- **Heterogeneous data**: in addition to having to deal with multiple media types and hence with multiple formats, we also have different languages and, what is worse, different alphabets, some of them very large (for example, Chinese or Japanese Kanji).

Most of these problems (such as the variety of data types and poor data quality) are not solvable simply by software improvements. In fact, many of them will not change (and they should not, as in the case of language diversity!) because they are problems (also features) intrinsic to human nature.

The second class of problems are those faced by the user during the interaction with the retrieval system. There are basically two problems: (1) how to specify a query and (2) how to interpret the answer provided by the system. Without taking into account the semantic content of a document, it is not easy to precisely specify a query, unless it is very simple. Further, even if the user is able to pose the query, the answer might be a thousand Web pages. How do we handle a large answer? How do we rank the documents? How do we select the documents that really are of interest to the user? In addition, a single document could be large. How do we browse efficiently in large documents?

So, the overall challenge, in spite of the intrinsic problems posed by the Web, is to submit a good query to the search system, and obtain a manageable and relevant answer. Moreover, in practice we should try to achieve the latter goal even for poorly formulated queries. In the rest of this chapter, we use the term Web pages for HTML documents (HTML is described in Chapter 6). To denote all possible data types available on the Web, we use the term Web documents.

13.3 Characterizing the Web

13.3.1 Measuring the Web

Measuring the Internet and in particular the Web, is a difficult task due to its highly dynamic nature. Nowadays, there are more than 40 million computers in more than 200 countries connected to the Internet, many of them hosting Web servers. The estimated number of Web servers ranges from 2.4 million according to NetSizer [597] (November 1998) to over three million according to the Netcraft Web survey [596] (October 1998). This wide range might be explained when we consider that there are many Web sites that share the same Web server using virtual hosts, that not all of them are fully accessible, that many of them are provisional, etc. Other estimations were made by sampling 0.1% of all Internet numeric addresses obtaining about 2 million unique Web sites [619] or by counting domain names starting with www which in July 1998 were 780,000 according to the Internet Domain survey [599]. However, since not all Web servers have this prefix, the real number is even higher. Considering that in July 1998 the number of Internet hosts was estimated at 36.7 million [599], there is about one Web server per every ten computers connected to the

Internet. The characterization of the Web is a new task of the Web Consortium [797].

In two interesting articles, already (sadly) outdated, Bray [114] and Woodruff *et al.* [834] studied different statistical measures of the Web. The first study uses 11 million pages while the second uses 2.6 million pages, with both sets gathered in November 1995. Their characterization of Web pages is partially reproduced in the following paragraphs. A first question is how many different institutions (not Web servers) maintain Web data. This number is smaller than the number of servers, because many places have multiple servers. The exact number is unknown, but should be more than 40% of the number of Web servers (this percentage was the value back in 1995). The exact number of Web pages is also not known. Estimates at the beginning of 1998 ranged from 200 to 320 million, with 350 million as the best current estimate (July 1998 [91]). The latter study used 20,000 random queries based on a lexicon of 400,000 words extracted from Yahoo!. Those queries were submitted to four search engines and the union of all the answers covered about 70% of the Web. Figure 13.1 gives an approximation of how the number of Web servers and the number of pages have changed in recent years. Between 1997 and 1998, the size of the Web doubled in nine months and is currently growing at a rate of 20 million pages per month. On the other hand, it is estimated that the 30,000 largest Web sites (about 1% of the Web) account for approximately 50% of all Web pages [619].

The most popular formats for Web documents are HTML, followed by GIF and JPG (both for images), ASCII text, and Postscript, in that order. The most popular compression tools used are GNU zip, Zip, and Compress. What is a typical HTML page? First, most HTML pages are not standard, meaning that they do not comply with all the HTML specifications. In ad-

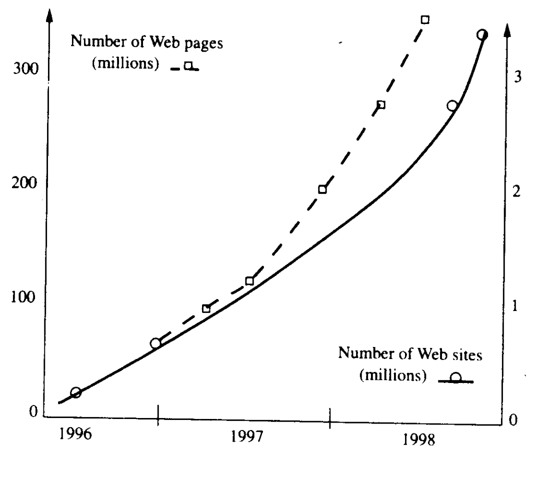


Figure 13.1 Approximate growth of the Web.

dition, although HTML is an instance of SGML, HTML documents seldom start with a formal document type definition. Second, they are small (around 5 Kbs on average with a median of 2 Kbs) and usually contain few images (between one and two on average with an average size of 14 Kb). The pages that have images use them for presentation issues such as colored bullets and lines. An average page has between five and 15 hyperlinks (more than eight links on average) and most of them are local (that is, they point to pages in their own Web server hierarchy). On average, no external server points to any given page (typically, there are only local links pointing to a given page). This is true even for home pages of Web sites. In fact, in 1995, around 80% of these home pages had fewer than ten external links pointing to each of them.

The top ten most referenced sites are Microsoft, Netscape, Yahoo!, and top US universities. In these cases we are talking about sites which are referenced by at least 100,000 places. On the other hand, the site with most links to outside sites is Yahoo!. In some sense, Yahoo! and other directories are the glue of the Web. Without them we would have many isolated portions (which is the case with many personal Web pages). If we assume that the average HTML page has 5 Kb and that there are 300 million Web pages, we have at least 1.5 terabytes of text. This is consistent with other measures obtained from search engines. Note that this volume does not include non-textual documents.

Regarding the languages used in Web pages, there have been three studies made. The first study was done by Funredes [637] from 1996 to 1998. It uses the AltaVista search engine and is based on searching different words in different languages. This technique might not be significant statistically, but the results are consistent with the second study which was carried out by Alis Technology [11] and is based on automatic software that can detect the language used. One of the goals of the study was to test such software (done in 8000 Web servers). The last study was done by OCLC in June of 1998 [619] by sampling Internet numeric addresses and using the SILC language identification software. Table 13.1 gives the percentages of Web pages written in each language (with the exception of the OCLC data that counts Web sites), as well as the number of people (millions) who speak the language. The variations for Japanese might be due to an inability to detect pages written in Kanji. Some languages, in particular Spanish and Portuguese, are growing fast and will surpass French in the near future. The total number of languages exceeds 100.

13.3.2 Modeling the Web

Can we model the document characteristics of the whole Web? Yes, as has already been discussed partially in Chapter 6. The Heaps' and Zipf's laws are also valid in the Web. In particular, the vocabulary grows faster (larger β) and the word distribution should be more biased (larger θ). However, there are no experiments on large Web collections to measure these parameters.

Language	Funredes (1998, %)	Alis Tech. (June 1997, %)	OCLC (June 1998, %)	Spoken by (millions)
English	76.4	82.3	71	450
Japanese	4.8	1.6	4	126
German	4.4	4.0	7	118
French	2.9	1.5	3	122
Spanish	2.6	1.1	3	266
Italian	1.5	0.8	1	63
Portuguese	0.8	0.7	2	175

Table 13.1 Languages of the Web.

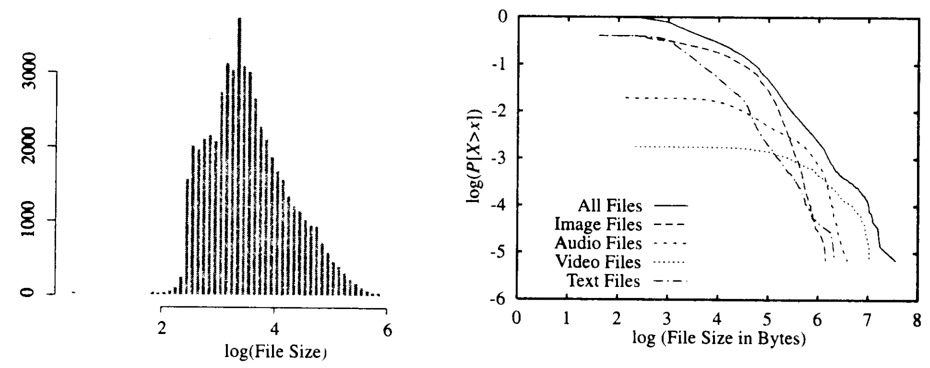


Figure 13.2 Left: Distribution for all file sizes (courtesy of M. Crovella, 1998). Right: Right tail distribution for different file types (from Crovella and Bestavros, 1996). All logarithms are in base 10.

An additional model is related to the distribution of document sizes. According to this model, the document sizes are self-similar [201], that is, they have a large variance (a similar behavior appears in Web traffic). This can be modeled by two different distributions. The main body of the distribution follows a logarithmic normal distribution, such that the probability of finding a document of size x bytes is given by

$$p(x) = \frac{1}{x\sigma\sqrt{2\pi}} \exp -(\ln x - \mu)^2/2\sigma^2$$

where the average (μ) and standard deviation (σ) are 9.357 and 1.318, respectively [59]. Figure 13.2 (left) shows the size distribution of the experimental data.

The right tail of the distribution is 'heavy-tailed.' That is, the majority of documents are small, but there is a non-trivial number of large documents. This is intuitive for image or video files, but it is also true for HTML pages. A good fit is obtained with the Pareto distribution

$$p(x) = \frac{\alpha k^\alpha}{x^{1+\alpha}}$$

where x is measured in bytes and k and α are parameters of the distribution [59] (see Figure 13.2 (right)). For text files, α is about 1.36, being smaller for images and other binary formats [201, 819]. Taking all Web documents into account, we get $\alpha = 1.1$ and $k = 9.3$ Kb [58]. That is, 9.3 Kb is the cut point between both distributions, and 93% of all the files have a size below this value. In fact, for less than 50 Kb, images are the typical files, from 50 to 300 Kb we have an increasing number of audio files, and over that to several megabytes, video files are more frequent. The parameters of these distributions were obtained from a sample of more than 54,000 Web pages requested by several users in a period of two months of 1995. Recent data collected in 1998 show that the size distributions have the same form, but parameters change [58]. Related information can be found on Web benchmarks such as WebSpec96 and the Sun/Inktomi Inkbench [395].

13.4 Search Engines

In this section we cover different architectures of retrieval systems that model the Web as a full-text database. One main difference between standard IR systems and the Web is that, in the Web, all queries must be answered without accessing the text (that is, only the indices are available). Otherwise, that would require either storing locally a copy of the Web pages (too expensive) or accessing remote pages through the network at query time (too slow). This difference has an impact on the indexing and searching algorithms, as well as on the query languages made available.

13.4.1 Centralized Architecture

Most search engines use a centralized crawler-indexer architecture. Crawlers are programs (software agents) that traverse the Web sending new or updated pages to a main server where they are indexed. Crawlers are also called robots, spiders, wanderers, walkers, and knowbots. In spite of their name, a crawler does not actually move to and run on remote machines, rather the crawler runs on a local system and sends requests to remote Web servers. The index is used in a centralized fashion to answer queries submitted from different places in the Web. Figure 13.3 shows the software architecture of a search engine based on the AltaVista architecture [17]. It has two parts: one that deals with the users,

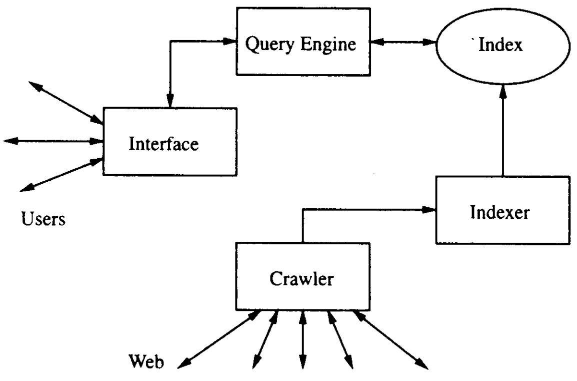


Figure 13.3 Typical crawler-indexer architecture.

consisting of the user interface and the query engine and another that consists of the crawler and indexer modules. In 1998, the overall AltaVista system was running on 20 multi-processor machines, all of them having more than 130 Gb of RAM and over 500 Gb of disk space. Only the query engine uses more than 75% of these resources.

The main problem faced by this architecture is the gathering of the data, because of the highly dynamic nature of the Web, the saturated communication links, and the high load at Web servers. Another important problem is the volume of the data. In fact, the crawler-indexer architecture may not be able to cope with Web growth in the near future. Particularly important is good load balancing between the different activities of a search engine, internally (answering queries and indexing) and externally (crawling).

The largest search engines, considering Web coverage in June 1998, were AltaVista [17], HotBot [380], Northern Light [608], and Excite [240], in that order. According to recent studies, these engines cover 28–55% [749] or 14–34% [490] of all Web pages, whose number was estimated at over 300 million in 1998. Table 13.2 lists the most important search engines and their estimated sizes along with their corresponding URLs. Beware that some search engines are powered by the same internal engine. For example, HotBot, GoTo, and Microsoft are powered by Inktomi [395] and Magellan by Excite's internal engine. Up to date information can be found in [749, 609].

Most search engines are based in the United States and focus on documents in English. Nevertheless, there are search engines specialized in different countries and/or languages, which are able, for instance, to query and retrieve documents written in Kanji (Chinese, Japanese, and Korean). Also there are search engines that take other approaches, like Ask Jeeves! which simulates an interview [34] or DirectHit [215] which ranks the Web pages in the answer in order of their popularity. We should also mention those search engines aimed at specific topics, for example the Search Broker [537] which allows us to search in many specific topics and DejaNews [212] which searches the USENET archives.

Search engine	URL	Web pages indexed
AltaVista	`www.altavista.com`	140
AOL Netfind	`www.aol.com/netfind/`	–
Excite	`www.excite.com`	55
Google	`google.stanford.edu`	25
GoTo	`goto.com`	–
HotBot	`www.hotbot.com`	110
Infoseek	`www.infoseek.com`	30
Lycos	`www.lycos.com`	30
Magellan	`www.mckinley.com`	55
Microsoft	`search.msn.com`	–
NorthernLight	`www.nlsearch.com`	67
WebCrawler	`www.webcrawler.com`	2

Table 13.2 URLs and estimated size (millions) of the largest search engines (May 1998).

There are also engines to retrieve specific Web pages such as personal or institutional home pages or specific objects such as electronic mail addresses, images, or software applets.

13.4.2 Distributed Architecture

There are several variants of the crawler-indexer architecture. Among them, the most important is Harvest [108]. Harvest uses a distributed architecture to gather and distribute data, which is more efficient than the crawler architecture. The main drawback is that Harvest requires the coordination of several Web servers.

The Harvest distributed approach addresses several of the problems of the crawler-indexer architecture, such as: (1) Web servers receive requests from different crawlers, increasing their load; (2) Web traffic increases because crawlers retrieve entire objects, but most of their content is discarded; and (3) information is gathered independently by each crawler, without coordination between all the search engines.

To solve these problems, Harvest introduces two main elements: gatherers and brokers. A gatherer collects and extracts indexing information from one or more Web servers. Gathering times are defined by the system and are periodic (i.e. there are harvesting times as the name of the system suggests). A broker provides the indexing mechanism and the query interface to the data gathered. Brokers retrieve information from one or more gatherers or other brokers, updating incrementally their indices. Depending on the configuration of gatherers and brokers, different improvements on server load and network traffic can be

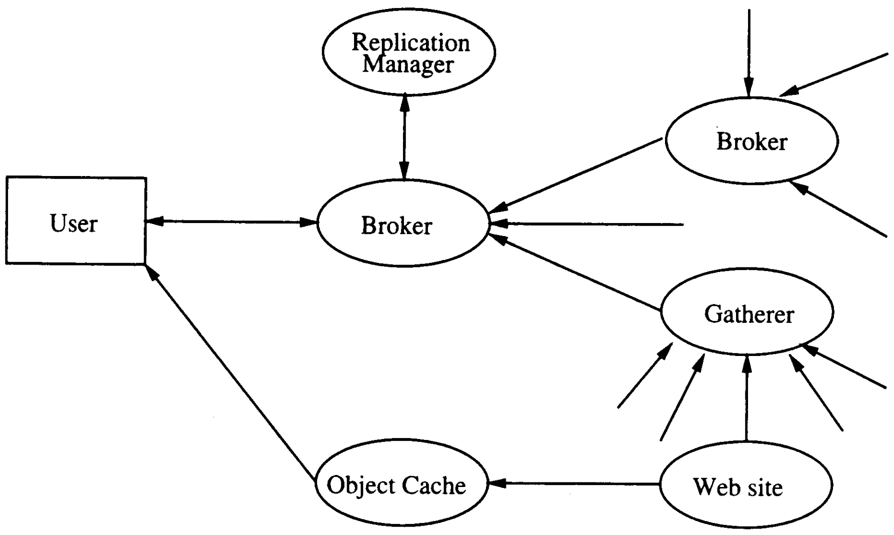


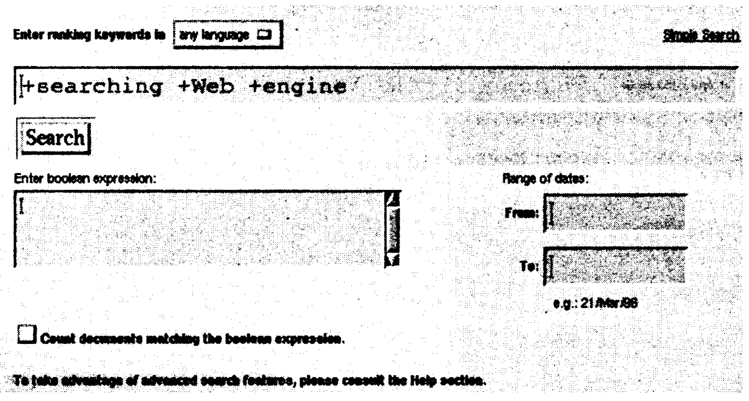
Figure 13.4 Harvest architecture.

achieved. For example, a gatherer can run on a Web server, generating no external traffic for that server. Also, a gatherer can send information to several brokers, avoiding work repetition. Brokers can also filter information and send it to other brokers. This design allows the sharing of work and information in a very flexible and generic manner. An example of the Harvest architecture is shown in Figure 13.4 [108].

One of the goals of Harvest is to build topic-specific brokers, focusing the index contents and avoiding many of the vocabulary and scaling problems of generic indices. Harvest includes a distinguished broker that allows other brokers to register information about gatherers and brokers. This is useful to search for an appropriate broker or gatherer when building a new system. The Harvest architecture also provides replicators and object caches. A replicator can be used to replicate servers, enhancing user-base scalability. For example, the registration broker can be replicated in different geographic regions to allow faster access. Replication can also be used to divide the gathering process between many Web servers. Finally, the object cache reduces network and server load, as well as response latency when accessing Web pages. More details on the system can be found in [108].

Currently, there are hundreds of Harvest applications on the Web (for example, the CIA, NASA, the US National Academy of Sciences, and the US Government Printing Office), as this software is on the public domain.‡ Netscape's Catalog Server is a commercial version of Harvest and Network Appliances' cache is a commercial version of the Harvest Cache.

‡ Information is available at `harvest.transarc.com`.


Enter ranking keywords in any language

Simple Search

+searching +Web +engine

Search

Enter boolean expression:

Range of dates:

From:

To:

e.g.: 21/Mar/96

☐ Count documents matching the boolean expression.

To take advantage of advanced search features, please consult the Help section.

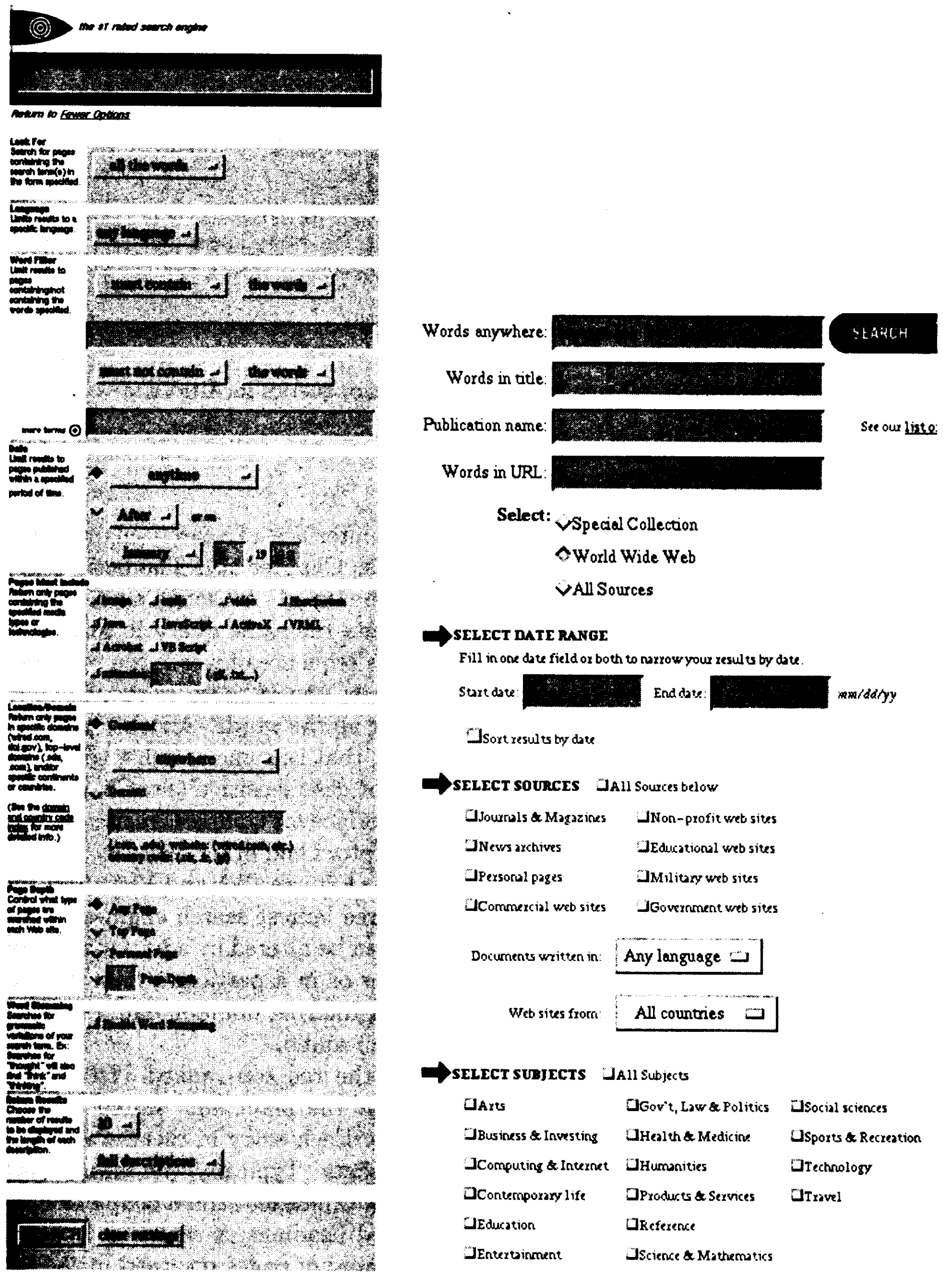
Figure 13.5 Query interface for complex queries in AltaVista.

13.4.3 User Interfaces

There are two important aspects of the user interface of search engines: the query interface and the answer interface (see also Chapter 10). The basic query interface is a box where one or more words can be typed. Although a user would expect that a given sequence of words represents the same query in all search engines, it does not. For example, in AltaVista a sequence of words is a reference to the union of all the Web pages having at least one of those words, while in HotBot it is a reference to the Web pages having all the words. Another problem is that the logical view of the text is not known, that is, some search engines use stopwords, some do stemming, and some are not case sensitive (see Chapter 7).

All search engines also provide a query interface for complex queries as well as a command language including Boolean operators and other features, such as phrase search, proximity search, and wild cards. Figures 13.5 and 13.6 show the query interfaces for complex queries for the three largest search engines. They provide several filtering functions. The results can be filtered by additional words that must be present or absent from the answer or in a particular field such as the URL or title, language, geographic region or Internet domain, date range, or inclusion of specific data types such as images or audio.

The answer usually consists of a list of the ten top ranked Web pages. Figure 13.7 shows the three top documents for the main four search engines for the query `searching` and `Web` and `engine`. Each entry in this list includes some information about the document it represents. Typically, the information includes the URL, size, the date when the page was indexed, and a couple of lines with its content (title plus first lines or selected headings or sentences). Some search engines allow the user to change the number of pages returned in the list and the amount of information per page, but in most cases this is fixed or limited to a few choices. The order of the list is typically by relevance, but sorting by URL or date is also available in some engines. In addition, most search engines also have an option to find documents similar to each Web page in the answer.



the #1 rated search engine

Return to Fewer Options

Look For
Search for pages containing the search term(s) in the form specified.
all the words

Language
Limits results to a specific language.
any language

Word Filter
Limit results to pages containing/not containing the words specified.
must contain | the words
must not contain | the words
more terms

Date
Limit results to pages published within a specified period of time.
◆ anytime
◇ After | or on
January | , 19

Pages Must Include
Return only pages containing the specified media types or technologies.
image | audio | video | Shockwave
Java | JavaScript | ActiveX | VRML
Acrobat | VB Script
extension (.gif, .txt...)

Location/Domain
Return only pages in specific domains (wired.com, doi.gov), top-level domains (.edu, .com), and/or specific continents or countries.
(See the domain and country code index for more detailed info.)
◆ Continent
anywhere
◇ Domain
(.com, .edu), website: (wired.com, etc.)
country code: (.uk, .fr, .jp)

Page Depth
Control what type of pages are searched within each Web site.
◆ Any Page
◇ Top Page
◇ Personal Page
◇ Page Depth

Word Stemming
Searches for grammatic variations of your search term. Ex: Searches for "thought" will also find "think" and "thinking".
Enable Word Stemming

Return Results
Choose the number of results to be displayed and the length of each description.
10
full descriptions

[illegible] | clear settings

Words anywhere: SEARCH

Words in title:

Publication name: See our list o

Words in URL:

Select: ◇ Special Collection
◆ World Wide Web
◇ All Sources

SELECT DATE RANGE
Fill in one date field or both to narrow your results by date.

Start date: End date: mm/dd/yy

☐ Sort results by date

SELECT SOURCES ☐ All Sources below

☐ Journals & Magazines	☐ Non-profit web sites
☐ News archives	☐ Educational web sites
☐ Personal pages	☐ Military web sites
☐ Commercial web sites	☐ Government web sites

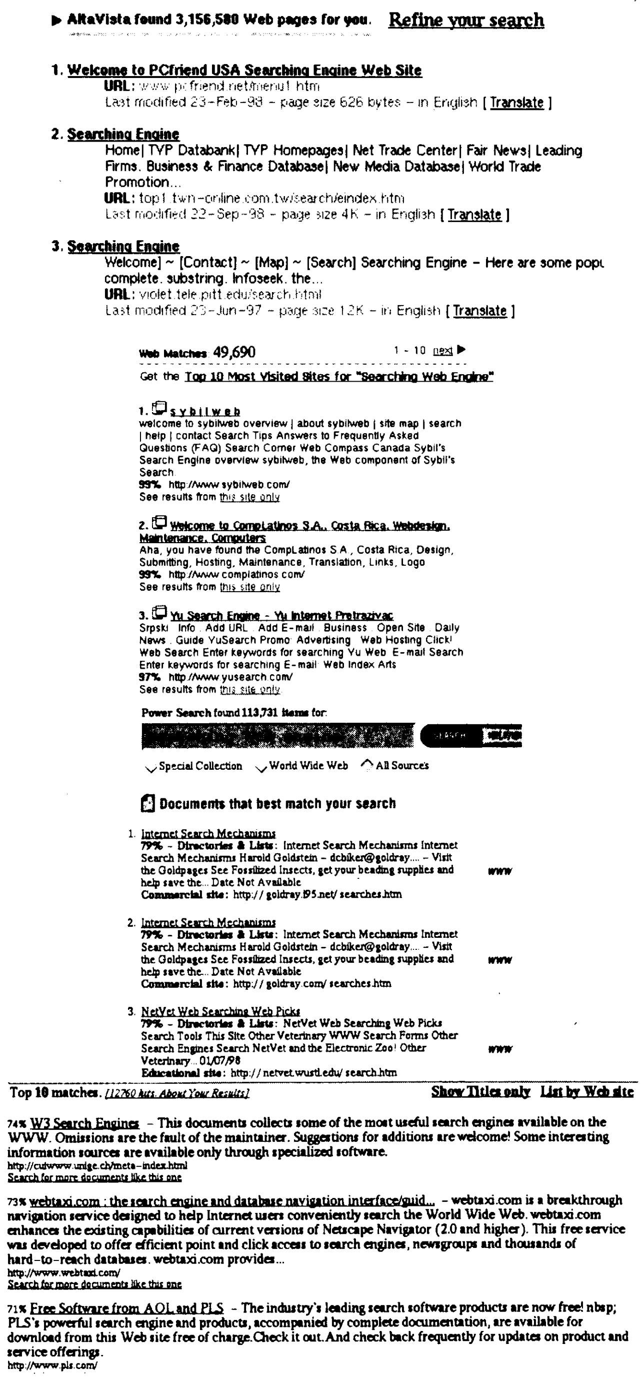
Documents written in: Any language

Web sites from: All countries

SELECT SUBJECTS ☐ All Subjects

☐ Arts	☐ Gov't, Law & Politics	☐ Social sciences
☐ Business & Investing	☐ Health & Medicine	☐ Sports & Recreation
☐ Computing & Internet	☐ Humanities	☐ Technology
☐ Contemporary life	☐ Products & Services	☐ Travel
☐ Education	☐ Reference	
☐ Entertainment	☐ Science & Mathematics	

Figure 13.6 Query interface for complex queries for HotBot (left) and NorthernLight (right).



▶ AltaVista found 3,156,580 Web pages for you. Refine your search

1. Welcome to PCfriend USA Searching Engine Web Site
   URL: www.pcfriend.net/menu1.htm
   Last modified 23-Feb-98 - page size 626 bytes - in English [Translate]

2. Searching Engine
   Home| TYP Databank| TYP Homepages| Net Trade Center| Fair News| Leading Firms. Business & Finance Database| New Media Database| World Trade Promotion...
   URL: top1.twn-online.com.tw/search/eindex.htm
   Last modified 22-Sep-98 - page size 4K - in English [Translate]

3. Searching Engine
   Welcome] ~ [Contact] ~ [Map] ~ [Search] Searching Engine - Here are some popu complete. substring. Infoseek. the...
   URL: violet.tele.pitt.edu/search.html
   Last modified 23-Jun-97 - page size 12K - in English [Translate]

Web Matches 49,690 1 - 10 next ▶

Get the Top 10 Most Visited Sites for "Searching Web Engine"

1. sybilweb
welcome to sybilweb overview | about sybilweb | site map | search | help | contact Search Tips Answers to Frequently Asked Questions (FAQ) Search Corner Web Compass Canada Sybil's Search Engine overview sybilweb, the Web component of Sybil's Search
99% http://www.sybilweb.com/
See results from this site only

2. Welcome to CompLatinos S.A., Costa Rica, Webdesign, Maintenance, Computers
Aha, you have found the CompLatinos S.A., Costa Rica, Design, Submitting, Hosting, Maintenance, Translation, Links, Logo
99% http://www.complatinos.com/
See results from this site only

3. Yu Search Engine - Yu Internet Pretrazivac
Srpski . Info . Add URL . Add E-mail . Business . Open Site . Daily News . Guide YuSearch Promo: Advertising . Web Hosting Click! Web Search Enter keywords for searching Yu Web . E-mail Search Enter keywords for searching E-mail: Web Index Arts
97% http://www.yusearch.com/
See results from this site only

Power Search found 113,731 items for:

Special Collection World Wide Web All Sources

Documents that best match your search

1. Internet Search Mechanisms
79% - Directories & Lists: Internet Search Mechanisms Internet Search Mechanisms Harold Goldstein - dcbiker@goldray.... - Visit the Goldpages See Fossilized Insects, get your beading supplies and help save the... Date Not Available www
Commercial site: http:// goldray.f95.net/ searches.htm

2. Internet Search Mechanisms
79% - Directories & Lists: Internet Search Mechanisms Internet Search Mechanisms Harold Goldstein - dcbiker@goldray.... - Visit the Goldpages See Fossilized Insects, get your beading supplies and help save the... Date Not Available www
Commercial site: http:// goldray.com/ searches.htm

3. NetVet Web Searching Web Picks
79% - Directories & Lists: NetVet Web Searching Web Picks Search Tools This Site Other Veterinary WWW Search Forms Other Search Engines Search NetVet and the Electronic Zoo! Other Veterinary... 01/07/98 www
Educational site: http:// netvet.wustl.edu/ search.htm

Top 10 matches. *[12760 hits. About Your Results]* Show Titles only List by Web site

74% W3 Search Engines - This documents collects some of the most useful search engines available on the WWW. Omissions are the fault of the maintainer. Suggestions for additions are welcome! Some interesting information sources are available only through specialized software.
http://cuiwww.unige.ch/meta-index.html
Search for more documents like this one

73% webtaxi.com : the search engine and database navigation interface/guid... - webtaxi.com is a breakthrough navigation service designed to help Internet users conveniently search the World Wide Web. webtaxi.com enhances the existing capabilities of current versions of Netscape Navigator (2.0 and higher). This free service was developed to offer efficient point and click access to search engines, newsgroups and thousands of hard-to-reach databases. webtaxi.com provides...
http://www.webtaxi.com/
Search for more documents like this one

71% Free Software from AOL and PLS - The industry's leading search software products are now free! nbsp; PLS's powerful search engine and products, accompanied by complete documentation, are available for download from this Web site free of charge.Check it out.And check back frequently for updates on product and service offerings.
http://www.pls.com/

Figure 13.7 Output for the query `searching and Web and engine` for the four main search engines; from top to bottom: AltaVista, HotBot, NorthernLight, and Excite.

The user can also refine the query by constructing more complex queries based on the previous answer.

The Web pages retrieved by the search engine in response to a user query are ranked, usually using statistics related to the terms in the query. In some cases this may not have any meaning, because relevance is not fully correlated with statistics about term occurrence within the collection. Some search engines also taking into account terms included in metatags or the title, or the popularity of a Web page to improve the ranking. This topic is covered next.

13.4.4 Ranking

Most search engines use variations of the Boolean or vector model (see Chapter 2) to do ranking. As with searching, ranking has to be performed without accessing the text, just the index. There is not much public information about the specific ranking algorithms used by current search engines. Further, it is difficult to compare fairly different search engines given their differences, and continuous improvements. More important, it is almost impossible to measure recall, as the number of relevant pages can be quite large for simple queries. Some inconclusive studies include [327, 498].

Yuwono and Lee [844] propose three ranking algorithms in addition to the classical tf-idf scheme (see Chapter 2). They are called Boolean spread, vector spread, and most-cited. The first two are the normal ranking algorithms of the Boolean and vector model extended to include pages pointed to by a page in the answer or pages that point to a page in the answer. The third, most-cited, is based only on the terms included in pages having a link to the pages in the answer. A comparison of these techniques considering 56 queries over a collection of 2400 Web pages indicates that the vector model yields a better recall-precision curve, with an average precision of 75%.

Some of the new ranking algorithms also use hyperlink information. This is an important difference between the Web and normal IR databases. The number of hyperlinks that point to a page provides a measure of its popularity and quality. Also, many links in common between pages or pages referenced by the same page often indicates a relationship between those pages. We now present three examples of ranking techniques that exploit these facts, but they differ in that two of them depend on the query and the last does not.

The first is WebQuery [148], which also allows visual browsing of Web pages. WebQuery takes a set of Web pages (for example, the answer to a query) and ranks them based on how connected each Web page is. Additionally, it extends the set by finding Web pages that are highly connected to the original set. A related approach is presented by Li [512].

A better idea is due to Kleinberg [444] and used in HITS (Hypertext Induced Topic Search). This ranking scheme depends on the query and considers the set of pages S that point to or are pointed by pages in the answer. Pages that have many links pointing to them in S are called authorities (that is, they should have relevant content). Pages that have many outgoing links are called hubs (they should point to similar content). A positive two-way feedback exists:

better authority pages come from incoming edges from good hubs and better hub pages come from outgoing edges to good authorities. Let $H(p)$ and $A(p)$ be the hub and authority value of page p. These values are defined such that the following equations are satisfied for all pages p:

$$H(p) = \sum_{u \in S \mid p \rightarrow u} A(u) , \qquad A(p) = \sum_{v \in S \mid v \rightarrow p} H(v)$$

where $H(p)$ and $A(p)$ for all pages are normalized (in the original paper, the sum of the squares of each measure is set to one). These values can be determined through an iterative algorithm, and they converge to the principal eigenvector of the link matrix of S. In the case of the Web, to avoid an explosion of the size of S, a maximal number of pages pointing to the answer can be defined. This technique does not work with non-existent, repeated, or automatically generated links. One solution is to weight each link based on the surrounding content. A second problem is that the topic of the result can become diffused. For example, a particular query is enlarged by a more general topic that contains the original answer. One solution to this problem is to analyze the content of each page and assign a score to it, as in traditional IR ranking. The link weight and the page score can be included on the previous formula multiplying each term of the summation [154, 93, 153]. Experiments show that the recall and precision on the first ten answers increases significantly [93]. The order of the links can also be used by dividing the links into subgroups and using the HITS algorithm on those subgroups instead of the original Web pages [153].

The last example is PageRank, which is part of the ranking algorithm used by Google [117]. PageRank simulates a user navigating randomly in the Web who jumps to a random page with probability q or follows a random hyperlink (on the current page) with probability $1 - q$. It is further assumed that this user never goes back to a previously visited page following an already traversed hyperlink backwards. This process can be modeled with a Markov chain, from where the stationary probability of being in each page can be computed. This value is then used as part of the ranking mechanism of Google. Let $C(a)$ be the number of outgoing links of page a and suppose that page a is pointed to by pages p_1 to p_n. Then, the PageRank, $PR(a)$ of a is defined as

$$PR(a) = q + (1 - q) \sum_{i=1}^{n} PR(p_i)/C(p_i)$$

where q must be set by the system (a typical value is 0.15). Notice that the ranking (weight) of other pages is normalized by the number of links in the page. PageRank can be computed using an iterative algorithm, and corresponds to the principal eigenvector of the normalized link matrix of the Web (which is the transition matrix of the Markov chain). Crawling the Web using this ordering has been shown to be better than other crawling schemes [168] (see next section).

Therefore, to help ranking algorithms, page designers should include informative titles, headings, and meta fields, as well as good links. However, keywords should not be repeated as some search engines penalize repeating words (spamming). Using full terms instead of indirect ways to refer to subjects should also be considered.

13.4.5 Crawling the Web

In this section we discuss how to crawl the Web, as there are several techniques. The simplest is to start with a set of URLs and from there extract other URLs which are followed recursively in a breadth-first or depth-first fashion. For that reason, search engines allow users to submit top Web sites that will be added to the URL set. A variation is to start with a set of populars URLs, because we can expect that they have information frequently requested. Both cases work well for one crawler, but it is difficult to coordinate several crawlers to avoid visiting the same page more than once. Another technique is to partition the Web using country codes or Internet names, and assign one or more robots to each partition, and explore each partition exhaustively.

Considering how the Web is traversed, the index of a search engine can be thought of as analogous to the stars in an sky. What we see has never existed, as the light has traveled different distances to reach our eye. Similarly, Web pages referenced in an index were also explored at different dates and they may not exist any more. Nevertheless, when we retrieve a page, we obtain its actual content. How fresh are the Web pages referenced in an index? The pages will be from one day to two months old. For that reason, most search engines show in the answer the date when the page was indexed. The percentage of invalid links stored in search engines vary from 2 to 9%. User submitted pages are usually crawled after a few days or weeks. Starting there, some engines traverse the whole Web site, while others select just a sample of pages or pages up to a certain depth. Non-submitted pages will wait from weeks up to a couple of months to be detected. There are some engines that learn the change frequency of a page and visit it accordingly [175]. They may also crawl more frequently popular pages (for example, pages having many links pointing to them). Overall, the current fastest crawlers are able to traverse up to 10 million Web pages per day.

The order in which the URLs are traversed is important. As already mentioned, the links in a Web page can be traversed breadth first or depth first. Using a breadth first policy, we first look at all the pages linked by the current page, and so on. This matches well Web sites that are structured by related topics. On the other hand, the coverage will be wide but shallow and a Web server can be bombarded with many rapid requests. In the depth first case, we follow the first link of a page and we do the same on that page until we cannot go deeper, returning recursively. This provides a narrow but deep traversal. Only recently, some research on this problem has appeared [168], showing that good ordering schemes can make a difference if crawling better pages first (using the PageRank scheme mentioned above).

Due to the fact that robots can overwhelm a server with rapid requests and can use significant Internet bandwidth (in particular the whole bandwidth of small domains can be saturated), a set of guidelines for robot behavior has been developed [457]. For this purpose, a special file is placed at the root of every Web server indicating the restrictions at that site, in particular the pages that should not be indexed. Crawlers can also have problems with HTML pages that use frames (a mechanism to divide a page in two or more parts) or image maps (hyperlinks associated to images). In addition, dynamically generated pages cannot be indexed as well as password protected pages.

13.4.6 Indices

Most indices use variants of the inverted file (see Chapter 8). In short, an inverted file is a list of sorted words (vocabulary), each one having a set of pointers to the pages where it occurs. Some search engines use elimination of stopwords to reduce the size of the index. Also, it is important to remember that a logical view of the text is indexed. Normalization operations may include removal of punctuation and multiple spaces to just one space between each word, uppercase to lowercase letters, etc. (see Chapter 7). To give the user some idea about each document retrieved, the index is complemented with a short description of each Web page (creation date, size, the title and the first lines or a few headings are typical). Assuming that 500 bytes are required to store the URL and the description of each Web page, we need 50 Gb to store the description for 100 million pages. As the user initially receives only a subset of the complete answer to each query, the search engine usually keeps the whole answer set in memory, to avoid having to recompute it if the user asks for more documents.

State of the art indexing techniques can reduce the size of an inverted file to about 30% of the size of the text (less if stopwords are used). For 100 million pages, this implies about 150 Gb of disk space. By using compression techniques, the index size can be reduced to 10% of the text [825]. A query is answered by doing a binary search on the sorted list of words of the inverted file. If we are searching multiple words, the results have to be combined to generate the final answer. This step will be efficient if each word is not too frequent. Another possibility is to compute the complete answer while the user requests more Web pages, using a lazy evaluation scheme. More details on searching over an inverted file can be found in Chapter 8.

Inverted files can also point to the actual occurrences of a word within a document (full inversion). However, that is too costly in space for the Web, because each pointer has to specify a page and a position inside the page (word numbers can be used instead of actual bytes). On the other hand, having the positions of the words in a page, we can answer phrase searches or proximity queries by finding words that are near each other in a page. Currently, some search engines are providing phrase searches, but the actual implementation is not known.

Finding words which start with a given prefix requires two binary searches in the sorted list of words. More complex searches, like words with errors, arbitrary wild cards or, in general, any regular expression on a word, can be performed by doing a sequential scan over the vocabulary (see Chapter 8). This may seem slow, but the best sequential algorithms for this type of query can search around 20 Mb of text stored in RAM in one second (5 Mb is more or less the vocabulary size for 1 Gb of text). Thus, for several gigabytes we can answer those queries in a few seconds. For the Web this is still too slow but not completely out of the question. In fact, using Heaps' law and assuming $\beta = 0.7$ for the Web, the vocabulary size for 1 Tb is 630 Mb which implies a searching time of half a minute.

Pointing to pages or to word positions is an indication of the granularity of the index. The index can be less dense if we point to logical blocks instead of pages. In this way we reduce the variance of the different document sizes, by making all blocks roughly the same size. This not only reduces the size of the pointers (because there are fewer blocks than documents) but also reduces the number of pointers because words have locality of reference (that is, all the occurrences of a non-frequent word will tend to be clustered in the same block). This idea was used in Glimpse [540] which is at the core of Harvest [108]. Queries are resolved as for inverted files, obtaining a list of blocks that are then searched sequentially (exact sequential search can be done over 30 Mb per second in RAM). Glimpse originally used only 256 blocks, which was efficient up to 200 Mb for searching words that were not too frequent, obtaining an index of only 2% of the text. By tuning the number of blocks and the block size, reasonable space-time trade-offs can be achieved for larger document collections (for more details see Chapter 8). These ideas cannot be used (yet) for the Web because sequential search cannot be afforded, as it implies a network access. However, in a distributed architecture where the index is also distributed, logical blocks make sense.

13.5 Browsing

In this section we cover Web tools which are based on browsing and searching, in particular Web directories. Although the Web coverage provided by directories is very low (less than 1% of all Web pages), the answers returned to the user are usually much more relevant.

13.5.1 Web Directories

The best and oldest example of a Web directory is Yahoo! [839], which is likely the most used searching tool. Other large Web directories include eBLAST, LookSmart, Magellan, and NewHoo. Some of them are hybrids, because they also provide searches in the whole Web. Most search engines also provide subject categories nowadays, including AltaVista Categories, AOL Netfind, Excite Channels, HotBot, Infoseek, Lycos Subjects, and WebCrawler Select. are specific to some areas. For example, there are Web sites focused on business, news,

Web directory	URL	Web sites	Categories
eBLAST	`www.eblast.com`	125	–
LookSmart	`www.looksmart.com`	300	24
Lycos Subjects	`a2z.lycos.com`	50	–
Magellan	`www.mckinley.com`	60	–
NewHoo	`www.newhoo.com`	100	23
Netscape	`www.netscape.com`	–	–
Search.com	`www.search.com`	–	–
Snap	`www.snap.com`	–	–
Yahoo!	`www.yahoo.com`	750	–

Table 13.3 URLs, Web pages indexed and categories (both in thousands) of some Web directories (beginning of 1998).

Arts & Humanities	Local
Automotive	News
Business & Economy	Oddities
Computers & Internet	People
Education	Philosophy & Religion
Employment	Politics
Entertainment & Leisure	Recreation
Games	Reference
Government	Regional
Health & Fitness	Science & Technology
Hobbies & Interests	Shopping & Services
Home	Social Science
Investing	Society & Culture
Kids & Family	Sports
Life & Style	Travel & Tourism
Living	World

Table 13.4 The first level categories in Web directories.

and, in particular, research bibliography. Web directories are also called catalogs, yellow pages, or subject directories. Table 13.3 gives the URLs of the most important Web directories (not including the search engines already listed in section 13.4).

Directories are hierarchical taxonomies that classify human knowledge. Table 13.4 shows the first level of the taxonomies used by Web directories (the number of first level categories ranges from 12 to 26). Some subcategories are also available in the main page of Web directories, adding around 70 more topics. The largest directory, Yahoo!, has close to one million pages classified, followed by LookSmart, which has about 24,000 categories in total. Yahoo! also offers

14 regional or country specialized directories in other languages including Chinese, Danish, French, German, Italian, Japanese, Korean, Norwegian, Spanish, and Swedish. In most cases, pages have to be submitted to the Web directory, where they are reviewed, and, if accepted, classified in one or more categories of the hierarchy. Although the taxonomy can be seen as a tree, there are cross references, so it is really a directed acyclic graph.

The main advantage of this technique is that if we find what we are looking for, the answer will be useful in most cases. On the other hand, the main disadvantage is that the classification is not specialized enough and that not all Web pages are classified. The last problem becomes worse every day as the Web grows. The efforts to do automatic classification, by using clustering or other techniques, are very old. However, up to now, natural language processing is not 100% effective in extracting relevant terms from a document. Thus, classification is done manually by a limited number of people. This is a potential problem with users having a different notion of categories than the manmade categorization.

Web directories also allow the user to perform a search on the taxonomy descriptors or in the Web pages pointed to by the taxonomy. In fact, as the number of classified Web pages is small, we can even afford to have a copy of all pages. In that case they must be updated frequently, which may pose performance and temporal validity problems. In addition, most Web directories also send the query to a search engine (through a strategic alliance) and allow the whole Web to be searched.

13.5.2 Combining Searching with Browsing

Usually, users either browse following hypertext links or they search a Web site (or the whole Web). Currently, in Web directories, a search can be reduced to a subtree of the taxonomy. However, the search may miss related pages that are not in that part of the taxonomy. Some search engines find similar pages using common words, but often this is not effective.

WebGlimpse is a tool that tries to solve these problems by combining browsing with searching [539]. WebGlimpse attaches a small search box to the bottom of every HTML page, and allows the search to cover the neighborhood of that page or the whole site, without having to stop browsing. This is equivalent to following hypertext links that are constructed on the fly through a neighborhood search. WebGlimpse can be useful in building indices for personal Web pages or collections of favorite URLs.

First, WebGlimpse indexes a Web site (or a collection of specific documents) and computes neighborhoods according to user specifications. As a result, WebGlimpse adds the search boxes to selected pages, collects remote pages that are relevant, and caches those pages locally. Later, the users can search in the neighborhood of a page using the search boxes. As the name suggests, WebGlimpse uses Glimpse as its search engine [540].

The neighborhood of a Web page is defined as the set of Web pages that are reachable by a path of hypertext links within a maximum predefined distance. This distance can be set differently for local and remote pages. For example, it can be unlimited locally, but be only three at any remote site. The neighborhood can also include all the subdirectories of the directory where the Web page is. The result is a graph of all the neighborhoods of the Web site or collection, and for each Web page, a file with all the Web pages in its neighborhood. When searching, any query in the whole index can be intersected with a neighborhood list, obtaining the relevant Web pages. A nice addition to WebGlimpse would be to visualize the neighborhoods. This problem is the topic of the next section.

13.5.3 Helpful Tools

There are many software tools to help browsing and searching. Some of them are add-ons to browsers, such as Alexa [10]. Alexa is a free Web navigation service that can be attached as a toolbar at the bottom of any browser and accompanies the user in his surfing. It provides useful information about the sites that are visited, including their popularity, speed of access, freshness, and overall quality (obtained from votes of Alexa users). Alexa also suggests related sites helping one's navigation. Another navigation service and searching guide is WebTaxi [805].

There are other tools that use visual metaphors, which can be broadly classified into two types: tools designed to visualize a subset of the Web and tools designed to visualize large answers. Both cases need to represent a large graph in a meaningful way. Specific commercial examples of tools to visualize Web subsets are Microsoft's SiteAnalyst (formerly from NetCarta), MAPA from Dynamic Diagrams, IBM's Mapuccino (formerly WebCutter [527], shown in Figure 10.22), SurfSerf, Merzscope from Merzcom, CLEARweb, Astra SiteManager, WebAnalyzer from InContext, HistoryTree from SmartBrowser, etc. Non-commercial works include WebMap [220], Sitemap, Ptolomeaus, and many earlier research [234, 578, 564, 20]. We have not included more generic visualization software, where Web visualization is just a particular case, or other related visualization tools such as Web usage analysis [642, 294, 737].

Metaphors to visualize large answers are covered in Chapter 10. Visual tools are not yet deployed in the whole Web because there is no standard way of communicating visualizers and search engines. One possible approach is to use a markup language based on XML (see Chapter 6), as proposed in [15].

13.6 Metasearchers

Metasearchers are Web servers that send a given query to several search engines, Web directories and other databases, collect the answers and unify them. Examples are Metacrawler [715] and SavvySearch [383, 223]. The main advantages of metasearchers are the ability to combine the results of many sources and the fact that the user can pose the same query to various sources through a single common interface. Metasearchers differ from each other in how ranking

Metasearcher	URL	Sources used
Cyber 411	`www.cyber411.com`	14
Dogpile	`www.dogpile.com`	25
Highway61	`www.highway61.com`	5
Inference Find	`www.infind.com`	6
Mamma	`www.mamma.com`	7
MetaCrawler	`www.metacrawler.com`	7
MetaFind	`www.metafind.com`	7
MetaMiner	`www.miner.uol.com.br`	13
MetaSearch	`www.metasearch.com`	–
SavvySearch	`savvy.cs.colostate.edu:2000`	>13

Table 13.5 URLs of metasearchers and number of sources that they use (October 1998).

is performed in the unified result (in some cases no ranking is done), and how well they translate the user query to the specific query language of each search engine or Web directory (the query language common to all of them could be small). Table 13.5 shows the URLs of the main metasearch engines as well as the number of search engines, Web directories and other databases that they search. Metasearchers can also run on the client, for example, Copernic, EchoSearch, WebFerret, WebCompass, and WebSeeker. There are others that search several sources and show the different answers in separate windows, such as All4One, OneSeek, Proteus, and Search Spaniel.

The advantages of metasearchers are that the results can be sorted by different attributes such as host, keyword, date, etc; which can be more informative than the output of a single search engine. Therefore browsing the results should be simpler. On the other hand, the result is not necessarily all the Web pages matching the query, as the number of results per search engine retrieved by the metasearcher is limited (it can be changed by the user, but there is an upper limit). Nevertheless, pages returned by more than one search engine should be more relevant.

We expect that new metasearchers will do better ranking. A first step in this direction is the NEC Research Institute metasearch engine, Inquirus [488, 489]. The main difference is that Inquirus actually downloads and analyzes each Web page obtained and then displays each page, highlighting the places where the query terms were found. The results are displayed as soon as they are available in a progressive manner, otherwise the waiting time would be too long. This technique also allows non-existent pages or pages that have changed and do not contain the query any more to be discarded, and, more important, provides for better ranking than normal search engines. On the other hand, this metasearcher is not available to the general public.

Measure	Average value	Range
Number of words	2.35	0 to 393
Number of operators	0.41	0 to 958
Repetitions of each query	3.97	1–1.5 million
Queries per user session	2.02	1–173,325
Screens per query	1.39	1–78,496

Table 13.6 Queries on the Web: average values.

The use of metasearchers is justified by coverage studies that show that a small percentage of Web pages are in all search engines [91]. In fact, fewer than 1% of the Web pages indexed by AltaVista, HotBot, Excite, and Infoseek are in all of those search engines. This fact is quite surprising and has not been explained (yet). Metasearchers for specific topics can be considered as software agents and are covered in section 13.8.2.

13.7 Finding the Needle in the Haystack

13.7.1 User Problems

We have already glanced at some of the problems faced by the user when interacting with the query interfaces currently provided by search engines. First, the user does not exactly understand the meaning of searching using a set of words, as discussed in Chapter 10. Second, the user may get unexpected answers because he is not aware of the logical view of the text adopted by the system. An example is the use of uppercase letters when the search engine is not case sensitive. Hence, a word like 'Bank' loses part of its semantics if we search for 'bank.' Simple experiments also show that due to typos or variations of a word, even if correctly capitalized, 10–20% of the matches can be lost. Similarly, foreign names or words that are difficult to spell may appear incorrectly which may result in a loss of up to 50% of the relevant answers, as mentioned in section 13.2.

Another problem is that most users have trouble with Boolean logic. In natural language, sometimes we use 'and' and 'or' with different meaning depending on the context. For example, when choosing between two things, we use an exclusive 'or,' which does not match the Boolean interpretation. Because of this, several studies show that around 80% of the queries do not use any Boolean or other operation. For these reasons many people have trouble using command query languages, and query forms should clearly specify which words must or must not be contained in a document that belongs to the answer.

There are a few surveys and analyses of query logs with respect to the usage of search engines [647, 403, 728]. The latter reference is based on 285 million user sessions containing 575 million queries. Table 13.6 gives the main results of that study, carried out in September 1998. Some of the strange results might be due to queries done by mechanized search agents. The number of queries submitted per day to AltaVista is over 13 million. Users select a search engine mainly based on ease of use, speed, coverage, relevance of the answer, and habit. The main purposes are research, leisure, business, and education. The main problems found are that novice users do not know how to start and lack the general knowledge that would help in finding better answers. Other problems are that search engines are slow, that the answer is too large, not very relevant, and not always up to date. Also, most people do not care about advertising, which is one of the main sources of funding for search engines. When searching, 25% of the users use a single keyword, and on average their queries have only two or three terms. In addition, about 15% of the users restrict the search to a predefined topic and most of them (nearly 80%) do not modify the query. In addition, most users (about 85%) only look at the first screen with results and 64% of the queries are unique. Also, many words appear in the same sentence, suggesting that proximity search should be used. There are also studies about users' demographics and software and hardware used.

13.7.2 Some Examples

Now we give a couple of search examples. One problem with full-text retrieval is that although many queries can be effective, many others are a total deception. The main reason is that a set of words does not capture all the semantics of a document. There is too much contextual information (that can be explicit or even implicit) lost at indexing time, which is essential for proper understanding. For example, suppose that we want to learn an oriental game such as Shogi or Go. For the first case, searching for Shogi will quickly give us good Web pages where we can find what Shogi is (a variant of chess) and its rules. However, for Go the task is complicated, because unlike Shogi, Go is not a unique word in English (in particular, because uppercase letters are converted to lowercase letters, see Chapter 7). The problem of having more than one meaning for a word is called *polysemy*. We can add more terms to the query, such as `game` and `Japanese` but still we are out of luck, as the pages found are almost all about Japanese games written in English where the common verb go is used. Another common problem comes from synonyms. If we are searching for a certain word, but a relevant page uses a synonym, we will not find it.

The following example (taken from [152]) better explains the polysemy problem, where the ambiguity comes from the same language. Suppose that we want to find the running speed of the jaguar, a big South American cat. A first naive search in AltaVista would be `jaguar speed`. The results are pages that talk about the Jaguar car, an Atari video game, a US football team, a local network server, etc. The first page about the animal is ranked 183 and is a fable, without information about the speed. In a second try, we add the term `cat`. The answers are about the Clans Nova Cat and Smoke Jaguar, LMG Enterprises, fine cars, etc. Only the page ranked

25 has some information on jaguars but not the speed. Suppose we try Yahoo!. We look at '`Science:Biology:Zoology:Animals:Cats:Wild_Cats`' and '`Science:Biology:Animal_Behavior`.' No information about jaguars there.

13.7.3 Teaching the User

Interfaces are slowly improving in assisting the user with the task of acquiring a better grasp of what Web pages are being retrieved. Query forms must specify clearly if one or all the words must be in a page, which words should not be in a page, etc., without using a written Boolean query language. Second, users should try to give as many terms as possible, in particular terms that must be or should not be in the pages. In particular, a user should include all possible synonyms of a word. If the user can restrict the search to a field (for example, the page title) or limit some attribute (date, country), this will certainly reduce the size of the answer. In case of doubt, the user should remember to look at the help information provided by the search engine. If he cannot find where one of the relevant terms is in a page, he can use the Find option of the browser.

Even if we are able to pose a good query, the answer can still be quite large. Considering that the visual tools mentioned before are not yet available for the general public, the user must learn from experience. There are many strategies for quickly finding relevant answers. If the user is looking for an institution, he can always try to guess the corresponding URL by using the `www` prefix followed by a guessed institution acronym or brief name and ending with a top level domain (country code or `com`, `edu`, `org`, `gov` for the US). If this does not work, the user can search the institution name in a Web directory.

If we are looking for work related to a specific topic, a possible strategy is: (1) select an article relevant to the topic, if possible with non-common author surnames or title keywords (if it is not available, try any bibliographic database or a Web directory search for a first reference); and (2) use a search engine to find all Web pages that have all those surnames and keywords. Many of the results are likely to be relevant, because we can find: (a) newer papers that reference the initial reference, (b) personal Web pages of the authors, and most important, (c) pages about the topic that already contain many relevant references. This strategy can be iterated by changing the reference used as better references appear during the search.

As mentioned at the beginning of this chapter, the Web poses so many problems, that it is easier and more effective to teach the user how to properly profit from search engines and Web directories, rather than trying to guess what the user really wants. Given that the coverage of the search engines is low, use several engines or a metasearcher. Also, remember that you have to evaluate the quality of each answer, even if it appears to be relevant. Remember that anybody can publish in the Web, and that does not mean that the data is correct or still valid. The lessons learned in the examples shown above are: (1) search engines still return too much hay together with the needle; and (2) Web directories do not have enough depth to find the needle. So, we can use the following rules of thumb:

- **Specific queries** Look in an encyclopedia, that is the reason that they exist. In other words, do not forget libraries.
- **Broad queries** Use Web directories to find good starting points.
- **Vague queries** Use Web search engines and improve the query formulation based on relevant answers.

13.8 Searching using Hyperlinks

In this section we cover other paradigms to search the Web, which are based on exploiting its hyperlinks. They include Web query languages and dynamic searching. These ideas are still not widely used due to several reasons, including performance limitations and lack of commercial products.

13.8.1 Web Query Languages

Up to this point, queries have been based on the content of each page. However, queries can also include the link structure connecting Web pages. For example, we would like to search for all the Web pages that contain at least one image and are reachable from a given site following at most three links. To be able to pose this type of query, different data models have been used. The most important are a labeled graph model to represent Web pages (nodes) and hyperlinks (edges) between Web pages, and a semi-structured data model to represent the content of Web pages. In the latter model, the data schema is not usually known, may change over time, may be large and descriptive, etc. [2, 129].

Although some models and languages for querying hypertext were proposed before the Web appeared [563, 72, 184], the first generation of Web query languages were aimed at combining content with structure (see also Chapter 4). These languages combine patterns that appear within the documents with graph queries describing link structure (using path regular expressions). They include W3QL [450], WebSQL [556, 33], WebLog [476], and WQL [511]. The second generation of languages, called Web data manipulation languages, maintain the emphasis on semi-structured data. However, they extend the previous languages by providing access to the structure of Web pages (the model also includes the internal structure) and by allowing the creation of new structures as a result of a query. Languages in this category include STRUQL [253], FLORID [373], and WebOQL [32]. All the languages mentioned are meant to be used by programs, not final users. Nevertheless, there are some examples of query interfaces for these languages.

Web query languages have been extended to other Web tasks, such as extracting and integrating information from Web pages, and constructing and restructuring Web sites. More details about Web query languages can be found in the excellent survey by Florescu, Levy, and Mendelzon [258].

13.8.2 Dynamic Search and Software Agents

Dynamic search in the Web is equivalent to sequential text searching. The idea is to use an online search to discover relevant information by following links. The main advantage is that you are searching in the current structure of the Web, and not in what is stored in the index of a search engine. While this approach is slow for the entire Web, it might be used in small and dynamic subsets of the Web. The first heuristic devised was the *fish search* [113], which exploits the intuition that relevant documents often have neighbors that are relevant. Hence, the search is guided by following links in relevant documents. This was improved by *shark search* [366], which does a better relevance assessment of neighboring pages. This algorithm has been embedded in Mapuccino (see section 13.5.3), and Figure 10.22 shows a Web subset generated by this type of search. The main idea of these algorithms is to follow links in some priority, starting from a single page and a given query. At each step, the page with highest priority is analyzed. If it is found to be relevant, a heuristic decides to follow or not to follow the links on that page. If so, new pages are added to the priority list in the appropriate positions.

Related work includes software agents for searching specific information on the Web [602, 477]. This implies dealing with heterogeneous sources of information which have to be combined. Important issues in this case are how to determine relevant sources (see also Chapters 9 and 15, as well as section 10.4.4) and and how to merge the results retrieved (the fusion problem). Examples are shopping robots such as Jango [401], Junglee [180], and Express [241].

13.9 Trends and Research Issues

The future of the Web might surprise us, considering that its massive use started less than five years ago. There are many distinct trends and each one opens up new and particular research problems. What follows is a compilation of the major trends as we have perceived them.

- **Modeling:** Special IR models tailored for the Web are needed [308, 155, 652]. As we have seen, Web user queries are different. We also have the pull/push dichotomy: Will we search for information or will the information reach us? In both cases we need better search paradigms and better information filtering [782].
- **Querying:** Further work on combining structure and content in the queries is needed as well as new visual metaphors to pose those queries and visualize the answers [44]. Future query languages may include concept-based search and natural language processing, as well as searching by example (this implies document clustering and categorization on the Web [810, 120, 157]).
- **Distributed architectures:** New distributed schemes to traverse and search the Web must be devised to cope with its growth. This will have an impact on current crawling and indexing techniques, as well as caching

techniques for the Web. Which will be the bottleneck in the future? Server capacity or network bandwidth?

- **Ranking:** Better ranking schemes are needed, exploiting both content and structure (internal to a page and hyperlinks); in particular, combining and comparing query-dependent and independent techniques. One problem related to advertisements is that search engines may rank some pages higher due to reasons that are not based on the real relevance of a page (this is called the search engine persuasion problem in [543]).
- **Indexing:** Which is the best logical view for the text? What should be indexed? How to exploit better text compression schemes to achieve fast searching and get lower network traffic? How to compress efficiently word lists, URL tables, etc. and update them without significant run-time penalty? Many implementation details must be improved.
- **Dynamic pages:** A large number of Web pages are created on demand and current techniques are not able to search on those dynamic pages. This is called the hidden Web.
- **Duplicated data:** Better mechanisms to detect and eliminate repeated Web pages (or pages that are syntactically very similar) are needed. Initial approaches are based on resemblance measures using document fingerprints [121, 120]. This is related to an important problem in databases: finding similar objects.
- **Multimedia:** Searching for non-textual objects will gain importance in the near future. There are already some research results in the literature [579, 80, 136].
- **User interfaces:** Better user interfaces are clearly needed. The output should also be improved, for example allowing better extraction of the main content of a page or the formulation of content-based queries [766].
- **Browsing:** More tools will appear, exploiting links, popularity of Web pages, content similarity, collaboration, 3D, and virtual reality [384, 638, 385, 421]. An important trend would be to unify further searching with browsing.

An important issue to be settled in the future is a standard protocol to query search engines. One proposal for such a protocol is STARTS [316], which could allow us to choose the best sources for querying, evaluate the query at these sources, and merge the query results. This protocol would make it easier to build metasearchers, but at the same time that is one of the reasons for not having a standard. In that way, metasearchers cannot profit from the work done by search engines and Web directories. This is a particular case of the federated searching problem from heterogeneous sources as it is called in the database community [656]. This is a problem already studied in the case of the Web, including discovery and ranking of sources [161, 845, 319]. These issues are also very important for digital libraries [649] (see also Chapter 15) and visualization issues [15]. A related topic is metadata standards for the Web (see Chapter 6) and their limitations [544]. XML helps [436, 213, 306], but semantic integration is still needed.

Hyperlinks can also be used to infer information about the Web. Although this is not exactly searching the Web, this is an important trend called Web mining. Traditionally, Web mining had been focused on text mining, that is, extracting information from Web pages. However, the hyperlink structure can be exploited to obtain useful information. For example, the ParaSite system [736] uses hyperlink information to find pages that have moved, related pages, and personal Web pages. HITS, already mentioned in Section 13.4.4, has also been used to find communities and similar pages [444, 298]. Other results on exploiting hyperlink structure can be found in [639, 543, 154]. Further improvements in this problem include Web document clustering [810, 120, 162] (already mentioned), connectivity services (for example, asking which Web pages point to a given page [92]), automatic link generation [320], extracting information [100, 115], etc.

Another trend is intranet applications. Many companies do not want their private networks to be public. However, for business reasons they want to allow Web users to search inside their intranets obtaining partial information. This idea leads to the concept of *portals* for which there are already several commercial products. New models to see Web sites as databases and/or information systems are also important.

13.10 Bibliographic Discussion

There are hundreds of books about the Web. Many of them include some information about searching the Web and tips for users. A recent book edited by Abrams includes a chapter on searching the Web [3]. Other sources are [682], the special numbers of *Scientific American* on the Internet (March 1997) and IEEE's *Internet Computing on Search Technologies* (July/August 1998). For details about crawlers and other software agents see [166, 817].

In addition, the best source for references to the Web is the Web itself. To start with, there are many Web sites devoted to inform and rate search engines and Web directories. Among them we can distinguish Search Engine Watch [749] and Search Engine Showdown [609]. A survey about Web characterizations is given by Pitkow [641] and a good directory to Web characteristics is [217]. Other Web pages provide pointers and references related to searching the Web, in particular the World Wide Web Consortium (`www.w3.org`), the World Wide Web journal (`w3j.com`) and WWW conferences. These and other pointers are available in the Web page of this book (see Chapter 1).

Acknowledgements

We would like to thank the following for their helpful comments: Omar Alonso, Eric Brown, Pablo de la Fuente, Monika Henzinger and Gonzalo Navarro.

Chapter 14
Libraries and Bibliographical Systems

by Edie M. Rasmussen

14.1 Introduction

Despite the image sometimes presented of libraries as archaic collections of dusty books accessed through a card catalog, libraries were among the earliest institutions to make use of information retrieval systems. This early adoption took two main forms: searching remote electronic databases provided by commercial vendors in order to provide reference services to patrons, and the creation and searching of catalog records for materials held within the library. Each of these applications followed different developmental paths resulting in different products and functionality. According to Hildreth [372],

> Proceeding along different paths, the developmental histories of on-line public access catalogs (OPACs) and conventional information retrieval (IR) systems differed in three respects: origins of systems development, file and database content, and intended users. (p.10)

Initial development of information retrieval systems was carried out by government laboratories in support of research in science and technology, based on bibliographic databases containing largely textual information, with trained search intermediaries as the intended users. OPACs were developed initially inhouse by large, usually academic, library systems, and later by commercial vendors of turnkey systems.† They used standardized record formats, generally the MARC record with minimal subject information (title, a few subject headings, and a classification number); and unlike commercial IR systems, they were intended from the outset for end users (library patrons). These factors led to significant differences between commercial IR systems and OPACs.

† 'Turnkey systems' include software (and often hardware) and are usually developed with a specific library type and size in mind; within the constraints of the system, some customizing to suit the particular library is often possible.

Developed independently of each other, information retrieval systems and OPACs are quite different in character and use, and will be treated separately in this chapter. For these applications, a brief history, overview of current trends, some sample records and search examples will be given, and profiles of well-known systems will be presented. (Topics related to the *use* of IR systems in libraries, through Reference and Technical Services departments, and the techniques by which reference librarians perform the reference function, are beyond the scope of this chapter.) An important recent phenomenon, the digital library (see Chapter 15), has the potential to integrate information retrieval functions in the library under a common interface, eliminating the distinction between locally held and remote resources. Some examples of libraries which have attempted this integration will be discussed.

14.2 Online IR Systems and Document Databases

A synergistic relationship exists between the producers and vendors of document databases‡ (see Figure 14.1). In general, database producers create a product which they license to the database vendors. These vendors or search services provide search software and access to their customers, who benefit from the ability to search multiple databases from a single source.

It is common to speak of the online database industry, since production of databases has usually been undertaken by corporations, organizations, or government on a for-profit or cost-recovery basis. These database producers have seen databases as products for sale or lease, often to libraries, and usually by a third party or database vendor. The role of database vendor is to license databases from their producers and add value by making them available to users. Database vendors provide some degree of standardization to the record formats, create indexes (usually in the form of inverted files), and provide a common interface for searching multiple databases. Examples of well known database vendors are DIALOG, LEXIS-NEXIS, OCLC, and H.W. Wilson; profiles are given in Figure 14.2. Some database producers choose to serve as their own search service providers, leading to a degree of vertical integration within the database industry; examples are the National Library of Medicine (NLM), which provides free access to its Medline database through the Web, and the H.W. Wilson Company, which markets its own series of databases.

A significant aspect of these major commercial services is the very large size of their databases and the need for rapid, reliable service for many simultaneous users. In a description of their computing complex, LEXIS-NEXIS [510] give their database size as 1.3 billion documents, with 1.3 million subscribers, and 120 million annual searches. They return an answer set within six to ten seconds,

‡ 'Database' is commonly used by producers and vendors of document databases when referring to their product. These databases lack the tabular structure of relational databases and contain bibliographic information and/or the full-text of documents. This usage will be followed in this chapter.

Database Producers:

- design database structure
- collect in-scope literature
- enter bibliographic information in standard form
- abstract (or edit authors' abstracts)
- index with (usually) controlled vocabulary
- generate file updates at regular intervals
- market backfile and updates to vendors

Database Vendors:

- create search software
- license databases from producers
- standardize (as possible) record structure
- mount databases, creating inverted indexes
- update databases as appropriate (daily, weekly, monthly)
- provide documentation for searchers
- market to clients
- provide service and training to client base

Figure 14.1 Role of database producers and vendors.

with a claimed availability above 99.99% and reliability of 99.83%. Similarly, DIALOG claims to be over 50 times the size of the Web.

14.2.1 Databases

The history of commercial online retrieval systems begins with the creation of databases of bibliographic information in electronic form. In fact, Neufeld and Cornog claim 'databases can almost be said to have created the information industry as we now know it' [600]. Abstracting and indexing tools in printed form were available in the nineteenth century and became increasingly available in the twentieth century. Professional organizations, commercial firms, and government bodies served as publishers, selecting relevant materials from the world's literature, creating bibliographic records for them, and providing abstracts and indexing information. These databases were concentrated in the sciences, with titles such as *Chemical Abstracts*, *Biological Abstracts*, and *Engineering Index*, but humanities (*Historical Abstracts*) and social sciences (*PsycINFO*) products soon became available.

As publishers of abstracts and indexes turned to computer-assisted typesetting and printing for their products, the resulting magnetic tapes of information began to be used for information retrieval purposes. Today virtually all print abstracting and indexing products are also available in electronic form, and many new products are available solely in electronic form, without a print equivalent. As storage costs have dropped dramatically, many of these electronic databases

- **The DIALOG Corporation**

  DIALOG, 'the world's largest online information company,' contains about 500 databases covering a full range of subjects, including science, technology and medicine, humanities, business, and electronic newspapers. Bibliographic and full-text databases are included. Some databases are also available in CD-ROM versions for onsite searching.
  URL: `http://www.dialog.com`

- **LEXIS-NEXIS**

  LEXIS-NEXIS markets full-text databases to the legal and business community. LEXIS provides access to 4800 legal research products including state and federal case law, statutes, and regulations. NEXIS covers over 18,000 news and business sources.
  URL: `http://www.lexis-nexis.com`

- **OCLC**

  OCLC (the Online Computer Library Center, Inc.), which began as a bibliographic utility for cooperative cataloging of library materials, now offers access to over 70 databases and 1.5 million full-text articles. Features include an interface oriented to end-users and links to documents as well as to an inter-library loan module.
  URL: `http://www.oclc.org/oclc/menu/fs.html`

- **H.W. Wilson**

  H.W. Wilson began producing print indexes in 1898, and now offers 40 databases to the public, school, and college library market. Wilson has electronic, CD-ROM, magnetic tape, and Web-based versions of its databases.
  URL: `http://www.hwwilson.com/default.htm`

Figure 14.2 Profiles of database vendors.

have expanded to include not only bibliographic information about documents, but the text of the documents themselves. These are referred to as full-text databases, and include databases of journal articles and newspapers as well as reference materials such as encyclopedias and directories. Characteristics of some common databases (as available on DIALOG) are given in Figure 14.3.

Databases and Indexing

In general, bibliographic databases are expensive to produce, because they require rigorous selection and analysis of the documents that they cover. Some databases cover materials in a specific group of journals, others attempt to be comprehensive, collecting the world's literature within the defined subject scope. Every item must be examined for relevance to the database's goals, indexed,

- **CA SEARCH: Chemical Abstracts**

  Coverage: bibliographical records for worldwide literature of chemistry and its applications
  File size: 14 million records; weekly updates of 11,000 records

- **MEDLINE**

  Coverage: the broad field of biomedicine, including clinical and experimental medicine, dentistry, nursing, pharmacology, psychiatry, etc. It indexes articles from 3,700 journals worldwide
  File size: about 9.5 million records; weekly updates of 7700 records

- **New York Times - Fulltext**

  Coverage: full-text of *New York Times* from 1981 to the present
  File size: 1.8 million records; daily updates

- **PsycINFO: Psychological Abstracts**

  Coverage: bibliographic records for materials in psychology and related behavioral and social sciences, including psychiatry, sociology, anthropology, education, pharmacology, and linguistics; 1887 to the present
  File size: 1.5 million records; monthly updates of 5000 records

Figure 14.3 Characteristics of some well known databases on DIALOG.

abstracted, and entered in the system. Despite the promise of SGML tagging of materials by primary producers, most of this work is still done by the database producer, with a clerical staff to handle data input and subject specialists to abstract (more commonly, edit the author's abstract) and index the material.

Each bibliographic database is a unique product designed to meet the information needs of a particular user group. Therefore, there is no single standard for the content of a database record. Typically, it contains tagged information that includes a record key, bibliographic data such as author, title, and source of the document, an abstract, and subject indicators such as indexing terms or category codes. In full-text databases (see Chapters 2 and 4), the text of the document is also included. Sample database records from *BIOSIS PREVIEWS* (*Biological Abstracts*) and *Historical Abstracts* are shown in Figures 14.4 and 14.5. Note that the vocabulary (descriptors and codes) used for subject description is very dependent on the field of study (in this case, biology and history).

As these database records show, the subject information they contain is of two types: so-called 'natural language' or 'free text' information found in the title or abstract field, and terms from an indexing or controlled vocabulary which are assigned by human indexers. Most databases include indexing terms in a descriptor field, usually taken from a database-specific thesaurus (e.g., for *PsycINFO*, the *Thesaurus of Psychological Index Terms*). Other types of codes or indexing may be applied as relevant to the database (for instance, biosystematic

```
DIALOG(R)File 5:BIOSIS PREVIEWS(R)
(c) 1998 BIOSIS. All rts. reserv.
13165209 BIOSIS Number: 99165209
Population genetics of the Komodo dragon Varanus komodoensis
Ciofi C; Bruford M; Swingland I R
D.I.C.E., Univ. Kent, Kent, UK
Bulletin of the Ecological Society of America 77 (3 SUPPL. PART 2). 1996. 81.
Full Journal Title: 1996 Annual Combined Meeting of the Ecological Society of
America on Ecologists/Biologists as Problem Solvers, Providence, Rhode Island,
USA, August 10-14, 1996. Bulletin of the Ecological Society of America
ISSN: 0012-9623
Language: ENGLISH
Document Type: CONFERENCE PAPER
Print Number: Biological Abstracts/RRM Vol. 048 Iss. 010 Ref. 171812
Descriptors/Keywords: MEETING ABSTRACT; VARANUS KOMODOENSIS;
KOMODO DRAGON; MONITOR LIZARD; GENETIC DIVERGENCE; GENE
FLOW; EVOLUTION; GENETIC DIVERSITY; SPECIES RANGE; POPU-
LATION SIZE; POPULATION GENETICS; LESSER SUNDA REGION; IN-
DONESIAN ISLANDS; ORIENTAL REGION; KOMODO; RINCA; FLORES;
GILI MOTANG; INDONESIA
Concept Codes:
03506 Genetics and Cytogenetics-Animal
03509 Genetics and Cytogenetics-Population Genetics (1972- )
07508 Ecology; Environmental Biology-Animal
62800 Animal Distribution (1971- )
00520 General Biology-Symposia, Transactions and Proceedings of Conferences,
Congresses, Review Annuals
Biosystematic Codes:
85408 Sauria
Super Taxa:
Animals; Chordates; Vertebrates; Nonhuman Vertebrates; Reptiles
```

Figure 14.4 Sample record: *BIOSIS PREVIEWS*.§

codes in *BIOSIS PREVIEWS*, historical time periods in *Historical Abstracts*). The assignment of these subject terms contributes significantly to the cost of database production. Obviously an automated indexing system would be of interest to database producers, though production systems currently in use are best described as performing 'machine-assisted' rather than automatic indexing.

§ With permission of BIOSIS UK. The format of this record has now changed as BIOSIS now use New Relational Indexing

```
DIALOG(R)File 39: Historical Abstracts
(c) 1998 ABC-CLIO. All rts. reserv.
1488625 47A-9910
THE U.S.S. KEARSARGE, SIXTEEN IRISHMEN, AND A DARK AND
STORMY NIGHT.
Sloan, Edward W
American Neptune 1994 54(4): 259-264.
NOTE: Based on primary sources, including the Official Records of the Union
and Confederate Navies in the War of the Rebellion, Series I and II (1894-1927);
28 notes.
DOCUMENT TYPE: ARTICLE
ABSTRACT: Tells the story of the Union navy's Kearsarge, a sloop-of-war that
patrolled English seas looking for Confederate commerce raiders. Upon dock-
ing at the Irish port of Cobh (Queenstown) in November 1863, 16 locals stowed
away. They were subsequently returned to Cobh, but in the meantime Captain
John Winslow temporarily enlisted the men in order, he said, that they be jus-
tifiably clothed and fed, although other ship diaries indicate that the ship was
short-handed and Winslow intended a real enlistment. Whatever the reality, the
captain inadvertently created an international crisis since his action technically
violated the British Foreign Enlistments Act. It is unclear whether Confederates
plotted the incident to embarrass the Union in Britain because there are dispar-
ities between official accounts and the diaries of individual crewmen. (S )
DESCRIPTORS: USA ; Civil War ; Ireland -(Cobh) ; Kearsarge -(vessel) ; Po-
litical Crisis ; Military Service ; Stowaways ; 1862-1864
HISTORICAL PERIOD: 1860D 1800H
HISTORICAL PERIOD (Starting): 1862
HISTORICAL PERIOD (Ending): 1864
```

Figure 14.5 Sample record: *Historical Abstracts*. From ABC-CLIO,CA,USA.

A subject of early (and ongoing) research has been the relative value of 'free text' and controlled vocabulary terms in contributing to retrieval performance. This subject was addressed in the Cranfield studies in the 1960s [415], and has continued to be examined by researchers up to the present time; good reviews of this research have been presented by Svenonius [752], Lancaster [479], and Rowley [688]. No definitive answer has been found, though later studies seem to suggest a complementarity between the two types of indexing in promoting good retrieval.

14.2.2 Online Retrieval Systems

The use of the computer for bibliographic information retrieval was first demonstrated in the 1950s, and initiated by the National Library of Medicine in 1964 using batch processing [107]. Also in the 1960s, federally funded projects were carried out to develop prototype online systems which were then implemented in government research laboratories. The first production service, Lockheed's

DIALOG system, was implemented for NASA and subsequently made available to other government locations before becoming a commercial activity in the early 1970s and undergoing several changes in ownership. Today DIALOG operates worldwide with databases offered via the Internet to libraries and other organizations as well as individuals.

With a few exceptions, database vendors do not produce information but rather make it available to searchers via a common search interface. Database vendors license databases from the producers, process the databases to introduce as much standardization as is feasible (e.g., standard field names), mount the database through the creation of inverted indexes, create database descriptions and aids to searchers in a standard format, and conduct training sessions for clients (see Figure 14.1). These organizations offer a value-added service by providing a common gateway to multiple databases. A database vendor may offer cross-database searches; for example, DIALOG allows the searcher to search simultaneously a predetermined or searcher-selected grouping of databases to create a merged set of references, then process the set to remove duplicates.

14.2.3 IR in Online Retrieval Systems

Since the inception of these online retrieval services, their retrieval functionality has been based primarily on the Boolean model for retrieval, in contrast to research in the IR field which has focused on improving retrieval performance through non-Boolean models, such as the vector space model (see Chapter 2). A number of factors guided the choice of the Boolean model as the basis for these services. Research in indexing and retrieval at the time, particularly the Cranfield studies, a series of experiments comparing natural and controlled vocabulary indexing, suggested that 'natural language' retrieval provided a level of retrieval performance comparable to manual indexing. Boolean logic was already being used in some libraries for manual retrieval systems, such as edge-notched cards and optical coincidence cards, and seemed to offer a natural mechanism for implementing retrieval based on combinations of words in documents. Research on alternate retrieval models was in its infancy, and the effectiveness of these models had not been proven for large databases. Most significantly, perhaps, the limited processing and storage capability of the computers of the time, while enough to support the inverted file structures and logical operations required for Boolean retrieval in an online environment, could not provide real time retrieval performance for other retrieval models which were more computationally intensive.

Despite developments in IR research which suggested that alternative models might provide improved retrieval performance, Boolean retrieval has remained the commonest access method offered by database vendors, although in recent years some systems have added a form of natural language input with ranked output processing as an alternative access method. Reasons that have been suggested for the predominance of Boolean searching include financial considerations (cost of major changes in search software and database structures), service issues (a client community trained on existing systems), and lack of evidence in support of viable alternatives in operational environments [662].

In general, database vendors use proprietary search software which is specific to their system, so that information professionals who search multiple systems are required to learn a different command vocabulary for each. A standard has been developed for a Common Command Language, NISO Z39.58 or ISO 8777, as described in Chapter 4, but it does not substitute for the advanced search features which are unique to individual search systems. The basic functionality for an IR search system is the ability to search for single terms or phrases, or Boolean combinations of them, to create sets of documents that can be further manipulated, then printed or displayed. Typically the system will also offer the option of using proximity operators to specify term relationships (A adjacent to B, A within n words of B, etc.) as discussed in Chapter 5, and to specify the location of the search term within the record (A occurring in title field, B occurring in the descriptor field, etc.). Of course, these capabilities require the storage of a significant amount of positional information within the inverted index. Other functions that may be available are the ability to browse the database index to select search terms (see Chapter 10) or to follow the term relationships within a database thesaurus to find candidate search terms (see Chapter 7). Other, more sophisticated functions, perhaps associated with a specific category of database, are also available, such as the ability to conduct structural searches for compounds in a chemistry database.

As a term is entered by a searcher, the system creates a 'set' corresponding to all documents containing that term, and assigns a set number for the searcher's use. Multiple sets of retrieved documents are maintained in temporary storage. These set numbers serve as surrogates for the document set when issuing search commands, and Boolean logic can be used to manipulate existing sets. A display command allows the searcher to review the search history and return to previous sets. Based on data about the size of a set retrieved with search term or expression, and a review of the associated documents and their indexing, searchers continually revise a search until they feel they have achieved the best possible outcome. This iterative process is as much art as science, and its success is highly dependent on the skill and subject knowledge of the searcher.

A typical Boolean search on DIALOG is shown in Figure 14.6. In this search, the user requests a specific database (file 61, *Library and Information Science Abstracts*) and then uses the 'Select Steps' or `ss` command to create sets of records. The '`(w)`' represents a proximity operator, so set 5 (S5) will contain all records containing the phrases 'document retrieval' or 'text retrieval' or 'information retrieval.' Set 13 (S13) will contain all records containing the term 'OPAC' or the phrase 'online public access catalog.' The '`?`' is a truncation operator, and '`? ?`' limits truncation to one letter, so alternate spellings and plural of 'catalog' and the singular or plural of 'OPAC' will be retrieved. The two sets are combined with a Boolean AND operator, and finally the set is further limited to records that contain the terms in the title (`ti`) or descriptor (`de`) field, resulting in 100 records for review.

```
begin 61
File  61:LISA(LIBRARY&INFOSCI)  1969-1998/May
       (c) 1998 Reed Reference Publishing

      Set  Items  Description
      ---  -----  -----------
? ss (document or information or text)(w)retrieval
      S1    7363  DOCUMENT
      S2   92299  INFORMATION
      S3    6219  TEXT
      S4   29302  RETRIEVAL
      S5   15338  (DOCUMENT OR INFORMATION OR TEXT)(W)RETRIEVAL
? ss opac? ? or online(w)public(w)access(w)catalog?
      S6    1111  OPAC? ?
      S7   20922  ONLINE
      S8   32238  PUBLIC
      S9   16388  ACCESS
     S10   18798  CATALOG?
     S11     424  ONLINE(W)PUBLIC(W)ACCESS(W)CATALOG?
     S12    1246  OPAC? ? OR ONLINE(W)PUBLIC(W)ACCESS(W)CATALOG?
? s s5 and s12
           15338  S5
            1246  S12
     S13     146  S5 AND S12
? s s13/ti,de
     S14     100  S13/TI,DE
```

Figure 14.6 A DIALOG search.

14.2.4 'Natural Language' Searching

To ensure their place in the market, database vendors continually develop new features that they feel will be of value to their client group, as well as add new database products. In general, these new features are augmentations to the existing Boolean search engine — removal of duplicates, sophisticated ranking or sorting within the retrieved set. However, about five years ago several of the major database vendors announced they were adding 'natural language' search functionality to their systems. WESTLAW (a legal resources vendor) introduced its WIN system, DIALOG offered TARGET, and LEXIS-NEXIS announced a system called FREESTYLE [758, 653]. WIN and FREESTYLE accept a natural language query; TARGET requires the searcher to eliminate terms that are not useful for searching. All three systems provide ranked lists of retrieved documents. The 'natural language' systems are offered as auxiliary modules to standard Boolean searching, and are not intended to replace it. A sample TARGET search is shown in Figure 14.7.

In this search in BIOSIS, the searcher is first provided with a series of instructions on dealing with phrases, synonyms, etc. The searcher enters a series of search terms (up to 25) at the '?' prompt, in this case '`komodo dragon food`

```
? target
Input search terms separated by spaces (e.g., DOG CAT FOOD). You
can enhance your TARGET search with the following options:
   - PHRASES are enclosed in single quotes
       (e.g., 'DOG FOOD')
   - SYNONYMS are enclosed in parentheses
       (e.g., (DOG CANINE))
   - SPELLING variations are indicated with a ?
       (e.g., DOG? to search DOG, DOGS)
   - Terms that MUST be present are flagged with an asterisk
       (e.g., DOG *FOOD)
Q = QUIT   H = HELP
? komodo dragon food diet nutrition
Your TARGET search request will retrieve up to 50 of the
statistically most relevant records.
Searching 1997-1998 records only
...Processing Complete
    Your search retrieved 50 records.
Press ENTER to browse results  C = Customize display  Q = QUIT
H = HELP
```

Figure 14.7 A TARGET search on DIALOG.

`diet nutrition`'. By default the search is limited to the most recent two years of the file, and the 50 highest scoring records are available for display in ranked order. In this example no restrictions are made on the search terms but as the on-screen instructions indicate, Boolean logic can be imposed on the search terms, resulting in a Boolean search with ranked output.

14.3 Online Public Access Catalogs (OPACs)

Library catalogs serve as lists of the library's holdings, organized as finding tools for the collection. For many years the card catalog served this function, and later computer-produced catalogs in book, microfilm, and microfiche form. Online catalogs were implemented in libraries during the 1970s, although these first catalogs were usually modules linked to the automated circulation system and had brief catalog records and very limited functionality. (The circulation system was the first component of what are now called library management systems (LMSs) or integrated library systems (ILSs) to be introduced). By the 1980s, true online public access catalogs had been implemented.

Hildreth [372] has described the history of online catalogs by classifying them according to three generations. In the first generation, OPACs were largely known-item finding tools, typically searchable by author, title, and control number, and contained relatively short, non-standard bibliographic records. As is typical of technologies in their infancy, they were basically an old technology (the card catalog) in an automated form. In the second generation, increased search functionality included access by subject headings and, latterly, keyword, some basic Boolean search capability, and ability to browse subject headings. Second generation catalogs also offered a choice of display formats (e.g., short, medium, long) and improved usability (for instance, different dialogs for novices and experts, more informative error messages, etc.). According to Hildreth, problems with second generation systems included failed searches, navigational confusion, problems with the subject indexing vocabulary and excessively large, badly organized retrieval sets.

Needed enhancements for third generation systems, as delineated by Hildreth, included search strategy assistance, integrated free text/controlled vocabulary approaches, augmented cataloging records, cross-database access, natural language input, individualized displays and context-sensitive error correction. For many years library catalogs remained on what Hildreth referred to as the 'second generation plateau.' One of the barriers to innovation in OPAC development has been the cost of developing new systems and the need for a reliable customer base. From the perspective of the library, selecting and migrating to a new system is a costly process, and with library budgets traditionally squeezed, libraries have been cautious in selecting new and untried systems. They have learned to be wary of the *'it's in the next release'* syndrome, while system developers have required a stable customer base to fund new systems.

Third generation systems are now appearing, and with features not envisioned by Hildreth, who was speaking in a pre-Web environment. The availability of electronic resources on the Web has blurred the distinction between local and global resources, and between cataloging information and other electronic databases. According to a recent vendor survey [632],

> Automated system vendors have a vested interest in the transition of libraries to a mixed digital/print environment. Many see their own survival dependent upon their ability to help libraries thrive in this mixed arena. (p.47)

Therefore, much of the emphasis in recent library systems development has been on the deployment of functionality for library management systems within new open systems architectures [351]. Features appearing in these new systems include improved graphical user interfaces (GUIs), support for Z39.50, electronic forms, hypertext links and Dublin Core (a developing metadata standard for multimedia materials), and incorporation of Java programming. Systems are also beginning to move beyond the basic Boolean search functionality, and some, like EOSi's Q series (described in section 14.3.3) have advanced search features.

14.3.1 OPACs and Their Content

Libraries use standardized systems for cataloging and classifying the materials (texts and other media) they hold. Typically, they follow the Anglo-American Cataloging Rules to describe these materials, an organizational scheme (such as Library of Congress or the Dewey Decimal Classification) to assign subject codes, and use a subject heading list (such as the Library of Congress Subject

```
00723cam  22002418a 4500001001300000008004100013005001700054
0100018000710200033000890400013001220500026001350820017001 61
1000020001782450074001982500012002722600052002843000034003 36
5040064003706500041004 34
001  0013   97002718
008  0041   970417s1997    ilua     b    001 0 eng
005  0017   19971128134653.1
010  0018     $a   97002718
020  0033     $a0838907075 (acid-free paper)
040  0013     $aDLC$cDLC
050  0026   00$aZ699.35.M28$bH34 1997
082  0017   00$a025.3/16$221
100  0020   1 $aHagler, Ronald.
245  0074   14$aThe bibliographic record and information
            technology / $cRonald Hagler.
250  0012     $a3rd ed.
260  0052     $aChicago :$bAmerican Library Association,$c1997.
300  0034     $axvi, 394 p. :$bill. ;$c24 cm.
504  0064     $aIncludes bibliographical references (p.375-380)
            and index.
650  0041    0$aMachine-readable bibliographic data.#
```

Figure 14.8 Sample MARC record.

Headings) to assign a series of subject descriptors. Given this standardization, cooperative cataloging ventures by library consortia have the potential to lower the cost per unit to catalog library materials, broaden access through shared databases, and facilitate the sharing of materials. Thus library cataloging relies on centralized and shared information through bibliographic utilities such as the Online Computer Library Center (OCLC). (OCLC is also a database vendor with characteristics shown in Figure 14.2.)

The structure that underlies this cooperation among many libraries supporting distinct online catalogs is the MARC Record. MARC (Machine Readable Cataloging Record) is a data format that implements national and international standards, such as the Information Interchange Format (ANSI Z39.2) and the Format for Information Exchange (ISO 2709). With some variations (USMARC, UKMARC, etc.) it is used worldwide. A sample MARC record is shown in Figure 14.8.

The MARC record has three parts: a fixed length (24 character) leader; a record directory showing the 3-digit tag for each field contained in the record with the length of that field in characters; and the data-containing fields and subfields themselves. Subfields are indicated by codes (e.g., '$a') within the field and are specific to each field. For instance, field 260 contains publication information and may have subfields for place, publisher, and date. (To improve readability the record here has been reformatted slightly, so that the field tag (e.g., 001) and field length (e.g., 0013) from the directory are repeated with the data for each field). A recent innovation is the adoption of the 856 field for holdings information to include URLs, allowing the specification of Web hyperlinks.

14.3.2 OPACs and End Users

Probably the greatest challenge for designers of OPACs is to create usable systems. OPACs are found in every type of library, and while users of research libraries might be expected to be knowledgeable about library practices in organizing and accessing information, elsewhere the end user could as easily be a schoolchild, college undergraduate, or patron of a local public library with little or no formal training in library use (what Borgman calls 'perpetual novices' [105]). The underlying record structure (the MARC record) is detailed and complex, and the organizational structures (LCSH, LC classification scheme) are far from intuitive.

The most common type of searching in OPACs is subject searching, and failures by users in topical searching are well documented [484]. Common failures are null sets ('zero results'), or at the other extreme, information overload in which more references are retrieved than can easily be examined [484]. According to one study of transaction logs for the MELVYL catalog [252], 82% of in-library users had a zero retrieval for one or more searches. Interestingly, over 25% of users continued their search through ten or more tries, and another 25% did not appear to retrieve any useful information. Writing in 1986, Borgman [104] raised the question, 'Why are online catalogs hard to use?,' and in 1996, revisited the problem with 'Why are online catalogs *still* hard to use?' [105]. She argues the reason is that they do not incorporate knowledge about user behavior, and place too heavy a burden on the searcher for query specification. Greater contextual assistance for searchers has been suggested by a number of researchers [105, 252, 371].

14.3.3 OPACs: Vendors and Products

The OPAC market is a specialized one, and products are developed and marketed by a limited number of vendors who compete for market position. While it is rare to find a library of any size that does not have a library management system, libraries are constantly in a state of flux, upgrading their systems as old ones become obsolete or unsupported, and introducing new systems. For example, many academic libraries had OPACs based on the venerable mainframe-based NOTIS software, and have undertaken to identify a suitable replacement. Most of the vendors target niche markets: academic libraries, public libraries, and school and special libraries. Profiles of three such vendors are found in Figure 14.9. Fuller details of these and other systems can be found in [351] and [61].

14.3.4 Alternatives to Vendor OPACs

While early OPACs were developed inhouse, sometimes by enthusiastic amateurs at considerable expenditure of time and money, and a significant risk of failure,

- **Endeavor Information Systems, Inc.**

  With a significant academic library clientele, Endeavor has replaced a number of NOTIS systems. Its system, Voyager, is based on a multi-tier architecture with Oracle as the DBMS. The public access client and server are Z39.50 compliant. The search engine supports natural language queries and relevance ranking to display results.
  URL: `http://www.endinfosys.com`

- **Innovative Interfaces, Inc. (III)**

  A large company for this industry, III has an academic library customer base, and also a public library presence. Its newest system, Millennium, is based on its INNOPAC library management system but adds a thin client architecture with modules developed in Java. In addition to its own search engine, INNOPAC uses one licensed from Fulcrum Technologies. In Millennium, relevance ranking is available for full-text searching.
  URL: `http://www.iii.com`

- **EOS International (EOSi)**

  EOSi markets to smaller libraries; it has a large special library clientele plus a significant academic, public, and school library customer base. Its Q series of library management system tools uses a three-tier, client/server architecture. The search engine is Excalibur RetrievalWare, on license from Excalibur Technologies. Standard Boolean searching is available but greater functionality is supplied by natural language entry, dictionary-based query expansion, fuzzy search for bad data, and relevance ranked output.
  URL: `http://www.eosintl.com`

Figure 14.9 Library management system vendors.

today's environment supports turnkey systems developed by a third party. However, there are some instances of systems developed with a research focus for implementation in academic libraries. Notable examples are the Okapi system [416] at City University, London, MARIAN [264] at Virginia Tech, the MELVYL system at the University of California [526], and the Cheshire II system [486] for a UC Berkeley branch library.

The Cheshire II system was designed for the UC Berkeley Mathematics, Statistics and Astronomy library using standards such as Z39.50 and SGML. It provides integrated access to bibliographic, full-text and multimedia resources. The search engine offers both probabilistic ranking and Boolean searches, which can be combined in a single search. Cheshire II was designed as a research as well as an operational environment, and issues such as combining probabilistic and

Boolean models, and design of the client interface to support searching with a variety of Z39.50 servers while minimizing cognitive overload on searchers [486].

14.4 Libraries and Digital Library Projects

Libraries are concerned with enhanced, seamless access to electronic information from all sources. These libraries [351]

> see the library's Web pages, not the OPAC, as the entry point for library users. Through the web pages the user gains access to the library catalog, networked information resources, and locally created information. (p.5)

Through the Web, a single interface can provide access to the local OPAC and reference materials, as well as to remotely accessible databases in the sciences, humanities, and business, including full-text journals, newspapers, and directories. Special collections, in multimedia as well as text formats, become available to the user through the same gateway. Many libraries, particularly academic and large public libraries, have undertaken digital library projects to achieve interoperability, ease of use, and equity of access (see Chapter 15). Two such projects, the Los Angeles Public Library's Virtual Electronic Library project (`http://www.lapl.org`), and University of Pennsylvania's Digital Library (`http://www.library.upenn.edu`) are described in [351].

The Web not only provides integration in terms of resources and collections, but the accompanying standards which support interoperability lead to a uniform search architecture. With this approach, the traditional distinction between information retrieval from OPACs and from remote electronic databases is beginning to disappear.

14.5 Trends and Research Issues

With a few exceptions, librarians are consumers of information systems, whether information retrieval systems provided by database vendors, or turnkey OPACs. Even in the digital library environment, their emphasis is on providing integrated access to a diversity of modules for information retrieval. Their interest therefore is in obtaining and using systems which offer ease of integration in their automated environment, and ease of use for themselves and their patrons. The former goal is approached through standards such as SGML and Z39.50, and the development and application of these standards is an important trend in the design of IR systems for libraries. For the latter goal, ease of use, the trend toward user-centered research and design is significant because it offers the potential to answer Borgman's query, 'Why are online catalogs *still* hard to use?' [105]. Much of the recent research interest is in cognitive and behavioral issues (as reviewed in [482]). Developing an understanding of information need, either in general or for a specific client group, has been an important component of this work.

Researchers are also interested in the searching behavior of users. Obviously, there is no single 'user' group, and studies have focused on groups such as trained intermediaries, children, and subject specialists, in both the search service and OPAC environment. One such project conducted over two years is the Getty Online Search Project which studied the end user search behavior of humanities scholars [67]. The interest in end user behavior also extends to an examination of relevance, since an understanding of the criteria by which users determine if retrieved information meets their information need is critical to achieving user-centered design.

14.6 Bibliographic Discussion

The early history of online databases and systems makes interesting reading, and Hahn's 'Pioneers of the Online Age' is a good place to start [331]. The early history of online systems is also described by Bourne [107], and the history of electronic databases by Neufeld and Cornog [600]. The current status of the online industry is profiled annually in the May 1 issue of *Library Journal* (see, for example, [757]).

An overview of research issues in OPACs is provided by Large and Beheshti [482]. A 1996 issue of the *Journal of the American Society for Information Science* was a special topic issue on 'Current Research in Online Public Access Systems' [68]. Comparative information on OPACs (and other library management system software) is readily available. A Council on Library and Information Resources report profiles 12 major vendors and their products [351]. The April 1 issue of *Library Journal* each year includes an 'Automated System Marketplace' update which discusses trends in library management systems, provides company and product information, and tabulates sales. *Library Technology Reports* frequently publishes 'consumer reports' of online systems; for instance, one issue was devoted to a survey of Z39.50 clients [813].

Recent monographs by Allen [13] and Marchionini [542] address the issues of user-centered design and electronic information seeking behavior.

Chapter 15
Digital Libraries

by Edward A. Fox and Ohm Sornil

The benefits of digital libraries will not be appreciated unless they are easy to use effectively. [525]

15.1 Introduction

Information retrieval is essential for the success of digital libraries (DLs), so they can achieve high levels of effectiveness while at the same time affording ease of use to a diverse community. Accordingly, a significant portion of the research and development efforts related to DLs has been in the IR area. This chapter reviews some of these efforts, organizes them into a simple framework, and highlights needs for the future.

Those interested in a broader overview of the field are encouraged to refer to the excellent book by Lesk [501] and the high quality papers in proceedings of the ACM Digital Libraries Conferences. Those more comfortable with online information should refer to *D-Lib Magazine* [280]; the publications of the National Science Foundation (NSF), Defense Advanced Research Projects Agency (DARPA), and National Aeronautics and Space Administration (NASA) 'Research on Digital Libraries Initiative' (DLI) [349]; or online courseware [268]. There also have been special issues of journals devoted to the topic [265, 267, 710]. Recently, it has become clear that a global focus is needed [270] to extend beyond publications that have a regional [55] or national emphasis [221].

Many people's views of DLs are built from the foundation of current libraries [683]. Capture and conversion (digitization) are key concerns [160], but DLs are more than digital collections [634]. It is very important to understand the assumptions adopted in this movement towards DLs [509] and, in some cases, to relax them [29].

Futuristic perspectives of libraries have been a key part of the science fiction literature [811] as well as rooted in visionary statements that led to much of the work in IR and hypertext [135]. DLs have been envisaged since the earliest days of the IR field. Thus, in *Libraries of the Future*, Licklider lays out many of the challenges, suggests a number of solutions, and clearly calls for IR-related efforts [516]. He describes and predicts a vast expansion of the world of publishing, indicating the critical need to manage the record of knowledge, including search, retrieval, and all the related supporting activities. He notes that to handle this problem we have no underlying theory, no coherent representation scheme, no unification of the varied approaches of different computing specialties — and so must tackle it from a number of directions.

After more than 30 years of progress in computing, we still face these challenges and work in this field as a segmented community, viewing DLs from one or another perspective: database management, human-computer interaction (HCI), information science, library science, multimedia information and systems, natural language processing, or networking and communications. As can be seen in the discussion that follows, this practice has led to progress in a large number of separate projects, but has also made interoperability one of the most important problems to solve [624].

Since one of the threads leading to the current interest in DLs came out of discussions of the future of IR [264], since people's needs still leave a rich research agenda for the IR community [197], and since the important role of Web search systems demonstrates the potential value of IR in DLs [711], it is appropriate to see how IR may expand its horizons to deal with the key problems of DLs and how it can provide a unifying and integrating framework for the DL field. Unfortunately, there is little agreement even regarding attempts at integrating database management and text processing approaches [325]. Sometimes, though, it is easier to solve a hard problem if one takes a broader perspective and solves a larger problem. Accordingly we briefly and informally introduce the '5S' model as a candidate solution and a way to provide some theoretical and practical unification for DLs.

We argue that DLs in particular, as well as many other types of information systems, can be described, modeled, designed, implemented, used, and evaluated if we move to the foreground five key abstractions: streams, structures, spaces, scenarios, and societies. 'Streams' have often been used to describe texts, multimedia content, and other sequences of abstract items, including protocols, interactive dialogs, server logs, and human discussions. 'Structures' cover data structures, databases, hypertext networks, and all of the IR constructs such as inverted files, signature files, MARC records (see Chapter 8 for more details), and thesauri. 'Spaces' cover not only 1D, 2D, 3D, virtual reality, and other multidimensional forms, some including time, but also vector spaces, probability spaces, concept spaces, and results of multidimensional scaling or latent-semantic indexing. 'Scenarios' not only cover stories, HCI designs and specifications, and requirements statements, but also describe processes, procedures, functions, services, and transformations — the active and time-spanning aspects of DLs. Scenarios have been essential to our understanding of different DL user communities' needs [525], and are particularly important in connection with social issues [48]. 'Societies' cover these concerns especially regarding authors, librarians, annotators, and other stakeholders. For the sake of brevity we omit further direct discussion of this abstraction, especially since anthropologists, communication researchers, psychologists, sociologists, and others are now engaging in DL research.

Since the 5S model can be used to describe work on databases, HCI, hyperbases, multimedia systems, and networks, as well as other fields related to library and information science, we refer to it below to help unify our coverage and make sure that it encompasses all aspects of DLs. For example, the 5S model in general, and scenarios in particular, may help us move from a paper-centered framework for publishing and communicating knowledge [195] to a hybrid paper/electronic one with a variety of streams and spaces. The 5S model is a simple way to organize our thinking and understand some of the changes that DLs will facilitate:

> The boundaries between authors, publishers, libraries, and readers evolved partly in response to technology, particularly the difficulty and expense of creating and storing paper documents. New technologies can shift the balance and blur the boundaries. [525]

To ground these and other subsequent discussions, then, we explore a number of definitions of DLs, using 5S to help us see what is missing or emphasized in each.

15.2 Definitions

Since DL is a relatively new field, many workshops and conferences continue to have sessions and discussions to define a 'digital library' [266, 347]. Yet, defining DLs truly should occur in the context of other related entities and practices [315]. Thus, a 'digital archive' is like a DL, but often suggests a particular combination of space and structure, and emphasizes the scenario of preservation, as in 'digital preservation' that is based upon digitization of artifacts. Similarly, 'electronic preservation' calls for media migration and format conversions to make DLs immune to degradation and technological obsolescence. Maintaining 'integrity' in a DL requires ensuring authenticity, handled by most regular libraries, as well as consistency, which is a concern whenever one must address replication and versioning, as occurs in database systems and in distributed information systems.

While these concerns are important, we argue that 'DL' is a broader concept. Because it is true that the 'social, economic, and legal questions are too important to be ignored in the research agenda in digital libraries' [525], we really prefer definitions that have communities of users (societies) as part of a DL:

> DLs are constructed — collected and organized — by a community of users. Their functional capabilities support the information needs and uses of that community. DL is an extension, enhancement, and integration of a variety of information institutions as physical places where resources are selected, collected, organized, preserved, and accessed in support of a user community. [48]

This definition has many aspects relating to 5S, but largely omits streams, and only indirectly deals with spaces by calling for extensions beyond physical places. Its coverage of scenarios is weak, too, only giving vague allusion to user support. In contrast, definitions that emphasize functions and services are of particular importance to the development community [299], as are definitions concerned with distributed multimedia information systems:

> The generic name for federated structures that provide humans both intellectual and physical access to the huge and growing worldwide networks of information encoded in multimedia digital formats. [97]

While brief, this definition does tie closely with 5S, though it is weak on scenarios, only mentioning the vague and limited concept of 'access.'

To the IR community a DL can be viewed as an extended IR system, in the context of federation and media variations [48]. Also, DLs must support (large) collections of documents, searching, and cataloging/indexing. They bring together in one place all aspects of 5S, and many of the concerns now faced by IR researchers: multilingual processing, search on multimedia content, information visualization, handling large distributed collections of complex documents, usability, standards, and architectures, all of which are explored in the following sections.

15.3 Architectural Issues

Since DLs are part of the global information infrastructure, many discussions of them focus on high level architectural issues [611]. On the one hand, DLs can be just part of the 'middleware' of the Internet, providing various services that can be embedded in other task-support systems. In this regard they can be treated separately from their content, allowing development to proceed without entanglement in problems of economics, censorship, or other social concerns.

On the other hand, DLs can be independent systems and so must have an architecture of their own in order to be built. Thus, many current DLs are cobbled together from pre-existing pieces, such as search engines, Web browsers, database management systems, and tools for handling multimedia documents.

From either perspective, it is helpful to extend definitions into more operational forms that can lead to specification of protocols when various components are involved. Such has been one of the goals of efforts at the Corporation for National Research Initiatives (CNRI), as illustrated in Figure 15.1.

Thus, Kahn and Wilensky proposed one important framework [426]. Arms *et al.* have extended this work into DL architectures [28, 31]. One element is a digital object, which has content (bits) and a handle (a type of name or identifier) [189], and also may have properties, a signature, and a log of transactions that involve it. Digital objects have associated metadata, that can be managed in sets [472]. Repositories of digital objects can provide security and can respond to a repository access protocol [30]. Significant progress has been made toward adopting a scheme of digital object identifiers, first illustrated by the Online

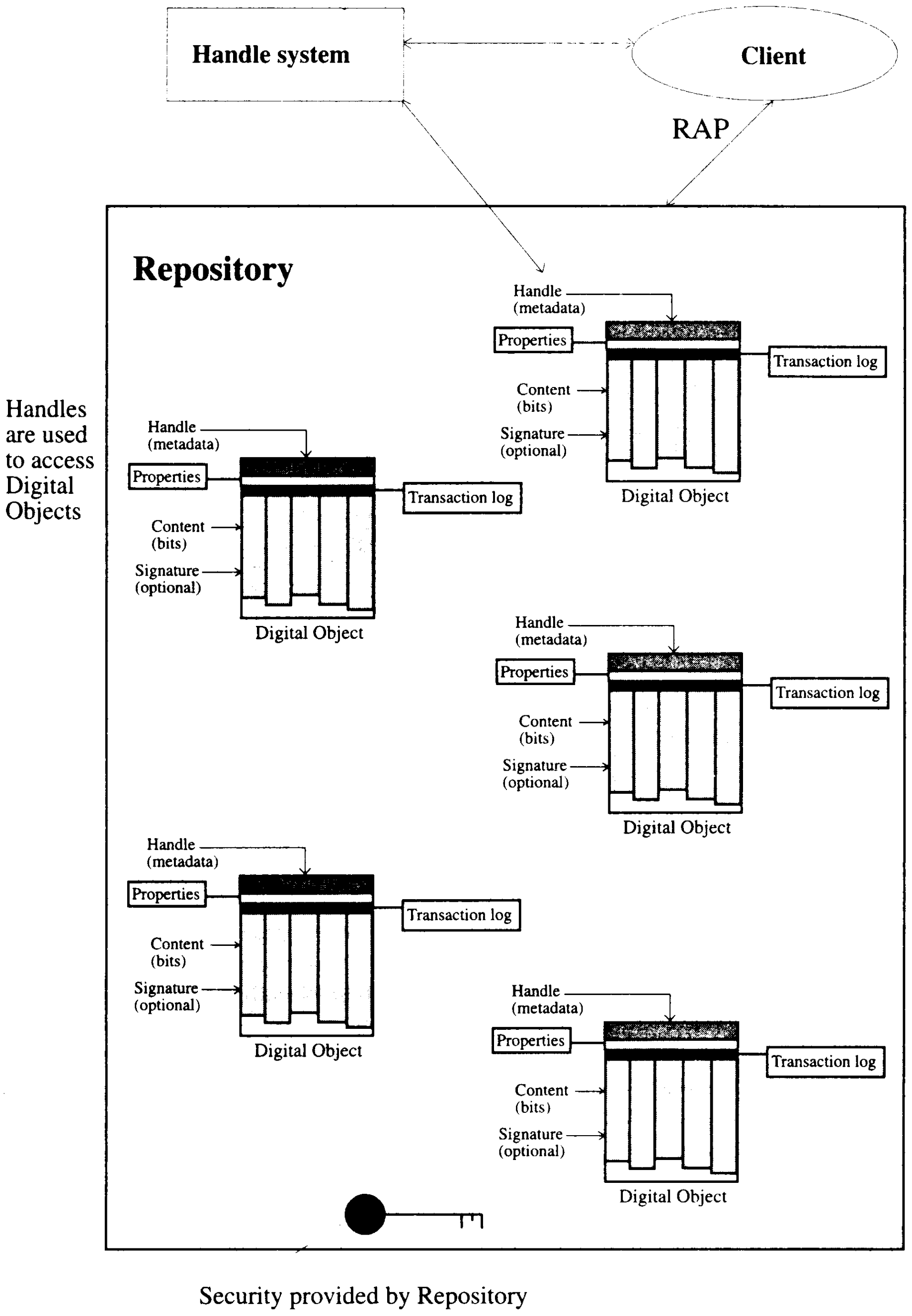


Figure 15.1 Digital objects, handles, and repositories (adapted from [426, 28, 31, 30]).

Computer Library Center, Inc. (OCLCs) Persistent URLs (PURLs) [654], and agreement seems likely on a standard for Digital Object Identifiers (DOIs) [396].

Other implementation efforts have focused more on services [473] and security [475]. A useful testbed for this work has been computer science reports [210], most recently through the Networked Computer Science Technical Reference Library, NCSTRL [471].

Two large Digital Libraries Initiative (DLI) projects have devoted a good deal of attention to architecture, taking radically different approaches. At Stanford, the key concern has been interoperability [624]. Their 'InfoBus' [625] allows a variety of information resources to be connected through suitable mediators and then used via the shared bus through diverse interfaces. At the University of Michigan, the emphasis has been on agent technologies [97]. This approach can have a number of classes of entities involved in far-flung distributed processing. It is still unknown how efficiently an agent-based DL can operate or even be built.

Ultimately, software to use in DLs will be selected as a result of comparisons. One basis for such comparisons is the underlying conceptual model [820]. Another basis is the use of metrics, which is the subject of recent efforts towards definition and consensus building [499]. In addition to metrics traditionally used in IR, dealing with efficiency, effectiveness, and usability, a variety of others must be selected, according to agreed-upon scenarios. Also important to understand is the ability of DLs to handle a variety of document types (combinations of streams and structures), to accurately and economically represent their content and relationships (structures), and to support a range of access approaches and constraints (scenarios).

15.4 Document Models, Representations, and Access

Without documents there would be no IR or DLs. Hence, it is appropriate to consider definitions of 'document' [709], and to develop suitable formalizations [508], as well as to articulate research concerns [505]. For efficiency purposes, especially when handling millions of documents and gigabytes, terabytes, or petabytes of space, compression is crucial [825]. While that is becoming more manageable, converting very large numbers of documents using high quality representations [151] can be prohibitively expensive, especially relative to the costs of retrieval, unless items are popular. All of these matters relate to the view of a document as a stream (along with one or more organizing structures); alternatively one can use scenarios to provide focus on the usage of documents. These problems shift, and sometimes partially disappear, when one considers the entire life cycle and social context of a document [124, 353] or when DLs become an integral part of automation efforts that deal with workflow and task support for one or more document collections.

15.4.1 Multilingual Documents

One social issue with documents relates to culture and language [633]. Whereas there are many causes of the movement toward English as a basis for global scientific and technical interchange, DLs may actually lead to an increase in availability of non-English content. Because DLs can be constructed for a particular institution or nation, it is likely that the expansion of DLs will increase access to documents in a variety of languages. Some of that may occur since many users of information desire it from all appropriate sources, regardless of origin, and so will wish to carry out a parallel (federated) search across a (distributed) multilingual collection.

The key aspects of this matter are surveyed in [613]. At the foundation, there are issues of character encoding. Unicode provides a single 16-bit coding scheme suitable for all natural languages [783]. However, a less costly implementation may result from downloading fonts as needed from a special server or gateway, or from a collection of such gateways, one for each special collection [208].

The next crucial problem is searching multilingual collections. The simplest approach is to locate words or phrases in dictionaries and to use the translated terms to search in collections in other languages [387]. However, properly serving many users in many languages calls for more sophisticated processing [612]. It is likely that research in this area will continue to be of great importance to both the IR and DL communities.

15.4.2 Multimedia Documents

From the 5S perspective, we see that documents are made up of one or more streams, often with a structure imposed (e.g., a raster organization of a pixel stream represents a color image). Multimedia documents' streams usually must be synchronized in some way, and so it is promising that a new standard for handling this over the Web has been adopted [379].

At the same time, as discussed in Chapters 11 and 12, IR has been applied to various types of multimedia content. Thus, at Columbia University, a large image collection from the Web can be searched on content using visual queries [158]. IBM developed the *Query By Image Content* (QBIC) system for images and video [257] and has generously helped build a number of important image collections to preserve and increase access to key antiquities [300].

Similarly, the Carnegie Mellon University DLI project, Informedia [146], has focused on video content analysis, word spotting, summarization, search, and in-context results presentation [146]. Better handling of multimedia is at the heart of future research on many types of documents in DLs [354]. Indeed, to properly handle the complexity of multimedia collections, very powerful representation, description, query and retrieval systems, such as those built upon logical inference [283], may be required.

15.4.3 Structured Documents

While multimedia depends on the stream abstraction, structured documents require both the abstractions of streams and structures. Indeed, structured documents in their essence are streams with one or more structures imposed, often by the insertion of markup in the stream, but sometimes through a separate external structure, like pointers in hypertext.

Since Chapter 6 of this book covers many of the key issues of document structure, we focus in this section on issues of particular relevance to DLs [288]. For example, since DLs typically include both documents and metadata describing them, it is important to realize that metadata as in MARC records can be represented as an SGML document (see Chapter 6 for more details) and that SGML content can be included in the base document and/or be kept separately [293].

Structure is often important in documents when one wants to add value or make texts 'smart' [167]. It can help identify important concepts [626]. SGML is often used to describe structure since most documents fall into one or more common logical structures [750], that can be formally described using a Document Type Definition (DTD). Another type of structure that is important in DLs, as well as earlier paper forms, results from annotation [548]. In this case stream and structure are supplemented by scenarios since annotations result from users interacting with a document collection, as well as collaborating with each other through these shared artifacts [680].

Structure is also important in retrieval. Macleod was one of the first to describe special concerns related to IR involving structured documents [533]. Searching on structure as well as content remains one of the distinguishing advantages of IR systems like OpenText (formerly 'PAT' [38]). Ongoing work considers retrieval with structured documents, such as with patterns and hierarchical texts [439]. An alternative approach, at the heart of much of the work in the Berkeley DLI project [775], shifts the burden of handling structure in documents to the user, by allowing multiple layers of filters and tools to operate on so-called 'multivalent documents' [774]. Thus, a page image including a table can be analyzed with a table tool that understands the table structure and sorts it by considering the values in a user-selected column.

Structure at the level above documents, that is, of collections of documents, is what makes searching necessary and possible. It also is a defining characteristic of DLs, especially when the collections are distributed.

15.4.4 Distributed Collections

Though our view of DLs encompasses even those that are small, self-contained, and constrained to a personal collection with a suitable system and services, most DLs are spread across computers, that is spanning physical and/or logical spaces. Dealing with collections of information that are distributed in nature is one of the common requirements for DL technology. Yet, proper handling of such collections is a challenging problem, possibly since many computer scientists are poorly equipped to think about situations involving spaces as well as the other aspects of 5S.

Of particular concern is working with a number of DLs, each separately constructed, so the information systems are truly heterogeneous. Integration requires support for at least some popular scenarios (often a simple search that

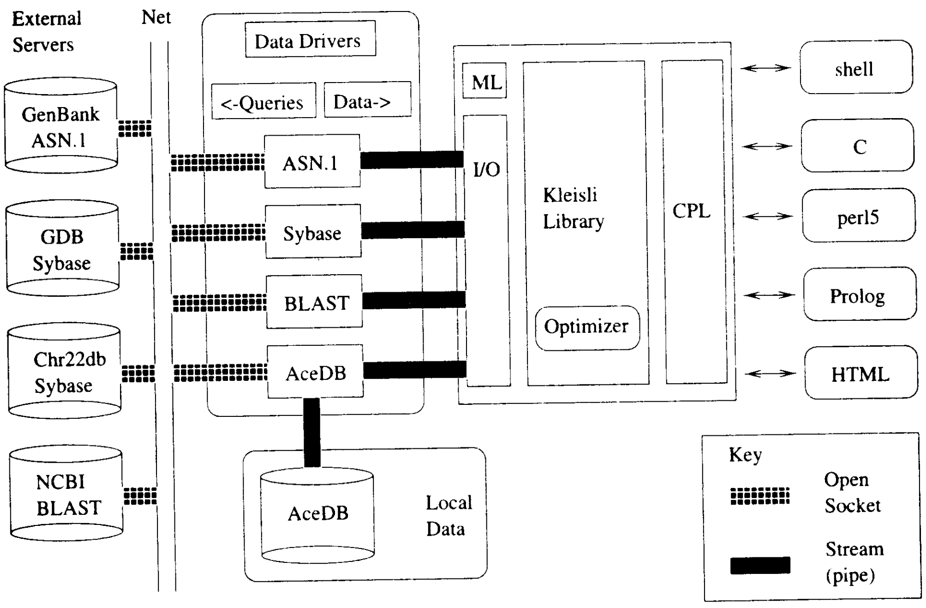


Figure 15.2 Architecture of the BioKleisli system (adapted from [829, 128]).

is a type of least common denominator) by systems that expect differing types of communication streams (e.g., respond to different protocols and query languages), have varying types of streams and structures, and combine these two differently in terms of representations of data and metadata. To tackle this problem, one approach has been to develop a description language for each DL and to build federated search systems that can interpret that description language [161].

However, when DL content is highly complex (e.g., when there are 'unstructured' collections, meaning that the structure is complex and not well described), there is need for richer description languages and more powerful systems to interpret and support highly expressive queries/operations [828, 209, 128]. An architecture of this type is illustrated in Figure 15.2 for the BioKleisli system [829].

In addition to these two approaches – namely reducing functionality for end users in order to give DL developers more freedom and increasing functionality by making the federated system smarter and able to use more computational resources on both servers and clients – there is the third approach of making each DL support a powerful protocol aimed at effective retrieval. This third course is supported by the Computer Interchange of Museum Information (CIMI) effort [570], wherein a Z39.50 interface exists on a number of museum information servers and clients [570]. While Z39.50 was aimed at the needs of libraries desiring interoperability among library catalogs, it does support many of the needs for DLs. Thus, the CIMI interoperability demonstration, with its support for multimedia content, is of great import, but does leave open further improvement in supporting richer DL interaction scenarios, involving more powerful federated searchers.

15.4.5 Federated Search

Federated search work has often been prompted by challenging application requirements. For example, to allow computer science technical reports from around the world to become accessible with minimal investment and maximal local control, the NSF-funded Wide Area TEchnical Report Service (WATERS)initiative was launched [279]. This was then integrated with an effort begun earlier with DARPA funding, the Computer Science Technical Report (CSTR) project [260], leading to a hybrid effort, the Networked CS Technical Reference (previously, Report) Library (NCSTRL) [471]. At the heart of NCSTRL is a simple search system, a well-thought-out open federated DL protocol and the Dienst reference implementation, developed at Cornell University [210]. While this system was custom-built with little dependence on other software, its type of operation could be constructed more rapidly atop various supports like CORBA [788].

Federated search has had an interesting history, with workers adopting a variety of approaches. First, there are those interested in collecting the required information, often through Web crawling of various sorts [715]. Second, there are those focusing on intelligent search [27]. One example is work emphasizing picking the best sites to search [126]. These efforts often assume some integrated information organization across the distributed Internet information space [393].

Third, there is work on fusion of results. This can be viewed in the abstract, regardless of whether the various collections are nearby or distributed, with the target of improving retrieval by culling from a number of good sources [76]. One approach adopts a probabilistic inference network model [139]. Another views the problem as database merging [791]. Alternatively, one can assume that there are a number of search engines distributed to cover the collection, that must be used intelligently [292].

Fourth, there are commercial solutions, including through special Web services [223]. Probably the most visible is the patented, powerful yet elegant, approach by Infoseek Corporation [394].

Finally, there is a new line of work to develop comprehensive and realistic architectures for federated search [219, 218]. The long-term challenge is to segment the collection and/or its indexes so that most searches only look at a small number of the most useful sources of information, yet recall is kept high. Ultimately, however, there are rich types of use of DL content, once one of these approaches to search is carried out.

15.4.6 Access

When priceless objects are described by DL image collections [300], when collections are large and/or well organized so as to appear of value to communities of users, or when there are valuable services in information manipulation (searching, ordering, reporting, summarizing, etc.) afforded by a DL, some method of payment is often required [194, 191, 49, 251]. Though previously access to scientific literature was not viewed as a commodity as it is today [328], DLs clearly must manage intellectual property [559]. These services must support agreed-upon principles [586], copyright practices [705], as well as contracts and other agreements and laws [346].

Though technology is only part of the picture [822], a key to the implementation of policies for access management [30] is having trusted systems [746]. Security is one topic often ignored by the IR community. However, many aspects of security can be of fundamental importance in DLs [302, 301]. Just as encryption is essential to support electronic commerce, watermarking and stronger mechanisms are crucial in DLs to protect intellectual property rights and to control the types of access afforded to different user groups. Scenarios are important here, to ensure that suitable constraints are imposed on processing, all the way from input to output. For example, secret documents may not even be made visible in searches through metadata. On the other hand, advertising full documents as well as allowing locating and viewing metadata records is appropriate when the purpose of security is to enforce payment in 'pay by the drink' document downloading systems. Inference systems can be used for complicated rights management situations [16]. A deeper understanding of these requirements and services can be obtained by considering representative DL projects, such as those mentioned in the next section.

15.5 Prototypes, Projects, and Interfaces

Though numerous efforts in the IR, hypertext, multimedia, and library automation areas have been underway for years as precursors of today's DL systems, one of the first new efforts aimed at understanding the requirements for DLs and constructing a prototype from scratch was the ENVISION project, launched in 1991 [269]. Based on discussions with experts in the field and a careful study of prospective users of the computer science collection to be built with the assistance of ACM, the ENVISION system was designed to extend the MARIAN search system [264] with novel visualization techniques [273, 360]. Careful analysis has shown its 2D approach to management of search results is easy to use and effective for a number of DL activities [610].

The CORE project, another early effort, is an electronic library prototype on chemical journal articles. Its collection included, for each article, both scanned images and an SGML marked-up version, as well as indexes for full-text Boolean searching. It was undertaken by the American Chemical Society, Chemical Abstracts Service, OCLC, Bellcore, and Cornell University, along with other partners [237]. This project also was concerned with collection building as well as testing of a variety of interfaces that were designed based on user studies.

One of the most visible project efforts is the Digital Libraries Initiative, initially supported by NSF, DARPA, and NASA [349]. Phase 1 provided funding for six large projects over the period 1994–1998. These projects spanned a wide range of major topics in developing the National Information Infrastructure (NII) and addressed future technological problems. The Illinois project [777] focused on manually structured text documents in full systems with many users; the Berkeley project [775] emphasized automatically recognized image documents, also with large systems. The Santa Barbara [776] and Carnegie Mellon [146] projects investigated the ability to manipulate new media; Carnegie Mellon focused on segmenting and indexing video using speech recognition and program structure, and Santa Barbara concentrated on indexing maps using image processing and region metadata. Stanford [745] and Michigan [784] investigated the intermediaries to perform operations on large digital libraries; Stanford investigated interoperability of different search services, and Michigan concentrated on interacting software agents to provide services to users [710].

Since these projects have been described elsewhere in depth, it should suffice here to highlight some of the connections of those projects with the IR community. First, each project has included a component dealing with document collections. The Illinois project produced SGML versions of a number of journals while the Berkeley project concentrated on page images and other image classes. Santa Barbara adopted a spatial perspective, including satellite imagery, while Carnegie Mellon University (CMU) focused on video. Stanford built no collections, but rather afforded access to a number of information sources to demonstrate interoperability. At the University of Michigan, some of the emphasis was on having agents dynamically select documents from a distributed set of resources.

Second, the DLI projects all worked on search. Text retrieval, and using automatically constructed cross-vocabulary thesauri to help find search terms, was emphasized in Illinois. Image searching was studied at Berkeley and Santa Barbara while video searching was investigated at CMU. Michigan worked with agents for distributed search while Stanford explored the coupling of a variety of architectures and interfaces for retrieval.

Finally, it is important to note that the DLI efforts all spent time on interface issues. Stanford used animation and data flows to provide flexible manipulation ana integration of services [192]. At Michigan, there were studies of the PAD++ approach to 2D visualization [70]. Further discussion of interfaces can be found below in subsection 15.5.2.

It should be noted that these projects only partially covered the 5S issues. Structures were not well studied, except slightly in connection with the Illinois work on SGML and the Berkeley work on databases. Scenarios were largely ignored, except in some of the interface investigations. Similarly, spaces were not investigated much, except in connection with the vocabulary transfer work at Illinois and the spatial collection and browsing work at Santa Barbara. Other projects in the broader international scene, some of which are discussed in the next section, may afford more thorough 5S coverage.

Since the announcement of DLI, activities and interest related to digital libraries have increased dramatically. The six DLI projects were highly visible and grew in scope; however, it was quickly realized that DLI still needed additional direction and coherence. During the initial funding period of the DLI program, additional workshops were created to develop consensus on the directions and boundaries with discussions from various communities. An important aspect that many people realized from the workshops is the importance of efforts in domains outside computer and information science to the advances in digital libraries research [324].

A follow-on program, Digital Libraries Initiative – Phase 2 (DLI-2), jointly supported by NSF, DARPA, NASA, the National Library of Medicine (NLM), the Library of Congress (LoC), the National Endowment for the Humanities (NEH), and others, was announced in the spring of 1998 focus less on technology research than DLI, but, more importantly, supporting research across the information life cycle, from content creation, access, and use to preservation and archiving, moving towards the concept of digital libraries as human-centered systems. DLI-2 will emphasize the study of interactions between digital libraries and humans, fuller understanding of and improving access to digital content and collections, and interoperability and integration toward flexible information environments at the level of individual, group, and institution [324, 216]. The program will involve people not only from science and engineering but also from arts and humanities.

15.5.1 International Range of Efforts

DL efforts, accessible over the Internet, can now lead to worldwide access. Since each nation wishes to share the highlights of its history, culture, and accomplishments with the rest of the world, developing a DL can be very helpful [86]. Indeed, we see many nations with active DL programs [270], and there are many others underway or emerging.

One of the largest efforts is the European ERCIM program [239]. This is enhanced by the large eLib initiative in the UK [778]. There are good results from activities in New Zealand [601] and Australia [389]. In Singapore, billions are being invested in developing networked connectivity and digital libraries as part of educational innovation programs [729]. For information on other nations, see the online table pointing to various national projects associated with a recent special issue on this topic [270].

As mentioned briefly above, many nations around the world have priceless antiquities that can be more widely appreciated through DLs [300]. Whether in pilot mode or as a commercial product, *IBM Digital Library* [390], with its emphasis on rights management, has been designed and used to help in this regard.

These projects all require multimedia and multilingual support, as discussed earlier. Different scenarios of use are appropriate in different cultures, and different structures and spaces are needed for various types of collections. Indeed, many international collections aim for global coverage, but with other criteria defining their focus. Thus, the Networked Digital Library of Theses and Dissertations (NDLTD) [594] is open to all universities, as well as other supporting organizations, with the aim of providing increased access to scholarly resources as a direct result of improving the skills and education of graduate students, who directly submit their works to the DL.

15.5.2 Usability

Key to the success of DL projects is having usable systems. This is a serious challenge! Simple library catalog systems were observed in 1986 to be difficult to use [104], and still remain so after a further decade of research and development [105].

The above mentioned ENVISION project's title began with the expression 'User-Centered' and concentrated most of its resources on work with the interface [360]. A 1997 study at Virginia Tech of four digital library systems concluded that many have serious usability problems [434], though the design of the Illinois DLI system seemed promising. The Virginia Tech study uncovered an important aspect of the situation, and suggested that it will be years before DL systems are properly understood and used. A pre-test asked about user expectations for a DL, and found that very few had worked with a DL. The post-test showed that user expectations and priorities for various features changed dramatically over the short test period. Thus, it is likely that in general, as DL usage spreads, there will be an increase in understanding, a shift in what capabilities users expect, and a variety of extensions to the interfaces now considered.

Early in the DLI work, DL use was perceived as a research focus [98], and understanding and assessing user needs became a key concern [382]. For two years, a workshop was held at the Allerton conference center of the University of Illinois on this topic. Since the 1995 event [313] had a diverse group of researchers, it was necessary to understand the various perspectives and terminologies. There were discussions of fundamental issues, such as information, from a human factors perspective [214], as well as specific explorations of tasks like document browsing [528].

The 1996 event was more focused due to greater progress in building and studying usability of DLs [314]. Thus, there was discussion of Stanford's SenseMaker system which supports rapid shifting between contexts that reflect stages of user exploration [51]. Social concerns that broaden the traditional IR perspective were highlighted [367]. In addition, there was movement towards metrics (see discussion earlier about DL metrics) and factors for adopting DLs [429].

DL interfaces and usability concerns have been central to many efforts at Xerox PARC. Some of the research considers social issues related to documents [354] while other research bridges the gap between paper and digital documents [353]. There are many issues about documents, especially their stability and how multimedia components as well as active elements affect retrieval, preservation, and other DL activities [506]. Some insight into DL use may result from actual user observation as well as other measures of what (parts of) documents are read [507]. There also has been collaboration between PARC and the UCB DLI team, which has extended the Xerox magic filter work into multivalent documents (discussed earlier) as well as having developed results visualization methods like

TileBars where it is easy to spot the location of term matches in long documents [355].

Further work is clearly needed in DL projects to improve the systems and their usability. But for these systems to work together, there also must be some emphasis on standards.

15.6 Standards

Since there are many DL projects worldwide, involving diverse research, development, and commercial approaches, it is imperative that standards are employed so as to make interoperability and data exchange possible. Since by tradition any library can buy any book, and any library patron can read anything in the library, DLs must make differences in representation transparent to their users. In online searching as well, data that can be understood by clients as well as other DLs should be what is transferred from each information source. At the heart of supporting federated DLs, especially, is agreement on protocols for computer-computer communication.

15.6.1 Protocols and Federation

In the 1980s it became clear that as library catalog systems proliferated, and library patrons sought support for finding items not locally available through inter-library loan or remote cataloging search, some protocol was needed for searching remote bibliographic collections. The national standard Z39.50, which later became an international standard as well, led to intensive development of implementations and subsequent extensive utilization [515]. One example of widespread utilization was the WAIS system (based on Z39.50), very popular before the World Wide Web emerged. Ongoing development of Z39.50 has continued, including its application to DLs, as demonstrated in the CIMI project described earlier, where a number of different clients and server implementations all worked together.

Also mentioned earlier is the NCSTRL effort, starting with CS technical reports, in which the Dienst protocol was developed [210]. This is a 'lighter' protocol than Z39.50, designed to support federated searching of DLs, but also connected to the centralized preprint service (CoRR) at Los Alamos National Laboratory. Dienst seems suitable for electronic theses and dissertations as well as technical reports, and so it has been considered in regard to NDLTD.

These protocols assume that each server and client will be changed to use the protocol. A less intrusive approach, but one harder to implement and enforce, is to have some mechanism to translate from a special server or gateway system to/from each of the information sources of interest. The STARTS protocol [316] was proposed to move in this direction, but competition among search services on the Internet is so severe that acceptance seems unlikely. Though this is unfortunate, simple federated schemes have been implemented in the DLI projects at Stanford and Illinois, and a simple one is in use in NDLTD. Yet, even more important than new protocols for DL federated search is agreement on metadata schemes, which does seem feasible.

15.6.2 Metadata

In the broadest sense, metadata can describe not only documents but also collections and whole DLs along with their services [50]. In a sense, this reflects movement toward holistic treatment like 5S. Yet in most DL discussions, metadata just refers to a description of a digital object. This is precisely the role played by library catalog records. Hence, cataloging schemes like MARC are a starting point for many metadata descriptions [514].

While MARC has been widely used, it usually involves working with binary records which must be converted for interchange. One alternative is to encode MARC records using some readable coding scheme, like SGML [293]. Another concern with MARC is that there are a number of national versions with slight differences, as well as differences in cataloging practices that yield the MARC records. USMARC is one such version. It is very important in the DL field, and can be encoded using SGML, or easily converted to simpler metadata schemes like the 'Dublin Core' [513]. Other 'crosswalks' exist between Dublin Core (DC), MARC, and schemes like GILS, proposed for a Government Information Locator Service [598]. A mapping also exists between DC and the Z39.50 protocol discussed in the previous section [503].

DC is a simple scheme, with 15 core elements that can be used to describe any digital object. What is of real import is that it has been widely accepted. That is because there have been years of discussion and development, focused around international workshops [806, 620, 560, 833, 333]. The core elements include seven that describe content (Title, Subject, Description, Source, Language, Relation, and Coverage). There are four elements that deal with intellectual property issues (Creator, Publisher, Contributor, and Rights). Finally, to deal with instances of abstract digital objects, there are four other types (Data, Type, Format, and Identifier).

Since digital objects and their metadata often have to be interchanged across systems, the problem of packaging arises. The Warwick Framework, which evolved out of the same type of discussions leading to DC, deals with packages and connections between packages [472]. In general, such discussion about metadata is crucial to allow the move from traditional libraries (with their complex and expensive cataloging), past the Web (with its general lack of cataloging and metadata), to a reasonable environment wherein metadata is available for all sorts of digital objects (suitable to allow the organization of vast collections in DLs [734]).

Because the Web has need of such organization, this has become an interest of its coordinating body, the WWW Consortium [84]. In 1996, as concern increased about protecting children from exposure to objectionable materials, metadata schemes became connected with censoring and filtering requirements. The problem was renamed for the more general case, in keeping with Harvest's treatment of 'resource discovery,' to 'resource description.' The Resource Description Framework (RDF) thus became an area of study for the Consortium [753]. It should be noted that RDF can lead to header information inside digital objects, including those coded in SGML or HTML, as well as XML (see Chapter 6 for more details). In the more general case, however, RDF is essentially a scheme for annotating digital objects, so alternatively the descriptions can be stored separately from those objects. These options bring us back to the Warwick Framework where there may be multiple containers, sometimes connected through indirection, of packages of metadata, like MARC or DC.

We see that DLs can be complex collections with various structuring mechanisms for managing data and descriptions of that data, the so-called metadata. However, coding may combine data with metadata, as is specified in the guidelines of the Text Encoding Initiative (TEI) [670]. This reminds us of the complexities that arise when combining streams and structures, where there are many equivalent representations. We also see that for DL standards to be useful, such as appears to be the case for DC, the structures involved must be relatively simple, and have well understood related scenarios of use. While this now appears to work for data interchange, further work is required for interoperability, i.e., interchange through the streams involved in protocols.

15.7 Trends and Research Issues

There are many remaining challenges in the DL field. While TEI provides guidance in complex encoding situations, and has been advocated by the University of Michigan for electronic theses and dissertations, it is unclear how far the rest of the scholarly community will move towards the thorough markup and description of digital objects that characterize humanistic study [670]. Though such markup is valuable to support context-dependent queries as well as electronic document preservation, it will only be generally feasible when there are less expensive tools and more efficient methods for adding in such markup and description, which may occur as XML usage expands. Then, too, the IR community must provide guidance regarding automatic indexing of marked up documents, metadata, full-text, multimedia streams, and complex hypermedia networks so that the rich and varied content of DLs can be searched.

On a grander scale are the problems of handling worldwide DLs, in the context of varying collection principles, enormous difference in response time between local and remote servers, and the needs of users for different views [474]. Thus, one type of scenario might deal with searching all dissertations worldwide, another might be concerned with finding recent results from a particular research group, a third might consider only freely available works in a particular specialty area, a fourth might deal with seeking the new works recently highly rated by a distributed group of close friends, and yet another might involve the most readable overviews in an unknown area.

Other key research challenges have been highlighted in various workshops aimed at establishing an agenda for investigation [525]. Of central concern is covering the range from personal to global DLs, the so-called 'scaling' problem. At the same time, the problem of interoperability must be faced [624]. As argued earlier, we view the solution to these problems to be the acknowledgement of the role of 5S in the DL arena and the focus of research and development on treating streams, structures, spaces, scenarios, and societies as first class objects and building blocks for DLs. We will continue to explore this approach in future work, and believe that, to the extent that integrated support for 5S is developed, real progress will be made towards the next generation of digital libraries.

15.8 Bibliographical Discussion

As explained in section 15.1, there are many good sources of information about digital libraries. The best pair are the book by Lesk [501] and the online *D-Lib Magazine* [280]. Pointers to the latest information and sources can be found through online courseware [268]. New books will appear from MIT Press and other publishers. Large funding initiatives, programs, and projects (e.g., [216, 778, 349]) involving the US National Science Foundation (see e.g., the call for Digital Libraries Initiative - Phase 2, NSF 98-63, `http://www.dli2.nsf.gov`) and other sponsors, and becoming more and more international in nature (e.g., International Digital Libraries Collaborative, NSF 99-6, will lead to a continuing stream of reports on workshops (e.g., [266, 313, 314, 333, 833, 525]) and high quality research presentations at premiere events like the ACM Digital Libraries conferences (e.g. [50, 192, 382, 507, 548, 705, 791]).

Acknowledgements

The preparation of this chapter and work described therein was supported in part by US Department of Education grant P116B61190 and by NSF grants CDA-9303152, CDA-9308259, CDA-9312611, DUE-9752190, DUE-975240, and IRI-9116991.

Appendix
Porter's Algorithm

The rules in the Porter algorithm are separated into five distinct phases numbered from 1 to 5. They are applied to the words in the text starting from phase 1 and moving on to phase 5. Further, they are applied sequentially one after the other as commands in a program. Thus, in what follows, we specify the Porter algorithm in a pseudo programming language whose commands take the form of rules for suffix substitution (as above). This pseudo language adopts the following (semi-formal) conventions:

- A consonant variable is represented by the symbol C which is used to refer to any letter other than a,e,i,o,u and other than the letter y preceded by a consonant.
- A vowel variable is represented by the symbol V which is used to refer to any letter which is not a consonant.
- A generic letter (consonant or vowel) is represented by the symbol L.
- The symbol ϕ is used to refer to an empty string (i.e., one with no letters).
- Combinations of C, V, and L are used to define *patterns*.
- The symbol $*$ is used to refer to zero or more repetitions of a given pattern.
- The symbol $+$ is used to refer to one or more repetitions of a given pattern.
- Matched parentheses are used to subordinate a sequence of variables to the operators $*$ and $+$.
- A generic pattern is a combination of symbols, matched parentheses, and the operators $*$ and $+$.

- The substitution rules are treated as commands which are separated by a semicolon punctuation mark.
- The substitution rules are applied to the suffixes in the current word.
- A conditional *if* statement is expressed as *'if (pattern) rule'* and the rule is executed only if the pattern in the condition matches the current word.
- A line which starts with a % is treated as a comment.
- Curly brackets (braces) are used to form compound commands.
- A 'select rule with longest suffix' statement selects a single rule for execution among all the rules in a compound command. The rule selected is the one with the largest matching suffix.

Thus, the expression $(C)^*$ refers to a sequence of zero or more consonants while the expression $((V)^*(C)^*)^*$ refers to a sequence of zero or more vowels followed by zero or more consonants which can appear zero or more times. It is important to distinguish the above from the sequence $(V * C)$ which states that a sequence must be present and that this sequence necessarily starts with a vowel, followed by a subsequence of zero or more letters, and finished by a consonant. Finally, the command

$$\text{if } (*V * L) \text{ then ed} \longrightarrow \phi$$

states that the substitution of the suffix ed by nil (i.e., the removal of the suffix ed) only occurs if the current word contains a vowel and at least one additional letter.

The Porter algorithm is applied to each word in the text (simple formulation) and is given by the following procedure.

```
% Phase 1: Plurals and past participles.
select rule with longest suffix {
    sses $\longrightarrow$ ss;
    ies $\longrightarrow$ i;
    ss $\longrightarrow$ ss;
    s $\longrightarrow$ $\phi$; }

select rule with longest suffix {
    if $((C)^*((V)^+(C)^+)^+(V)^*$eed) then eed $\longrightarrow$ ee;
    if (*V*ed or *V*ing) then {
            select rule with longest suffix {

                ed ⟶ $\phi$;
                ing ⟶ $\phi$; }

            select rule with longest suffix {

                at ⟶ ate;
                bl ⟶ ble;
                iz ⟶ ize;
                if (($^*C_1C_2$) and ($C_1 = C_2$) and
                ($C_1 \notin$ {l,s,z})) then $C_1C_2 \longrightarrow C_1$;
                if ((($C$)$^*$(($V$)$^+$($C$)$^+$)$C_1V_1C_2$) and
                ($C_2 \notin$ {w,x,y})) then $C_1V_1C_2 \longrightarrow$
                $C_1V_1C_2$e; }

            }

        }

    if (*V*y) then y ⟶ i;
    if (($C$)$^*$(($V$)$^+$($C$)$^+$)$^+$($V$)$^*$) then
    select rule with longest suffix {

        ational ⟶ ate;
        tional ⟶ tion;
        enci ⟶ ence;
        anci ⟶ ance;
        izer ⟶ ize;
        abli ⟶ able;
        alli ⟶ al;
        entli ⟶ ent;
        eli ⟶ e;
        ousli ⟶ ous;
        ization ⟶ ize;
        ation ⟶ ate;
        ator ⟶ ate;
        alism ⟶ al;
        iveness ⟶ ive;
        fulness ⟶ ful;
        ousness ⟶ ous;
        aliti ⟶ al;
        iviti ⟶ ive;
        biliti ⟶ ble; }

    if (($C$)$^*$(($V$)$^+$($C$)$^+$)$^+$($V$)$^*$) then
    select rule with longest suffix {

        icate ⟶ ic;
        ative ⟶ $\phi$;
        alize $\longrightarrow$ al;
        iciti $\longrightarrow$ ic;
        ical $\longrightarrow$ ic;
        ful $\longrightarrow$ $\phi$;
        ness $\longrightarrow$ $\phi$; }
    if $((C)^*((V)^+(C)^+)((V)^+(C)^+)^+(V)^*)$ then
    select rule with longest suffix {
        al $\longrightarrow$ $\phi$;
        ance $\longrightarrow$ $\phi$;
        ence $\longrightarrow$ $\phi$;
        er $\longrightarrow$ $\phi$;
        ic $\longrightarrow$ $\phi$;
        able $\longrightarrow$ $\phi$;
        ible $\longrightarrow$ $\phi$;
        ant $\longrightarrow$ $\phi$;
        ement $\longrightarrow$ $\phi$;
        ment $\longrightarrow$ $\phi$;
        ent $\longrightarrow$ $\phi$;
        ou $\longrightarrow$ $\phi$;
        ism $\longrightarrow$ $\phi$;
        ate $\longrightarrow$ $\phi$;
        iti $\longrightarrow$ $\phi$;
        ous $\longrightarrow$ $\phi$;
        ive $\longrightarrow$ $\phi$;
        ize $\longrightarrow$ $\phi$;
        if (*s or *t) then ion $\longrightarrow$ $\phi$; }
    select rule with longest suffix {
        if $((C)^*((V)^+(C)^+)((V)^+(C)^+)^+(V)^*)$ then e $\longrightarrow$ $\phi$;
        if $(((C)^*((V)^+(C)^+)(V)^*)$ and not $((*C_1V_1C_2)$ and $(C_2 \notin$
        $\{$w,x,y$\})))$ then e $\longrightarrow$ *nil*; }
    if $((C)^*((V)^+(C)^+)((V)^+(C)^+)^+V^*$ll$)$ then ll $\longrightarrow$ l;
```