am very lucky to have such a wife.

Curriculum Vitæ

Ron A. Dolin

Born July 1, 1961 in Cleveland, Ohio

Education

University of California, Santa Barbara — Santa Barbara, CA
June, 1998

Ph.D. in Computer Science: "Pharos: A scalable distributed architecture for locating heterogeneous information sources".

University of California, Berkeley — Berkeley, CA
June, 1986

B.A. in Physics and Pure Mathematics

Research Experience

UCSB, Computer Science Department — Santa Barbara, CA
1995 – Present

Alexandria Digital Library Project: work included assisting in the design and implementation of a large digital library focusing on geo-referenced data. Papers and demonstrations of some of this work may be found on the Web at 'http://www.alexandria.ucsb.edu/'. Independent work included the development of *Pharos*, a scalable distributed architecture for locating heterogeneous information sources. This work included theoretical results on network architectures for resource discovery and a running demonstration of scalable automated Library of Congress classification of Internet newsgroups. Papers and demonstrations of some of this work may be found on the Web at 'http://pharos-.alexandria.ucsb.edu/'.

1993 – 1995

Assisted in the development of *GILED*, a syntax-directed editor for *Graphical Interval Logic (GIL)*; system was built in LISP using GARNET. Developed a generalized specification language and accompanying parsing algorithm for visual languages based on attribute grammars and efficient token permutations.

Professional Experience

UCSB, Computer Science Department — Santa Barbara, CA
1992 – Present

Research Assistant (described above). Teaching Assistant for Networking; Data Structures and Algorithms; and Programming Methods.

Internet Café — Santa Barbara, CA
1994 – Present

Co-founder and President: ISP; WWW and Internet consulting (http://www-.internet-cafe.com/).

Bell Communications Research (Bellcore) — Morristown, NJ
Summer 1994

Intern for the Computer Graphics and Interactive Media Group. Assisted in the design and development of *ShowBiz*, a dataflow visual language with syntax-directed editor for building workflows, built with LISP and CLOS.

Technical Research Associates, Inc. — Camarillo, CA
1994 – Present

Consultant: taught university-level 40-hour seminars on UNIX for Programmers.

UCSB, Physics Department — Santa Barbara, CA
1990 – 1992

System Manager, System Programmer for High-Energy Physics Group.

European Organization for Nuclear Research (CERN)
Geneva, Switzerland
1986 – 1990

System Manager, System Programmer
L3 Experiment (1987 – 1990), Aleph Experiment (1986-1987): These experiments involved hundreds of physicists from over 30 countries building detectors costing several hundred million dollars. Designed, constructed, and maintained a multi-building data acquisition LAN including mainframes, workstations, PC's, electronics (VME, Fastbus, CAMAC, etc.), and so forth. Developed device drivers; wrote programs for security, system, and environmental monitoring; trained new system managers; acted as liaison between L3 and CERN's online computing groups; and oversaw general online computer management,

assistance, acquisitions, maintenance, and operational activities. Responsible for an annual equipment budget of approximately $100K.

UC Berkeley, Geology Department — Berkeley, CA
1985 – 1986
System Manager, System Programmer for the Theoretical Geochemistry Group. Work included a student project to model fluid flow through porous media.

UC Berkeley, Space Sciences Laboratory — Berkeley, CA
1984 – 1985
Programmer for the Infrared Spatial Interferometer Project. Work included converting a graphics package from VAX/VMS Fortran to Unix. Also built a data simulator for a clock/motor interface.

Jet Propulsion Laboratory — Pasadena, CA
1981 – 1983
Programmer: Tracking Systems and Applications Section and Space Physics Section. Work included the design and implementation of a program to calculate angular drifts between baselines derived from the analysis of Very Long Baseline Interferometry (VLBI) data.

California State University, Northridge — Northridge, CA
1981 – 1982
Intern in the Physics Department: built a device driver and graphics package on an Apple 2E for a flatbed plotter.

Computer Science Publications

1. D. Andresen, L. Carver, R. Dolin, C. Fischer, J. Frew, M. Goodchild, O. Ibarra, R. Kothuri, M. Larsgaard, B. Manjunath, D. Nebert, J. Simpson, T. Smith, T. Yang, and Q. Zheng, “The WWW prototype of the Alexandria Digital Library”, in Proceedings of ISDL’95: International Symposium on Digital Libraries, Japan, 22- 25 August 1995.
2. Smith, T.R., Andresen, D., Carver, L., Dolin R., Fischer, C., Frew, J., Goodchild, M., Ibarra, O., Kemp, R.B., Kothuri, R., Larsgaard, M., Manjunath, B.S., Nebert, D., Simpson, J., Wells, A., Yang, T., Zheng, Q. “The Alexandria Digital Library: Overview and WWW Prototype”, IEEE Computer, 54-60, May, 1996 (author errata in July, 1996).

3. Cheng, X., Dolin, R., Kothuri, R., Neary, M., Prabhakar, S., Wu, D. Agrawal, D., El Abbadi, A., Freeston, M., Singh, A., Smith, T., Su, J., "Scalable Access Within the Context of Digital Libraries", in Advanced Digital Libraries Forum, 97. Proceedings of the IEEE forum on Research and Technology Advances in Digital Libraries–ADL '97, Washington, DC, May 1997, pp. 70-81.
4. Dolin, R., Agrawal, D., El Abbadi, A., "Classifying Network Architectures for Locating Information Sources", Proceedings of the Fifth DASFAA Conference, Melbourne, Australia, April, 1997, 31-40.
5. Hill, L.L., Dolin, R., Frew, J., Kemp, R.B., Larsgaard, M., Montello, D.R., Rae, M-A, Simpson, J, "User Evaluation: Summary of the Methodologies and Results for the Alexandria Digital Library, University of California at Santa Barbara", Proceedings of the American Society for Information Science (ASIS) Annual Meeting, Washington D.C., November 1997.
6. Dolin, R., Agrawal, D., El Abbadi, A., Dillon, L., "Pharos: A Scalable Distributed Architecture for Locating Heterogeneous Information Sources", Proceedings of the Sixth CIKM Conference, Las Vegas, Nevada, November, 1997, 348-355.
7. Cheng, X., Dolin, R., Kothuri, R., Neary, M., Prabhakar, S., Wu, D. Agrawal, D., El Abbadi, A., Freeston, M., Singh, A., Smith, T., Su, J., "Scalable Access Within the Context of Digital Libraries", in International Journal of Digital Libraries, 1997.
8. Dolin, R., Agrawal, D., El Abbadi, A., Pearlman, J., "Using automated classification for summarizing and selecting heterogeneous information sources", D-Lib Magazine, January, 1998; http://www.dlib.org/.

Physics Publications[1]

1. Bales, B.L., Dolin, R.A., Schwartz, R.N., "An effective proton hyperfine tensor for di-tertbutylnitroxide", in Liquid Crystals and Ordered Fluids. Proceedings of the American Chemical Society Symposium. Edited by: Griffin, A.C., Johnson, J.F. New York, NY, USA: Plenum, 1984. p. 579-96 of xiii+1157 pp. 25 references.

[1]Work for these papers involved computing, not physics. L3 Collaboration papers have several hundred authors.

2. L3 Collab., B. Adeva et al., Nucl. Instr. and Methods A 289 (1990) 35, L3 preprint 000, 1989, “The Construction of the L3 Experiment”.
3. L3 Collab., B. Adeva et al., Phys. Lett. B 231 (1989) 509, L3 preprint 001, October 11, 1989, “A Determination of the Properties of the Neutral Intermediate Vector Boson Z^0”.
4. L3 Collab., B. Adeva et al., Phys. Lett. B 233 (1989) 530, L3 preprint 002, October 15, 1989, “Mass Limits for Scalar Muons, Scalar Electrons, and Winos from e^+e^- Collisions near $\sqrt{s} = 91$ GeV”.
5. L3 Collab., B. Adeva et al., Phys. Lett. B 236 (1990) 109, L3 preprint 003, November 24, 1989, “Measurements of g_A and g_V, the Neutral Current Coupling Constants to Leptons”.
6. L3 Collab., B. Adeva et al., Phys. Lett. B 237 (1990) 136, L3 preprint 004, December 24, 1989, “Measurement of Z^0 Decays to Hadrons, and a Precise Determination of the Number of Neutrino Species”.
7. L3 Collab., B. Adeva et al., Phys. Lett. B 238 (1990) 122, L3 preprint 005, February 5, 1990, “A Measurement of the Z^0 Leptonic Partial Widths and the Vector and Axial Vector Coupling Constants”.
8. L3 Collab., B. Adeva et al., Phys. Lett. B 241 (1990) 416, L3 preprint 006, February 20, 1990, “Measurement of $Z^0 \to b\bar{b}$ Decay Properties”.
9. L3 Collab., B. Adeva et al., Phys. Lett. B 247 (1990) 177, L3 preprint 007, June 20, 1990, “Mass Limits for Excited Electrons and Muons from Z Decay”.
10. L3 Collab., B. Adeva et al., Phys. Lett. B 247 (1990) 473, L3 preprint 008, June 21, 1990, “A Determination of Electroweak Parameters from $Z^0 \to \mu^+\mu^-\gamma$”.
11. L3 Collab., B. Adeva et al., Phys. Lett. B 249 (1990) 341, L3 preprint 009, July 14, 1990, “A Precision Measurement of the Number of Neutrino Species”.
12. L3 Collab., B. Adeva et al., Phys. Lett. B 248 (1990) 203, L3 preprint 010, June 23, 1990, “Search for Neutral Higgs Boson in Z^0 Decay”.
13. L3 Collab., B. Adeva et al., Phys. Lett. B 248 (1990) 464, L3 preprint 011, July 7, 1990, “Determination of α_s from Jet Multiplicities Measured on the Z^0 Resonance”.

14. L3 Collab., B. Adeva et al., Phys. Lett. B 248 (1990) 227, L3 preprint 012, July 14, 1990, “A Test of QCD based on 4-Jet Events from Z^0 Decays”.
15. L3 Collab., B. Adeva et al., Phys. Lett. B 250 (1990) 199, L3 preprint 013, August 3, 1990, “Test of QED in $e^+e^- \to \gamma\gamma$ at LEP”.
16. L3 Collab., B. Adeva et al., Phys. Lett. B 250 (1990) 205, L3 preprint 014, August 6, 1990, “Search for Excited τ’s from Z^0 Decays”.
17. L3 Collab., B. Adeva et al., Phys. Lett. B 251 (1990) 311, L3 preprint 015, August 25, 1990, “Search for the Neutral Higgs Bosons of the Minimal Supersymmetric Standard Model from Z^0 Decays”.
18. L3 Collab., B. Adeva et al., Phys. Lett. B 251 (1990) 321, L3 preprint 016, August 24, 1990, “A Search for Heavy Charged and Neutral Leptons from Z^0 Decays”.
19. L3 Collab., B. Adeva et al., Phys. Lett. B 250 (1990) 183, L3 preprint 017, August 20, 1990, “A Determination of Electroweak Parameters from Z^0 Decays into Charged Leptons”.
20. L3 Collab., B. Adeva et al., Phys. Lett. B 252 (1990) 511, L3 preprint 018, September 14, 1990, “Search for the Charged Higgs Boson in Z^0 Decay”.
21. L3 Collab., B. Adeva et al., Phys. Lett. B 252 (1990) 518, L3 preprint 019, September 18, 1990, “Search for a Low Mass Neutral Higgs Boson in Z^0 Decay”.
22. L3 Collab., B. Adeva et al., Phys. Lett. B 252 (1990) 703, L3 preprint 020, November 2, 1990, “A Measurement of $B^0\overline{B^0}$ Mixing in Z^0 Decays”.
23. L3 Collab., B. Adeva et al., Phys. Lett. B 252 (1990) 525, L3 preprint 021, October 2, 1990, “Search for Excited Neutrinos from Z^0 Decays”.
24. L3 Collab., B. Adeva et al., Phys. Lett. B 252 (1990) 713, L3 preprint 022, November 5, 1990, “A Measurement of the $Z^0 \to b\bar{b}$ Forward-Backward Asymmetry”.
25. L3 Collab., B. Adeva et al., Phys. Lett. B 257 (1991) 469, L3 preprint 023, December 6, 1990, “Determination of α_s from Energy-Energy Correlations Measured on the Z^0 Resonance”.
26. L3 Collab., B. Adeva et al., Phys. Lett. B 257 (1991) 450, L3 preprint 024, December 17, 1990, “Search for the Neutral Higgs Boson”.

Abstract

Pharos: A Scalable Distributed Architecture for Locating Heterogeneous Information Sources

by

Ron A. Dolin

Information retrieval over the Internet increasingly requires the filtering of thousands of information sources. As the number of sources increases, new ways of automatically summarizing, discovering, and selecting sources relevant to a user's query are needed. We introduce Pharos, a highly scalable distributed architecture for locating heterogeneous information sources. Its design is hierarchical, thus allowing it to scale well as the number of information sources increases. We demonstrate the feasibility of the Pharos architecture using 2500 USENET newsgroups as separate collections. Each newsgroup is summarized via automated Library of Congress classification. We show that using Pharos as an intermediate retrieval mechanism provides acceptable accuracy of source selection compared to selecting sources using complete classification information, while maintaining good scalability.

Contents

List of Tables

List of Figures

Preface

research: careful, systematic, patient study and investigation in some field of knowledge, undertaken to discover or establish facts or principles.

technology: 1. the science or study of the practical or industrial arts, applied sciences, etc. 2. applied science. 3. a method, process, etc. for handling a specific technical problem. 4. the system by which a society provides its members with those things needed or desired.

progress: 1. a moving forward or onward. 2. forward course; development. 3. advance toward perfection or to a higher or better state; improvement.

> "The little farmers watched debt creep up on them like the tide. They sprayed the trees and sold no crop, they pruned and grafted and could not pick the crop. And the men of knowledge have worked, have considered, and the fruit is rotting on the ground, and the decaying mash in the wine vat is poisoning the air. And taste the wine – no grape flavor at all, just sulphur and tannic acid and alcohol.
>
> "This little orchard will be a part of a great holding next year, for the debt will have choked the owner.
>
> "This vineyard will belong to the bank. Only the great owners can survive, for they own the canneries, too. And four pears peeled and cut in half, cooked and canned, still cost fifteen cents. And the

canned pears do not spoil. They will last for years.

"The decay spreads over the State, and the sweet smell is a great sorrow on the land. Men who can graft the trees and make the seed fertile and big can find no way to let the hungry people eat their produce. Men who have created new fruits in the world cannot create a system whereby their fruits may be eaten. And the failure hangs over the State like a great sorrow."

John Steinbeck, The Grapes of Wrath

"...The last clear definite function of man – muscles aching to work, minds aching to create beyond the single need – this is man. To build a wall, to build a house, a dam, and in the wall and house and dam to put something of the wall, the house, the dam; to take hard muscles from the lifting, to take the clear lines and form from conceiving. For man, unlike any other thing organic or inorganic in the universe, grows beyond his work, walks up the stairs of his concepts, emerges ahead of his accomplishments. This you may say of man – when theories change and crash, when schools, philosophies, when narrow dark alleys of thought, national, religious, economic, grow and disintegrate, man reaches, stumbles forward, painfully, mistakenly sometimes. Having stepped forward, he may slip back, but only half a step, never the full step back. This you may say and know it and know it. This you may know when the bombs plummet out of the black planes on the market place, when prisoners are stuck like pigs, when the crushed bodies drain filthily in the dust. You may know it in this way. If the step were not being taken, if the stumbling-forward ache were not alive, the bombs would not fall, the throats would not be cut. Fear the time when the bombs stop falling while the bombers live – for every bomb is proof that the spirit has not died. And fear the time when the strikes stop while the great owners live – for every little beaten strike is proof that the step is being taken. And this you can know – fear the time when Manself will not suffer and die for a concept, for this one quality is the foundation of Manself, and this one quality is man, distinctive in the universe."

John Steinbeck, The Grapes of Wrath

"It is a grave duty which I now face. In preparing for it, I have tried to enquire: what great principle or ideal is it that has kept this Union so long together? And I believe that it was not the mere matter of separation of the colonies from the motherland, but that sentiment in the Declaration of Independence which gave liberty to the people of this country and hope to all the world. This sentiment was the fulfillment of an ancient dream, which men have held through all time, that they might one day shake off their chains and find freedom in the brotherhood of life. We gained democracy, and now there is a question of whether it is fit to survive.

"Perhaps we have come to the dreadful day of awakening, and the dream is ended. If so, I am afraid it must be ended forever. I cannot believe that ever again will men have the opportunity we have had. Perhaps we should admit that, and concede that our ideals of liberty and equality are decadent and doomed. I have heard of an eastern monarch who once charged his wise men to invent him a sentence which would be true and appropriate in all times and situations. They presented him the words, 'And this too shall pass away.'

"That is a comforting thought in time of affliction – 'And this too shall pass away.' And yet – let us believe that it is not true! Let us live to prove that we can cultivate the natural world that is about us, and the intellectual and moral world that is within us, so that we may secure an individual, social and political prosperity, whose course shall be forward, and which, while the earth endures, shall not pass away...."

Abraham Lincoln, Farewell Address, 1861, before assuming the Presidency and the beginning of the U.S. Civil War (as told by Kurt Vonnegut in "Timequake")

Chapter 1

Introduction

Information retrieval over the Internet increasingly requires the filtering of thousands of heterogeneous information sources. Important sources of information include not only traditional databases with structured data and queries, but also increasing numbers of non-traditional, semi- or un-structured collections such as Web sites, FTP archives, and newsgroups. As the number and variety of information sources increase, new ways of summarizing, discovering, and selecting collections relevant to a user's query are needed. One such method involves the use of classification schemes, such as the Library of Congress Classification (LCC) [Lib86], with which a collection may be represented based on aspects of its content, irrespective of the structure of the actual data or documents. For such a system to be useful in a large-scale distributed environment, it must be easy to use for both collection managers and users. For collection managers, it must be possible to classify collections automatically within a classification scheme. Furthermore, there must be a straightforward and intuitive interface with which the user may use the scheme to assist in information retrieval (IR). Finally, once the collections are summarized, this information must be distributed across the network within a clearly defined architecture.

The Alexandria Digital Library (ADL) Project [ACD$^+$95] focuses on geo-referenced information, whether text, maps, aerial photographs, or satellite images. As a result, we are interested in techniques that work with both text and non-text, such as combined textual and graphical queries, multi-dimensional indexing, and IR methods that are not solely dependent on words or phrases. Part of this work involves locating relevant online sources of information. In particular, we have designed and tested aspects of a distributed architecture, Pharos, which we believe will scale up to $\sim 10^6$ heterogeneous sources [DAE97, DAED97, DAEP98]. Pharos accommodates heterogeneity in content and format, both across multiple sources as well as within a single source. That is, we consider sources to include Web sites, FTP archives, newsgroups, and full digital libraries; all of these systems can include a wide variety of content and multimedia data formats.

This dissertation focuses on the selection of *collections* of documents rather than on particular documents themselves. Thus, we are not directly addressing the problem of a user finding a particular document via author, title, keyword, etc. However, a particular author or title is generally included in one or more information domains, such as subject area, geographical region, time period, image feature, type of business, etc. By focusing on finding collections which are characterized by such domains, we believe that users are likely to find those authors and titles which are relevant to the concepts underlying specialized queries. The reason for such a change in focus is that we believe that locating collections by domains is a more scalable methodology than locating particular authors, titles, etc.

Pharos is based on the use of hierarchical classification schemes. These include not only well-known 'subject' (or 'concept') based schemes such as the Dewey Decimal Classification and the LCC, but also, for example, geographic classifications, which might be constructed as layers of smaller and smaller hierarchical longitude/latitude boxes. Pharos is designed to work with sophisticated queries that utilize subjects, geographical locations, temporal specifications, and other information domains. The Pharos architecture requires that hierarchically structured *collection* metadata be extracted so that it can be partitioned in such a way as to greatly enhance scalability. Automated classification is important to Pharos because it allows information sources to extract automatically the requisite collection metadata that must be distributed.

1.1 Significance of the Research

This research focuses on two main themes: scalability and flexibility.

1.1.1 Scalability

We define *scalability* as the ability (of a system) to accommodate growth while maintaining acceptable performance (up to a sufficiently large environment). In the areas of distributed information systems and resource discovery, scalability refers both to source parameters as well as to user parameters. Thus, we require that Pharos is able to accommodate an increasing number of information sources, increasing sizes of collections, etc. In addition, we require Pharos to be able to handle an increasing number of users. Our research has been oriented toward specific scalability goals, such as being able to handle 10^6 sources and 10^8 users – in other words, the Internet. We have also attempted to estimate rigorously the necessary resource utilization.

In many IR systems involving multiple collections, it is straightforward to allow for growth by allowing retrieval accuracy to degrade (e.g. [MZ94]). However, scalability requires acceptable performance, including acceptable query results. This research is important in that a high level of scalability is achieved while maintaining good retrieval accuracy. In order to accomplish this, we have completely specified the network architecture of Pharos, including the metadata structure and placement, and the mechanisms for metadata propagation and retrieval. Pharos achieves scalability by using a hierarchical metadata structure and a highly decentralized metadata distribution, storage, and retrieval mechanism.

1.1.2 Flexibility

In one sense, flexibility can be viewed as another type of scalability – namely, in accommodating the growing diversity of data and the growing uses and complexity of information systems. Pharos has been designed around a very general framework that can work with data that is heterogeneous in both content and format, including text, maps, images, etc. User queries in Pharos are expressible as combinations of different information domains, such as keyword/subject, geographical region, and time period. In addition, Pharos is designed to extend user source selection beyond linear ranking, and to give control of the selection mechanism to the user, rather than some intermediate network server. This allows the user to include not only factors such as number of relevant documents for a given collection, but also non-content factors such as network parameters, use charges, etc. These values are made available to the user interface, which can then make them available to the user in customizable ways to aid in source selection. Finally, Pharos provides a significant level of autonomy to the information sources. It allows them a fairly general framework within which to describe their collections and also permits them to determine their own metadata update frequency. Pharos is intended to work with heterogeneity not only *across* collections, but also *within* collections. This diversity includes both data format and data content.

1.2 Research Objectives

The main goals of this research, as previously discussed, are 1) to develop a distributed architecture that scales well, and 2) to enhance the selection of information sources and query flexibility. The architecture must be able to work with non-text entities such as images, while still being applicable to text entities. In particular, besides content keywords and/or concepts, queries should be able to incorporate temporal and geographical specifications.

As an outgrowth of the need to evaluate the effectiveness of attaining the primary objectives, a secondary goal of the research is to develop evaluation metrics to measure the accuracy of the selection of information sources in the context of high scalability. Finally, another objective is to encourage and enhance the relationship between library science and computer science in the area of digital library research.

1.3 Assumptions

This design is based on several assumptions. First, we assume that filtering sources iteratively with finer and finer grain metadata will yield a sufficient number of sources with relevant content. Second, we assume that the classification system used to summarize collections is reasonably accurate. Third, we assume that querying sources that have a large *total number* of 'relevant' documents, a large *percentage* of 'relevant' documents, or some combination of the two, are likely to have documents that contain information that satisfy a user's query. Fourth, we assume that sending a query to a small number of good sources is sufficient to acquire desired information. Of these assumptions, we attempt to verify the first two. We do not address the second two in this dissertation.

1.4 Outline

In the next chapter, we discuss background and related work. Then, in Chapter 3, we motivate the problem of large-scale information discovery and retrieval, and present an overview of the methodology used by Pharos. Next, we describe the Pharos architecture in detail in Chapter 4. One long-range goal of this work is the construction of a distributed prototype system. However, before beginning a prototype, we first study the feasibility of our architecture. This study has two main components. The first is to compare the scalability of Pharos with that of some other existing or proposed architectures; this is described in Chapter 5. Chapter 6 then describes the second feasibility study: to analyze the expected accuracy of Pharos query results using simulated queries and information sources. Given the results of this work, we describe the beginning of a prototype in Chapter 7, including an evaluation of its query performance and scalability. Finally, after summarizing the work in Chapter 8, we discuss future directions in Chapter 9.

Chapter 2

Background and Related Work

There are several components to this research in both the design of the architecture and the development of the prototype. There has been work done in many areas that relate to one or more of these components. In particular, the following general areas are relevant to this research: classification, information retrieval, search engines, and resource discovery and digital libraries.

2.1 Classification

Generally speaking, a *classification* is an arrangement according to some systematic division into distinct categories because of certain likenesses or common traits. Pharos is based on the use of hierarchical classification schemes to summarize collections. Moreover, Pharos uses such schemes for many types of information domains (concepts, geographical regions, images, etc.). While many well-known classification schemes exist, not all of them are hierarchical. For example, wines are usually classified along several orthogonal components such as type (table, sparkling, and fortified), color (white, red, rosé), character (sweet, dry), region of origin (Bordeaux, Chianti, Rioja, etc.), and/or type of grape (Chardonnay, Pinot Noir, etc.). For images (e.g. artwork, photographs, satellite remote sensing, etc.), there are also several classification schemes. These schemes are based on anything from art historical perspectives to subject matter [Jör96], as well as mathematical processes such as wavelet analysis ("texture") and color composition [MM98].

A good example of hierarchical classification is in biology, described in Microsoft's Encarta Encyclopedia 98 [Mic97, article entitled "Classification"] as follows:

> Classification, in biology, [is] the identification, naming, and grouping of organisms into a formal system. The vast numbers of living forms are named and arranged in an orderly manner so that biologists all over the world can be sure they know the exact organism that is being examined and discussed. Groups of organisms must be defined by the selection of important characteristics, or shared traits, that make the members of each group similar to one another and unlike members of other groups. Modern classification schemes also attempt to place groups into categories that will reflect an understanding of the evolutionary processes underlying the similarities and differences among organisms. Such categories form a kind of pyramid, or hierarchy, in which the different levels should represent the different degrees of evolutionary relationship. The hierarchy extends upward from several million species, each made up of individual organisms that are closely related, to a few kingdoms, each containing large assemblages of organisms, many of which are only

distantly related.

Besides its epistemological use as a description of knowledge, classification has also been used for the purpose of organizing and retrieving documents. Francis Miksa [Mik98, p. 33] states the following:

> The classification of information-bearing entities is as old as libraries themselves. Evidence of attempts to group such artifacts can be found among collections of clay tablets in ancient times. These older efforts to classify information-bearing entities were characterized by their relative simplicity, where categories commonly reflected practical storage expediencies such as the size of the items or contemporaneous educational curricula. It was not until the post-Renaissance modern period, and especially the late nineteenth century, that library classification achieved anything of the complexity that is now associated with it, especially in theory and techniques.

Of particular relevance to our own research is the potential use of classification schemes, including traditional library classification, for the specific purpose of electronic document retrieval. Library classification has typically been associated simply with placing and finding a document on a shelf. In discussing this issue, Jolande Goldberg of the Library of Congress concludes with this [Gol96, p. 41]:

> Despite such statements made by Librarians of Congress (in particular, Herbert Putnam, at the inception of the LCC, and Luther Evans, during the planning state of the Law Classification) that the LCC is not intended to serve as a general classification for American libraries, the LCC, in fact, has become the preferred classification for libraries and educational institutions in the United States and for many institutions abroad. Their voiced concerns for stability of

> shelf arrangements by classification has, in the past, markedly inhibited the inventive remodelling of the LCC. Ultimately, all efforts will have to concentrate on the envisioned function of the electronic LCC as an online retrieval tool. For online browsing and navigation of electronically stored information, including the segregation of whole portions of one class and transfer to another, a knowledge-based, field-specific structure of the classification is of utmost importance. So, also, is the separation from the shelving function.

We make a distinction between traditional classification (partitioning into unique categories) and our approach (fractional allocation among shared categories). We are free from atomic allocation because we are not 'putting a book on a shelf'. We use what we might call probabilistic fractional classification. It is probabilistic because we are not certain to what degree an object belongs in a particular category, but rather make estimates based on an automated process. We require automated classification due to the scale of the problem we are addressing – namely, documents and collections on the Internet. By fractional classification, we mean that, rather than considering the objects to be classified as discrete entities, we assign fractions of an object to different categories. We will describe this methodology in detail in Chapter 7.

2.2 Information Retrieval

Information retrieval (IR) is concerned with the systems, evaluation methodology, and user models involved with a person's locating needed information, usually in an electronic environment. Standard IR techniques include document indexing and summarization, similarity determination between a query and a document, and query expansion and relevance feedback [Kan94]. A common problem is that of retrieving a set of documents from a given collection that are 'relevant' to a user's query. Although problematic, it is often assumed that such relevant documents are identifiable. In reality, the operational definition of relevancy is some form of mathematical similarity function applied between a query and the documents in the collection. The query is considered to be a specification against which documents in a collection are compared. In general, there is not a tight semantic relationship between the automated comparison function and what the user might consider to be document relevancy.

There are, in general, three broad views of relevancy [Sch94]: 1) the system view, 2) the information view, and 3) the situation view. The first view follows along the line of a mathematical similarity function and is independent of any user's perspective. The second view is based on human judgment between a particular query and document. The third view, simultaneously the most useful and most subjective, focuses on the degree to which a query result addresses the overall information needs of a user in a particular instance.

The concept of relevancy as one component of a searcher's overall information behavior is discussed in [Sch94], while [EY72] presents information systems as one component of general decision making. Relevancy is not in general the only criteria by which users might want to retrieve documents. Other factors include, for example, document length, contextual information such as author or publication date, and so on. In particular, a decision to stop searching is more likely determined when the results are *satisficing*, that is, good enough [Sim82]. This criteria often relates more to the time and effort required for further searching than to any objective measure of the result set.

Despite its limitations, measurement of the relevancy of retrieval documents to a given query is used to evaluate the performance of an IR system. Two common benchmarks are *precision* and *recall* [Sal89]. Precision is the percentage of returned documents that are relevant to the query. Recall is the percentage of total relevant documents in the collection that are returned. For example, suppose there are T documents in a collection relevant to a user's query. M documents are returned to the user, R of which are relevant to the query. Then precision is defined as R/M, and recall is defined as R/T. In general, if we attempt to increase recall (i.e. return a larger percentage of relevant documents with respect to the entire collection), we do so at the cost of decreasing precision (i.e. we simultaneously return a larger percentage of non-relevant documents with respect to the retrieval set). This is due to the fact that, in general, in order to retrieve more relevant documents, we have to loosen the document selection criteria, which allows more non-relevant documents to be retrieved. Similarly, if we attempt to increase precision, we do so by tightening the document specification. This generally has the undesirable effect of filtering out relevant documents, and thus the recall drops. Typically, IR systems return approximately 50% precision for a fixed 50% recall [Har95]. That is, by adjusting the document specification until half of the relevant documents in a collection are returned, on average about half of the total retrieved documents are relevant to the query. There has been much work done in the area of automatic text retrieval [Sal89], for example SMART [SM83], Latent Semantic Indexing (LSI) [BDO95], and others involved with the Text REtrieval Conferences (TREC)

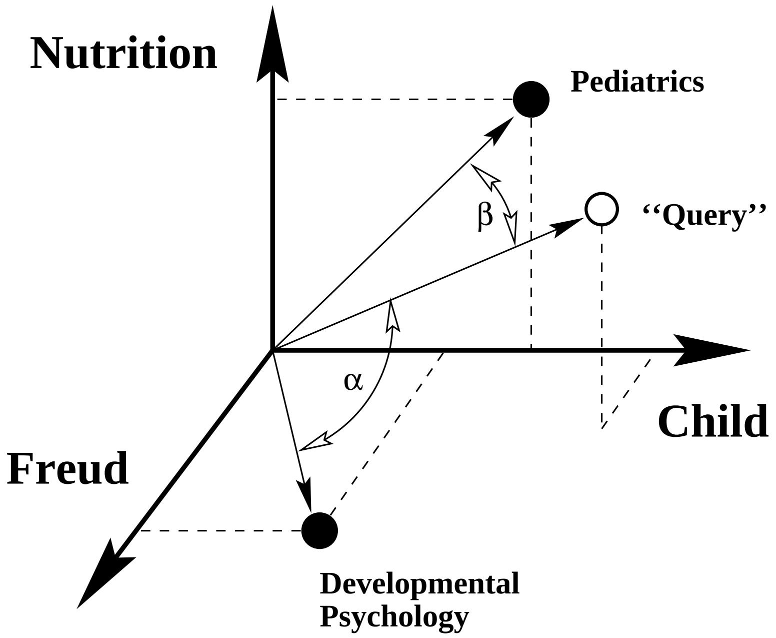


Figure 2.1: Vector Space Model of Information Retrieval

[Har95] run by the National Institute of Standards and Technology (NIST).[1]

One common IR method, used in our experiments, employs what is called the *vector space model* [Sal89]. This model constructs a high-dimensional vector space such that each term occurring in any document of a collection is represented by a different dimension. Each document is assigned a point, or vector, in this space. The component of a document's vector is determined, for example, by the number of times that the term for that dimension is used in the document.

[1]http://www.nist.gov/ and http://trec.nist.gov/

Figure 2.1 shows a simplified instance of the vector space model with only three dimensions: ‘Child’, ‘Freud’, and ‘Nutrition’. A document in the collection discussing Developmental Psychology, containing the words ‘Freud’ and ‘child’, is shown lying on the Freud/child plane. Another document, discussing Pediatrics and containing the words ‘child’ and ‘nutrition’, is shown lying on the child/nutrition plane. Suppose that we want to find documents similar to a particular ‘query’ document, and that this document contains many uses of the words ‘nutrition’ and ‘child’, and only a very few instances of ‘Freud’. We place the query document in the vector space in the same fashion as the documents in the collection. We then take the cosine of the angle between the query document and all the other documents in the collection. In this case, we see that the angle β between the query and Pediatrics is less than the angle α between the query and Psychology. As a result, the cosine of β is larger than that of α, and we conclude that Pediatrics is more similar to the query document than Developmental Psychology. The result is that we give documents a higher rank depending on their angular proximity to the query in the vector space. This simplistic version of the vector space model can be generalized using methods such as weighting terms based on their frequency within the entire collection, taking into account document length, and even using Singular Value Decomposition (SVD) to reduce the dimensionality of the space [BDO95].

2.2.1 Networked Information Retrieval

Much work has been done in the area of IR using multiple collections, including, for example, [ACM96, CLC95, CH95, FY95, VF95]. The predominant feature of the majority of this work is that the techniques used are generally limited to text documents, for example distributing inverted term lists. Another aspect of these systems is that it is unclear to what degree they are scalable in terms of the number of sources, the number of users, and the number of documents. None of them experiment with more than a few hundred collections. Finally, although many systems provide for the transfer of collection metadata, they do not specify exactly the content and/or format of the metadata, where specific pieces of metadata are to be placed within the network architecture, or the exact metadata transmission methodology involved in the handling of queries and updates. These factors are all important in the design of an Internet-scale system.

2.2.2 Applicability of Evaluation Measures

The question arises as to how appropriate are evaluation measures such as precision and recall with respect to the documents and knowledge structures found within the Internet. Thomas Walker [Wal96, p. 324] states the following:

> Are measurements of recall and precision feasible here? Recall, a ratio used to describe the ability of a system to retrieve a percent of relevant documents from all relevant documents in a system, is not useful here because the total number of relevant documents is

not knowable. Documents are very fluid and changing. Being part of such an unsystematic system, the presence and quality [of] many sources depend on the sustained interest of an individual or organization. For instance, in a reference class, I had students use the Web to locate reproductions of "The Scream" (or "The Cry") by Edvard Munch. One particular site was especially fruitful: a person, obviously fascinated by the different forms of this image, had created a "Scream" site, which provided links to many digital versions of the work. This is the work of a volunteer who may be excited about Munch this moment, but who may not even have a Net account tomorrow or in ten years. There are thousands of such passionate collectors. Although students found some images easily, it is impossible to know how many they missed.

Precision, which describes the ability of a system not to retrieve irrelevant documents, may be a more satisfying measure. Many users have already waded through a considerable amount of Web "trash" and have already carried out informal measurements of precision. Another complication, commonly encountered in recall/precision measurements, is that it is difficult to define relevance because it has always been very personal. Measures of recall and precision depend on relevance, which can be so unpredictable, so subjective, that it is difficult or impossible to verify. For one individual, a document or information source may be "close enough" to a subject or "good enough" for a particular use, even though it is not the best or even close to the best. If a fee is required, as they are for some of the highest quality Net resources, will a given user be less likely to use it? If it is necessary to register, will a user be less likely to use a very good source, even if no fee is involved? Will a frustrated user switch topics rather than carry out an exhaustive Internet search? Will a novice user take the time to learn how the different search engines work? Several Web search engines allow for Boolean searching, but in at least one case, the default operator is "OR," which of course has the potential of delivering results that are hardly precise. These problems are addressed in part by variations of the Principle of Least Effort: a solution will be judged by a user to be satisfactory if it is easily found, even if it is not the best available solution, and perhaps even if it is not a solution at all.

2.3 Search Engines

Current WWW indexes, such as AltaVista[Dig97], Lycos [Lyc96], and Yahoo [Yah97], which are designed for locating information on the Internet, are limited in several ways. Notably, these systems do not scale to handle increasingly large numbers of search requests due to limited network bandwidth and server power. Furthermore, because they focus on keyword matching, they are usually unable to provide adequate simultaneous precision and recall. Existing WWW indexing systems utilize either small, hand-made, hierarchical lists, as in Yahoo, or word-matching on huge document-spaces, as in AltaVista. Standard techniques of automatic text retrieval, such as relevance feedback and local/global term weightings [Sal89], aid in matching query terms to relevant documents; however, these techniques are seldom used for locating sources. Another factor that limits most existing WWW indexes is their inability to share index information. This lack of sharing results in highly duplicated document fetching and indexing. As the number of indexing sites grows, determining which ones contain the most useful information for a given query becomes increasingly difficult. Furthermore, few WWW indexing systems give sources control over how often they are indexed, or provide for automatic updates when information becomes stale. Another problem with many WWW indexes is that they index the actual texts of documents. This approach is not appropriate for indexing catalogs, especially those which list documents which are not available online, and it does not extend to non-text documents, such as maps, images, and sound and video files. Information sources that make their documents available only as query results have no mechanism of including their contents within such indexing systems.

2.4 Resource Discovery and Digital Libraries

Problems of locating sources of information, one form of *resource discovery* [BDMS94], are of particular interest to the digital library community [FAFL95]. Such emerging systems must deal with storing, locating, cataloging, propagating, and retrieving large numbers of diverse documents in a distributed, heterogeneous environment. Several projects in the NSF/NASA/DARPA Digital Library Initiative Program [CAC95, Nat93] are therefore investigating these problems. The Alexandria Digital Library (ADL) Project [ACD$^+$95] is focusing on indexing spatial information, as well as on the storage, distribution, and retrieval of large (spatial) images. The Illinois Digital Library Project [Sch95, SMC$^+$96] is building a prototype of the *Interspace*, which presents the Internet as a single space of highly interlinked, distributed information. The University of Michigan Digital Library Project [Cru95, ABD$^+$96] is designing an agent-based system; user interface agents, mediation agents, and collection agents cooperate to allow concurrent searching of multiple collections. Within the Stanford Digital Library Project [Sta95, PCGM$^+$96], the *gGlOSS* resource discovery system [GGM95] represents sources by vectors of term frequencies. This work has been extended to the STARTS proposal [GCGMP96] in which sources extract their own metadata and pass it to intermediaries. Outside the digital library work, Content Routing [SDW$^+$94] is a system in which queries get routed to the available servers based on the expected relevance of the server to the query. Harvest [BDH$^+$94] automatically indexes documents within a source and distributes these indexes. Harvest will be discussed in more detail in Chapter 4.

Chapter 3

Motivation and Design Overview

3.1 Motivation

We consider a modern Internet environment with a large number of users and information sources, including WWW sites, FTP archives, digital libraries, file systems, etc. Attempting to locate a small set of sources that best fit user queries requires detailed knowledge of the holdings at each source. There are two extreme approaches for acquiring this metadata. The first, which we informally call the *remote approach*, is to query each source dynamically in turn (or possibly in parallel) without storing any local metadata. The second, which we informally call the *local approach*, is for each user to store locally detailed information about the holdings of each source and not to request anything remotely (until a final set of sources has been selected for direct access). Both of these extremes are impractical. The remote approach would take too long even for a single user query, let alone a large number of simultaneous ones. The local approach needs to have sufficient information for any given query; we believe that the amount of metadata required for accurate source comparison is too large to be practical.

Our approach is therefore a hybrid one in which we attempt to balance the location of the metadata's storage with network traffic by using an iterative, multi-level query method. A limited amount of metadata about each source is massively replicated among "high-level" servers. These servers are locally situated within an organization such as a university campus or a corporate network. They receive requests from local users about most or all known sources and reply with only enough metadata to allow for a rough comparison of sources. More detailed metadata is stored in specialized, sparsely replicated "mid-level" servers. These servers receive requests from remote users about a relatively small set of sources and supply greater detail than the high-level servers. All high-level and mid-level servers store information about each source. However, while all high-level servers store the same metadata, each mid-level server (up to replication) stores unique metadata within a sub-area of a single, pre-specified, domain-specific classification tree, or *taxonomy.* For example, we expect separate mid-level servers specializing in subject areas such as history, art, physics; in geographical regions such as North America, Africa, Asia; etc. We call any such sub-area of a particular information domain a *sub-domain.* This design solves the problem of the remote approach by providing a fast, scalable method of receiving any needed metadata. It also solves the storage problem of the local approach by distributing the metadata throughout the network and requiring only minimal storage within a user's local environment. Finally, network traffic is reduced in several ways. Smaller local storage reduces the amount of broadcast-type traffic generated by the sources for update purposes, while the mid-level servers reduce the amount of query traffic to the sources generated by users. Furthermore, having users communicate with local servers for initial source filtering also reduces network traffic.

3.2 Example Query: An Iterative Approach

As an example query, suppose that a student is studying the following topic: "A Comparison of Political Music in 1967: San Francisco, Shanghai, and Cairo." We performed a search through AltaVista, Lycos, Yahoo, and the University of California's Melvyl online general catalog (see Appendix A). AltaVista, Lycos, and Yahoo found little of value for this study. Furthermore, the precision was very low, on the order of 1%, requiring excessive viewing of many irrelevant documents in order to locate the rare relevant ones. Melvyl, on the other hand, returned 44 items of potential interest, such as "When the music's over: the story of political pop," "Female college students in China," "Rhythm and resistance: explorations in the political uses of popular music," "Qira'at munawi'ah" ("Opposition Readings"), and several recordings. How do we inform the student that, in this case, Melvyl is a potentially good information source?

In our proposed system, depicted in Figure 3.1, the student attempts to find sources as follows. Initially, at the client-side user interface (UI), she specifies the query in terms of relevant information taxonomies and significant sub-domains within them. The taxonomies involved in this search, for example, include subject, geographical region, and time period. Within each of these general taxonomies, the student is prompted to provide more specific sub-domains that match her query. For the subject domain, the UI attempts to locate the query terms or their synonyms within the taxonomy. For example, the phrase *political music* might be covered by political science, history, and music; sources that excel in these areas are potentially useful for finding information about her query. For the geography taxonomy, the student specifies a longitude/latitude bounding region, or a place-name which is checked against a *gazetteer*[1] to obtain the bounding coordinates.

Once the student has specified the appropriate sub-domains within the three taxonomies, the UI then queries the local *high-level* server and gives highest weights to those sources that contain relatively more information 1) within the subject taxonomy, in, say, politics, history, or music; 2) within the time period taxonomy, about the 1960's; and 3) within the geography taxonomy, in SW North America, SE Asia, or NE Africa. From the weightings, the student selects some initial set of sources, say the best thousand based on the high-level metadata. Next, the UI queries the relevant seven remote *mid-level* servers for more detailed metadata about the sources included in the initial set: a political subject server, a history subject server, a music subject server, a 1960-1969 time period server, and the three geographic region servers covering SW North America, SE Asia, and NE Africa. These mid-level servers return their

[1]A standard *gazetteer* is an index of place names; we are referring to a specific type containing a mapping from place names to geographical coordinates.

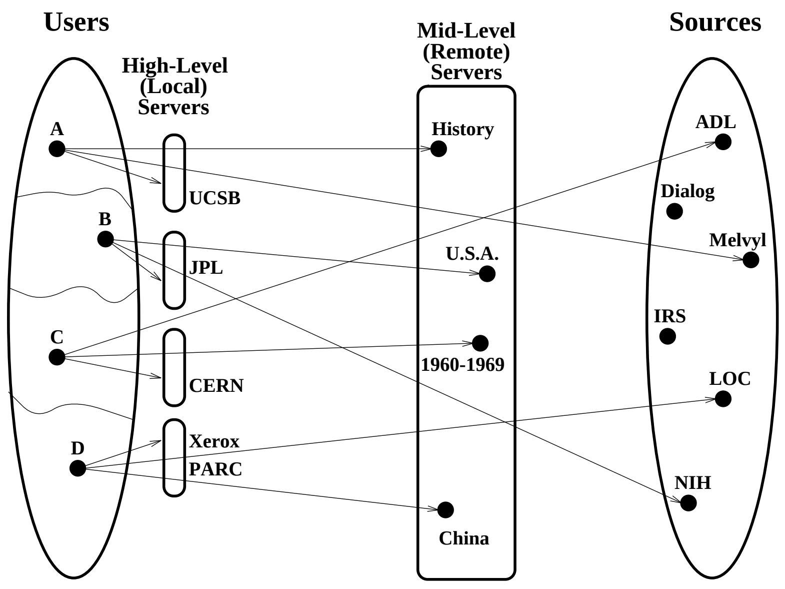


Figure 3.1: Retrieval Design Overview

detailed and specialized metadata about the requested sources. For example, the political subject server provides metadata about the percentage of holdings among the initial set of sources particular to political music; the 1960-1969 time period server provides metadata about the percentage of holdings covering 1967; and the SW North America geographic region server provides metadata about the percentage of holdings dealing with the California area. The UI then merges the information returned from the various mid-level servers and presents them to the student. This information can be used to select a small set of sources for further (direct) querying. As we discussed in the Introduction, Pharos is not designed to query information sources directly with individual user requests. Rather, it is designed to select a sufficiently small set of sources that can then be used by other systems for this purpose, utilizing, for example, traditional IR methods.

It is important to recognize why the user cannot simply query the mid-level servers initially, bypassing the high-level server completely. A mid-level server dealing with source information about, say, the subject of music, does not store any other information. It does not have information about, for example, sources that may have documents geographically related to San Francisco. A user might query the music mid-level server to find out about the best 100 sources for political music, and a “California” mid-level server to find out about the best 100 sources with content geographically related to San Francisco. The problem is that there would likely be little or no overlap between these sets. The user is trying to find sources that have documents related to both aspects of the query: the subject of political music and the geographical region near San Francisco. The only way to accomplish this is to first query the high-level server, which can return a list of sources that are known to contain both types of information. This list is then passed to the mid-level servers for more detailed information. Thus a final list of 100 can be selected that are hopefully among the best dealing with both aspects of the query.[2]

Given such a retrieval system, we need a corresponding metadata distribution system such that the high-level metadata is distributed widely and the mid-level metadata is distributed selectively. The wide distribution mechanism in Pharos is modeled after the distribution of USENET news [Hor83] via NNTP

[2]An alternative approach would be to have the mid-level servers communicate with each other. We have not explored this possibility due to the anticipated scalability costs it would incur.

[KL86]. Unlike the storage mechanism for high-level metadata, mid-level metadata from different taxonomies in Pharos are not all stored together. Each mid-level server stores all the metadata about each source that is related to a unique sub-domain of a particular information taxonomy. These servers are not generally resident within users' local areas, but are sparsely located at remote sites. When a source sends out its mid-level metadata, it sends the different components to the corresponding servers in a point-to-point manner. The distribution of the mid-level metadata in Pharos is modeled after the distribution of indexes in Harvest [BDH$^+$94], which supports efficient point-to-point metadata transfer, server replication, caching, and structured querying of servers. Metadata distribution will be discussed further in Chapter 4.

Chapter 4

Architecture

In this chapter, we describe the various components of the Pharos architecture. We first describe the structure of the metadata and the manner in which it is partitioned for distribution. We briefly introduce the process through which the metadata is extracted. This will be discussed in detail in Chapter 7. We then describe the User Interface (UI). The UI is designed to prompt initially for information about the user which is stored between sessions. Next, the UI must prompt for query information. During the query process, the UI will request source metadata from the various servers as needed. Finally, it needs to present the metadata in a concise and understandable manner during each phase of the search. Between the extraction and retrieval of the metadata, we have the massively replicated, localized high-level servers and the sparsely replicated, remote mid-level servers, each receiving and sending appropriate metadata with very different network characteristics, and hence different underlying network protocols.

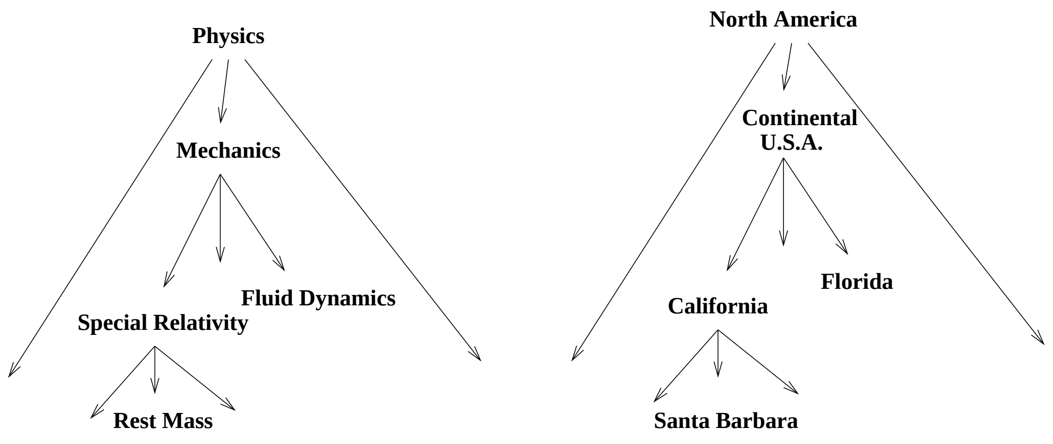
4.1 Metadata Structure

The metadata must be designed to support a multi-level information system and it should match the intended queries as well as the retrieval system. Therefore, metadata should be grouped according to its informational relationship and designed around a hierarchical structure, which lends itself to progressively refined queries. Pharos employs information classifications that fit common query methods, such as a subject-based hierarchy, a geographical hierarchy, and a temporal hierarchy. Once these hierarchies have been defined, we include the top portions in the high-level metadata, and lower portions in the mid-level metadata.

4.1.1 Information Hierarchies

An *information hierarchy* is a topic-based classification tree, as shown in Figure 4.1. In order to describe the collection at a source, we quantify the number of documents with content relevant to each part of an information hierarchy. As mentioned above, the initial design of Pharos employs three information hierarchies: subject, geographical, and temporal hierarchies.

The subject hierarchy is modeled after the Library of Congress's *LC Classification* (LCC) [Lib86], which contains a controlled, hierarchically structured



(a) Subject Domain (b) Geography Domain

Figure 4.1: Information Hierarchies

set of categories. This classification scheme has several advantages: it is fairly extensive; it is familiar to most of the library classification community; it is open-ended (in the sense that the structure allows for the inclusion of new topics); it is revised frequently; and it is used by most academic libraries in the U.S.A. [Gol96]. While there are problems with using a fixed list for query terms [Dum91], there are several problems with using vocabularies that are based on the terms found within each document collection. For example, it is difficult to compare sources that do not use the same terms, and we cannot expect different collections to yield the same vocabularies. Moreover, without a controlled set of terms, it is difficult to construct comparable hierarchical structures that can be used to build source metadata within a multi-level approach. For a given query, the relevant categories must be deduced. The Pharos UI must bridge the gap between the words used by the user and the controlled terms in the subject hierarchy. Techniques such as those described in [CL92, Dum91] address the problem of finding the relationships between query terms and potentially different document terms; these techniques also apply to matching query terms with subject categories. We therefore attempt to place users' concepts within appropriate sub-domains in the subject hierarchy, while at the same time controlling the categories so that it is easier to accurately compare sources. This component of the UI is included in the prototype discussed in Chapter 7.

A simple geography information hierarchy can be composed by tiling the Earth's surface into progressively smaller, hierarchical, longitude/latitude squares. If the highest level of the hierarchy represents the regions of the Earth's surface defined by $45^\circ \times 45^\circ$ grid lines, as in Figure 4.2, the result is 32 top-level tiles. In this figure, the actual data density of the $\sim$421,000 documents of the May, 1996 Alexandria Digital Library (ADL) holdings is shown: the lighter the area in a tile, the more documents it contains. The next level of the hierarchy is formed by sub-dividing each top-level tile by $5^\circ \times 5^\circ$ grid lines yielding 2,592 sub-tiles. Each top-level tile has 81 sub-tiles. This sub-tiling process is continued to create a spatial hierarchy of desired levels of detail.

The temporal domain allows queries to specify date ranges of content (e.g. history of the sixteenth century) as well as publication/creation dates (e.g. aerial photographs taken over a geographical region every 10 years from 1900 to present). As a simplification, we refer to content dates. The temporal information hierarchy needs to cover all references to time, including future time. The majority of temporal references, however, deal with the last 300 years.

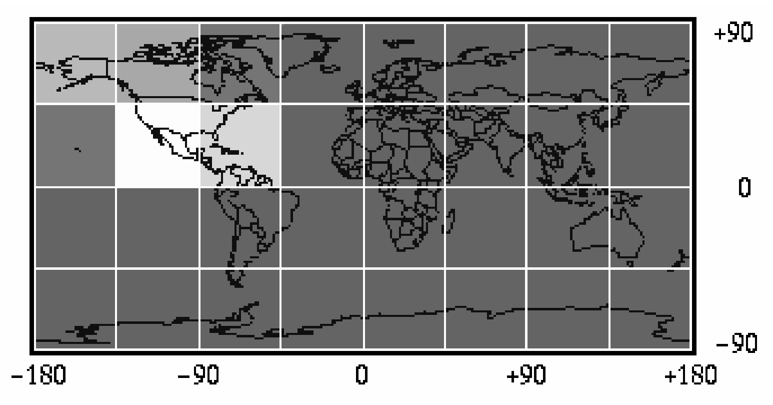


Figure 4.2: 45° × 45° Grid, 32 Elements

For example, Microsoft's Encarta Encyclopedia 98 [Mic97] allows articles to be looked up by "time". Of the 16,366 temporally referenced articles, 4,046 relate to time before the year 1700; 1,879 relate to the 1700's; 6,159 relate to the 1800's; and 8,180 relate to the 1900's. We therefore prefer to have a higher level of granularity for the time periods nearer to 'now' than those further away on a time-line. By using a modified logarithmic scale, which is approximately linear between the years 1700 and 2000, we can achieve a more even distribution of time-referenced documents (see Appendix B).

4.1.2 Information Taxonomies

An information *taxonomy* is defined as a controlled hierarchy within which documents can be classified, usually according to their content. In addition, a taxonomy is assumed to be a tree structure such that each node in the tree has two attributes: a label and a numerical value. The label denotes a word, term, or phrase from an information hierarchy. The numerical value of each node of a taxonomy is called the *coverage* of the node. It is used to describe a document collection by quantifying the number of documents with content relevant to the meaning of the label of the node. Since most documents cannot be classified completely within a single node of a classification tree, we assign portions of a document to different nodes in the tree. Thus for each node in the taxonomy there are a total number of *document equivalents* associated with it. Intuitively, a node's coverage indicates what fraction of a document collection is classifiable within a node's label. In order to calculate this value, we first allocate fractions of each document to nodes in the tree such that not more than 100% of the document is allocated. We allocate as much of a document as we can to nodes closest to the leaves of the tree. In this way, a document is classified as much as possible within the most specific sub-domains of the hierarchy. Once all the documents in a collection have been fractionally allocated within the tree, the coverage values can be computed. The children of a parent node have labels that represent information sub-domains that are wholly or partly contained within the sub-domain represented by the label of their parent. By using the notion of coverage, this containment property is captured numerically by defining the coverage at a node to be the sum of the coverages of its children plus the fraction of the collection assigned to it but not to its children. For a leaf node, this is just the fraction of the collection assigned to it. The number of document equivalents associated with a node is its coverage times the total number of documents in the collection.

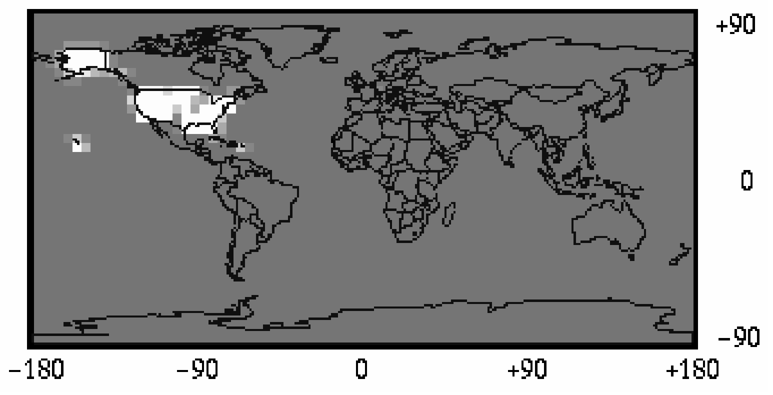


Figure 4.3: 5° × 5° Grid, 2,592 Elements

As an example, consider the geography information hierarchy discussed previously. We create a taxonomy by assigning the 45° × 45° tiles to the depth-1 nodes, the 5° × 5° tiles as depth-2 nodes, etc. Within the ADL collection of $\sim 421,000$ documents shown in Figure 4.2, the region over Hawaii, defined by the (longitude, latitude) coordinate pair [upper-left=$(-180°, 45°)$, bottom-right=$(-135°, 0°)$], contains $\sim 5,700$ document equivalents. Thus, within the geography taxonomy, this node is assigned a coverage of ∼0.014. To calculate the coverages of the depth-2 nodes, we break down the depth-1 nodes into $5° \times 5°$ tiles, as shown in Figure 4.3. This figure shows another view of the same ADL holdings as in Figure 4.2. The three depth-2 nodes which contain the Hawaiian Islands contain approximately 1500, 2400, and 1100 document equivalents, and so are assigned coverages of ∼0.0038, ∼0.0060, and ∼0.0028, respectively.

Clearly Figure 4.3 gives a much more precise display of the ADL holdings. However, the 2,592 data elements prohibit that level of detail from being included in the high-level metadata, as discussed in more detail in the next section. On the other hand, Figure 4.2 does not necessarily include enough detail to be useful. As a case in point, consider a user looking for information about Hawaii. The large-scale region over Hawaii does not appear to contain a sufficient amount of data to make the ADL an attractive source; the average coverage of the 81 depth-2 children is only $\sim$0.00017, implying that there are only approximately 220 documents for Hawaii. On closer examination, however, the more detailed map shows that the ADL is potentially a very good source for such a user; there are actually over 5000 documents for that region. With only 5% of the depth-1 node's children containing 97% of its data, a high-level comparison involving the ADL might exclude it and the depth-2 metadata would never be requested. This problem of statistically losing lower level information, and hence potentially valuable sources, can occur within any information taxonomy whenever a small fraction of a node's children contain a vast majority of the node's coverage. Even if we kept track of the variance among a node's descendents, we would not be able to determine which ones held the majority of the information without acquiring more metadata. This situation can be considered a *special collection* within a source's taxonomy. We handle special collections by providing a special metadata attribute.

The geography taxonomy is very different from the subject taxonomy in that it is not a word-based hierarchy, but a spatial hierarchy. To put this in perspective, Microsoft's Encarta Encyclopedia 98 [Mic97] allows articles to be looked up by "place". However, while a query on 'North America' yields 910 articles, a query on 'United States' yields 5,816 articles. These results do not occur under a geography taxonomy, which would place 'United States' within 'North America'; our definition of a taxonomy requires that the spatial nodes covering North America include at least all those articles assigned to the nodes covering the continental United States.

4.1.3 Metadata Levels

We now describe the relevant metadata records that need to be included in the high- and mid-level servers. The main constraint in regard to the high-level server is size. The total size T_H of information distributed to the high-level servers is the size S_H of each high-level metadata record describing a given source times the total number N_S of sources described. The overriding constraint is that T_H not grow beyond some maximum value. Using the newsgroup distribution as an example, we assume that each high-level server stores under 10 GB of data. Assuming that we want to be able to handle a large number of sources, say $N_S \leq 10^6$, we derive a value of S_H of roughly 10 KB. A high-level metadata record needs to summarize the coverage information related to various nodes in an information taxonomy. Given these values for N_S and S_H, and the width of the upper levels of the taxonomy trees, we can determine which nodes of the taxonomies should be included in the high-level metadata.[1]

The size of a high-level metadata record is dependent on the number of tax-

[1]In the future, we may allow N_S to grow larger by using a single metadata record to represent a hierarchical collection of related sources (e.g., a 'UC' source representing all of the University of California campuses). In order to allow this, the structure of a metadata record must be independent of the number of sources described.

onomies used and the number of nodes that are included from each taxonomy. Although the number of taxonomies a source uses might vary, for a given taxonomy all sources must always include the same nodes in the high-level metadata. High-level metadata refers to 1) any taxonomy-independent source description, and 2) the high-level portions of all taxonomies within which a source's collection has been classified. We determine the number of nodes to include based on our estimated size, S_H, of a high-level record, the number of taxonomies included, N_T, and the size, S_I, of the taxonomy-independent metadata. For simplicity, we assume that each taxonomy's node metadata is the same size, S_N, and that the same number of nodes, $N_H(T)$, will be included from each taxonomy T. Thus we have $S_H = S_I + S_N * N_H(T) * N_T$, or $N_H(T) = (S_H - S_I)/(S_N * N_T)$. Assuming $S_H \approx$ 10KB; $S_I \approx$ 1KB; $S_N \approx$ 100B; and $N_T \leq 4$, we conclude that $N_H(T) \approx 22$. That is, we assume that we can include on the order of 20 to 30 nodes from each taxonomy in our high-level metadata records.

Tables 4.1 and 4.2 show the portion of the high-level metadata that is independent of the taxonomies (of size S_I). In the `Description` column, we illustrate the corresponding attribute with an example. Other than the source identification information, these attributes can be viewed as quality and cost factors that are presented to the user through the UI. The user can then consider a range of factors in determining specific sources relevant to current queries. The first attribute, `HL_Desc_Ver`, is the version of the metadata schema and is used to allow for backward compatibility. The `Src_*` attributes describe basic source identification information. `Siz_Col` describes the size of the collection in MB, pages, and items. These units are used to calculate the fractional assignment of documents in three units. The reason that we use multiple units is that they may yield very different ratios. For example, an image library whose images may range from over 100MB/image to less than 10KB/image will show very different geographic coverages in MB than in items. `Siz_Cat` describes the size of the catalog in MB and items. `Count_Taxon` is set to the number of taxonomies used for classification by the source. `Spec` is the list of character strings used to describe special collections, discussed earlier. The set of parameters `Net_*` are included for connectivity estimation between the users and the sources [GS95]. `Count_Avg_Hits` describes the average number of accesses the source receives per day. Additional attributes are available for various library policies relevant to charging, lending, etc. Finally, `Count_Src` denotes the number of sources included in a particular metadata record to allow condensed multi-source records. If this value is greater than one, some of the other values might not be included, such as the network parameters.

Tables 4.3 and 4.4 show the taxonomy-dependent part of the high-level metadata. Within the full metadata record for a source, the attribute-values in Table 4.3 are repeated once for each taxonomy. `Taxon_ID` and `Taxon_Desc_Ver` identify the taxonomy name and version, respectively. The next attribute, `Cov_Root`, gives the fractional amount of the collection that has been classified within the taxonomy, with respect to MB, pages, and items. The next four attributes give statistical information about the root's children (the depth-1 nodes in the taxonomy). Furthermore, within each taxonomy, the attribute-values of Table 4.4 are repeated once for each high-level node in the tree.

The geography taxonomy is treated as a special case in order to include factors

Attribute Name	**Type**	**Range**	**Description:** *Example*
HL_Desc_Ver	String	-	Metadata descriptor version number: *0.0.7*
Src_ID	String	-	Name of source: *Alexandria Digital Library*
Src_Type	String	-	Type of source: *DLIB*
Src_Loc	String	-	Geographical location of source (if relevant): *Paris, Texas*
Src_Sch	String	-	Availability schedule of source (if relevant): *Closed Mondays*
Siz_Col	Real[3]	0.0+	Collection size in MB, pages, and items: *1.843E6,,8.91E5*
Siz_Cat	Real[2]	0.0+	Catalog size in MB and items: *2.172E6,3.892E6*

Table 4.1: Taxonomy-Independent High-Level Metadata, Part A

Attribute Name	**Type**	**Range**	**Description:** *Example*
Count_Taxon	Integer	0+	Number of taxonomies used: *3*
Spec	String[M]	-	List of M Special Collections: *Geography:BE16*
Net_Band	Real	0+	Network bandwidth (Kbps): *1.5E3*
Net_Avg_Util	Real	0.0 - 1.0	Avg. network utilization: *0.47*
Net_Avg_Delays	Integer[N]	0+	Avg. network delay (ms) to N beacons [GS95]: *,200,,521,,,147,*
Net_Avg_Thru	Integer[N]	0+	Avg. network throughput (Kbps) to N beacons [GS95]: *,117,,,,,246,*
Count_Avg_Hits	Real	0+	Avg. number of accesses (hits/day): *1.237E5*
Pol_Charg	String	-	Charging Policy: *See http://www.lib.ucsb.edu/*
Pol_Lend	String	-	Lending Policy: *UC affiliate only*
Count_Src	Integer	1+	Number of sources: *1*

Table 4.2: Taxonomy-Independent High-Level Metadata, Part B

Attribute Name	**Type**	**Range**	**Description**
Taxon_ID	String	-	Name of taxonomy
Taxon_Desc_Ver	String	-	Taxonomy version number
Cov_Root	Real[3]	0.0 - 1.0	Root node's coverage value (MB, pages, items)
Cov_Avg	Real[3]	0.0 - 1.0	Avg. coverage of depth-1 nodes
Cov_SD	Real[3]	0.0 - 1.0	Std. Dev. of coverage of depth-1 nodes
Cov_Min	Real[3]	0.0 - 1.0	Minimum coverage of depth-1 nodes
Cov_Max	Real[3]	0.0 - 1.0	Maximum coverage of depth-1 nodes

Table 4.3: High-Level Metadata for Each Taxonomy T_i

Attribute Name	**Type**	**Range**	**Description**
Node_ID	String	-	Node's label
Cov_Node	Real[3]	0.0 - 1.0	Node's coverage value (MB, pages, items)
Cov_Avg	Real[3]	0.0 - 1.0	Avg. coverage of children
Cov_SD	Real[3]	0.0 - 1.0	Std. Dev. of coverage of children
Cov_Min	Real[3]	0.0 - 1.0	Minimum coverage of children
Cov_Max	Real[3]	0.0 - 1.0	Maximum coverage of children

Table 4.4: Metadata for Each High-Level Node in T_i

such as image or map resolution, which are not relevant to most other taxonomies. This taxonomy is a superset of that in Tables 4.3 and 4.4. Additional attributes are included to describe image resolution statistics that summarize all the documents that are associated with the geographical region covered by each node.

Mid-level metadata is not subject to the same space constraints as the high-level metadata for several reasons. First, because mid-level metadata is only sparsely replicated, there are not as many network constraints in distributing it. Second, each mid-level server handles only a specific sub-domain within a single taxonomy, so it needs to store only a very limited subset of all of the mid-level metadata records for any source. Third, because users request this metadata only after they have already gone through a primary filtering using the high-level metadata, mid-level query results contain data for only a relatively small number of sources. The structure of mid-level metadata is similar to the taxonomy-dependent components of the high-level metadata, but covers nodes of greater depth in the information taxonomies.

4.2 Metadata Extraction

The process of extracting the metadata from a source begins with a document collection, as portrayed in Figure 4.4. This collection is processed by hand or automatically. For text-based documents, it can be processed by one of several possible automatic text analysis tools, such as LSI, SMART [Sal89], or others that are used, for example, in the NIST Text REtrieval Conferences (TREC) [Har95]. Many of these systems read through a collection and build a matrix of weighted frequencies of the number of times a given term occurs in each document in the collection. From this matrix, documents can be placed in a multi-dimensional term-vector space. Nodes in a taxonomy can also be placed in this space and then be taken as centroids of document clusters. Depending on the relative closeness of the centroids to the various documents, we assign the fractional allocation of each document among the nodes in the taxonomy needed to calculate each node's coverage value. High-level automatic subject classification has been shown to be fairly successful, with over a 90% accuracy rate [LH95]. LSI has been shown to be an effective tool for automated classification [Hul94], and experiments on automatically classifying large document sets within the LC Classification system have yielded correct classifications, under some conditions, of over 80% of newly entered documents [Lar92]. These results indicate that subject-based automated text classification is sufficiently accurate to characterize sources for comparison purposes, where even order of magnitude estimates can greatly aid in filtering out most irrelevant sources. We will discuss details of such automated classification in Chapter 7.

While the TREC and related work focuses on subject-based text retrieval, one could use a gazetteer or time-name table to identify geographic or temporal references within text-based documents, or use more sophisticated techniques such as those used in GIPSY [WP94]. For maps, aerial photographs, and satellite images, documents are often cataloged with spatial extents, and the geographic clustering and classification process is much simpler. Automatically extracting

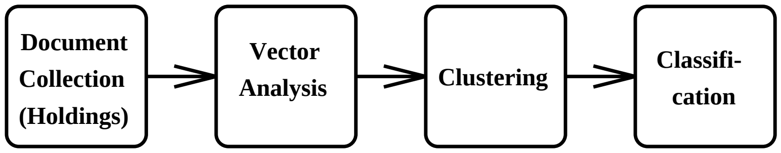


Figure 4.4: Metadata Extraction

subject information from maps and images, such as identifying vegetation in a map or a particular object in an image, would allow such documents to be automatically classified within, for example, the LC Classification system; such capabilities are ongoing research issues [Mos94, Ric93]. Yet another application area of this process is the incorporation of image feature vectors [MM98]. Given a hierarchical feature vector thesaurus, one could automatically classify images in much the same way as text. Although there are several methods of characterizing and classifying image features (textures, colors, shapes, etc.), Pharos works equally well with any (hierarchical) image classification scheme.

4.3 User Interface and Metadata Retrieval

The User Interface (UI) is designed to assist the user in an iterative decision-making process [EY72] in which a final small set of sources is chosen through a series of refinements on an initial large set. This process uses information about the user, called the *User Profile*, information about the query, called the *Task Profile* [Shn92], and the high- and mid-level metadata about the sources. Although the client acquires source metadata through the network from the high-

and mid-level servers, it acquires user and query information through the UI. Besides acquiring this information, the UI must also present metadata about many sources to the user in a concise and understandable manner. This allows the user to interact with that data and to select source subsets for further metadata retrieval. Figure 4.5 illustrates the key components of the UI subsystem in Pharos. User and Task Profiles are initialized by the user. When a query is issued, these profiles are used to send an initial metadata request to a local high-level server. The metadata returned includes information about the likely candidate sources that best match the query, subject to the User and Task Profile. The metadata is then visualized to the user within the UI. The visualization includes default but tailorable quality and cost factors. Given this information, the user then selects a set of sources, perhaps those with the highest 'value'. From this set, a new metadata request is issued to the specific mid-level servers that store information relevant to the user's query. The returned metadata is again visualized, but with greater accuracy and detail. Finally, the query results are narrowed down to a reasonably small set of sources, which are then directly queried.

The User Profile information in Pharos is used to determine which metadata parameters to retrieve, how to compare sources, and how to display the results. The user can assign relative weights to any parameter to adjust the comparison results. Other profile information includes interface parameters such as which taxonomies to use by default. As the set of sources becomes small enough, factors can be considered that require direct communication with each source, such as the number of network hops between the user and the sources. The Task

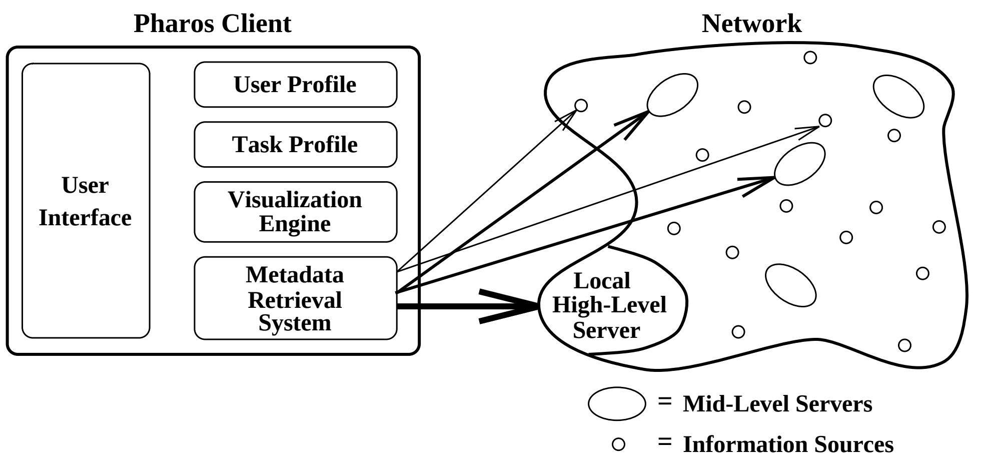


Figure 4.5: Pharos Client Components

Profile includes all parameters needed to specify the source search criteria. The user must decide which taxonomies to use. For each taxonomy included, the user selects one or more sub-domains either by entering keywords, or by traversing the tree directly and selecting particular sub-domains. The UI must deal with keyword entries differently for each taxonomy. For the subject hierarchy, techniques used in text retrieval, such as by LSI [BDO95], can aid in matching query terms with terms found in the hierarchy. The geography taxonomy requires a gazetteer, and the time-period taxonomy requires the equivalent of a gazetteer for time-names. Finally, as the set of sources becomes small enough, the UI attempts to match keywords with any special collections included in the set.

Since there are a large number of sources, it is impractical to list the results of a query in a text-based, tabular fashion. The level of detail of the display of each

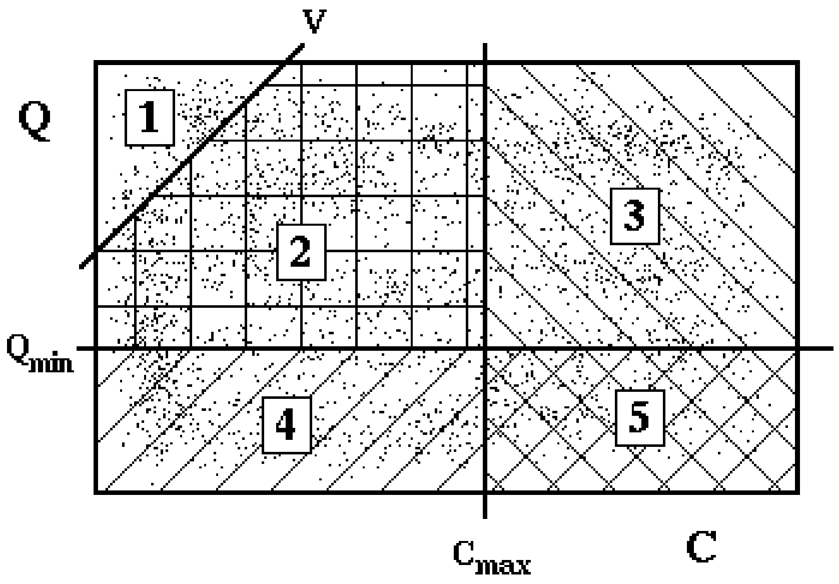


Figure 4.6: Simple Metadata Visualization

source is adjusted depending on the total number of sources being displayed. Initially, we can display each source as a point in a two-dimensional scatter plot, as depicted in Figure 4.6. There are several ways of assigning positions to sources. One possibility is to use a quality versus cost approach to visualize the economic value of the sources [Whi76]. The quality and cost dimension of each source is computed as a weighted sum of the relevant attributes and each source is depicted as a point in the Quality/Cost plane. The user can then graphically select sources which are above a minimum quality (regions 1, 2, and 3), below a maximum cost (regions 1, 2, and 4), and/or have the highest value (region 1). Many other forms of weightings and attribute combination are possible. For example, the ranges of values that each attribute may attain vary widely both between different attributes and also, for a given attribute, between different sources. We therefore can use normalized attribute values by dividing each one by the maximum value of that attribute among all the sources. Furthermore, for values such as network bandwidth and collection size, we can compare the normalized logarithm of the values so that large values do not overly dominate. Similarly, many other visualizations are possible, especially as the number of sources is reduced after filtering.

4.4 High-Level Metadata Servers

Once the metadata at each site has been compiled, it needs to be distributed over the network according to the intended storage and retrieval architecture. As previously stated, the high-level metadata needs to be widely distributed and replicated, while the mid-level metadata is very selectively distributed. Because of these storage differences, different distribution protocols are more efficient for the different metadata levels. All high-level metadata is sent to each high-level server.

We briefly compared the distribution of the high-level metadata to the distribution of USENET news [Hor83] via NNTP [KL86] in Chapter 3. NNTP uses a flooding protocol which has proved to be very robust and efficient for widely distributed, massively replicated data. In fact, rather than create a completely new protocol and transport system, we plan to distribute the high-level metadata in Pharos by using a new newsgroup hierarchy. This can be accomplished, for example, by having each source send out their high-level metadata as a news article. There are several advantages to this approach. First of all, it fulfills the basic design criteria of massive metadata replication. Second, NNTP provides a fast, reliable, and reasonably efficient mechanism of developing a prototype system. Third, this approach poses little or no degradation to the news system because news servers not wishing to store this information may selectively not receive a 'feed' for it, just as they can turn off reception of any other newsgroup hierarchy. Fourth, USENET provides a hierarchical naming convention among the newsgroups, which could be exploited by the source metadata design. For example, collections could be characterized based on theme, such as dl.medicine for MEDLINE, and dl.physics and dl.computer-science for INSPEC, so that servers within specific organizations can retrieve metadata tailored to their interests. Finally, those sites running a news server within their local area are prime candidates for high-level server locations.

A potential problem with this type of posting is that anyone could post articles with bogus metadata about a given source, such as the Library of Congress or the National Institutes of Health. This problem can be avoided by using moderated newsgroups and a standard electronic signature. When a source registers itself, it provides either a public encryption key or a place on the network where it can be found. Each posting is sent to an automated moderator that checks the signature against the source's key for verification before posting it.

Another consideration, as with the mid-level metadata, is the amount of information that can be handled. NNTP traffic generally generates approximately 400 MB per day or more. By cutting off unwanted newsgroups, news servers can substantially reduce the traffic coming to them. Recall that the total size of all the high-level metadata combined is on the order of several gigabytes. We imagine that most sources' statistical compositions of documents probably do not change dramatically over a short period of time. Therefore weekly or monthly updates are sufficient to keep high-level servers up-to-date to the level needed to assist users in the high-level, rough filtering of sources. We anticipate that this entire news hierarchy would not generate more than a small fraction of the normal NNTP traffic.

4.5 Mid-Level Metadata Servers

Mid-level metadata is not massively replicated, but rather stored on one or perhaps a few mirror sites. As such, a flood protocol is neither necessary nor efficient. Instead, a point-to-point metadata distribution system is more appropriate. Each source registers at each mid-level server for which the source has relevant metadata, and sends updates as needed. The sources in Pharos are responsible for keeping their distributed metadata up-to-date. This approach minimizes network traffic and guarantees that the mid-level servers receive new metadata only when necessary.

Harvest [BDH$^+$94] provides a suitable transport mechanism for distributing and storing mid-level metadata in Pharos. Harvest is a generalized system for automatically indexing documents within a source and distributing the metadata. In Harvest terminology, sources of information are called *providers.* Index metadata is extracted by *gatherers.* The gatherer is broken down into two components, one that extracts the metadata, and the other, *gatherd* (gather daemon), that handles the gatherer-side communication necessary for distributing it. The structured indexing information that the gatherer collects is represented as a list of attribute-value pairs using the *Summary Object Interchange Format (SOIF).* This metadata is then served to *brokers*, who collect it via *collectors* (which communicate with a gatherer's gatherd) and provide a query interface to the indexes. Brokers can collect metadata either from gatherers or from other brokers, and thus lend themselves to hierarchical metadata propagation.

Although we must perform our own metadata extraction (or *gathering*), incorporating the results into SOIF records is straightforward. Each source constructs a SOIF record for each taxonomy sub-tree used by a mid-level server. Each source builds a standard Harvest SOIF database and runs gatherd. Each mid-level server runs a standard broker that collects the appropriate records from each source. This collection is efficient in Harvest because of the broker's ability to use structured queries that can specify boolean combinations of attribute/value pairs. For example, a broker would retrieve a record by specifying `'(Taxon_ID : Subject) AND (Top_Node :  Physics)'`. Finally, the brokers running on the servers are designed not only to collect the metadata, but also to handle sophisticated queries. Therefore, this software already facilitates the communication between the UI and mid-level servers.

There are several benefits of using Harvest for distributing and storing mid-level metadata. First, Harvest uses very efficient network protocols and compression for exchanging metadata records. Second, it allows us to place several logical mid-level servers on the same physical server; we need only change the query between the server's broker and the sources so that multiple SOIF records are transferred. Additionally, Harvest includes metadata timestamps and automatic server updates for expired information. Each source is able to set its own timestamps and thus regulate its own update frequency. Further, Harvest's ability to chain brokers allows us to incorporate a more sophisticated mid-level network model than requiring that each source directly communicate with each server. Instead, we can arrange the sources in a hierarchical network structure by placing intermediate brokers between the sources and the servers.[2] Finally,

[2]In this sense, the mid-level servers can be viewed as the point between physically structuring the metadata based on network topology and physically structuring it based on infor-

Harvest includes both a replication and caching system. The replication allows us to automate the process of mirroring the mid-level servers. Moreover, the caching allows us to take advantage of possible speed-ups by grouping together similar sources as pre-packaged sets within the UI; such query results might stay in a server's cache.

mation topology.

Chapter 5

Comparison to Other Models

As previously mentioned, prior to constructing a prototype, we first wanted to determine the feasibility of the Pharos architecture presented in the previous chapter. The feasibility study is comprised of two components. The first is a comparison between Pharos and other architectures, presented in this chapter. The second, presented in the next chapter, is an estimate of the accuracy of Pharos query results. This chapter presents a systematic description of possible network architectures that support the discovery of information sources, and analyzes their differences. The first section describes relevant concepts such as query routing and the extraction, propagation, and retrieval of metadata. Based on these concepts, different models of locating and querying relevant information sources are presented within three broad classes. Finally, we estimate several important characteristics of these models and classes as well as their expected scalability.

5.1 Terminology

We first describe the terms that will be used in the network models. We define a *document* operationally as an atomic entity that can be searched for, retrieved, and viewed, though not necessarily stored, as a single unit, electronically or otherwise. A document is not required to be stored atomically, nor is it necessarily always used atomically; although one can access sections of a document, a document must be available as an atomic entity. A *collection* refers to a well-defined set of documents. A *query* refers to a document specification against which documents in a collection can be compared for relative similarity. A *query generator* is a system that electronically creates and transmits queries; they are represented as circles, $\circ$, in the model diagrams. A *search engine* is an interface that accepts as input a query about a particular collection and returns as output a *query result*, which is a subset of the collection that the search engine has determined contains documents relevant to the query. The subset may not contain the actual documents, but rather pointers to, and possibly descriptions of, the documents. A single collection may have multiple search engines associated with it, but by definition a search engine can work on only a single collection (i.e. we do not require a collection to be an input parameter to a search engine). An *information source*, or *source*, refers to a particular [collection, search engine] pair. Sources are represented as triangles, $\triangle$, in the model diagrams. We point out that *collection* refers only to a set of documents, while *source* refers more generally to a machine, including its documents, search engine, metadata, etc.

Metadata refers to the description of an object. We make a distinction between *document metadata* and *collection metadata*. The former is information about individual documents, while the latter is summary information about an entire collection and possibly other descriptive information about the machine, network parameters, etc. Unless otherwise stated, *metadata* will refer to *collection metadata* in this discussion. Therefore, a query result is the document metadata that would be returned by a search engine given a particular query. It is not necessarily returned directly by a search engine, but by any system that gives the same document subset, as explained below.

Finally, an *intermediary*, drawn as a small or large box, □, in the model diagrams, is a logical machine on the network that provides a level of indirection between a request for information and the original supplier of the result, usually an information source. A *query intermediary* acts as a pseudo search engine; it takes queries as input and provides query results as output. However, a query intermediary differs from a search engine in that 1) it does not necessarily have direct access to any document collection, and 2) it may return query results for several sources simultaneously. A *metadata intermediary* accepts a request for either document or collection metadata and supplies the relevant result. Intermediaries may store information locally, or may dynamically request information either from another intermediary or from a source.

Our network models describe several common activities. *Query propagation* or *query routing* is the process of passing a query from a query generator to either a source or a query intermediary. This action is expected to result in *result retrieval*, which returns a query result to the query generator. *Result merging* involves combining query results for a query sent simultaneously to multiple sources. Documents in a result set are often ranked within that set; result merging can include *rank merging* where the union of the multiple result sets are re-ranked for the entire union. Result merging can also include *duplicate detection*, in which an attempt is made to detect if the same document has been returned by multiple sources.

Metadata extraction denotes the process of deriving either document or collection metadata. Document metadata extraction can take place either at a source or at an intermediary which has direct access to a source's documents. Collection metadata extraction is generally done only directly at a source, though there is no requirement for this. *Metadata propagation* involves the 'pushing' of metadata by a source to any other machine. *Metadata retrieval* involves the 'pulling' of metadata by an intermediary or a query generator that either stores it locally for long-term use or else uses it immediately to help resolve a query and then discards it. The models that follow have examples of both long-term and short-term use of both document and collection metadata. An *update* is the process of bringing up-to-date all stored metadata across the entire network about a particular source.

5.2 Network Models

This section describes three classes of network models that encompass the discovery and querying of information sources. The classes, in increasing complexity, use 1) no intermediary, 2) a single intermediary, and 3) multiple intermediaries.

5.2.1 Direct

The models in Class 1, the *Direct* models, do not include an intermediary. We describe three architectures; these are not intended to be realistic for a large-scale network, but rather exemplary of particular problems of scalability which will be discussed in Section 5.3.

The Simple Model is the most basic model, which contains one or more query generators but only a single information source. Queries are sent directly to the information source and the results are sent back directly to the generator of each query. This model does not differentiate between simple, one-site sources, and complex, multi-site sources with a single query point. The Simple Model is overly simplistic and does not shed much light on scalability problems; we therefore do not consider it further. Next, the Remote Model allows for multiple sources by simply adding them in and not worrying about which sources are queried nor how a query generator should merge the results. It is implicitly assumed that each query will be propagated to all sources.

Neither the Simple Model nor the Remote Model include the use of any collection metadata. The next model, the Local Model, requires that each query generator pre-collect sufficient metadata about each source so that the best sources can be decided locally at the generator without first contacting any remote site. To handle a query, the only traffic generated in this model is that

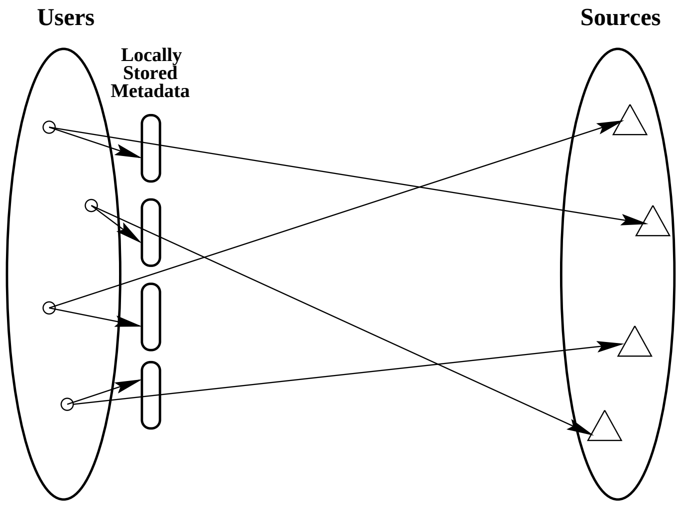


Figure 5.1: Local Model Query

used to query the most relevant sources and retrieve their results. This model assumes that there is a standard method with which metadata can be retrieved and merged by the query generators. This model is shown in Figure 5.1.

5.2.2 Single Intermediary

Within Class 2, the *Single Intermediary* models, we describe two architectures: the Brute-Force (BF) Model and the STARTS Model [GCGMP96]. Both these models maintain a single intermediary between the query generators and sources. This intermediary handles both query and metadata traffic, though there is not necessarily any synchronization between its reception of a query and its metadata retrieval. Although there may be several intermediaries on the network simultaneously in this class of models, they do not communicate or coordinate with each other. A query generator must decide which intermediary to use for a given query, though by definition, no method is provided for assisting in this decision. This uncertainty affects estimates of query network traffic, as will be discussed in Section 5.3.2.

The simpler of the two models described in this class, the BF Model, is the system used by most, if not all, current WWW index sites and is depicted in Figure 5.2. In this model, an intermediary performs metadata extraction by gathering all documents available at all sources. Current implementations extract only limited document metadata to reduce storage space, and no collection metadata. In principle some collection metadata could be extracted also. This process is driven completely by the intermediary; the sources have no control over the frequency of metadata collection nor over the type of metadata extracted. Queries are sent to the intermediary, which uses its pre-collected metadata to determine which of the documents it has analyzed are relevant, and then returns a (possibly ranked) result set to the query generator. Queries are not propagated to the sources, and results include information only about documents which have been analyzed directly by the intermediary.

Another approach is the STARTS Model [GCGMP96], shown in Figure 5.3. In this model, built on a *gGlOSS* [GGM95] framework, each source performs its own metadata extraction. The intermediary then gathers this metadata rather than extracting its own. Queries in the STARTS Model are of a keyword matching nature, although the design allows for extensions to this query structure. When an intermediary receives a query, it analyzes its pre-collected metadata and chooses a small set of 'best' sources. The query is then propagated to these

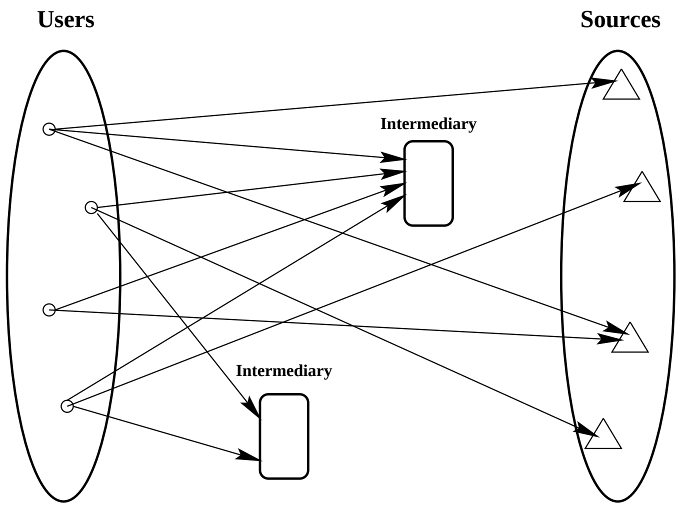


Figure 5.2: Brute-Force Model Query

sources in a standard format and they return the results (document metadata retrieval), also in a standard format, back to the intermediary. The intermediary is then responsible for merging the results from all the queried sources and passing a single, ranked list of results back to the query generator.

The major differences that the STARTS Model has from the BF Model are that 1) it allows (actually, requires) each source to extract its own metadata, and 2) there are standard formats for information exchange (metadata, queries, and results). The STARTS Model does not require that an intermediary store document metadata as does the BF Model. Instead, it retrieves document metadata at query time, including, for example, an abstract where applicable.

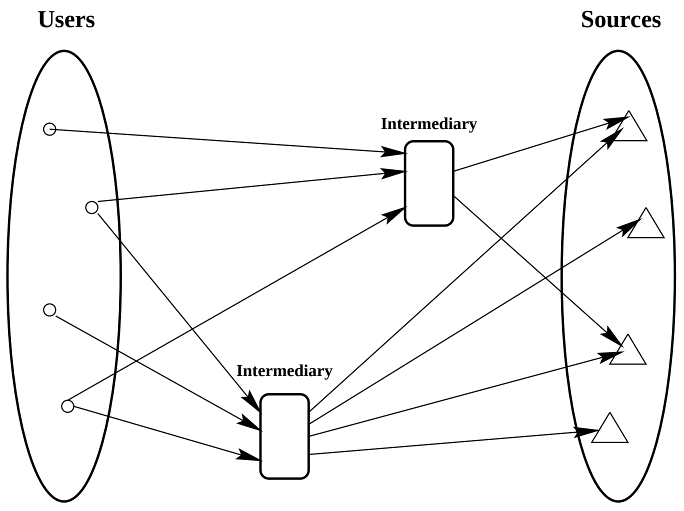


Figure 5.3: STARTS Model Query

5.2.3 Multiple Intermediaries

Within Class 3, the *Multiple Intermediaries* models, there are several known frameworks such as Content Routing [SDW$^+$94], *hGlOSS* [GGM95], and Harvest [BDH$^+$94]. These systems provide for multiple intermediaries within a DAG-like network structure by allowing intermediaries to be nested; collections of source summaries can be viewed as single collections by higher-level intermediaries. However, these systems do not describe the actual network architecture. Furthermore, although *hGlOSS* takes its metadata structure from the underlying text-based vector spaces, the other two do not specify the nature of the metadata. We seek to exemplify this class with a model that highlights and benefits from the relationship between the metadata structure and the network architecture.

Thus we define three extensions to standard metadata, which have been incorporated into the Pharos architecture described in Chapter 4. The first extension is *hierarchical metadata*, which means that the collection metadata is organized in a tree-like structure. Information at a parent node in the tree contains some form of summary of the information of all its children, though possibly with less detail. The second extension is that of *shared metadata*, in which metadata is partitioned, rather than simply replicated, between several *cooperating* intermediaries. Thus, depending on the architecture, an intermediary can collect unique metadata about many sources; other systems can then retrieve that specialized information if and when they need it without having to communicate with the separate sources directly. The last extension is that of *non-text metadata*, which describes non-text aspects of documents and collections. For example, documents that either include or consist entirely of images, sounds, or maps may not be best described by the text associated with them (if any). Many of these non-text documents, as well as many text documents, are also characterized by, for example, image feature-vectors, geographical coordinates, temporal information, etc. All these metadata extensions are used by Pharos, shown in Figure 3.1.

Pharos, described in detail in the last chapter, is designed to select a small set of (highly relevant) sources from among a large set. While Pharos separates the query activity (query propagation and result retrieval) from the metadata propagation, as in the previous models, it differs from them in that it synchronizes this activity with the collection metadata retrieval. Metadata is extracted by the source, as in the STARTS Model, but here the metadata is composed of a small amount of non-hierarchical information and a relatively large amount of hierarchical information defined by one or more information taxonomies. A source propagates its metadata as needed to the relevant intermediaries.

5.3 Model Comparison

Some of the criteria that we might use to compare these different models and classes, especially from an implementation point of view, are scalability, flexibility, and complexity. Increases in design and processing complexity should be warranted by corresponding increases in either scalability or flexibility. The performance of these models depends on factors such as the number of queries, the number of sources, and the size of a collection. We attempt to estimate several important characteristics of each model as well as the three classes in general. We have selected several model parameters and estimated their corresponding order of magnitude values. We then use these parameters to derive equations for network traffic, storage requirements, and the number of accesses. Next, we use the sample values to check that these equations are realistic and to identify potential problems. Finally, we discuss how these characteristics are likely to change with the future growth of the Internet.

5.3.1 Model Parameters

Tables 5.1 and 5.2 list all model parameters used in our derivations. All values in this and the following tables are written in $\log_{10}(x)$. We first list the values

Symbol	**Small**	**Large**	**Typical**	**Description**
N_D	2	8	4	Number of documents in a single collection
N_{RD}	1	4	2	Number of relevant documents from a single collection
N_S	2	7	5	Number of sources
N_{BS}	0	2	1	Number of 'best' sources
N_{RS}	0	5	2	Number of relevant sources (i.e. sources with any relevant documents): assumed to be $\sim N_S/500$
N_{QG}	2	8	6	Number of query generators
N_{II}	1	4	2	Number of independent intermediaries (Class 2)
N_{MLI}	1	3	2	Number of unique mid-level intermediaries (Pharos)
N_{IM}	0	2	0	Number of duplicate mid-level intermediaries per 'topic' (Pharos)
N_{HLI}	2	5	3	Number of high-level intermediaries (Pharos)
N_{HLF}	1	4	3	Number of sources selected by high-level filter (Pharos)
N_{QI}	0	1	0	Number of taxonomy nodes in a query (Pharos)

Table 5.1: Model Parameters, Part A (all values are $\log_{10}(x)$)

Sym	**Sml**	**Lrg**	**Typ**	**Description**
S_D	2	8	5	Size of a non-specific document
S_{DT}	2	6	4	Size of a text (ASCII) document
S_C	4	14	9	Size of a collection of non-specific documents: $N_D * S_D$
S_{CT}	4	12	8	Size of a collection of text (ASCII) documents: $N_D * S_{DT}$
S_Q	1	3	2	Size of a query
S_M	1	4	3	Size of a query result (returned metadata record) for a single document (e.g. [title, author, date, keywords, abstract])
S_R	1	8	5	Size of a query result (all relevant documents) from a single source: $N_{RD} * S_M$
S_{CTS}	4	9	6	Size of full (text) collection metadata at a source (STARTS): see Equations 5.3.1–5.3.3
S_{CTI}	6	14	11	Size of full collection metadata for all sources, at an intermediary (STARTS): see Equation 5.3.4
S_{TN}	0	2	1	Size of information for each node in a taxonomy (Pharos)
S_{CPS}	4	6	5	Size of full collection metadata at a source (Pharos): $S_{TN}*$ (nodes/taxonomy) $*$ (taxonomies/source)
S_{HPS}	1	4	3	Size of full collection high-level metadata at a source (Pharos): extracted from ~25 depth-1 nodes of the taxonomy trees
S_{MPS}	2	5	4	Size of full collection mid-level metadata at a source (Pharos): extracted from $\sim 25^2$ depth-2+ nodes of the taxonomy trees
S_{HLI}	4	10	8	Size of metadata at a high-level intermediary for all sources (Pharos): $\sim S_{HPS} * N_S$
S_{MLI}	6	11	9	Size of combined metadata of all unique mid-level intermediaries for all sources (Pharos): $\sim S_{MPS} * N_S$

Table 5.2: Model Parameters, Part B (all values are $\log_{10}(x)$; sizes in bytes)

used to describe quantities such as the number of documents in a collection and the number of sources. Next we list the values used to describe, in bytes, quantities such as the size of a document and the size of a query result. These tables include estimates of what might be considered a parameter's small, large, and typical order of magnitude. For example, N_D, the number of documents in a single collection, is estimated to range from 10^2 to 10^8, and a typical collection is estimated to contain 10^4 documents.[1]

Several of these parameters are specialized and need some further explanation. In Pharos, queries are based on one or more topics chosen from several information classifications, and the number of topics chosen per query, N_{QI}, affects how many mid-level intermediaries are involved in handling a query: one intermediary per topic. We differentiate S_{DT}, the size of an ASCII text document, from S_D, the size of a non-specific document, because architectures such as the BF and STARTS models are designed for text documents; ASCII documents tend to be smaller than, for example, images.

All sizes in Table 5.2 after S_R are related to metadata. It is difficult to estimate these sizes; as an example, we derive two of them. For text-based vector analysis [GGM95, Sal89], the size S_{CTS} of the full collection metadata at a source is a function of the number of unique words, t, in its collection, which is generally a function of the total number of words in the collection, N. For $t = f(N)$, clearly $f(N)$ is greater than $\log(N)$ and less than N. Salton [Sal89] gives $t = kN^\beta$, for constants k and β, $10 \leq k \leq 20$, and $0.5 \leq \beta \leq 0.6$. We take $k = 15$

[1]The 1987 Annual Report of the Library of Congress lists approximately 86 million items in its collection; INSPEC contains well over a million articles.

and $\beta = 0.55$. Next, we must estimate N as a function of S_{CT}, the size of a collection of ASCII text documents. Salton points out that, while average word length for distinct English words is 8.1 characters, the average word length in ordinary English text (with many repeated words) is 4.7 characters. Since we are not assuming ordinary English text necessarily (or even English, let alone text), we take the average number of bytes per word, L, as 6. Finally, assuming that each word requires at least three 4-byte numbers in the metadata[2], we derive the following estimate:

$$\begin{aligned} S_{CTS} &\gtrsim (12 + L) * t && (5.3.1)\\ &\approx [((12 + L) * k)/L^{\beta}] * (S_{CT})^{\beta} && (5.3.2)\\ &\approx 100 * (S_{CT})^{0.55} && (5.3.3) \end{aligned}$$

with $t = k * N^{\beta}$ and $N = S_{CT}/L$.

An intermediary is required to maintain this metadata for each source, regardless of whether the words from a source have already appeared in an intermediary's global list of unique words. Thus

$$S_{CTI} \approx S_{CTS} * N_S \qquad (5.3.4)$$

where S_{CTI} is the size of metadata at an intermediary and N_S is the number of sources from which the intermediary has collected information.

[2]The metadata may require, for example, the number of documents in which each word occurs, the total number of occurrences of the word in the entire collection, fields in which the word occurs, etc.

5.3.2 Analysis

Estimates for several characteristics of each model are given in Estimate Tables 5.3–5.6, where all expressions are derived from the model parameters described in the previous section. Values are derived by applying the constant 'typical' values in Tables 5.1 and 5.2 to the expressions in each Estimate Table. Any column which is not applicable to the corresponding model is marked 'N/A'; such is the case, for example, of metadata quantities for the Remote Model, which has no metadata. This is different from columns which potentially apply to a model but for which the particular model has a zero quantity, marked '-none-'.

Table 5.3 gives estimates for remote network traffic generated by a single query and by a single update. We assume that network traffic is sent efficiently (e.g. text compression is $\sim$50%), but this does not greatly affect order-of-magnitude estimates. We specify *remote* traffic because we do not include in these estimates any traffic assumed to travel between machines within the same local area. This excludes, for example, traffic in the Pharos Model between a query generator and its corresponding local high-level metadata server. Table 5.4 gives estimates for long-term metadata storage requirements. Metadata storage is required at the query generator in the Local Model only. However, Pharos stores some metadata at high-level servers within a query generator's local area; there are a total of N_{HLI} such servers, and this storage requirement is listed as residing at the query generator. In our numerical estimates, we assume that each Class 2 intermediary retrieves metadata from every source. In reality, some intermediaries collect

Model	Single Query		Single Update		Comments
	Expression	Typical Value	Expression	Typical Value	
Remote	$N_S * (S_Q + S_R)$	10	N/A	N/A	No metadata
Local	$N_{BS} * (S_Q + S_R)$	6	$N_{QG} * S_{CTS}$	12	Assumes STARTS-style metadata (i.e. text only)
BF	$S_Q + (N_{RS} * S_R)$	7	$N_{II} * S_{CT}$	10	Query result assumes no duplicate detection
STARTS	$(S_Q * (1 + N_{BS})) + (2 * S_R * N_{BS})$	6	$N_{II} * S_{CTS}$	8	
Pharos	$(\sim N_{HLF} * N_{QI} * S_{TN}) + (N_{BS} * (S_Q + S_R))$	6	$(N_{HLI} * S_{HPS}) + (N_{IM} * S_{MPS})$	6	

Table 5.3: Estimates of Remote Network Traffic (in bytes; all values are $\log_{10}(x)$)

	Combined		**Typical Values**		
	Query	**Combined**	**Average per**		
Model	**Generators**	**Intermediaries**	**Storage Site**	**Total**	**Comments**
Remote	-none-	N/A	-none-	-none-	No metadata
Local	$N_{QG} * S_{CTI}$	N/A	11	17	Assumes STARTS-style metadata (i.e. text only)
BF	-none-	$N_{II} * (N_D * N_S * S_M)$	11	13	Assumes smaller S_M
STARTS	-none-	$N_{II} * S_{CTI}$	11	13	
Pharos	$N_{HLI} * S_{HLI}$	$N_{IM} * S_{MLI}$	8	11	

Table 5.4: Estimates of Metadata Storage Requirements (in bytes; all values are $\log_{10}(x)$)

	Combined		Typical Values			
	(Remote)	Combined	Average per			
Model	Intermediaries	Sources	Intermediary	Source	Total	Comments
Remote	N/A	$N_{QG} * N_S$	N/A	6	11	
Local	N/A	$N_{QG} * N_{BS}$	N/A	2	7	
BF	N_{QG}	$(N_{QG} * N_{RS} * N_{RD})/1000$	4	2	7	Assumes retrieval of 0.1% documents
STARTS	N_{QG}	$N_{QG} * N_{BS}$	4	2	7	
Pharos	$N_{QI} * N_{QG}$	$N_{QG} * N_{BS}$	4	2	7	

Table 5.5: Estimates of Query Accesses (one query per generator; all values are $\log_{10}(x)$)

Model	**Combined (Remote) Intermediaries**	**Typical Values**		**Comments**
		Average per Intermediary	**Total**	
Remote	N/A	N/A	N/A	No metadata
Local	$N_{QG} * N_S$	5	11	Query generator is intermediary
BF	$N_D * N_{II} * N_S$	9	11	
STARTS	$N_{II} * N_S$	5	7	
Pharos	$(N_{HLI} * N_S) +$ $(N_{MLI} * N_{IM} * N_S)$	5	8	

Table 5.6: Estimates of Update Accesses (one update per source; all values are $\log_{10}(x)$)

more than others, and as a result, many queries are propagated to multiple intermediaries by users trying to find as good information as is practical. The amount of duplicate query generation is difficult to estimate and we do not attempt to do so. Tables 5.5 and 5.6 give the total number of accesses generated assuming that each query generator issues a single query, and each source issues a single update, respectively. Thus the total number of queries and updates are determined by the number of query generators, N_{QG}, and sources, N_S, respectively.

The BF Model returns metadata about relevant documents from all relevant sources – a potentially much larger set than that returned by STARTS or Pharos, which deal only with the 'best' sources. If the BF Model does not accurately rank the documents in its result set, users must fetch many documents in order to determine their relevancy. Even with a ranked list, however, a lack of document duplicate detection would require needless document fetches. We account for the need to check documents in our estimates of the BF Model by assuming that the top 0.1% of the documents returned by an intermediary is checked for relevance, and therefore included in the query access count in Table 5.5.

As an example, we derive the Single Query Expression for the STARTS Model in Table 5.3. The query, of size S_Q, is first propagated to the intermediary, which matches it against its metadata and selects the best sources to query directly. These N_{BS} sources are then handed the query. So far the query has generated $S_Q * (1 + N_{BS})$ bytes of remote network traffic. The next action is the result retrieval process. Each queried source will send back its own query result assumed to be of size S_R. The intermediary in STARTS will perform only limited result merging: rank merging, but no duplicate detection. Thus, the same amount of traffic sent to the intermediary is bundled together and re-sent to the query generator, for a total result retrieval of $(2 * S_R * N_{BS})$ bytes. While one might consider the factor of 2 to be superfluous, it is negligible in the order-of-magnitude estimate anyway. Thus, the total amount of traffic generated by a single query in the STARTS Model is $(S_Q * (1 + N_{BS})) + (2 * S_R * N_{BS})$ bytes, as shown in the table. In fact, in all the models, traffic caused by queries is dominated by multiplicative factors of S_R, with the best being $S_R * N_{BS}$ in STARTS and Pharos.

5.3.3 Scalability Estimates

We first discuss our estimates assuming the parameters take on values that might be currently considered typical, as were listed in Tables 5.1 and 5.2. After that, we increase three key values and re-compare some of the models.

Typical Current Values

We intuitively expect that Class 1 should show the greatest scalability problems. Each model in this class breaks down in a different way. If the characteristics we are estimating are the most relevant ones, we expect that some of these reasons will show up in one or more of the Estimate Tables 5.3–5.6. For example, Table 5.5 shows that sources in the Remote model receive orders of magnitude more query accesses than in the other models because each query is propagated to every source. The last model discussed in this class, the Local Model, shows problems with, for example, the amount of traffic needed to perform a single update. The model also does poorly in terms of storage and total update accesses. Total storage is a problem because the model must assume that each query generator has its own copy of all metadata in order to be in Class 1.

The models discussed in Class 2 are the closest to present working systems. It is important to distinguish behavior evidenced in a particular model from behavior inherent in the class. Both BF and STARTS, for example, have high storage requirements, as seen in Table 5.4. However, this is a result of the fact that the size of the collection metadata used in these models is a function of the size of the collection. Pharos uses a classification-based metadata structure whose size is independent of the collection size, and thus yields smaller storage values. Such a metadata structure is not inconsistent with Class 2. The BF Model is particularly problematic in the number of update accesses, shown in Table 5.6. This is because the model requires that the intermediaries separately fetch each document from a source rather than fetching a single, albeit large, collection metadata record as in the STARTS Model. STARTS achieves the reasonable goal that each intermediary is accessed no more than once per update.

One of the advantages of Class 3 in general is that it allows intermediary metadata to be less than linear with respect to the number of sources; this is not possible in Class 1 or 2. Several resource discovery models [BDH$^+$94, GGM95, SDW$^+$94], including Pharos, allow intermediaries to collect information from other intermediaries, including metadata describing a collection of *collections*, rather than just a collection of *documents*. An appropriate network hierarchy of intermediaries could arbitrarily reduce the size of the metadata, at the cost of loss of discrimination between different sources. Furthermore, grouping sources behind particular intermediaries reduces the number of update accesses per intermediary, at the cost of maintaining more of them.

The only Class 3 model we have discussed is Pharos. Pharos performs well in the estimates for several reasons. Network traffic is minimized for updates for the same reason that storage is minimized; as stated, the metadata size is independent of the collection size. Pharos stratifies metadata within a collection. Because of this layering, it can send pieces of metadata selectively to different intermediaries; this allows a large number of sources to be coarsely described in a compact way. This type of metadata partitioning is possible only if the information itself is hierarchically structured. It can be utilized only within Class 3, since its benefit arises from having intermediaries hierarchically arranged in a way that matches the hierarchy within which a collection is classified.

Typical Future Values

We have chosen typical values for our estimates based more or less on the current (1997) Internet environment. It is generally expected that several parameters will experience fairly rapid growth over the next few years, some improving system performance and others degrading it. For example, performance is improved as network bandwidth, storage capacity, and processing power increase, while the growth in the volume of data and numbers of users and sources tend to degrade it. However, we expect that factors relating to computer-human interaction such as query response time and browsable result size cannot drastically change if a resource discovery system is to continue to be useful. Therefore, it is important to recognize which models are likely to experience future scalability problems based on parameter dependencies. We ignore the Class 1 models in this section since they were already problematic with the previous parameter values.

First, we change the three model parameters which most closely relate to the number of users, the number of sources, and the data volume: N_{QG}, N_S, and N_D, respectively. We increase the first two by two orders of magnitude and the third by one, yielding approximately 10^8 query generators, 10^7 sources, and an average of 10^5 documents per collection. These changes require changes in seven other parameters: N_{RS}, S_C, S_{CT}, S_{CTS}, S_{CTI}, S_{HLI}, and S_{MLI}.

We now allow for two cases of the models to handle these increases. In the first, Case 1, we keep the number of intermediaries the same; in the second, Case 2, we increase them. Increasing the number of intermediaries in Class 2 is a simple matter of increasing N_{II}, which we do by two orders of magnitude to 10^4. For Pharos, we have two coordinated sets of intermediaries. The mid-level servers are the most similar to the intermediaries in the other models. Thus for Case 2 we increase N_{IM}, the number of times each unique mid-level server is replicated, by two orders of magnitude to 10^2. The high-level servers, however, act like news servers and are expected to serve a local area network. Moreover, any replication of this type of service within a local area is irrelevant to our model, which is looking only at remote traffic. Therefore, we increase the number of high-level servers, N_{HLI}, only one order of magnitude to 10^4; that is, we roughly expect that the increase in the number of high-level servers grows approximately as the square root of the growth in the number of users.

The results are shown in Table 5.7. This table shows the re-computed estimates of all values in Tables 5.3–5.6 as a result of the increases in N_{QG}, N_S, and N_D. For each model, the column labeled 'Orig' shows the values from the original tables. The column labeled 'Case 1' shows the Case 1 values, where the number of intermediaries is left as it was. The column labeled 'Case 2' shows the values for which the number of intermediaries was increased as previously discussed. Certain scalability problems become evident in this table. For example, because the metadata in BF and STARTS grows with the size of the collection as well as with the number of intermediaries, the amount of traffic generated by a single update grows by as much as three orders of magnitude. In Pharos, the increase is dominated by the extra number of high-level servers, and so the result is much smaller: only one order of magnitude. Pharos also scales well with respect to storage; this is not only because, as before, the metadata size per source is constant, but also because only a portion of it is sent to any particular intermediary. Finally, STARTS scales the best for the number of update accesses, since it requires only one access between each source and intermediary, and has relatively few intermediaries.

	BF			STARTS			Pharos		
		New Values			New Values			New Values	
Estimate	**Orig**	**Case 1**	**Case 2**	**Orig**	**Case 1**	**Case 2**	**Orig**	**Case 1**	**Case 2**
Network Traffic (Tables 5.3):									
Query	7	9	9	6	6	6	6	6	6
Update	10	11	13	8	9	11	6	6	7
Metadata Storage (Table 5.4):									
Avg	11	14	14	11	14	14	8	10	10
Total	13	16	18	13	16	18	11	13	14
Query Accesses (Table 5.5):									
Avg Int	4	6	4	4	6	4	4	6	4
Avg Src	2	4	4	2	2	2	2	2	2
Total	7	11	11	7	9	9	7	9	9
Update Accesses (Table 5.6):									
Avg Int	9	12	12	5	7	7	5	7	7
Total	11	14	16	7	9	11	8	10	11

Table 5.7: Estimates for Large Parameters (all values are $\log_{10}(x)$)

5.4 Discussion

This chapter presented a classification of network architectures for locating information sources. These models have been grouped into three broad classes: Direct, Single Intermediary, and Multiple Intermediaries. We discussed the relevant parameters for these models, derived estimates for several of their characteristics, and compared them based on these estimates. Each class has been shown to have certain scalability characteristics for all its models. We believe that each increase in design and processing complexity among the classes and models is warranted by the corresponding increase in scalability. In general, the estimates we derived indicate the need for collection metadata whose size is not a function of the size of the collection. Moreover, utilizing multiple intermediary models that rely on metadata summaries can avoid a linear growth of metadata size with respect to the number of sources. Limiting the growth of collection metadata in these two ways should more easily accommodate the expected expansion of the Internet.

Chapter 6

Simulation Studies

We now complete the feasibility study of the Pharos architecture. The last chapter presented a comparison of Pharos with other models. In this chapter, we estimate the accuracy of Pharos query results.

6.1 Simulation Parameters

In order to evaluate the expected performance of Pharos, we tested it by generating simulated sources and queries and estimating the degree of success of finding the 'best sources' for each query.[1] Such an estimate requires the quantification of a few parameters. When dealing with a standard, single text document database, *precision* usually gives a measure of how many of the returned docu-

[1]For the simulations, we define 'best sources' as the sources with the greatest number of documents associated with the same node in a taxonomy as the query node.

ments are considered to be relevant to the query [Har95]. This definition does not extend naturally to the problem of locating sources. A source could be considered relevant if it contains even a single relevant document, resulting in a relevance test that is too broad and unintuitive. Instead, we are interested in a measure that is higher for sources that contain a relatively larger number of relevant documents. For simulation simplicity, we consider measures similar to the weights associated with documents based on vector analysis [Dum91, Sal89], which lead to a ranked list of documents. Assuming that we have a weighting system which will lead to a ranked list of sources, we have several options for defining the precision of the results. We first introduce two such definitions and show how they are problematic, leading us to a final, more intuitive definition. First, we could define source precision of a final set of k sources as the fraction of the k *best* sources; however, this gives too small a value when, for example, the $(k + 1)$ best source is substituted for the k^{th} best source in the final set. Next, if we define source precision so as to include the value of such sources, we might use

$$\frac{\sum_{i=1}^{k} i}{\sum_{i=1}^{k} est_i}$$

where est_i is the true rank of the estimated i^{th} best source. The resulting value would be unreasonably small, however, whenever a particularly poor source is selected, even if the other $k - 1$ sources are the best ones. Therefore, we define *source precision*, P_S, as follows:

$$P_S \equiv \frac{\sum_{i=1}^{k} 1/est_i}{\sum_{i=1}^{k} 1/i} \tag{6.1.1}$$

where est_i is the true rank of the estimated i^{th} best source, and k is the number of sources selected in the final set. Clearly $0.0 < P_S \leq 1.0$. As an example, if we select the best 10 out of 1000 sources, $k = 10$, and the denominator is $1/1 + 1/2 + \cdots + 1/10 = 2.93$. Suppose that for a given query, we return a set with the following source rankings: {1,3,4,7,9,14,17,20,27,103}. The numerator becomes $1/1 + 1/3 + 1/4 + \cdots + 1/103 = 2.06$. The source precision is then $2.06/2.93 = 0.70$. The first definition would have yielded 0.50 instead, and the second definition would have yielded 0.27.

The next parameters we need to quantify are the source weightings. Pharos is designed for a more complex method of source comparison than a linear ranking, as discussed in Chapter 4. However, for the purposes of simplifying the simulation, we restrict comparisons to such a ranking, based on the number of documents at each source which are associated with the query. Thus our current weighting is based solely on estimating which sources have the greatest number of associated documents.[2] A source's *true* weight is different from its *estimated* weight. The former is a function of the actual number of documents and in general is not determinable from a user's machine. The latter is based on

[2]Other methods, for example, involve using a conspectus rating or estimates of how well a source covers a subject, region, etc., as well as using any of the taxonomy-independent metadata (see Section 4.3).

the information available through the high-level and mid-level metadata. These weights are complicated by the fact that we cannot assume that all documents in a collection have been classified within each taxonomy. For simulation and evaluation, we compare the rankings based on the true source weights to those obtained from the estimated weights in order to calculate the source precision. We define the weight of a source for a query involving multiple taxonomies, but simplify it here by excluding multiple nodes within any single taxonomy. Queries are then considered to be one node from each of the one or more taxonomies used in the query, limited by the total number t of taxonomies known to Pharos. We currently limit t to 3. A multi-taxonomy query is considered to be a logical AND of the individual nodes within each separate taxonomy. Furthermore, we assume that documents are uniformly distributed among a query node's children within a taxonomy. Therefore, we estimate that the total number of documents that are relevant to a query is the product of the total number of documents in the collection times the individual fractions of the collection within each query component. For example, consider a two-taxonomy query with a subject component of political music and a geographical component of San Francisco. If 80% of the documents in a collection match the subject component and 30% of the documents in the collection match the geographical component, then we assume that 24% of the documents in the collection match the query. Therefore, we use the product of the coverages to derive the following *true source weighting* for a given query:

$$S_C \ * \ \prod_{i=1}^{t} \gamma_i \tag{6.1.2}$$

where S_C is the size of the collection at the source, t is the number of taxonomies in the query, and γ_i is the coverage at the query node n_i for taxonomy i.

The final parameter we need to define is the *estimated* source weighting. This weight is calculated from the high-level and mid-level metadata. For the taxonomy nodes which define a given query, the corresponding metadata are the high-level and mid-level ancestors of those nodes in the respective taxonomies. If the query node is actually a high-level node, the mid-level 'ancestor' is defined to be the query node itself. We make a slight modification from Equation 6.1.2 to account for the fact that the metadata gives a more accurate estimate of nodes which are higher in the taxonomy than those which are lower. The number of nodes which are represented by a high-level (mid-level) node is a function of two factors: 1) Δh, the difference in height between the query node and its high-level (mid-level) ancestor, and 2) N_c, the number of children per node in the taxonomy. For simplicity, we assume here that N_c is a constant for each taxonomy. The number of descendents of the query node's high-level (mid-level) ancestor which are at the query node's height in the tree is then $(N_c)^{\Delta h}$. In order to give more relative weight to query nodes which are higher in their respective trees, we divide the product factor in Equation 6.1.2 by $N_c * \Delta h$. This divisor is a compromise between just using the height difference and using the full dilution of $(N_c)^{\Delta h}$. Therefore, the high-level (mid-level) *estimated source weighting* is

$$S_C \; * \; \prod_{i=1}^{t} \left\{ \frac{\gamma_i'}{(N_c * \Delta h_i)} \right\} \tag{6.1.3}$$

where S_C is the size of the collection at the source, t is the number of taxonomies in the query, γ_i' is the coverage at the high-level (mid-level) ancestor a_i of the

Tax / Query	**Number of Queries**	**Average Precision**
1	320	0.76 ± 0.19
2	337	0.63 ± 0.21
3	343	0.58 ± 0.23

Table 6.1: Precision as a Function of the Number of Taxonomies per Query

query node n_i for taxonomy i, N_c is the number of children per node in i, and Δh_i is the difference in height between n_i and a_i, or $1/N_c$ if n_i equals a_i. It should be noted that the dilution factor used in this estimate makes no difference for single-taxonomy queries. Equation 6.1.3 is a heuristical equation which needs to be compared with other estimated weighting schemes.

6.2 Simulation Results

Using the performance measures developed in Equations 6.1.1-6.1.3, we ran experiments with 1000 simulated sources. The sources were generated with 3 taxonomies per source, 8 children per node, and a taxonomy depth of 4 (i.e. 4681 nodes per taxonomy). High-level (mid-level) metadata were considered to be the level-1 (level-2) nodes in the taxonomies. Collection sizes ranged from 10^2 to 10^8, with a higher probability[3] given to 10^5. Finally, the coverages of

[3]The exponent was chosen as the random variate by averaging two values taken uniformly between 2.0 and 8.0.

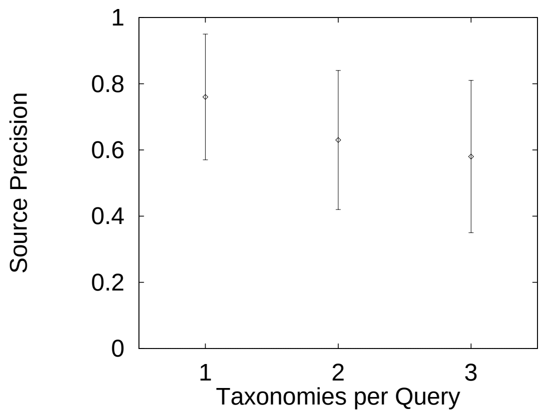


Figure 6.1: Simulation Precision

a node's children were chosen such that they summed exactly to the coverage of that node. However, we increased the variability of the children by using a non-uniform probability distribution. Our experiment generated 1000 queries. Each query was randomly chosen to use 1, 2, or 3 taxonomy query components. Furthermore, the actual taxonomy to use for the first, second, or third component was randomly selected as needed, although we disallowed using the same taxonomy more than once per query. Query nodes were chosen uniformly among all nodes in the tree, giving a higher probability that nodes were chosen from among the leaf nodes. By using Equation 6.1.3, we selected approximately 100 sources with the high-level metadata, and then from that set and the mid-level version of the same equation, selected a final set of 10 sources. For each query, we used Equation 6.1.2 to rank the 1000 sources. Finally, we derived the source precision for each query using Equation 6.1.1.

The overall average precision was 0.66 ± 0.22. In Table 6.1 we show the break-

Query Depth	**Number of Queries**	**Average Precision**
1	0	N/A
2	3	1.00 ± 0.00
3	39	0.84 ± 0.17
4	278	0.75 ± 0.19

Table 6.2: Precision as a Function of Query Node Depth

down of the results based on the number of taxonomies per query; the results are shown graphically in Figure 6.1. In Table 6.2 we show the breakdown of the results based on the node depth for the single taxonomy queries. We also calculated the average best and worst ranked source in the final selected sets. The former is 1.5; the latter is 324. These numbers indicate that in the vast majority of cases, the source which was assigned the highest or second highest true source weight was in the final set, and that on average the worst source in the final set was within the top 32%.

For proper perspective, precision measurements generally must be compared to their corresponding recall values, which we have not yet defined. While work from TREC [Har95] generally shows the state-of-the-art of document retrieval to be around 50% precision for 50% recall, these values are for document query results from single collections. We informally define *source recall* as the fraction of total documents in a sub-domain that are contained in the final set of the selected sources. Clearly this is a non-trivial value to determine. One study by the Research Libraries Group of top research libraries in the U.S.A. indicated that the best source alone can provide well over 70% source recall, and that a few sufficient sources can provide a source recall of over 90% [Mos85].

It is difficult to know how these values might be extrapolated from tens of research libraries to tens of thousands of digital information sources; they do, however, help put the simulation results in some perspective. They indicate the potential importance of finding several of the best sources, as our simulations do for most of the final sets. Considering that each source in a final set, once selected, would be directly handed the query, we find these results reasonable. They indicate that a sufficient number of good sources could be located with the Pharos architecture so that the majority of relevant documents could be retrieved. We note that there is a drop of precision as the number of taxonomies per query is increased; while this result is not unintuitive, further work is needed to more definitively explain it. It is also worth noting that the results for 1000 queries are not significantly different than for only 100 queries, including the standard deviations. This indicates that the results are stable, though clearly more experimentation is needed to clarify parameter dependencies. Finally, the next section presents simulation parameters and results for the case of collections that are only partially classified.

6.3 Simulation Parameters under Partial Classification

We distinguish a document that has been determined not to relate to a particular information hierarchy from one for which no determination has been made at all. Those documents that do not relate to a taxonomy can be considered to fall under a special 'excluded' child of the root node. But we must extrapolate from the *measured* coverage values within the taxonomy (i.e. from those documents that have been classified) to the *expected* coverages of the entire collection (i.e. including those documents that have not yet been classified). A simple way to do this would be to consider as unrelated all those documents not yet classified. Another simple method would be to ignore the fact that not all documents have been classified and to assume that the expected coverages is a simple linear extrapolation of the measured values. Neither of these approaches is likely to give the best estimate of the true coverages of the entire collection within a taxonomy.

Instead, since we have no reason to assume otherwise, we make the *a priori* assumption that the unclassified documents are most likely to be uniformly distributed within the taxonomy. After all, if only a small fraction of documents have yet been classified, it is difficult to accurately extrapolate to the entire collection. Correspondingly, if a large fraction of documents have already been classified, then assuming a uniform distribution of documents for the remainder of the collection will not change the existing coverages significantly. Let α_i be the fraction of classified documents within taxonomy i. For query node n_i

of i, the number of documents which would be counted in its coverage of the remaining $1 - \alpha_i$ documents under a uniform distribution is simply the number of nodes under n_i's sub-tree divided by the total number of nodes in i. We call this the *fractional area* of n_i and denote it as f_i. Therefore, in order to weight the sources for a given (possibly multi-taxonomy) query, we use the product derived in Equation 6.1.2 of Section 6.1 for those documents which have been classified, and a uniform distribution for the remainder of the documents. Thus we derive the following *true source weighting* for a given query:

$$S_C \ * \ \prod_{i=1}^{t} \Big\{ (\gamma_i \alpha_i) + \big[f_i * (1 - \alpha_i) \big] \Big\} \tag{6.3.1}$$

where S_C is the size of the collection at the source, t is the number of taxonomies in the query, γ_i is the coverage at the query node n_i for taxonomy i, α_i is the fraction of the collection which has been classified within i, and f_i is the fractional area of n_i.

The high-level (mid-level) *estimated source weighting*, then, becomes the following:

$$S_C \ * \ \prod_{i=1}^{t} \left\{ \frac{(\gamma_i' \alpha_i) + \big[f_i' * (1 - \alpha_i) \big]}{(N_c * \Delta h_i)} \right\} \tag{6.3.2}$$

where S_C is the size of the collection at the source, t is the number of taxonomies in the query, γ_i' is the coverage at the high-level (mid-level) ancestor a_i of the query node n_i for taxonomy i, α_i is the fraction of the collection which has been classified within i, f_i' is the fractional area of a_i, N_c is the number of children per node in i, and Δh_i is the difference in height between n_i and a_i, or $1/N_c$ if n_i equals a_i.

By using Equations 6.3.1-6.3.2 in place of Equations 6.1.2-6.1.3, respectively, and assigning a U[0,1] random variate representing the fraction of a collection's documents classified within a taxonomy, the average precision rose from 0.66 ± 0.22 to 0.79 ± 0.16. This new value is misleadingly high. The problem here is that by using a U[0,1] random variate, we are uniformly distributing, on average, half of our data. This tends to smooth out the variability of the simulated taxonomies, which erroneously makes the high- and mid-level estimates better. As an example of an extreme case, we ran 1000 queries against 1000 sources, but used a U[0,0.01] random variate (i.e. 99.5% of the data was uniformly distributed). The result, as expected, is that the precision went to 1.00 ± 0.00; both the estimated and true weightings were ranked solely on the collection sizes. We are currently experimenting with other methods of simulating partially classified collections.[4] Equations 6.3.1 and 6.3.2, however, are still accurate.

[4]In particular, we use two trees for each taxonomy: one for the true weights and the other for the estimated weights.

Chapter 7

Metadata Extraction: Prototype and Evaluation

Given that the results of the feasibility studies described in Chapters 5 and 6 were acceptable, we could then proceed to begin implementation of a prototype. Arguably the most difficult, yet important, component of Pharos is that which is responsible for the automated extraction of collection metadata. As will be described in detail below, we experimented with the use of newsgroups as collections. We built an initial prototype that automatically classified and summarized them within the Library of Congress Classification (LCC).[1] The prototype used electronic library catalog records as a 'training set' and Latent Semantic Indexing (LSI) [Dum91] for information retrieval (IR). We used the training set to build a rich set of classification terminology, and associated these

[1]The prototype can be tested at http://pharos.alexandria.ucsb.edu/demos/.

terms with the relevant categories in the LCC. This association between terms and classification categories allowed us to relate users' queries to nodes in the LCC so that users could select appropriate query categories. Newsgroups were similarly associated with classification categories. Pharos then matched the categories selected by users to relevant newsgroups. In principle, this approach allows users to exclude newsgroups that might have been selected based on an unintended meaning of a query term, and to include newsgroups with relevant content even though the exact query terms may not have been used. This work is extensible to other types of classification, including geographical, temporal, and image feature.

7.1 Source Summarization: Methodology

Hierarchical classification is fundamental to the way that Pharos uses metadata for collection summarization and selection. Pharos is dependent on each source extracting and distributing this information about its collection. For classification to be practical as an aid to distributed source selection, it must be an automated procedure at the source. Since we wanted to show that automated classification could be successfully applied, we processed our collections (e.g., newsgroups in our experiments) in the same manner that we would expect to happen at individual information sources. Geographical classification within a spatial database, where each document has spatial coordinates associated with it, is straightforward. However, classifying semi- or un-structured digital text within a subject classification is more difficult. As discussed in Chapter 4, automated text-based subject classification has been shown to be fairly successful, using techniques such as sublanguage terms [LH95], LSI [Hul94], and inverted term lists [Lar92]. These results indicate that at least some subject-based automated classification is sufficiently accurate to characterize sources for comparison purposes, where even order of magnitude estimates can greatly aid in filtering out most irrelevant sources.

In general, automated classification requires several components. First and foremost of these is the collection itself; clearly this must be in a digital form to facilitate content-based classification. The second component is a classification scheme, often a hierarchical tree, which organizes the concepts of a particular information domain. The third is a pre-classified training set of documents, which the system uses to characterize each node of the classification scheme. This characterization is generally some type of abstract space within which classification nodes are placed. The position of the nodes in this space serves to specify syntactically the semantics of the nodes. For example, such a space may consist of a large dimensional term space where documents are placed based on the term frequencies of their content. The fourth and final component is an IR system.

The IR system serves two purposes. First, it actually builds the abstract space via some type of mapping or index structure, and then places the classification nodes as reference points within the space. The second purpose of the IR system is to accept queries as input and return a set of ranked classification nodes as output, where the ranking is determined by the relevancy of the nodes to the query. This is the step which actually classifies new documents within the classification scheme. While these components are sufficient to automatically classify a single collection, Pharos additionally requires that a summarization, or *profile*, of each collection be built so that multiple collections can be compared. This is accomplished by taking the classification results of each document in a collection and aggregating them into the individual collection-wide profile.

For this experiment, we implemented this abstraction as follows. We used 2500 USENET newsgroups as individual collections, each newsgroup being considered a separate information source,[2] and used the LCC as the classification scheme. We used newsgroups for several reasons. First, they are an easily available source of thousands of different collections. Second, newsgroups are typically uncontrolled, and their content tends to be based on a distributed consensus; in other words, they are messy, consisting of many unrelated articles, 'spams', misspellings, etc.[3] In that sense, newsgroups represent an extreme of digital collection that could be most chaotic. Hence, if we can bring order to this chaos, then it should be easier to deal with more structured digital collections, which are typically administered by a professional (i.e., if we can work with newsgroups, we can work with anything). Third, the name of a newsgroup gives some quick check on the relevance of the returned collections to the users' queries.

For the classification scheme, we chose the top portion of the LCC scheme

[2]The newsgroups included the following hierarchies in alphabetical order: alt.politics, comp, misc, rec, sci, and soc. Only non-empty newsgroups were used, taken from a two-week period.

[3]A 'spam' is a message that is broadcast to hundreds or thousands of newsgroups, generally to advertise some product or service. These spams are more often than not completely unrelated to the central theme of the newsgroups to which they are posted.

(the LCC Outline). We use the LCC Outline because it is a wide, multi-topic hierarchy (at least it is reasonably hierarchical among the 4214 nodes in the upper part of the tree that form the Outline). As a training set, we used 1.5 million electronic catalog records from the UCSB library. These records were in "MAchine-Readable Cataloging" (MARC) format. Each MARC record contains information about a single holding, including its LCC call number (a notation based on LCC, chosen by a cataloger, representing the major topic of the item). MARC records also generally contain a holding's title, descriptions of its subject matter, an indication of its authorship and creation date, and other bibliographic information. The subject matter in MARC records is derived from the controlled vocabulary of the Library of Congress Subject Headings (LCSH). Finally, we used LSI [Dum91] for the IR system, a commonly used IR research tool.

7.1.1 Building an Online LCC Outline

As previously discussed, we require an online classification scheme in order to classify documents automatically. The LCC contains 21 top-level subject categories, such as "Science", "Law", and "Political Science". Each top-level category is assigned a single letter, such as "Q" for Science. Beneath each of these are sub-categories, usually with a two-letter notation; for example, "Q: Science" includes "QC: Physics", "QE: Geology", and ten others. After the two-letter notation, further differentiation is usually denoted by way of numerical ranges. For example, "QC 221-246" denotes "Acoustics, Sound", while "QC

501-766" denotes "Electricity and Magnetism". This hierarchy continues down many levels.

Once we chose the LCC, we could not find an electronic version of it.[4] Below, we describe some of the details of building an online version of the LCC Outline because some of the difficulties of doing so involved the structure of the LCC itself, rather than simply being a result of normal programming problems.

At first, we hoped to use only the first and second letters in the outline, such as "H: Social Sciences", and "HG: Finance". However, it soon became apparent that this would not provide sufficient detail. For example, "BF" is described as "Psychology, Parapsychology, Occult Sciences", putting too many disparate topics together in a single node. Computer Science is buried five levels down in the tree, beneath "QA: Mathematics"; in fact, it is deeper in the tree than "Slide Rules", which is only four levels down. Clearly we needed, as a minimum, the nodes listed in the full LCC Outline.

The development of a "machine-readable" version of the LCC is available from the Library of Congress' Cataloging Distribution Service and is entitled, "Classification Plus" [Gue96]. However, this distribution does not include a programming interface; it is distributed via CD-ROM and accessible only through the user interface provided. As a result, this information was basically useless to us. We located a few versions of the LCC Outline on the Web. These were in a flat format, with one page for each of the 21 major LCC categories. After parsing these pages to remove the HTML and correcting misspellings and numerical

[4]Although the Library of Congress publishes the complete LCC on CD-ROM, it is not built with a programming interface.

range errors, we were left with 21 lists of the form shown in Table 7.1.

While we were able to work around problems with the layout of this data and, in particular, decipher the nesting structure of the letters and numerical ranges, other problems were not as straightforward. Most of the LCC places classification numbers with two alphabetic characters as children of those with one alphabetic character in the tree; in the "K" section this extends to three beneath two. For example, in the above list, "AC" is a child of "A" in the tree. Similarly, "KGC" is a child of "KG". However, the exceptions to this rule make the automatic parsing of the data difficult. For example, "DAW" is a child of "D", placed between "DA" and "DB". "E" has no other letters beneath it at all, just numbers. Most top-level categories repeat the one-letter symbol as its first child, so that, for example, "R: Medicine (General)" is a child of "R: Medicine". The "K"s were by far the worst. For example, both "KK" and "KKA" are children of "KJ". Also troublesome is that several nodes in the tree were completely unlabeled and left out, so that a parent node and its grandchildren exist without an intermediate level being explicitly noted. In order to impose a somewhat rigorous tree structure, all these situations had to be manually discovered and corrected, including not only the data, but also the classification scheme itself.

7.1.2 Building an LCC Vector Space

After constructing an online version of the LCC Outline, we next needed to construct a relationship between terms and the LCC categories. For example, the category "RJ 1-570: Pediatrics" might be associated with terms such as "chil-

A General Works

AC 1-999 Collections, Series, Collected Works

1-195 Collections of Monographs, Essays, etc.

801-895 Inaugural and Program Dissertations

901-995 Pamphlet Collections

999 Scrapbooks

AE 1-90 Encyclopedias (General)

. . .

B Philosophy, Psychology, Religion

B 1-5739 Philosophy (General).

108-708 Ancient

. . .

Table 7.1: LCC Outline Data Format

dren", "hospital", and "measles", while the category "QA 75.5-76.95: Electronic Computers, Computer Science" might be associated with "database", "algorithm", and "cryptography". Although we chose to use a vector space model, which follows closely the techniques used by LSI in TREC [Har95], any IR system could have been substituted, as explained below. We present a more detailed description of the process of constructing such a term-category association using MARC records and LSI.

The general design of this component is shown in Figure 7.1. We want to associate each of the 4214 categories (nodes) in the LCC Outline with a set of terms that are representative of that category. In order to build such a term representation, we begin with a training set, such as MARC records. Each record is processed by a program ("Vector Builder") that extracts the appropriate terms from it and assigns them to the corresponding LCC category. We end up with a representative 'document' for each LCC category ("LCC Vectors") consisting of the terms from all the MARC records that were associated with that category. These LCC Vector documents are used as input to LSI. LSI constructs a high-dimensional vector space such that each unique term is represented by a different dimension. It places each LCC category as a point, or vector, in this space. Each component of a vector is determined by the number of times that the term for that dimension is used in the category's term-based representation (adjusted by various term weighting schemes). LSI then employs singular value decomposition to reduce the dimensionality to approximately 100 dimensions whose orientations are some linear combination of the original axes. LSI maintains an index of all the terms with their position vectors in the reduced

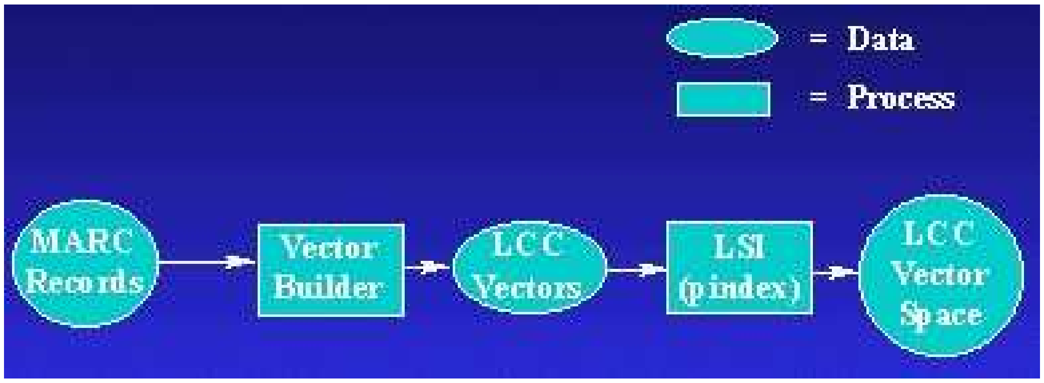


Figure 7.1: Building an LCC Term Vector Space

space. The LCC categories can then be viewed as items in a collection that LSI attempts to retrieve based on their relevance to queries. The end product of this component, the "LCC Vector Space", is used by LSI in both classifying newsgroups and in processing queries.

Suppose, for example, that instead of using an LSI-based vector space, we used an inverted term index, as does Cheshire[LMO+96]. We would need this index both at the source, to build the profile, and at the client, to handle the subject query-mapping. We would not, however, need to retrieve this index at query-time. The acquisition of the index (or LSI space, etc.) is required only once. In other words, Pharos is independent of the particular IR system used. As discussed in [IT95], other IR methods may have certain advantages over the vector-based LSI approach. As there are advantages and disadvantages with any IR system, and our work is not focused specifically on IR, LSI was sufficient for our purposes.

We prefer LSI because it indexes based on term groupings, rather than on individual terms. Thus “investment” might be associated with documents which contain “bank” and “finance” even if the word “investment” does not actually appear in all those documents. Also, LSI is language-independent, so it works equally well with both English and non-English terms. For example, by default LSI uses no word stemming, a technique that is generally highly dependent on language syntax such as pluralization rules. One big disadvantage of LSI is that it is not capable of working with multi-term phrases; each term is considered independently. This is not an inherent problem with vector-based IR, but simply with the implementation employed by LSI. By default, when LSI builds its term vector space, it first removes the following:

- punctuation;
- words of less than three characters;
- numbers not of the form 18nn or 19nn;
- entries in its standard stop-word list of 437 common words.

We added “journal” and “proceedings” to the stop-word list, as these words are common in MARC records and do not help much in differentiating between subjects. We also included hyphens as allowable punctuation, and kept numbers.

Building the relationships between the terms and the LCC nodes requires a training set which associates terms with LCC categories. One way of doing this is simply to use the descriptive text associated with each category, such as "Pediatrics" associated with "RJ 1-570" as described above. While we did incorporate these terms, they did not provide a rich enough set of words. In order to enhance this set, we used the 2 million MARC records for the items held at the UCSB library (of which 1.5 million were usable). MARC record format[5] is a national standard for the exchange and distribution of cataloging data, backed by the Library of Congress.[6] These records contain a series of numerically tagged fields and alphabetic subfields. For example, field 050 contains the LCC call number assigned by the Library of Congress. Within the subject fields, the 600's, subject terms are assigned from the Library of Congress Subject Headings (LCSH). Subfields for the 600's include, for example, $y for temporal information and $z for geographical information. From each record, we extracted the LCC number, title, and subject heading information. We do not take duplicate terms from within the title nor from within the subject headings, so that each MARC record can add a given term no more than twice. This restriction is intended to base term frequencies on term occurrences between many documents, rather than, for example, repetitive uses from multiple subject headings for the same document. We take hyphenated words as a special case. For example, the term "anglo-saxon" will generate the following terms: "anglo-saxon", "anglo", "saxon", and "anglosaxon". This last form is for including non-hyphenated versions of words such as "on-line" and "e-mail".

We assign the terms from the title and subject heading fields to the LCC cat-

[5]http://lcweb.loc.gov/marc/bibliographic/ecbdhome.html

[6]Technically, MARC is an international standard, and USMARC is the U.S. version. We work only with USMARC records.

Frequency	Term
141	children
101	child
67	infants
66	pediatrics
65	health
43	mental
42	feeding
30	nutrition
29	care
27	breast

Table 7.2: Ten Most Frequent Terms for "RJ 1-570: Pediatrics"

egory associated with the LCC number. In so doing, we acquired over 410,000 unique terms for the 4214 node LCC hierarchy. On average, there were 371 MARC records per node, with a median of 43; 414 nodes had zero MARC records. For example, there were 229 MARC records that were placed directly into "RJ 1-570: Pediatrics", as well as 1679 records distributed among its 14 children. These 229 records generated, after removing duplicates within each record, over 2600 terms. These consisted of, after removing stop-words, over 700 unique terms, including the ten most frequent terms shown in Table 7.2 in decreasing order by frequency.

The distribution of the 1.5 million MARC records among the nodes in the LCC

Outline, shown in Table 7.3, follows roughly Zipf's rank-frequency law [Sal89]. That is, first group LCC nodes which are assigned the same number of MARC records per node, M. Then sort these by M and number them in order, assigning a rank number, R. If we count the number of nodes, $N(R)$, in the LCC Outline which are placed in the same rank (i.e. received the same number of MARC records), we find that for the 1254 nodes which received less than 10 records each, the following holds: $R * N(R) \simeq 450 \pm 60$. In fact, averaging over all the 915 rank values yields $R * N(R) \approx 714$. This average is pushed up by the long tail of the curve, with 512 nodes having a large but unique number of MARC records, and thus each receiving a large, unique rank value.

Given that so many nodes had very few MARC records, we could increase this set of terms by using MARC records from, for example, the UC-wide Melvyl online catalog system. This sets contain tens of millions of MARC records from a variety of different types of libraries, and so we would be able to greatly enhance our term-based representation of each LCC category.

7.1.3 Automatically Classifying the Newsgroups

Once we have constructed the LSI term vector space, we use this data to characterize newsgroups within the LCC and use the resulting collection profile as required by Pharos. Each newsgroup, which is treated as a separate collection, requires its own profile. A profile is compiled by processing the individual news articles (treated as 'documents') within each newsgroup. The articles are passed as queries to (MARC/LCC seeded) LSI, which returns a ranked list of LCC cat-

Rank	**Number of MARC Records per Node**	**Number of R Nodes**		**Node Description**
R	M	$N(R)$	$R * N(R)$	**(if only one node)**
1	0	414	414	
2	1	178	356	
3	2	139	417	
4	3	103	412	
5	4	96	480	
6	5	90	540	
7	6	69	483	
8	7	63	504	
9	8	58	522	
10	9	44	440	
...	...	...	...	
913	36011	1	913	Spanish Literature
914	61986	1	914	English Literature
915	90067	1	915	American Literature

Table 7.3: Distribution of MARC Records within the LCC Outline

egories for each article. In effect, this procedure treats the article as if it were a query and automatically classifies the query – or article – into the LCC. The articles are then compiled into a profile in the form of an LCC tree where each node contains the percentage of articles in that newsgroup associated with that node in the tree.

We first pre-process each article, mainly to strip its headers (and remove punctuation). The only header information we keep is the content of the subject line, which in principle the author wrote intentionally to describe the message's content. The reason that we strip the headers is to avoid using the name of the newsgroup and the cross-posting groups, which appear in the headers, as an aid to classification. In this way, we attempt to be as unbiased as we reasonably can, since the purpose of the experiment is to attempt to classify by content only. We exclude articles that have no terms which match our list from the MARC records; these articles include rare aberrations, less than 0.1% of the articles, such as one article whose subject was "s" and whose entire content was "1".

LSI queries take the form of 'documents' or free text. The terms of the query define a new vector in the LCC vector space in the same manner as the original documents. LSI then returns a weighted list of similar documents. In our case, this list is a set of LCC categories whose weights indicate some measure of relevance between the query terms and the categories. In effect, this procedure automatically classifies the query into the LCC. As shown in Figure 7.2, each news article is given as a query to LSI, which returns the weighted list of LCC

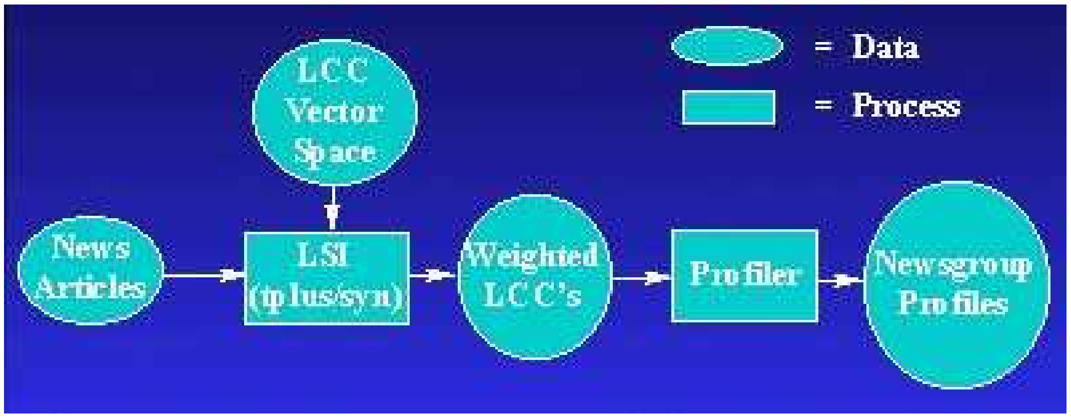


Figure 7.2: Automatically Classifying the Newsgroups

nodes.[7] We then keep all nodes which have a weight above a threshold value (currently 0.25). Suppose we are returned four nodes with the following weights: Node A: 0.8; Node B: 0.8; Node C: 0.4; and Node D: 0.1. We first drop Node D, since its weight is below our threshold value. We then normalize the remaining weights so that exactly 100% of the article is divided between the remaining nodes; thus the article is assigned 40% to Node A, 40% to Node B, and 20% to Node C.

For each newsgroup, we construct a classification-based collection profile in the form of an LCC tree where each node contains the percentage of articles in that newsgroup associated with that node in the tree. As an example, Figure 7.3 shows a collection profile where 19% of the collection falls under the Physics node or its children in the classification, 6% falls under Mechanics, etc. The values in parentheses denote articles which fall under a particular node but not

[7]These values range from 0 to 1 because we are using cosine weighting.

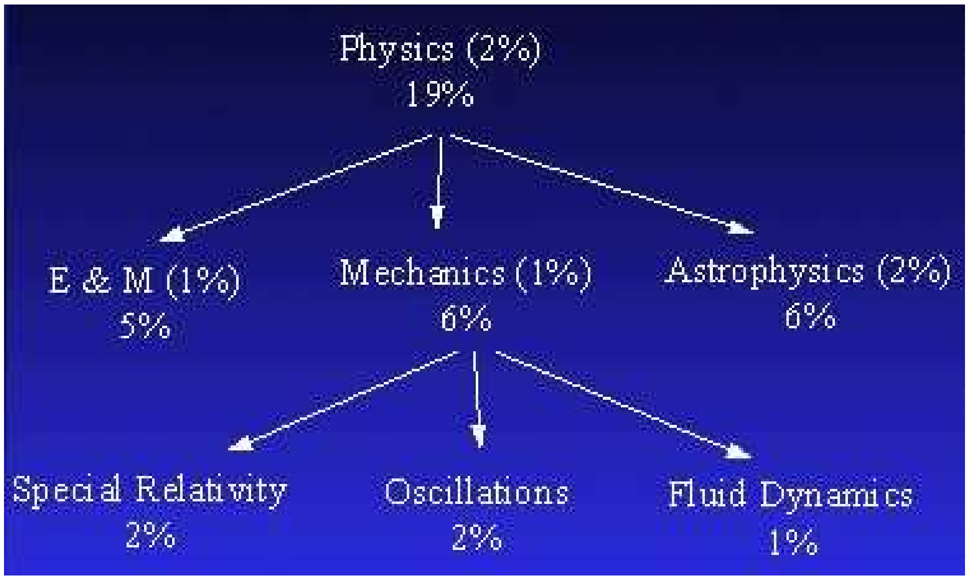


Figure 7.3: Newsgroup Summary: Pharos Taxonomy

under any of that node's children.

We process each article in the newsgroup as above, and then allocate it among the nodes such that 100% of it is added in. If we assume that there are 10 documents in a collection, then each document adds a total of 0.1 to its collection profile. Hence, if we were adding the article from the above example to this profile, we would add 0.04 to the document count of Node A, 0.04 to Node B, and 0.02 to Node C. After processing each article in a newsgroup, we end up with the total number of document equivalents associated with each of the 4214 nodes in the LCC tree. One newsgroup profile is generated in this manner for each newsgroup.

7.1.4 Pharos Query Processing

Given the newsgroup profiles, we must allow users to retrieve those newsgroups that are most relevant to their queries. This requires a semantic mechanism that enables users to map their query concepts into the LCC tree. From here, they can decide which nodes in the tree best represent their search criteria. We accomplish this in two ways. The obvious and perhaps most straightforward way is to provide users with an online version of the LCC and allow them to walk up and down the tree until they find the correct node. But as has been pointed out in the literature [Dum91], this is a difficult process for the user. For example, if someone is looking for the subject of prostate cancer, there are relevant nodes in distant parts of the tree. These include not only “surgery” and “internal medicine”, both beneath “medicine”, but also “immunology” and “anatomy”, beneath “science”. Thus, using the classification scheme effectively requires a more sophisticated understanding of its structure than most users, particularly casual ones, typically possess. Clearly, a more effective approach is required.

We, therefore, provide a more sophisticated mechanism of searching the LCC, as outlined in Figure 7.4. We first map the user’s query terms into the previously constructed LCC term vector space, and then return to the user a ranked list of nodes in the tree (LCC categories) that receive the highest weighting from LSI. These categories are linked to the online LCC Outline in the interface, so that the user can then navigate the specific parts of the tree that are likely to be relevant to the query. This component of the UI is similar to the work of

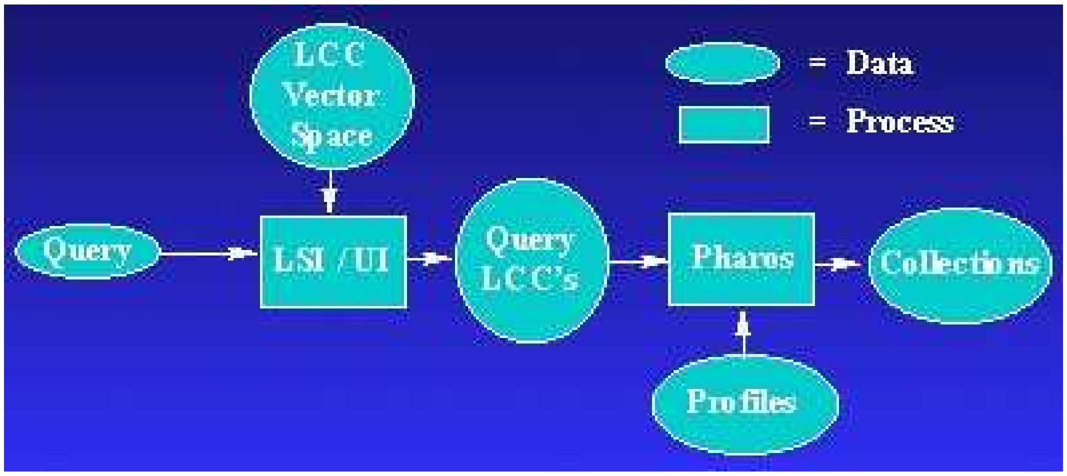


Figure 7.4: Pharos Query Processing

Chen [CLBN93].

Three questions should be re-examined: 1) why, given that the entire newsgroups are available to us online, do we pass queries through an intermediate structure such as a classification scheme? 2) given that we can not use an entire online collection, why use a classification mechanism rather than some other indexing scheme, in particular one which is built directly from the content of the individual collection? and 3) why, given that the entire taxonomies are available to us, do we use only part of that information? The basic answer to all these questions comes down to one main issue: extensive scalability. This experiment uses newsgroups an example of what could easily be over a million dynamic collections; it is not feasible to attempt to store them all locally, even if they were not constantly changing. Second, while using collection-dependent intermediate index structures might be more efficient as a means of retrieving actual documents from individual collections, the accumulated effect of doing so results in very large index sizes and increased complexity. Finally, even using a greatly reduced fixed index structure such as the LCC Outline incurs too great a storage and/or retrieval cost if the entire index must be available for searching at query time. Thus, in a real-world distributed environment with many large information sources, Pharos would process a keyword-based query by first helping a user map it to an appropriate classification tree (or trees) and then, within that tree, map it to relevant query nodes. This leads to the selection of appropriate collections for more direct searching.

Within the prototype, once the user has selected a node in the LCC tree, we then wish to return an appropriate set of newsgroups. This function of Pharos, in the simple version used in the current prototype, linearly ranks the newsgroups based on their collection profiles. The Pharos client would select the appropriate high-level and mid-level ancestor nodes of the query node in the tree.[8] It would then send the high-level node as a query to a high-level server available to each client within their local area (similar to a news server). This server would sort the collections based on the different weighting schemes (discussed below). It would then send back to the client one ranked list of sources for each weighting scheme. The user (or client) would then select a greatly reduced sub-set of the total number of known sources (via an appropriate graphical user interface), a reduction of, say, from $\sim 10^6$ to $\sim 10^3$.[9] The client would then send the mid-level node and the reduced lists to the mid-level server responsible for the mid-level

[8]If the query node is actually a high-level node, the mid-level 'ancestor' is defined to be the node itself.

[9]The reason that mid-level nodes are selected from the high-level results relates to multi-classification querying, as discussed in Chapter 3.

node. This mid-level server would pass back to the client the source lists it had received, but now sorted based on the weighting values for the particular mid-level node. The user (or client) would then select from these lists the final set of 10-100 sources to query directly, potentially independent of the original query node or classification structure. We present an example of this query mechanism in the next section.

We currently use three different weighting schemes. The first is to use the *absolute* counts of documents among all the profiles. Thus the newsgroups with the largest numbers of articles for the selected node are given the highest weights. However, we find that newsgroups with large numbers of articles tend to dominate this list, even if they are fairly irrelevant overall. So we also weight the newsgroups based on the *relative* counts. That is, we give the highest weights to those newsgroups which have the largest percentage of their articles contained in the query node. A third, more sophisticated *combined* weighting algorithm, multiplies the log of the absolute count (plus one) with the relative count. This is one of many possible ways of attempting to return those newsgroups with both large relative and large absolute document counts.

7.2 Source Selection and Evaluation

Once we have constructed Pharos taxonomies for each newsgroup, we need to evaluate the use of these taxonomies for source selection. Before examining the accuracy of the automated classification, we first demonstrate the use of these profiles in processing Pharos queries. Then, assuming for the purposes of the experiment that the classification is generally sound, we investigate the effectiveness of the taxonomy summaries as an indicator of relevant newsgroups. Such an evaluation seeks to determine how well the upper parts of the collection taxonomies estimate the lower parts of the taxonomies. This is important because the Pharos architecture widely distributes only the top parts of the taxonomies for a first-round filter (i.e. via high-level servers), then the next level down for a finer-grain filter (i.e. via mid-level servers), etc. If the upper parts of the taxonomies are not accurate indicators of the lower parts, Pharos will end up selecting incorrect sources. Thus, it is important to evaluate the accuracy of the Pharos multi-level querying mechanism.

7.2.1 Representative Query Results

As an example of a successful query, we show the result of querying the prototype with the keywords "prostate cancer". The top ten LCC nodes returned by the system are shown in Table 7.4, in decreasing order of relevance. In this case, the system was able to match the query terms quite accurately to nodes in the classification tree. These nodes are distributed beneath four main nodes in the tree: "RM: Therapeutics, Pharmacology", "RC: Internal Medicine, Practice of Medicine", and "QR: Microbiology". Upon choosing, for example, "RC 254-282: Neoplasms, Tumors, Oncology (including cancer and carcinogens)" as the query node, the next step is to select the relevant collections.

The top ten newsgroups selected by sorting based directly on the actual query

LCC ID	LCC Category Description
RM 270-282	Serum Therapy, Immunotherapy
RC 254-282	Neoplasms, Tumors, Oncology (including cancer and carcinogens)
QR 189-189.5	Vaccines
QR 201	Pathogenic Micro-Organisms, By Disease, A-Z
QR 186	Immune Response
QR 186.5-186.6	Antigens
RC 633-647.5	Diseases of the Blood and Blood-Forming Organs
QR 186.7-186.85	Antibodies, Immunoglobulins
QR 180-189.5	Immunology
QR 355-502	Virology

Table 7.4: Top Ten LCC Categories Related to 'Prostate Cancer'

node “RC 254-282” are shown in Table 7.5, in decreasing rank order for each weighting scheme. The majority of newsgroups suggested in this manner are relevant to the query and potentially good sources of information. However, as seen in list of newsgroups under the absolute weighting scheme, newsgroups such as misc.jobs.offered and rec.sport.pro-wrestling are not highly relevant overall. They are included because of the large number of articles in the newsgroup, some of which touch on topics related to the query (treated as two individual word parameters: “prostate” and “cancer”). In any case, although the selected newsgroups are fairly relevant overall, Pharos would not have that information directly available in a distributed environment. It would first use the high-level ancestor of this node, “R: Medicine”, and sort the newsgroups based on that node’s coverage values: the aggregated document count for all of its descendents in the taxonomy, for each of the three weighting schemes. Out of the top, say 250, newsgroups from these sorted lists, Pharos would then re-sort these subsets based on the mid-level node, “RC 31-1245: Internal Medicine, Practice of Medicine”. From this final sorting, we might select a final set of, say, 25 newsgroups for each weighting scheme. The top ten newsgroups selected by sorting based on high-level and mid-level nodes are shown in Table 7.6, in decreasing rank order for each weighting scheme. This example shows that in this case, Pharos is able to select many relevant collections. In particular, the highest ranked newsgroups were included in the Pharos estimated lists. As with previous results, however, absolute weighting continues to lead to several less relevant newsgroups.

Relative	**Combined**	**Absolute**
sci.med.diseases.cancer	sci.med.diseases.cancer	sci.med
sci.med.immunology	sci.med	misc.jobs.offered
sci.med.prostate.cancer	sci.med.pharmacy	sci.med.diseases.lyme
sci.med.aids	sci.med.diseases.lyme	sci.med.nutrition
sci.med.diseases.hepatitis	misc.health.alternative	sci.med.pharmacy
misc.health.aids	misc.health.aids	sci.med.diseases.cancer
sci.med.prostate.prostatitis	sci.med.diseases.hepatitis	rec.pets.dogs.health
sci.med.laboratory	sci.med.prostate.prostatitis	misc.health.alternative
sci.med.diseases.als	sci.med.aids	rec.arts.comics.marketplace
sci.med.pharmacy	sci.med.cardiology	rec.sport.pro-wrestling

Table 7.5: True Top Ten Newsgroups, by Weighting Scheme, for Query Node "RC 254-282: Neoplasms, Tumors, Oncology (including cancer and carcinogens)"

Relative	**Combined**	**Absolute**
sci.med.diseases.cancer	sci.med	misc.jobs.offered
sci.med.orthopedics	sci.med.pharmacy	misc.jobs.contract
sci.med.cardiology	sci.med.diseases.lyme	sci.med
sci.med.pharmacy	sci.med.diseases.cancer	rec.sport.pro-wrestling
sci.med.immunology	misc.health.alternative	sci.med.diseases.lyme
soc.support.depression.treatment	sci.med.nutrition	soc.men
soc.support.depression.manic	sci.med.cardiology	sci.med.pharmacy
sci.med.diseases.hepatitis	soc.support.depression.treatment	soc.women
sci.med.psychobiology	sci.med.diseases.hepatitis	rec.pets.dogs.behavior
sci.med.aids	misc.health.aids	rec.food.cooking

Table 7.6: Pharos Estimated Top Ten Newsgroups, by Weighting Scheme, for Query Node "RC 254-282: Neoplasms, Tumors, Oncology (including cancer and carcinogens)"

Windsurfing

Suppose, instead, a user is attempting to find newsgroups related to windsurfing. The query, “windsurfing”, returns with the top-ranked LCC node “GV 770.3–840: Water Sports: Canoeing, Sailing, Yachting, etc.”, which seems appropriate. However, in the lists of newsgroups suggested for this node, `rec.windsurfing` is ranked at most 96 (using the relative weighting scheme). In fact, this node is the top one within which this newsgroup gets classified, implying that our classification scheme is closely associating this newsgroup with this classification node. Even the node “GV 200.6: Water Oriented Recreation” ranks this newsgroup with a maximum of 48 (using the combined weighting scheme). In fact, the only node which ranks this newsgroup among its top ten is “G 540–550: Seafaring Life, Ocean Travel, etc.” The problem is that windsurfing gets lost in the more general topic of water-based recreational activities, even though the first two nodes mentioned are both at the bottom of the LCC Outline tree. In other words, these two nodes are as specific in these subjects as the classification outline gets. This is a problem of a mismatch between the specificity of the query with that of the level of the classification scheme used here. Using a more detailed version of LCC (beyond the Outline) would help in querying collections locally available such as with the prototype. However, it would not help in the distributed environment for which Pharos is designed, where nodes deep in the tree are not available to the high- and mid-level servers. Using more specialized trees would help in both instances, though (for this query, one that specializes in, say, recreation and leisure activities).

Investment Clubs

In another query, "investment clubs", a user was attempting to locate information about clubs that deal with personal financial investment. The query response included the seemingly appropriate LCC node "HG 179: Personal Finance". However, at the time the query was posed, the "misc" newsgroups had not yet been processed, including, for example, `misc.invest`, `misc.invest.-mutual-funds`, etc. Clearly the system can do no better than the content of the digital collections available to it. Once these newsgroups had been included, they showed up prominently in the set of suggested newsgroups. It is not clear though that even the investment newsgroups would be appropriate sources for this particular query. In this case, the system is limited by the sources available, and there is nothing that the system can do about it. This query is important in terms of the standard IR benchmark recall. As pointed out in the Introduction, the user has no way of knowing how many relevant sources may exist. A source that has not been included appears the same to a user as if that source did not exist.

History of Environmental Sciences

Suppose we want information about "History of Environmental Sciences", and by traversing the LCC, select "GE 50: History" (of environmental sciences). We find that the newsgroups suggested have little to do with the history of environmental science, but rather more to do with history in general. There are two reasons for this. The first is that we have no MARC records for this category in our training set, so the only association between the category and terms comes from the descriptive text of the category itself. This leads to the second reason for the problem – namely, that the term "history" is insufficient to describe "history of environmental science". Thus the only term associated with this category is "history". A larger or more diverse collection of MARC records would help prevent this problem. We could also include terms from parents and children of nodes for which we have no MARC records.

Jobs

One problem that occurs with the selection of newsgroups is that weightings which involve the absolute number of articles tend to be overly dominated by very large newsgroups. The suggested newsgroups are currently presented to the user in three columns, corresponding to the three different weighting schemes being used: relative, absolute, and combined ($rel * \log(1 + abs)$). It is apparent after looking over several of the results that one newsgroup completely dominates the absolute weighting: `misc.jobs.offered`. Even though there are on average approximately 400 articles per newsgroup that we have processed so far, the median is much lower, approximately 100. However, there are over 50,000 articles in `misc.jobs.offered`, by far the greatest amount of any of the currently processed newsgroups. Another problem is that this newsgroup is fairly heterogeneous, with job listings related to all areas of the classification. As a result, if there is even a slight relevance of a small fraction of these articles to any topic, this newsgroup can receive the highest absolute weighting. In fact, it often has a weighting one or two orders of magnitude higher than the next newsgroup, and therefore would even show up among the best newsgroups based on the combined weighting scheme. Using a different combined algorithm helps solve the problem. For example, we actually use a minimum relative weighting which is allowed in the combined weighting scheme; thus if a newsgroup does not receive, say, at least a 0.1% relative weighting, it can not be included in the combined weighting scheme. This type of restriction removes `misc.jobs.offered` from most of the LCC categories under the combined weighting scheme, except for those which have an emphasis in the newsgroup. While this problem might seem particular to this newsgroup, it is in fact likely to be a common problem among general digital sources of information, where collection sizes vary widely, as do their degree of heterogeneity. This problem will have to be addressed if we hope to direct users and their queries to sources that they will consider useful.

7.2.2 Classification Accuracy

We now describe an evaluation of the accuracy of the automated classification. The main component of this evaluation is to examine the ability of the system to classify the building blocks of the training set – the MARC records themselves. The second component of the evaluation is an attempt to estimate the accuracy of the news article classification.

Classification Accuracy of MARC Records

Due to size constraints, we first partitioned the 2 million MARC records (of which only 1.5 million were valid for our experiments) into 20 unique groups of 100,000 records each. From four such groups, we randomly selected at least 100 MARC records. This selection was generated differently for each of the four groups. We then removed the bad MARC records, as described below, and, if necessary, re-did the selection if we had to remove so many that we were left with under 100 usable MARC records. After making the measurements described below, we then re-did the experiments using more MARC records from each of the four groups in order to obtain a better confidence level for our values.

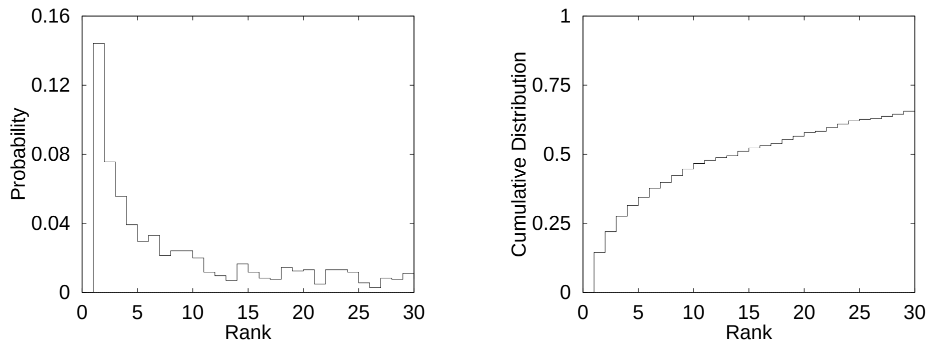
We excluded all MARC records that didn't have a classification category (a 050 or 090 tag). Also, we excluded MARC records for which the category was invalid: either it began with I, O, W, X, or Y (not part of the LCC), or else, other than the K's, excluded three alphabetic characters in a row in the beginning of the category label. We could in principle lose the DAW's (a relatively minor loss), but none of these appeared in the randomized samples anyway.

For each set of MARC records selected, we compared the classification category listed in the MARC record with the categories selected by the automated system. In this case, we used the entire MARC record as the query to LSI. That is, we used all fields and did not limit the number of times that any word in the records could be used. The first value we measured was the location of the true classification category in the ranked list generated by the automated procedure.

For example, the true value might show up 100th in the sorted list of 4200 categories. We then measured the mean, median, and mode of this value for each set of approximately 100 MARC records. A value of 1 means that the automated procedure placed the true category at the top of the list. If we randomly assigned ranks to the categories, we would expect that the mean, median, and mode would all be approximately 2100. A perfect classification would set all the values to 1. The average median came out to 12.9 ± 2.8. The mode was exactly 1. The average mean was 84 ± 19. The average sample size was 118 MARC records.

We then re-did the experiments using more MARC records. We first extracted 2700 records. After throwing out the bad records as described above, we ended up with the following usable number of records from each of the four sets: 2453, 2414, 891, and 1456. Thus we used 7214 MARC records total, with an average of 344 MARC records from each of the 21 major categories of the LCC. The median value was 216 MARC records per category. In this case, the (unweighted) average median came out to 13.0 ± 3.9, the mode, again, was exactly 1, and the average mean was 76 ± 19. Another important characteristic of these values is that approximately 2/3 of the time ($66.7\% \pm 5.1\%$), the best value was within an estimated rank of 30.

As an example of the ranks given to the assigned category, we show their distribution for the fourth set of MARC records. We used this set of MARC records because it contains fairly average values. Figure 7.5a shows the probability mass function, which plots the fraction of records that received a particular rank value. Figure 7.5b shows the cumulative density function, which sums the



(a) Probability Mass Function (b) Cumulative Distribution Function

Figure 7.5: Typical Distribution of Best Classification's Rank

first plot. This plot quickly approaches a value of 0.7 after only rank 30 out of 4214.

The median is more important than the mean for the classification evaluation. The important issue is whether or not a good source will receive a user's query. It either will or will not, and the issue of how close a source *almost* might have received the query is irrelevant to the user (and therefore to the architecture). Suppose that we send the query to the top, say, 15, sources in our estimated list of ranked sources. If, for example, we want the best source to receive the query, it must be among this set of top 15. Then we simply want to know what is the probability that this will be the case. In this case, we get a value of, say, 50%. If the user goes another round, sending the query to the second set of 15, then the probability goes to, say, 70% that the best source would be included. Of course, this is assuming that the rank distribution for the sample of MARC records is roughly the same as for the aggregated source summaries. The conclusion is the same, though. We care about the probability that the best source receives the query, not how far away in the list the best source is if it does not receive the query. This implies that we were able to automatically classify the MARC records with a high degree of accuracy.

Classification Accuracy of News Articles

Classifying MARC records is very different than classifying Internet news articles, and not just because the MARC records composed the training set. MARC records contain specific terms, such as subject headings, that are not only spelled correctly (for the most part), but also rich in meaning. This is quite a contrast to news articles, which, as was pointed out, contain many misspellings, spams, and off-topic comments and discussions.

After performing the previous analysis on the MARC records, we hoped to estimate the accuracy of the news article classification. Unlike the MARC records in the training set, the news articles were not previously classified. As a result, we did not have a 'true' classification category for them with which to compare the classification categories suggested by our system. We therefore hoped to use professional (i.e. human) catalogers from the UCSB library to classify a representative set of articles. Once these articles were classified, we could then compare the human classification with the automated classification to estimate the accuracy of the automated system. Unfortunately, the results of this experiment were inconclusive; they are discussed in Appendix C.

7.2.3 Source Precision

The question we address in this section is how well the selection of newsgroups via Pharos compares to that of selecting newsgroups based on the weights from the actual query node. This set of experiments consisted of selecting a set of nodes from the LCC Outline to use as Pharos queries against 2500 newsgroups, and then calculating the average evaluation metrics from the entire set. In these tests, for each query node we first selected 250 newsgroups using the high-level information which represented that node. That is, we selected the 250 newsgroups which had the highest aggregated weighting at the depth 1 ancestor of the query node. Then, from this set of 250, we further selected a final set of 25 newsgroups using the mid-level information representing that node. We then compared these 25 to the list of newsgroups sorted directly by the weighting of the query node itself. We use the same definition of source precision as given in Equation 6.1.1.

As previously noted, a query of the form “prostate cancer” would be mapped to an LCC node such as “RC 254-282: Neoplasms, Tumors, Oncology (including cancer and carcinogens)” before being propagated within the Pharos distributed retrieval mechanism. Therefore, it is possible to map all possible queries in this experiment to one of the nodes of the LCC Outline. As a result, we can estimate the overall accuracy of queries within this system by averaging the results of all 4214 nodes in the outline.

We show the general results in Tables 7.7 and 7.8. The different rows show the difference between the results of using only the high-level metadata compared

	Absolute	**Relative**	**Combined**
High-Level	0.74	0.44	0.49
Mid-Level	0.85	0.58	0.64

Table 7.7: Average Source Precision as a Function of Weighting Scheme

	Absolute	**Relative**	**Combined**
High-Level	1.4	38.8	10.9
Mid-Level	1.2	14.6	7.7

Table 7.8: Average Best Source as a Function of Weighting Scheme

to the results using both high- and mid-level metadata. The different columns show the results of using the three different weighting schemes. The first table shows the overall source precision averaged over all query nodes. These values can range from 0 to 1, with 1 being the best. The second table shows the average best source averaged over all query nodes. These values can range from 1 to ∼2500, with 1 being the best. In all cases, values improved in going from high-level to mid-level results. For the best source, the mid-level values are all well within the top 1% (i.e. 25) of the 2500 possible newsgroups selected.

For proper perspective, we recall the argument made at the end of Chapter 6. There we stated that, at least for research libraries, best source can alone provide well over 70% source recall, and that a few sufficient sources can provide a source recall of over 90% [Mos85]. To the degree that these values are a result of redundancy across collections, this is probably only minimally exemplary of newsgroups. On the other hand, to the degree that they are a result of large differences in collection sizes, this is likely to be even more the case in newsgroups. In any case, these values indicate the potential importance of finding several of the best sources, as is the case in most of the final sets. Considering that each source in a final set, once selected, would be directly handed the query, we find these results promising. They indicate that a sufficient number of good sources would be located with the Pharos architecture so that a large fraction of relevant documents could be retrieved.

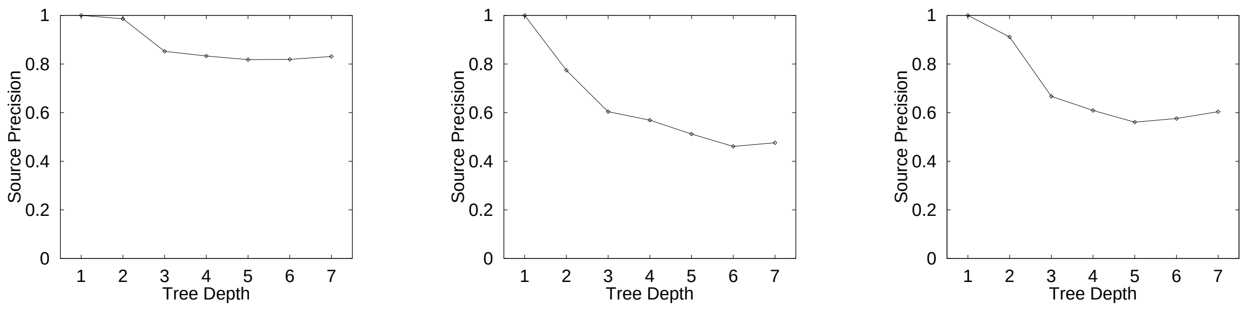
It is worth noting the difference between the different weighting schemes. At first glance, considering these metrics alone would clearly indicate that a preference be given to the absolute weighting scheme. It should be pointed out, however, that the effectiveness of the weighting schemes is dependent on the search capabilities of the information source. If we are selecting sources such as digital libraries, we expect that users should be able to select relevant documents even though these comprise a small fraction of the total number of documents in the collection. For newsgroups, however, this may not be the case. Note that in the 'prostate cancer' example, misc.jobs.offered shows up prominently; this is due to the fact that this newsgroup has 50,000 articles while half the newsgroups have under 100 articles. It would perhaps be challenging to extract the few articles out of the 50,000 that relate to the query. Therefore, even though the precision is lower for the other weighting schemes, it is still likely that users may find better collections using them, depending on the nature of the information sources included in the query set.

We expect absolute weighting to yield better precision than relative weighting due to its decreased sensitivity to minor differences. The spread among the weightings of the selected set of newsgroups using the absolute scheme is generally two orders of magnitude, while the spread using the relative scheme is usually within a factor of two. Thus small fluctuations of, say, 1%, make little difference when considering absolute document counts, but make large difference between relative document counts. This inevitably shows up as a deterioration of the source precision since the sensitivity of the upper levels of the taxonomies is affected by this. Consider the following example. Newsgroup A has 100 articles, the query node has an absolute weighting of 1.5, its parent 10.6, and its only sibling 9.1. Newsgroup B has 1000 articles, the query node has an absolute weighting of 11, its parent 107, and its only sibling 96. According to the absolute scheme, Pharos will select newsgroup B based on the parent node, and, since the actual query node has 11 compared to newsgroup A's 1.5, this is a correct selection. Using the relative scheme, Pharos will still select newsgroup B based on the parent node's 10.7% weight (107 out of 1000), as compared to 10.6% for newsgroup A (10.6 out of 100). In this case, however, the query node in newsgroup A has a higher relative value of 1.5% compared to newsgroup B's 1.1%. This weighting scheme would lead to the wrong newsgroup being selected, lowering the overall precision. Other, more complicated, functional dependencies, such as the depth of the query node and the number of MARC records used, are further examined in the next section. In summary, multi-level querying provides satisfactory result accuracy. However, it is important to compare this accuracy to the scalability requirements.

7.2.4 Functional Dependencies

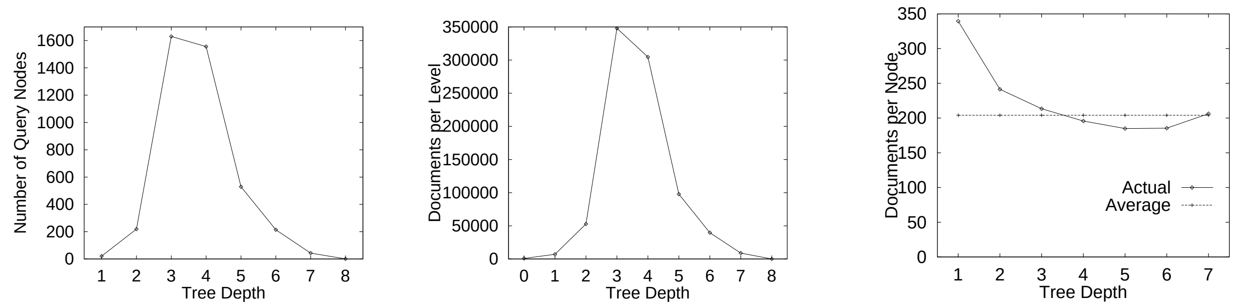
The evaluation metrics are potentially functionally dependent on many factors beyond the source weighting scheme used, such as the size and distribution of the classification training set, the depth of the query nodes in the LCC Outline, the number of documents in the collections, etc. We first show the dependence of source precision on query node depth. Figure 7.6 shows the average source precision as a function of the depth of the query nodes, for each of the three weighting schemes. It is clear from these plots that the depth of the query node in the classification tree has a potentially large effect on the resulting source precision. A query at depth 1 always yields a precision of 1.0 because it is its own high-level node, and thus Pharos always selects collections correctly. Similarly, the depth 2 nodes will yield much better results than nodes deeper in the tree because they are their own mid-level nodes. Since depth 8 has only a single node, we concentrate on depths 3-7. We first show the distribution of total and average documents from the newsgroups as a function of depth. Figure 7.7a shows the distribution of nodes in the tree. Dividing the total, Figure 7.7b, by the number of nodes, we derive the average per node for each level of depth in the tree, as shown in Figure 7.7c. The high-level and mid-level estimates in Pharos are determined by the total number of documents beneath them. Pharos assumes a uniform distribution of documents in making its estimates. If we look at the actual distribution of documents per node, we can compare the expected (average) value to the actual value. This ratio is shown in Figure 7.8. We see that depths 5 and 6 have almost 10% less documents per node than the expected average of 204. As a result, the high- and mid-level aggregate values are less accurate indicators for these levels, and the precision drops. This interpretation is supported by the rise in precision at depth 7, which has within 1% of the expected number of documents per node. It is important to remember that these are general trends across the entire classification tree, and that there can be sizable variability in precision between individual query nodes at the same depth due to peculiarities in the taxonomies, classification tree, etc. As explained below, however, there is another factor that tends to decrease precision with increasing depth.

Another factor we consider is the number of MARC records assigned to each node in the LCC Outline. Approximately 10% of the nodes have no MARC records, and thus the association between a document's content and these nodes is based solely on the textual descriptions within the LCC Outline. Some of these are fairly discriminatory, such as "BF 173-175: Psychoanalysis". Others, however, are poor representations of their semantics when taken out of context of the classification. For example, beneath "GE 1-140 Environmental Sciences" is "GE 50: History", which covers only the history of environmental sciences, not general history. Using this description alone to associate documents with this node would likely lead to classification errors. On the other hand, a node which has hundreds of MARC records assigned to it will involve a large vocabulary with which to relate it to documents, greatly aiding in the automated assignment of documents to that node. While the number of MARC records might clearly impact the effectiveness of the automated classification, it is less apparent how this might affect the query precision. We estimated this by varying our metrics as a function of the training set (TS) size (i.e. the number of MARC records



(a) Absolute (b) Relative (c) Combined

Figure 7.6: Average Source Precision as a Function of Depth



(a) Query Nodes per Level of Depth

(b) Documents per Level of Depth

(c) Documents per Node

Figure 7.7: Depth-related Parameters (Total per Level of Depth)

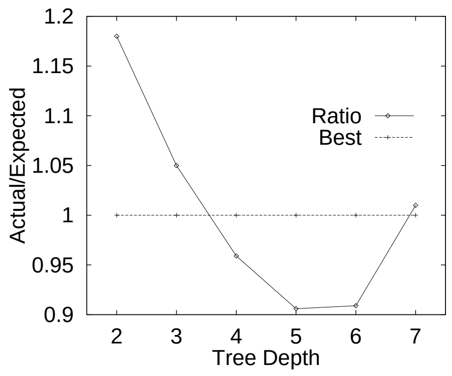
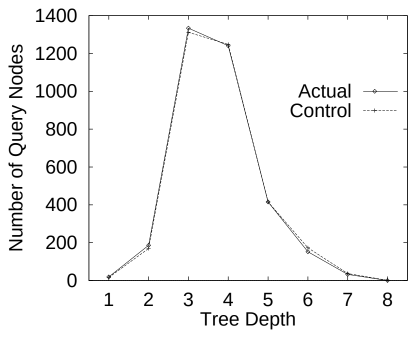


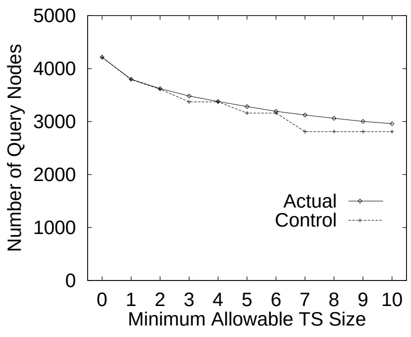
Figure 7.8: Ratio of Actual to Expected Documents per Node

per node).

The initial set of query nodes used was the entire 4214 nodes in the LCC Outline. The metrics were calculated separately for each of the three weighting schemes. We then removed the 414 nodes with zero MARC records and recalculated the metrics. Next, we additionally removed the 178 nodes with only one MARC record, and so on. We continued in this manner until we had removed the 1254 nodes with 9 or less MARC records, leaving 2960 query nodes. Clearly as we removed nodes from the LCC Outline due to an insufficient number of MARC records, we made the query set size smaller. It was possible that any change in evaluation metrics could have resulted simply from varying the size of the query set. Therefore, as a control set, we also removed a roughly equivalent number of elements more randomly and compared results with the actual sets. The control sets were chosen by calculating the largest N such that by removing every Nth node from the LCC Outline during a depth-first search traversal of the tree, we ended up with a smaller set than the actual. For example, the size



(a) Nodes per Level (TS Size > 4)



(b) Total Number of Nodes

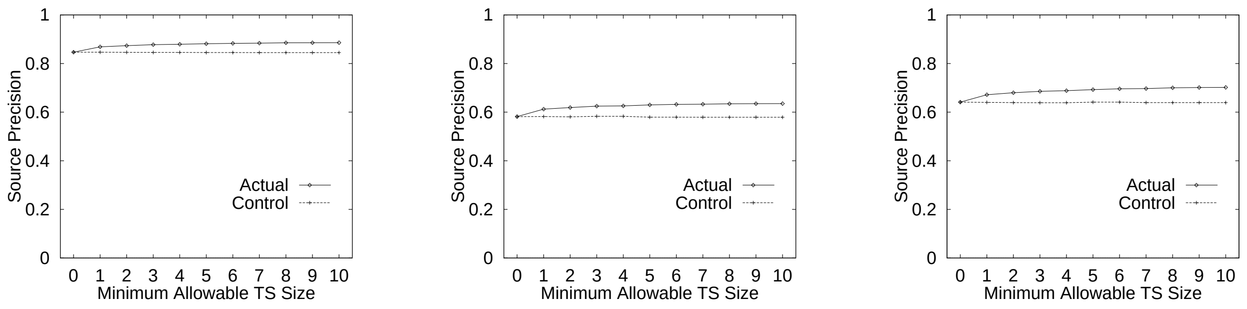
Figure 7.9: Comparison of Actual to Control Sets

of the query set consisting of all nodes with at least four MARC records is 3380. By removing every 5th node, we select a control set with 3372 nodes. These nodes are roughly equally distributed among tree depth as the actual set, but include nodes with arbitrary numbers of MARC records. Figure 7.9a shows the difference in the distributions of query node depths between the control set and the actual set of nodes with at least four MARC records each. It is clear that the distributions are basically equivalent. Figure 7.9b shows the difference in the total number of query nodes between the control sets and the actual sets with varying numbers of minimum training set sizes. These figures demonstrate that the control sets are very similar in nature to the actual sets except that they include arbitrary training set sizes per node.

Once we have established the control sets, we can determine the affect of TS size on average source precision. This is shown in Figure 7.10 for each of the three weighting schemes. The effects of too small a TS size show an overall degradation in source precision of approximately 0.05. Similarly, the effects on the best source are shown in Figure 7.11. These results indicate that for optimizing precision with Pharos-style querying, training sets sizes of at least 4 should be used. In fact, a minimal training set for each node is required for accurate classification anyway, although we not elaborate on that point here.

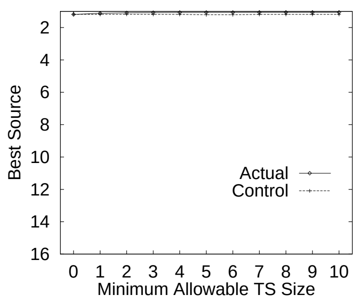
These effects manifest themselves into the relationship between precision and depth. The reason may be seen by looking at Figure 7.12. By dividing the total TS size per level by the number of nodes per level (Figure 7.7a), we see that the average TS size per node drops with increasing depth. As a result, it is more likely that there are query nodes with a small TS size at greater depths in the classification tree. Therefore, removing nodes with a small TS size increasingly improves the precision at greater depths. This trend is evidenced in Figure 7.13, where we compare nodes the TS size > 4 to a control set with roughly the same number of nodes with arbitrary TS sizes.

In conclusion, we have shown a few of the relevant functional dependencies which affect the overall precision of query handling in Pharos. We have also seen that the metrics we have used are at least roughly consistent with our intuitive understanding of the accuracy of query results.

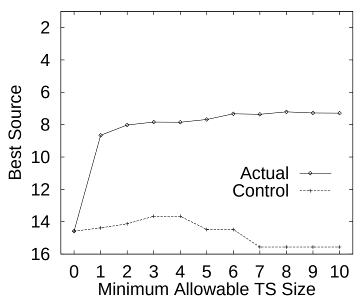


(a) Absolute (b) Relative (c) Combined

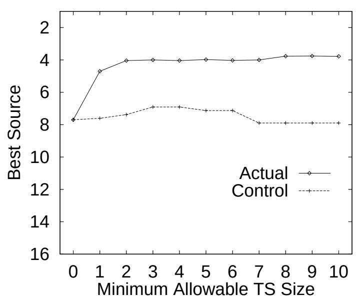
Figure 7.10: Average Source Precision: Actual vs. Control



(a) Absolute

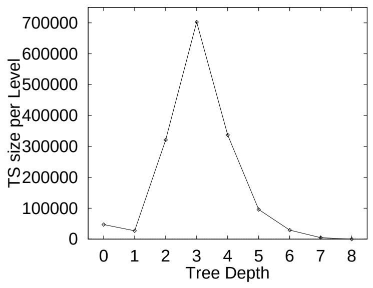


(b) Relative

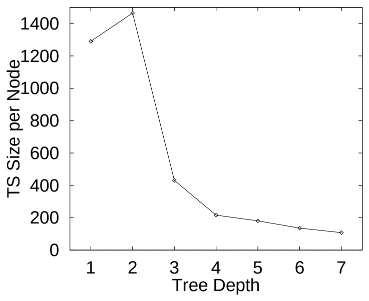


(c) Combined

Figure 7.11: Average Best Source: Actual vs. Control



(a) TS Size per Level

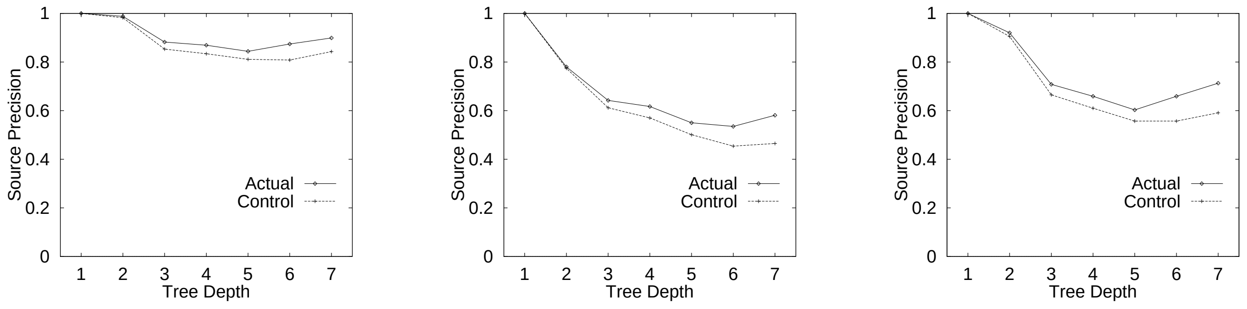


(b) TS Size per Node

Figure 7.12: Depth-related Parameters (Average per Node)

7.3 Scalability Analysis

In this section, we examine the relationship between source precision and resource utilization. Intuitively, one might expect that as we increase the size of collection metadata for each source, we should be able to summarize and select sources more accurately. Clearly this would come at the cost of increased storage and network requirements. Indeed, there are two extreme cases. The first case is to replicate completely all collections at every client. The precision would be high, but the costs of storage and network traffic due to updates are prohibitive. The other extreme case is to use no intermediate information at all, and simply pass all queries to all sources. Ignoring problems of result merging, the potentially high accuracy of this case is prohibited by the costs of network traffic at query time. Therefore, there is a trade-off between attainable query precision and reasonable resource utilization. Such a trade-off is seen, for exam-



(a) Absolute (b) Relative (c) Combined

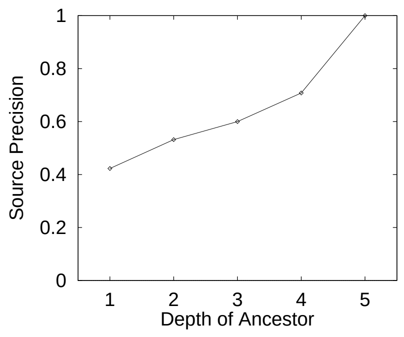
Figure 7.13: Average Source Precision vs. Depth (TS Size > 4)

ple, in a report from the Third TREC Conference [MZ94]. In order to evaluate this relationship in Pharos, we ran two types of experiments.[10]

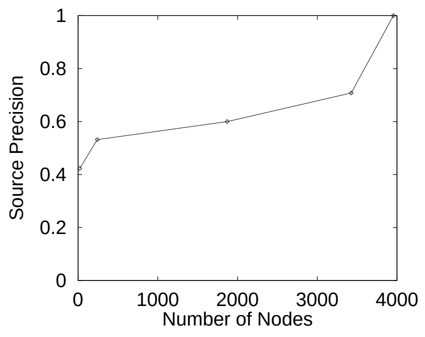
The first experiment ignores the mid-level servers in the Pharos architecture and assumes that all metadata must be sent to all high-level servers. The methodology involves first selecting a query node, Q, in the classification tree, say at depth 5. Assume that its depth 1 ancestor is node A. The true rankings of the sources in this case are based on Q, while the estimated rankings are based on A. We then calculate the source precision using Equation 6.1.1. Next, we use the depth 2 ancestor of node Q, compute the new estimated rankings for the sources, and calculate a new source precision. We continue doing this until A reaches Q in the tree, at which point the source precision, by definition, is exactly 1.0. We then perform the same procedure for all other depth 5 query nodes, and average the results. These results are presented in Figure 7.14a, which shows the average source precision for all query nodes at depth 5, as a function of the depth of the query node's ancestor.

At first glance, it might seem feasible to distribute the entire classification metadata. However, for metadata at the depth of A in the tree to be available to users, we require that all nodes in the tree at or above this depth be distributed to all the high-level servers across the network. The number of nodes in a classification tree grows roughly exponentially as a function of depth. Hence increasing the depth of A, even by one, substantially increases the number of nodes in the metadata. Figure 7.14b shows the same data as Figure 7.14a, but

[10]Results of both experiments are presented using the "relative" weighting scheme. This scheme shows the largest variability in precision and usually yields the most semantically relevant results.



(a) Per Query Depth

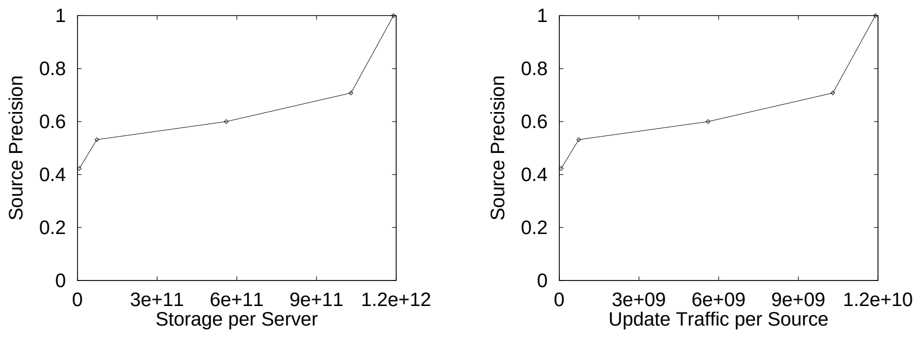


(b) Per Number of Nodes

Figure 7.14: High-Level Only: Average Source Precision for Query Nodes at Depth 5

as a function of the number of nodes at or above the depth of the ancestor. Such an increase in the number of nodes translates to a similar increase in the amount of collection metadata that must be transmitted to and stored at the servers. In order to estimate the resource costs of this metadata distribution, we must first assume the values of a few network parameters.

The storage requirements at a high-level server is equal to N_S, the number of sources, times S_H, the size of the high-level metadata at each source. We assume that N_S is $\sim 10^6$. From Chapter 4, $S_H = S_I + N_C * S_N * N_H$, where S_I is high-level metadata independent of classification summaries, N_C is the number of classification trees used at a source, S_N is the size of the metadata for a single node in the tree, and N_H is the number of nodes from each tree included in the high-level metadata. We assume that S_I is $\sim$ 1KB, N_C is 3, and S_N is $\sim$ 100B. N_H is taken from the data in Figure 7.14b. Figure 7.15a shows the estimated



(a) Server Storage (bytes)

(b) Metadata Update Network Traffic (bytes)

Figure 7.15: High-Level Only: Average Source Precision for Query Nodes at Depth 5

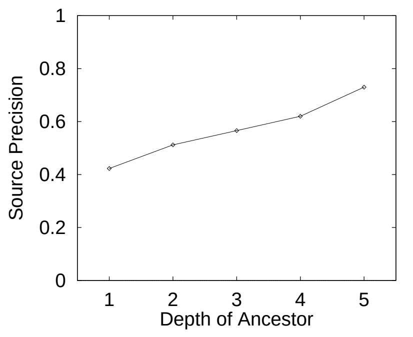
storage requirements for a single high-level server. It is clear that as soon as we go beyond the 21 nodes at depth 1 in the tree, we require at least 100 GB of storage at each high-level server.

Similarly, the network cost associated with a complete metadata update from a single source is S_H times the number of high-level servers. From Chapter 5, we take this to be 10^4 servers. Figure 7.15b shows the estimated network traffic generated by a single, complete metadata update from a source to all the servers. Again, as we go beyond the nodes at depth 1, updates require at least 1 GB of network traffic. As a result, we see that there is a large cost associated with distributing metadata from nodes deeper in the classification tree if we use only the high-level servers.

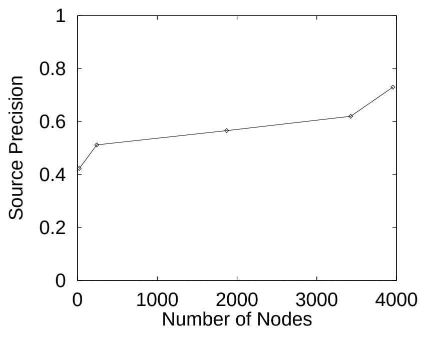
The second experiment incorporates both the high-level and mid-level servers in the Pharos architecture. We again select a query node, Q, in the classification tree, say at depth 5. In this case, however, we select the depth 1 ancestor (high-level node), A1, of Q, as well as the depth 2 ancestor (mid-level node), A2, of Q. We follow the Pharos procedure of first selecting 250 newsgroups of the 2500 based on A1, then selecting 25 newsgroups of the 250 based on A2. Again, we calculate the source precision as before. The next step is to drop A2 down one level in the tree, without changing A1, and re-calculating the source precision. We continue like this until A2 reaches Q. In this case, the best source precision that we can achieve is, in general, less than 1.0. This is because the sources selected based on A2 are a subset of the list generated from A1, and the A1 list does not necessarily include the best sources for Q. We show similar plots to the first experiment in Figures 7.16a and 7.16b, using the depth of A2 for the x-axis.[11]

The storage requirements at a mid-level server is equal to N_S, the number of sources, times S_M, the size of the mid-level metadata at each source. We again assume that N_S is $\sim 10^6$. The major difference here is that the mid-level metadata is not replicated among the mid-level servers. Instead, it is partitioned between them. That is, each mid-level server is responsible for a particular node or set of nodes, and no other mid-level server receives or stores that metadata. Since all the mid-level metadata comes directly from the classification trees, we have $S_M = N_C * S_N * N_M$, where N_C and S_N are the same as before, and N_M is the number of nodes sent to a particular mid-level server. We assume that

[11]For completeness, we show these plots beginning with the mid-level node starting at depth 1, which makes the first value the equivalent of ignoring the mid-level component.



(a) Per Query Depth



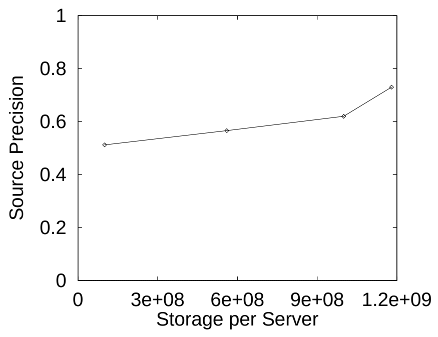
(b) Per Number of Nodes

Figure 7.16: High- and Mid-Level: Average Source Precision for Query Nodes at Depth 5

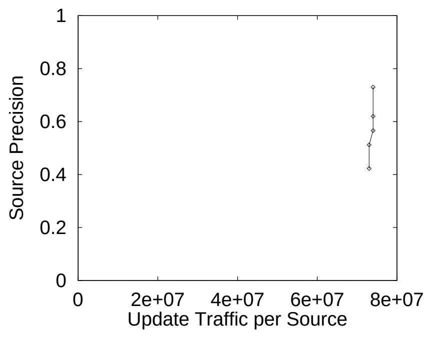
there are a maximum of 1000 mid-level servers.[12] As a result, with N_C set to 3, N_M is a maximum of 12 (that is, 4000 nodes per tree, with 3 trees, partitioned among 1000 servers). Figure 7.17a shows the estimated storage requirements for a single mid-level server. The high-level storage requirements remain at the same constant level as the depth 1 case from the first experiment. We see that, in the second experiment, the storage requirements for the mid-level servers is roughly 1 GB, substantially less than that required for the high-level servers while providing an opportunity of allowing for higher precision query results.

Similarly, the network cost associated with a complete metadata update is greatly reduced as well. In fact, most of this cost is associated with the distribution of the high-level metadata. The high-level is replicated and hence

[12]We do not allow the number of mid-level servers to be greater than the number of nodes used in the mid-level metadata.



(a) Server Storage (bytes)



(b) Metadata Update Network Traffic (bytes)

Figure 7.17: High- and Mid-Level: Average Source Precision for Query Nodes at Depth 5

the associated network traffic is linearly dependent on the number of high-level servers. The fact that the mid-level metadata is partitioned means that the mid-level network traffic is independent of the number of mid-level servers. Figure 7.17b shows the estimated network traffic generated by a single, complete metadata update from a source to all the servers. In this case, as we go beyond the nodes at depth 1, each node's metadata is transmitted only once, and hence it remains fairly small. As a result, we see that there is a relatively small cost associated with distributing metadata from nodes deeper in the classification tree if we use the mid-level servers.

The difference between the first and second experiment is that they incorporate very different network architectures. The design in the first experiment requires that all the metadata be sent to all the high-level servers on the network. The design in the second experiment, however, requires only that the mid-level metadata be partitioned and distributed only once to a single mid-level server. This allows us to keep the high-level metadata relatively small. Even though the original metadata is the same size, the resource utilization is much less. It is also worth noting that since this metadata is based on a static classification scheme, its size is independent of the size of the collection. As a result, it becomes possible to distribute fairly detailed collection metadata without greatly affecting scalability.

Chapter 8

Conclusion

We have presented Pharos, a scalable, distributed architecture for locating heterogeneous information sources. We demonstrated the feasibility of the architecture by first comparing its scalability with other systems, and second by showing the expected accuracy of Pharos query results via simulation. We then constructed a prototype of an automated classification system for extracting subject-based Pharos collection metadata from text collections. Based on the analysis of the classification accuracy, as well as the source precision of the Pharos system, we have demonstrated that Pharos is a feasible model for wide-scale, distributed discovery of information sources.

We decided to use subject classification of text documents in our prototype because it was crucial to demonstrate that the Pharos architecture could work with text and a keyword style query. However, as pointed out in Chapters 4 and 7, automated classification is equally applicable to other classification domains, such as geographical and temporal.

Automated classification is equally applicable to any digital text collection, including web sites, file systems, and FTP text archives. It is an interesting search technique that may add precision to many types of searches at web search engines. If integrated into existing search engines, it could provide another avenue into the mass of documents available for retrieval.

Another interesting result of this work is that job postings were classified. As mentioned, the newsgroup with the most articles was `misc.jobs.offered`, with over 50,000 articles for the two-week period for which we took our snapshot. The classification of this newsgroup indicated what type of jobs were being offered over the Internet during this time period. For example, the four LCC nodes which received the highest weightings in this newsgroup were "HF5546-5548.6: Office Organization and Management", "TS155-194: Production Management", "T58.6-58.62: Management Information Systems", and "QA76.75-76.765: Computer Software". Clearly these are popular positions in the current (1998) job market. This is a simple method of compiling a rough profile on the current job market. Furthermore, applying IR techniques directly to this newsgroup would assist job searchers in filtering the 50,000 job offers to extract ones which more or less meet their criteria. It is also interesting to note that the top categories of `misc.jobs.offered` had a very high overlap with the top categories of `misc.jobs.resumes`.

While this work is clearly important to the development of Pharos, it should be noted that it is also important in the overall development of a digital library framework. This work helps to bring some of the body of knowledge of library science into the electronic environment, and gathers together years of semantic development (the Library of Congress Classification) as an immediately useful classification tool for digital collections.

Chapter 9

Future Work

9.1 Implementation Issues

The implementation of Pharos consists of three main components: 1) metadata extraction at the information sources, 2) metadata distribution between the sources and the intermediate high-level and mid-level servers, and 3) the User Interface.

Metadata Extraction

Given an arbitrary collection at an information source, the high-level and mid-level metadata must be extracted before it can be propagated to the appropriate servers. As previously described, we have already developed a prototype for subject-based classification metadata within a text collection. Developing this into a cross-platform turn-key software module will require substantial software development. Furthermore, the current model is built on LSI, which may or may not be either the best IR method to use or licensable for free distribution. Beyond this type of classification, we also need to develop automated geographical and temporal classification systems. We would like to develop a complete software package for metadata extraction. In addition, we would like automatically to extract metadata from non-textual collections.

Intermediate Servers

Once the metadata at each site has been extracted, it needs to be distributed over the network according to the intended storage and retrieval architecture. As previously stated, the high-level metadata needs to be widely distributed and replicated, while the mid-level metadata is very selectively distributed. High-level metadata consists of the upper parts of the classification-based summaries, as well as source information such as network and usage statistics. This information must be formatted similarly to that described in Chapter 4. The distribution of high-level metadata will be based on the distribution of USENET news via NNTP by having each source post their high-level metadata as a news article. Thus the high-level metadata will be available at any news server wishing to include such metadata information. The distribution of the mid-level metadata is not replicated, and hence a point-to-point distribution scheme is more efficient. Harvest [BDH+94] provides a suitable transport mechanism for distributing and storing mid-level metadata in Pharos. While utilizing NNTP and Harvest for the metadata distribution saves development time and leverages existing technology, there are substantial design and implementation challenges to provide a smoothly running distribution scheme. For example, Harvest requires that data be transferred via *SOIF* records, where each record sent by a source contains the metadata stored at a particular topic-based mid-level server. Another important aspect of the server implementation is the manner in which the clients and sources locate the appropriate mid-level servers. Since this type of information is relatively stable, a single, possibly replicated, mid-level server directory broker, in conjunction with appropriate caching, is sufficient to handle this task.

The servers themselves need to be set up to handle server updates and client requests. The high-level servers need to regularly examine incoming source USENET postings, then extract and integrate new source metadata.[1] Although Harvest provides a framework and tools for the mid-level servers, it does not specify the format of stored data. Harvest provides a client–broker communication protocol for query handling, but does not specify the details required within the Pharos architecture.

User Interface

The User Interface (UI) serves three main functions. First, it interacts with the user, aiding in query formulation and source selection with corresponding user and task profiling. Query formulation requires aiding the user in selecting appropriate taxonomies. For example, if a user query contains the keywords

[1]This information cannot reside in the standard newsgroup directory structure since most news servers delete articles fairly quickly.

"software verification", the UI might suggest to the user that a subject taxonomy oriented around computer science is more appropriate than a general subject taxonomy such as the LCC. A utility such as this is similar in nature to the way that the existing Pharos prototype aids in deciding which classification category is appropriate. The IR techniques used in the prototype could be extended to this problem. Second, the UI is responsible for the query-time metadata retrieval between the client and the intermediate servers. Finally, the UI must visualize the retrieved metadata in a user-customizable manner. Although the UI is currently the least specified component of Pharos, the metadata retrieval mechanism, based on USENET and Harvest, is fairly straightforward. The other aspects of the UI should be able to build off of the latest versions of ADL, for example, which already have both graphical and textual components for user query specification and several types of result display. We would like to integrate Pharos into the ADL system.

9.2 Research Issues

Architecture Enhancements

In order to scale beyond 10^6 sources, the architecture may be enhanced in several ways. For example, the levels of hierarchy can be extended beyond two, up to perhaps four without unduly burdening users. Furthermore, the number of sources may be increased if several sources combine their metadata into single records. High-level metadata would then include multi-source records, while mid-level metadata would separate out the sources as needed. In order to accommodate a growing user base, the mid-level servers could easily be replicated. This feature is already built into the Harvest system and would therefore be fairly straightforward to implement. This replication, handled by the servers themselves, would have little or no impact on the update traffic delivered by the sources.

Extensions from Text to Software

In his Ph.D. dissertation [PD85], Prieto-Díaz lays out the groundwork for a multi-faceted classification system for computer software. Although this framework is poly-hierarchical, it could in principle be extended into the Pharos architecture. This would allow software modules to be retrieved in much the same manner as documents within any other information domain. Extending Pharos to software modules would greatly facilitate the brokering necessary in many current global computing models, which are only beginning to address resource discovery issues.

Enhanced Model Comparisons

In the process of developing the automated classification prototype, we built an association of terms with classification categories. As a result of this association data, we are now able to map keyword queries to subject queries. Such a mapping enables us to enhance the comparisons between networked information retrieval architectures described in Chapter 5. In particular, we can select random terms and associate them with subject categories. Query performance can then be compared between the keyword methodology of STARTS with the classification methodology of Pharos.

Enhancing the Automated Classification

One interesting research area is to attempt to continually update newsgroup snapshots so that the classification is always current. Discussions in the newsgroups vary widely even within a single newsgroup, and so the classification potentially changes regularly. The classification currently takes much too long for this to be feasible. It would be interesting to attempt to partially update the classification for each incoming news article.

Another topic of interest involves the interface into the classification categories. The current interface to this system first maps keywords to subject categories. It should be possible, similarly, to map author and institution information in the MARC records to LCC subject categories as an interesting and useful way of accessing not only Pharos sources, but also general catalog information in a library setting. Once the interface has mapped these into the LCC, we obtain the same scalable retrieval mechanism as before. Beyond these extensions, however, we would like to extract the geographical and temporal subfields from within the subject areas of the MARC records. We believe that this information could be used to construct geographical and temporal profiles which would allow the type of extended, multi-profile searching for which Pharos is designed.

It is perhaps worth noting that once an association between documents and a classification hierarchy has been made, the UI can be built in any language. There is no reason that the query side and the document side need to be in the same language, since they both get mapped into an intermediate tree structure which is independent of either side. The only requirement is the availability of a training set (e.g. MARC records) in the languages of choice.

Appendix A

Example Query Details

We searched for the topic of political music during 1967 in San Francisco, Shanghai, and Cairo. Each search engine requires a different format. Searches were performed on July 1, 1996. We describe each search separately. Because index sites are changing fairly rapidly, all descriptions below can be assumed to be accurate only up through July, 1996, including the number of documents referenced, the index site collection methodology, and so forth.

AltaVista regularly re-indexes the full text of the 30 million documents it references, and follows links to discover new pages. A 'simple search' on AltaVista consisting simply of the string 'political music' resulted in "about 500000 matching" documents which contained either the word 'political' or the word 'music'. While higher ranking was given to documents with both words, this approach was not appropriate for AltaVista. The top ten ranked documents yielded three related to political music: one was in Dutch about a Dutch foundation for political music and the other two were about current music. Changing the search to ' "political music" ' (i.e. double quoted) resulted in a match of 73 documents. The top ten documents included the following: an article from the online version of Mother Jones magazine, which was almost relevant to the original query, entitled "Rock 'n' Revolution: Dave Marsh on 20 years of political music" (the article consisted of a list of 20 songs very briefly annotated); an order form for CD's or descriptions for current bands or soloists; a magazine review site which listed two magazines partially related to political music; a long legal document on Vietnamese taxation which mentioned 'discs for political music programmes' in a sub-sub-section about turnover tax rates; a university dissertation about a 1977 Swedish musical dramatical production group; and the same Dutch foundation that was referenced in the unquoted query results. Several of the links in the set of 73 documents were no longer available, and some links pointed to the same document or copies thereof. Furthermore, the precision of the document set was well below 50%, especially since many documents contained the string "political music" once, but did not actually deal with the topic, and also because many pages were random people's personal home pages. An interesting document from East Tennessee State University described their archives of Appalachia music and included a 'Suggested Reading List'. There was also a brief description of a book entitled *Nineteenth Century Romanticism in Music, 3/e.* While several links were relevant to trends in, and activities around, current political music, there was little in the way of any comprehensive, let alone historical, discussion. Finally, an 'advanced search' of AltaVista was performed using the following search string: '(political NEAR music) AND (196) AND ("San Francisco" OR Shanghai OR Cairo)'. This query resulted in 3 documents, none of which had any bearing on political music or the 1960's. AltaVista does not support subject-based searching.

Lycos regularly re-indexes the full text of the 50 million documents it references, and follows links to discover new pages. The search on Lycos was less successful than on AltaVista. The simple search for 'political music' (unquoted), gave the highest weight to pages with the greatest number of occurrences of either word; the highest ranking went to a page with the word music in it several times but no occurrence of the word political. There were four documents of the top ten that contained both words, two of which dealt with some type of political music. One was a modern American satire group and the other was a publisher's brief summary of a book about a political musician that did not directly mention the exact years and place of the musician's compositions (though it appeared to be early Twentieth Century in Western Europe). Double quoting the phrase, ' "political music" ', did not affect the search results. A 'customized' search that required both words to appear on the same document resulted in 12 documents. Other than the top 4 documents which were included in the top ten of the previous query, the remaining 8 had little if anything to do with political music. The subject category '`Entertainment & Leisure:Music:Genres`' did not have a particular class for political music. The genres '`Pop/Rock/Alternative`', '`Country & Folk`', and '`Other`' had no mention of the word 'political' except for a single link to one artist. No other genres seemed to be related to political music.

Yahoo is more oriented toward subject-based searching than the other two index sites. It does not follow links to discover new pages, but automatically follows announcement site listings for new web sites. Yahoo does not regularly check links to validate that URL's are up-to-date, and does not state the number of pages it has indexed. One relevant topic, '`Arts:Humanities:History:Music`', had nothing about political music of the 1960's, with the possible exception of a link to someone compiling a list of all songs played at every Grateful Dead concert. A search of this section for the word 'political' yielded no matches. While there was no 'Political' topic under '`Entertainment:Music:Genres`', there was a 'Folk' topic; none of the links from this page, however, were relevant. A search of the entire Music section for 'political' resulted in 11 links, none of which were related to the original query. The topic '`Arts:Humanities:History:American History:20th Century:1960s`' had 4 links. Three were not relevant, but one, entitled "Wild Bohemians – an archive of historical information about Bohemian movements in the US during the 20th century," led, through a series of links, to the WWW site of the official "Museum of the City of San Francisco". This site included a chronology of Rock music and political events in the 1960's, and a complete bibliography about the Haight-Ashbury district, with over 60 highly relevant works, including, for example, "San Francisco nights: the psychedelic music trip, 1965-1968". A global search of all of the Yahoo pages for 'political music' resulted in 38 hits which contained both 'political' and 'music'. Yahoo does not seem to be able to specify 'political music' as a single two-word phrase. Furthermore, a Yahoo query searches only a document's title and one- or two-line description; there is no method available for full text searching, as with the other two indexes. None of the 38 hits appeared to be relevant to the original query.

Appendix B

Temporal Information Hierarchy

Table B.1 shows the top-level sub-domains of a temporal information hierarchy. This hierarchy has 33 top-level periods, which cover all past, present, and future time. Since the majority of existing documents have been written during or regarding the last few hundred years, these periods are given a higher level of granularity than, for example, cosmological, geological, or anthropological periods. Such a partitioning of the time-line provides for a more equal distribution of the number of documents likely to be classified within each period.

Period	**Range (year)**	**Period**	**Range (year)**	**Period**	**Range (year)**
1	$< -10^9$	2	$-10^9 \longleftrightarrow -10^7$	3	$-10^7 \longleftrightarrow -10^6$
4	$-10^6 \longleftrightarrow -10^5$	5	$-10^5 \longleftrightarrow -10^4$	6	$-10^4 \longleftrightarrow -1001$
7	$-1000 \longleftrightarrow 999$	8	$1000 \longleftrightarrow 1199$	9	$1200 \longleftrightarrow 1399$
10	$1400 \longleftrightarrow 1499$	11	$1500 \longleftrightarrow 1599$	12	$1600 \longleftrightarrow 1699$
13	$1700 \longleftrightarrow 1749$	14	$1750 \longleftrightarrow 1799$	15	$1800 \longleftrightarrow 1849$
16	$1850 \longleftrightarrow 1899$	17	$1900 \longleftrightarrow 1909$	18	$1910 \longleftrightarrow 1919$
19	$1920 \longleftrightarrow 1929$	20	$1930 \longleftrightarrow 1939$	21	$1940 \longleftrightarrow 1949$
22	$1950 \longleftrightarrow 1959$	23	$1960 \longleftrightarrow 1969$	24	$1970 \longleftrightarrow 1979$
25	$1980 \longleftrightarrow 1989$	26	$1990 \longleftrightarrow 1999$	27	$2000 \longleftrightarrow 2009$
28	$2010 \longleftrightarrow 2019$	29	$2020 \longleftrightarrow 2049$	30	$2050 \longleftrightarrow 2099$
31	$2100 \longleftrightarrow 2499$	32	$2500 \longleftrightarrow 2999$	33	≥ 3000

Table B.1: Top-Level Temporal Information Hierarchy

Appendix C

Evaluation of News Article Classification

The evaluation of the automated classification of the news articles was inconclusive. We discuss the the experiment for two reasons: 1) it is important to show that an attempt was made in this vein, and 2) it is worth noting some of the difficulties encountered when humans are included in the evaluation process.

Since we wanted to use human catalogers in determining the best classification category for news articles, we were very limited as to how many articles we could use in our experiments. We were able to have hand classified only a small fraction of the approximately 800,000 articles we had classified automatically. We randomly selected 42 articles to distribute to three catalogers. Each cataloger received their own unique set of 8 articles. Each pair of catalogers (i.e. three unique pairings of three catalogers) received a different set of 6 articles. Thus each cataloger received 20 articles, 8 unique, and a set of 6 shared with each of the other two catalogers. The overlap was for the purpose of cross-checking the human classification assignments.

For each article, we first asked the catalogers three questions. The first was to rate the degree to which the article was classifiable at all, on a scale from 1 to 5, with 1 meaning 'easy to classify' and 5 meaning 'impossible to classify'. Assuming that they found the article classifiable, we asked them to let us know their degree of competency of cataloging in the subject area of the article (as determined by them). Third, we asked them to determine the best classification category for the article. With each article, we also supplied a list of the 30 top ranked automatically generated classification categories. Given the results from the MARC record classification in the previous section, we assumed that there was approximately a 67% chance of finding the true best classification category for those articles which were found to be easily classifiable. We asked the catalogers to rate each of the 30 categories from 1 to 5, with 1 meaning 'very relevant classification category' and 5 meaning 'completely unrelated classification category'.

The results were inconclusive. First of all, we were only able to get two of the three sets of articles returned to us, leaving at most 34 articles to be hand classified. The next problem was basically our estimate of how long it would take for the catalogers to complete the task. We based our estimates on discussions with one of the three catalogers (cataloger "A") after she had looked over one of the sets of 20 articles. She estimated that it would take approximately 5 minutes per article, thus requiring at most 2 hours for the set. In the end, it appears that performing the tasks we had requested in such a time-frame yields very questionable results, as explained below. On the other hand, the other cataloger (cataloger "B") took much longer on each article, and stopped after processing only 6 articles. Even then, there were sufficient discrepancies that the results were invalidated.

As an example of some of the classification problems, we describe a few articles and their individual results. The first article, from misc.transport.rail.americas (although the cataloger was not told the newsgroup), is as follows:

> Subject: Wheel bearings again
>
> Yet more wheel bearing questions for the group. What is the typical axle load for heavy haul traffic? I.e. coal and mineral trains. How often does hot boxes and collapsed bearings occur on these types of trains? Particularly interested in hearing from experienced engineers with a few miles under their belts.

Cataloger A rated this article as classifiable with a 2 out of 5, with 1 being the best possible. She also rated herself as 3 out of 5 in terms of competency in classifying in this subject area. She then assigned the best LC classification category as "TF 600-606: Railroad Cars (Utilization and Care)", under "TF 501-668: Railway Operation and Management". This is under "TF 1-1620: Railroad Engineering and Operation". One problem with this classification is that it shows that the catalogers went deeper into the LCC than we had intended – we were looking for detail only down through the LCC Outline, which stops at "TF 501-668". Thus the catalogers were taking longer than necessary and decreased the probability that the category they chose would be in our list.

A more serious problem from this article is the rating given to the automatically generated categories. The cataloger gave a value of 3 out of 5 (with 1 the best) to “TF 501-668”, but gave a value of 2 out of 5 to “TF 1-1620”. Clearly this is inconsistent with her own selection of “TF 600-606” being the best, since the category closer to this in LCC was given the lower rating. This type of inconsistency was not uncommon. Another example of this is seen in an article about prostatitis. The cataloger assigned the best category as “RC 899: Prostatitis”, but gave the automatically generated category “RC 870-923: Diseases of the Genitourinary System, Urology” a 5 out of 5 (with 5 the worst).

The final example, from cataloger B, deals with an article about how “New Year cake” is made in Singapore. The cataloger rated this article as classifiable with a 1 out of 5, and rated her competency as 2. She assigned the category “TX 643-840: Cookery”. The automated classification system also gave this category the highest rank. However, when the cataloger rated the top automatically generated categories, she rated this category with a 2, with 1 the best. It is difficult to understand how this category would not qualify for a 1.

In the end, what was clear from this experiment was that including a conclusive user study in this research, even a limited one as we attempted, would have required much more time and effort than was available. We would need to work much more closely with the catalogers and develop a more consistent rating system. We would also need to find a better way to balance the time commitment of the catalogers with the need to have a sufficient number of articles accurately classified.

The evaluation of the classification accuracy of the MARC records gives an indication of the success of this component of the prototype. While definitive results from the catalogers would have been relevant to the evaluation of the prototype, they need to be viewed from the overall perspective of the Pharos architecture. The prototype is one instantiation of an overall framework, which is designed to accommodate a wide variety of data types, classification schemes, training sets, IR systems, and collections. The success or failure of a particular combination of parameter values in the design space is not necessarily indicative of the effectiveness of a different combination. In more general terms, the inter-model comparison (Chapter 5), the simulations (Chapter 6), and the scalability analysis (Section 7.3), are more relevant to the evaluation of the overall architecture.

Bibliography

[ABD$^+$96] D.E. Atkins, W.P. Birmingham, E.H. Durfee, E.J. Glover, T. Mullen, E.A. Rundensteiner, E. Soloway, J.M. Vidal, R. Wallace, and M.P. Wellman. Toward inquiry-based education through interacting software agents. *IEEE Computer*, 29(5):69–76, May 1996.

[ACD$^+$95] D. Andresen, L. Carver, R. Dolin, C. Fischer, J. Frew, M. Goodchild, O. Ibarra, R. B. Kemp, R. Kothuri, M. Larsgaard, B. S. Manjunath, D. Nebert, J. Simpson, T. R. Smith, A. Wells, T. Yang, and Q. Zheng. The WWW Prototype of the Alexandria Digital Library. In *Proceedings of the International Symposium on Digital Libraries*, Tsukuba, Japan, 1995. Also appeared in IEEE Computer, 29(5): 54-60, May 1996.

[ACM96] ACM SIGIR. Workshop on Networked Information Retrieval. http://www.ciir.cs.umass.edu/nir96/, 1996.

[BDH$^+$94] C.M. Bowman, P.B. Danzig, D.R. Hardy, U. Manber, and M.F.

Schwartz. The Harvest information discovery and access system. In *Proceedings of the Second International World-Wide Web Conference*, pages 763–771, Chicago, Illinois, October 1994. http://harvest.cs.colorado.edu/.

[BDMS94] C.M. Bowman, P.B. Danzig, U. Manber, and M.E. Schwartz. Scalable Internet resource discovery: Research problems and approaches. *CACM*, 37(8):98–107, August 1994.

[BDO95] M.W. Berry, S. Dumais, and G.W. O'Brien. Using linear algebra for intelligent information retrieval. *SIAM Review*, 37(4):573–595, December 1995. http://www.cs.utk.edu/~library/TechReports-/1994/ut-cs-94-270.ps.Z.

[CAC95] CACM. Digital Libraries (section on Digital Library Initiatives). *Communications of the ACM*, 38(4):57–64, April 1995.

[CH95] A. Chakravarthy and K. Haase. NetSerf: Using semantic knowledge to find Internet information archives. In *Proceedings of the 18th ACM SIGIR Conference*, pages 4–11, Seattle, Washington, U.S.A., 1995.

[CL92] H. Chen and K.J. Lynch. Automatic construction of networks of concepts characterizing document databases. *IEEE Transactions on Systems, Man, and Cybernetics*, 22(5):885–902, 1992. http://ai.bpa.arizona.edu/papers/ieee91/ieee91.html.

[CLBN93] H. Chen, K. Lynch, K. Basu, and T. Ng. Generating, integrating, and activating thesauri for concept-based document retrieval. *IEEE Expert, Special Series on Artificial Intelligence in Text-Based Information Systems*, 8(2):25–34, April 1993.

[CLC95] J. Callan, Z. Lu, and B. Croft. Searching distributed collections with inference networks. In *Proceedings of the 18th ACM SIGIR Conference*, pages 21–28, Seattle, Washington, U.S.A., 1995.

[Cru95] L. Crum. University of Michigan Digital Library Project. *CACM*, 38(4):63–64, April 1995.

[DAE97] R. Dolin, D. Agrawal, and A. El Abbadi. Classifying network architectures for locating information sources. In *Proceedings of the 5th International Conference on Database Systems for Advanced Applications (DASFAA '97)*, pages 31–40, Melbourne, Australia, April 1997.

[DAED97] R. Dolin, D. Agrawal, A. El Abbadi, and L. Dillon. Pharos: A scalable distributed architecture for locating heterogeneous information sources. In *Proceedings of the 6th International Conference on Information and Knowledge Management (CIKM '97)*, pages 348–355, Las Vegas, Nevada, November 1997.

[DAEP98] R. Dolin, D. Agrawal, A. El Abbadi, and J. Pearlman. Using automated classification for summarizing and selecting heterogeneous information sources. *D-Lib Magazine*, January 1998. http://www.dlib.org/dlib/january98/dolin/01dolin.html.

[Dig97] Digital Equipment Corporation. AltaVista. http://www.altavista-.digital.com/, 1997.

[Dum91] S. Dumais. Improving the retrieval of information from external sources. *Behavior Research Methods, Instruments, & Computers*, 23(2):229–236, 1991.

[EY72] R. Ernst and M. Yovits. Information science as an aid to decision-making. In H. Brinkers, editor, *Decision-Making: Creativity, Judgment, and Systems*. Ohio State University Press, 1972.

[FAFL95] E.A. Fox, R.M. Akscyn, R.K. Furuta, and J.J. Leggett. Digital libraries. *CACM*, 38(4):23–28, April 1995.

[FY95] D. Flater and Y. Yesha. ALIBI: a novel approach to resource discovery. *Internet Research: Electronic Networking Applications and Policy*, 5(4):17–30, 1995.

[GCGMP96] L. Gravano, K. Chang, H. García-Molina, and A. Paepcke. STARTS: Stanford Protocol Proposal for Internet Retrieval and Search. Stanford Digital Library Working Paper SIDL-WP-1996-0043, Computer Science Department, Stanford University, California, July 1996. http://www-diglib.stanford.edu/cgi-bin/WP-/get/SIDL-WP-1996-0043.

[GGM95] L. Gravano and H. García-Molina. Generalizing *GlOSS* to vector-space databases and broker hierarchies. In *Proceedings of the 21st VLDB Conference*, pages 78–89, Zurich, Switzerland, 1995.

[Gol96] J.E. Goldberg. Library of Congress Classification: Shelving device for collections or organization of knowledge fields? In *Proceedings of the Fourth Conference of the International Society for Knowledge Organization (ISKO)*, pages 33–42, Washington, D.C., July 1996.

[GS95] J. Guyton and M. Schwartz. Locating nearby copies of replicated Internet servers. In *ACM SIGCOMM '95*, volume 25 of *Computer Communication Review*, pages 288–298, October 1995.

[Gue96] R.S. Guenther. Bringing the Library of Congress Classification into the computer age: Converting LCC to machine-readable form. In *Proceedings of the Fourth Conference of the International Society for Knowledge Organization (ISKO)*, pages 26–32, Washington, D.C., July 1996.

[Har95] D. Harman. Overview of the Fourth Text REtrieval Conference (TREC-4). In *Fourth Text REtrieval Conference (TREC-4)*, Gaithersburg, MD, 1995. http://trec.nist.gov/pubs/trec4-/overview.ps.

[Hor83] M. Horton. Standard for interchange of USENET messages. RFC850, IETF Network Working Group, 1983.

[Hul94] D. Hull. Improving text retrieval for the routing problem using latent semantic indexing. In *Proceedings of the 17th ACM SIGIR Conference*, pages 282–291, 1994.

[IT95] M. Iwayama and T. Tokunaga. Cluster-based text categorization: a comparison of category search strategies. In *Proceedings of the 18th ACM SIGIR Conference*, pages 12–20, Seattle, Washington, U.S.A., 1995.

[Jör96] C. Jörgensen. The applicability of selected classification systems to image attributes. In *Proceedings of the Fourth Conference of the International Society for Knowledge Organization (ISKO)*, pages 189–197, Washington, D.C., July 1996.

[Kan94] P. Kantor. Information retrieval techniques. In *Annual Review of Information Science and Technology*, volume 29, pages 53–90. Learned Information, Inc. (on behalf of the American Society for Information Science), Medford, NJ, 1994.

[KL86] B. Kantor and P. Lapsley. Network News Transfer Protocol: A proposed standard for the stream-based transmission of news. RFC997, IETF Network Working Group, 1986.

[Lar92] R. Larson. Experiments in automatic Library of Congress Classification. *JASIS*, 43(2):130–148, 1992.

[LH95] R. Losee and S. Haas. Sublanguage terms: Dictionaries, usage, and automatic classification. *JASIS*, 46(7):519–529, 1995.

[Lib86] Library of Congress. *LC Classification Outline*. Washington, D.C., fifth edition, 1986.

[LMO$^+$96] R.R. Larson, J. McDonough, P. O'Leary, L. Kuntz, et al. Cheshire II: Designing a next-generation online catalog. *JASIS*, 47(7):555–567, 1996.

[Lyc96] Lycos. Lycos. http://www.lycos.com/, 1996.

[Mic97] Microsoft Corporation. Microsoft Encarta 98 Encyclopedia Deluxe Edition. CD-ROM, 1997. Part No. X03-22730.

[Mik98] Francis L. Miksa. *The DDC, the Universe of Knowledge, and the Post-Modern Library.* Forest Press, Albany, New York, 1998.

[MM98] W. Ma and B.S. Manjunath. A texture thesaurus for browsing large aerial photographs. *JASIS*, 49(7):633–648, May 1998.

[Mos85] P. Mosher. The nature and uses of the RLG verification studies. *C&RL News*, pages 336–338, July 1985.

[Mos94] J. Mostafa. Digital image representation and access. In *Annual Review of Information Science and Technology*, volume 29, pages 91–135. Learned Information, Inc. (on behalf of the American Society for Information Science), Medford, NJ, 1994.

[MZ94] A. Moffat and J. Zobel. Information retrieval systems for large document collections. In *Third Text REtrieval Conference (TREC-3)*, Gaithersburg, MD, 1994. http://trec.nist.gov/pubs-/trec3/papers/moffat-zobel.ps.

[Nat93] National Science Foundation. *Research on Digital Libraries, Program Guideline NSF 93-141.* Washington, D.C., September 1993. http://www.nsf.gov/ftp/CISE/program/nsf93141.txt.

[PCGM$^+$96] A. Paepcke, S.B. Cousins, H. García-Molina, S.W. Hassan, S.P. Ketchpel, M. Röscheisen, and T. Winograd. Using distributed objects for digital library interoperability. *IEEE Computer*, 29(5):61–68, May 1996.

[PD85] R. Prieto-Díaz. *A Software Classification Scheme.* PhD thesis, Department of Information and Computer Science, University of California, Irvine, 1985.

[Ric93] J. Richards. *Remote sensing digital image analysis: an introduction.* Springer-Verlag, Berlin, 2nd rev. and enlarged edition, 1993.

[Sal89] G. Salton. *Automatic Text Processing.* Addison-Wesley Publishing Company, Inc., Reading, MA, 1989.

[Sch94] L. Schamber. Relevance and information behavior. In *Annual Review of Information Science and Technology*, volume 29, pages 3–48. Learned Information, Inc. (on behalf of the American Society for Information Science), Medford, NJ, 1994.

[Sch95] B. Schatz. Building the Interspace: The Illinois Digital Library Project. *CACM*, 38(4):62–63, April 1995.

[SDW$^+$94] M. Sheldon, A. Duda, R. Weiss, J. O'Toole, and D. Gifford. Content routing for distributed information servers. In *Proceedings of* *the 4th International Conference on Extending Database Technology*, pages 109–122, 1994.

[Shn92] B. Shneiderman. *Designing the User Interface, Strategies for Effective Human-Computer Interaction.* Addison-Wesley, second edition, 1992.

[Sim82] H.A. Simon. From substantive to procedural rationality. In A.G. McGrew and M.J. Wilson, editors, *Decision-Making: Approaches and Analysis.* Manchester University Press, Manchester, 1982.

[SM83] G. Salton and M. McGill. *Introduction to Modern Information Retrieval.* McGraw-Hill Book Company, New York, 1983.

[SMC+96] B. Schatz, W.H. Mischo, T.W. Cole, J.B. Hardin, and A.P. Bishop. Federating diverse collections of scientific literature. *IEEE Computer*, 29(5):28–36, May 1996.

[Sta95] Stanford Digital Libraries Group. The Stanford Digital Library Project. *CACM*, 38(4):59–60, April 1995.

[VF95] C. Viles and J. French. Dissemination of collection wide information in a distributed information retrieval system. In *Proceedings of the 18th ACM SIGIR Conference*, pages 12–20, Seattle, Washington, U.S.A., 1995.

[Wal96] T.D. Walker. L'Apparition du computer: Epistemology and the impact of networked computers on society. In *Proceedings of the*

Fourth Conference of the International Society for Knowledge Organization (ISKO), pages 320–328, Washington, D.C., July 1996.

[Whi76] D. White. *Fundamentals of Decision Theory.* North-Holland Publishing Company, Amsterdam, 1976.

[WP94] A. Woodruff and C. Plaunt. GIPSY: Automated Geographic Indexing of Text Documents. *JASIS*, 45(9):645–655, 1994.

[Yah97] Yahoo! Inc. Yahoo. http://www.yahoo.com/, 1997.