KATHARINA KLEMPERER

Director of Library Automation
Dartmouth College
Hanover, New Hampshire

Delivering a Variety of Information in a Networked Environment

ABSTRACT

The volume and variety of electronic information resources, the increase in desktop computing power, and the pervasiveness of networks have combined to make access to information fundamentally different from that of a decade ago. This paper describes the nature of information resources that libraries are dealing with now and discusses the different needs of each with regard to access and delivery.

INTRODUCTION

Information science has undergone a fundamental change during the past 10 or 15 years. While libraries still provide the same service that they always have—access to information—the tools and skills are entirely different from those that were taught in library schools a decade ago. The volume and variety of electronic information resources, the increase in desktop computing power, and the pervasiveness of networks have combined to challenge the resources of information providers. This paper will describe the nature of information resources that libraries are dealing with now and will discuss the different needs of each with regard to access and delivery.

The different kinds of electronic information that we have available for delivery today can be divided into rough categories:

- text, including
  - indexes
  - structured full text
  - unstructured full text
- numeric
- multimedia, including
  - images
  - full-motion video
  - sound

Each of these has different needs in terms of access and delivery.

TEXT

Indexes

The kind of information that libraries have dealt with for years, and which they have handled with great success, is indexes to larger bodies of information. Among these we find card catalogs (and their online cousins), indexes to the journal literature, and catalogs of objects such as museum artifacts. The characteristics of this kind of textual information follow:

- It divides neatly into "records," all of which include roughly the same fields.
- It is highly structured, that is, each record is composed of distinct and identifiable fields such as authors and ID numbers.
- The records are of similar size.

A whole generation of systems grew up to support online catalogs, and because other types of indexes are so similar, it was easy to force them into systems that were designed to handle online catalog records, usually in MARC format. Each of these indexes has a reasonably small number of access points that can be indexed and reasonably short fields that can be displayed and comprehended easily.

Compare the following two examples of data structures from the Dartmouth College Library Online System. The first is from the online catalog, the second from the locally mounted MEDLINE database.

Author: Symposium on Immunology of Milk and the Neonate (1990 : Miami, Fla.)
Title: Immunology of milk and the neonate / edited by Jiri Mestecky, Claudia Blair, and Pearay L. Ogra.
Imprint: New York : Plenum Press, c1991.
Series: Advances in experimental medicine and biology ; v. 310.
Location: Dana RJ/216/S945/1990

Author(s): Wilson NW, Self TW, Hamburger RN
Title: Severe cow's milk induced colitis in an exclusively breast-fed neonate. Case report and clinical review of cow's milk allergy.
Source: Clinical pediatrics 1990 Feb;29(2):77-80.
NLM ID: 90150935
Location: Health Sciences Serial

Structured Full Text

There is a conceptual difference between index-type databases and full-text databases. Indexes *represent* a complete document, whether it be a book, a phonograph record, or a museum object. A full-text file *is* the document; once you have retrieved it you need look no further. Among full-text formats, we find ourselves on a continuum. At one end are highly structured data files that are in fact full text but that can be forced into the traditional online catalog database structure without too much effort. It may not consist of bibliographic data, but once new field names have been defined, search and retrieval can proceed basically as if one were retrieving catalog records. An example of highly structured full text is a dictionary entry. The following example is from the *American Heritage Electronic Dictionary*, as mounted at Dartmouth College Library:

Word: kin*dle (1)
Part of Speech: verb
Inflected Form: -dled, -dling, -dles.
Part of Speech: transitive verb
Sense: 1. a. To build or fuel (a fire).
1. b. To set fire to; ignite.
2. To cause to glow; light up, as in: The sunset kindled the skies.
3. a. To inflame; make ardent.
3. b. To arouse; inspire, as in: "No spark had yet kindled in him an intellectual passion"(George Eliot).

Part of Speech: intransitive verb
Sense: 1. To catch fire; burst into flame.
2. To become bright; glow.
3. a. To become inflamed.
3. b. To be stirred up; rise.
Etymology: Middle English kindelen < Old Norse kynda.
Derivative: kin'dler
Part of Speech: noun

These data, although in fact full text, display the characteristics of index data proposed above: they divide into records, they are highly structured, and the records are of similar size. Consequently, it was a fairly easy task to load the data into the same database manager that was used for the index-type databases. Notice however the repeating groups of fields (Part of Speech and Sense). This feature of the data is not usually found in index-type databases.

When the entire full text is indexed, plenty of new ways to access information present themselves. For example, in the *American Heritage Electronic Dictionary:*

- Find all the words of Norwegian derivation in the English language (sample results: floe, iceberg, fiord, ski, slalom, telemark, lemming, troll).
- Find all the verbs that have to do with "fire" (sample results: anneal, barbecue, beacon, blaze, burn, crackle, discharge, douse, ignite).
- Find all the six-letter words that end in "ism" (sample results: ageism, cubism, Nazism, nudism, sadism, [Uncle] Tomism).

Unstructured Full Text

At the other end of the text continuum are complete texts of literature. These texts are minimally structured. Most have sentences, paragraphs, and chapters, but the information is structured not as a *collection of records* but as an *ordered string of words.*

This changes the methods of access and display significantly. In record-oriented data, one searches for a known feature (usually a keyword located in a specific field) in the entire database, and the goal is to locate all the records containing this feature. Boolean searching means that two or more features will be found in the same record. Thus, a search for the author word *Hemingway* and the title word *sun* will retrieve a number of catalog records (mostly Hemingway's *The Sun Also Rises;* see Figure 1). Nobody really cares which catalog records happen to precede or follow these retrieved records; the database records are basically unrelated to each other except perhaps alphabetically. Display of retrieved records is straightforward: one provides a short,

medium, or long display of individual records, giving various depths of detail to allow users to scan the results and then look more closely at one or more retrieved records.

Full-text files have a completely different set of needs. Since full texts consist of a series of ordered words rather than unrelated records, it can be argued that a literary text is really nothing more than a long string of characters. Where "records" can be identified, they are of varying length and tend to consist of multiply occurring "fields," also of varying length. Fields may also be interleaved and must be displayed in their original order (e.g., chapter heading, subheading, multiple paragraphs containing multiple sentences; see Figure 2).

What we are searching for here is not so much *records* (e.g., paragraphs), but *matchpoints*. If I am searching for the word "rabbit"

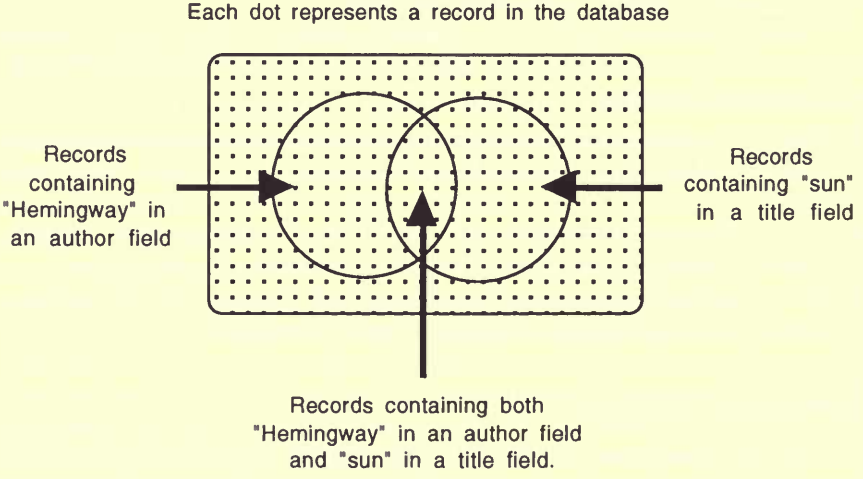

Figure 1. Boolean search of a record-oriented database

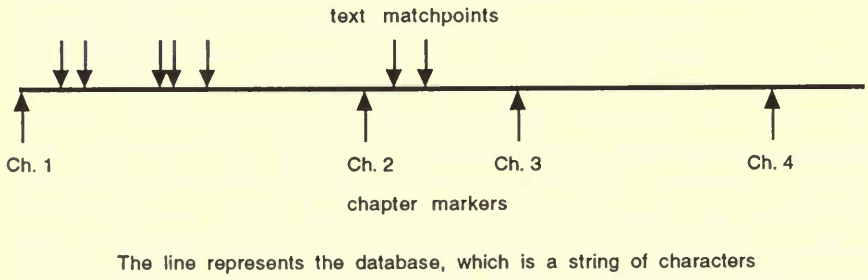

Figure 2. "Fields" in an unstructured full-text database

in the text of *Alice in Wonderland,* I don't want to find all the paragraphs or lines that contain the word, I want to locate the occurrences of the word itself and scan the text preceding and following the matchpoints I have found. Rather than Boolean combinations (find every "record" containing *rabbit* and *Alice*) full texts are better served by proximity operators, which locate occurrences within a certain distance of each other (find every occurrence of *rabbit* within five words of *Alice;* find every occurrence of *Alice* preceding *rabbit* by no more than 100 characters). Displays likewise have different requirements; rather than seeing a list of individual chapters or paragraphs, in which the target words might appear only after many lines, the user is better served by a display of matchpoints in context. The user needs to be able to see all the matchpoints in context at a glance, jump from one matchpoint to the next, and scroll forward and backward through the text from any given matchpoint. A further display requirement is to provide the ability to go to the beginning of the "segment" (e.g., paragraph, chapter, poem) for each matchpoint. An example will illustrate the initial display of matchpoints:

sorted by matchpoint:

18	suddenly a White	Rabbit with pink eyes ran close by her.
22	of the way to hear the	Rabbit say to itself, 'Oh dear! Oh dear!
25	natural); but when the	Rabbit actually TOOK A WATCH OUT
28	had never before seen a	rabbit with either a waistcoat-pocket, or
95	and the White	Rabbit was still in sight, hurrying down
99	to corner, but the	Rabbit was no longer to be seen:
07	CHAPTER 1 Down the	Rabbit-Hole Alice was beginning to get
30	it pop down a large	rabbit-hole under the hedge. In another
35	to get out again. The	rabbit-hole went straight on like a tunnel

sorted by appearance in the document:

07	CHAPTER 1 Down the	Rabbit-Hole Alice was beginning to get
18	suddenly a White	Rabbit with pink eyes ran close by her.
22	of the way to hear the	Rabbit say to itself, 'Oh dear! Oh dear!
25	natural); but when the	Rabbit actually TOOK A WATCH OUT
28	had never before seen a	rabbit with either a waistcoat-pocket, or
30	it pop down a large	rabbit-hole under the hedge. In another
35	to get out again. The	rabbit-hole went straight on like a tunnel
95	and the White	Rabbit was still in sight, hurrying down
99	to corner, but the	Rabbit was no longer to be seen: she

The surprise here is that this kind of display has long been with us, known as a KWIC display (KeyWord In Context). Its usefulness has not been lost.

The display above gives line numbers for each match; if the text were not broken into lines, the matches could just as easily be numbered sequentially. At this point, the user wants to zero in on one match and perhaps display a certain number of words around it. For example, displaying 100 words before and after the match at line 25 would result in the following:

> sleepy and stupid), whether the pleasure of making a daisy-chain would be worth the trouble of getting up and picking the daisies, when suddenly a White Rabbit with pink eyes ran close by her.
>
> There was nothing so VERY remarkable in that; nor did Alice think it so VERY much out of the way to hear the Rabbit say to itself, 'Oh dear! Oh dear! I shall be late!' (when she thought it over afterwards, it occurred to her that she ought to have wondered at this, but at the time it all seemed quite natural); but when the **Rabbit** actually TOOK A WATCH OUT OF ITS WAISTCOAT-POCKET, and looked at it, and then hurried on, Alice started to her feet, for it flashed across her mind that she had never before seen a rabbit with either a waistcoat-pocket, or a watch to take out of it, and burning with curiosity, she ran across the field after it, and fortunately was just in time to see it pop down a large rabbit-hole under the hedge.
>
> In another moment down went Alice after it, never once considering how in the world she was to get out again.

At this point, the user might want to keep scrolling forward or might want to go back to the beginning of the chapter in which this match was found. The whole procedure is more one of navigating around a text rather than looking at records in a file.

Between the strictly record-oriented indexes and the completely unstructured full texts, there is a wide variety of full texts that can be treated either as record oriented or as full-text files, for example, the Bible, collections of poems, plays, encyclopedias.

NUMERIC DATA

Beyond text files, there are completely different data formats that are now available electronically, each requiring different methods of access and display.

A completely different kind of data that has recently received a lot of attention, thanks to the U.S. government's decision to distribute

its census data on CD-ROM, and consequently in great quantity, is *numeric* data. Numeric data of course require an entirely different sort of database management system and user interface. The ideal here is not simply to provide access to tables as if they were pages out of printed volume, but to provide access to the raw data that can then be manipulated statistically. Rather than searching for occurrences of terms in records or full text, one would like to select a subset of the universe of data and then perform statistical tabulations on it and produce visually pleasing displays. For example, using the U.S. census as an example, select the universe to be all households living in towns with a population less than 5,000 in the state of Vermont. Then perform statistics on household annual income: mean, median, standard deviation, frequency by $10,000 increments. Produce a bar graph to illustrate these results. Now perform the same operations using the analogous universe in the states of New York, California, and Mississippi.

A living example of such statistical manipulations is Dartmouth's SPSS server, which in this simple and somewhat trivial case is showing frequencies of occurrence of signs of the Zodiac for birthdays of individuals in a specific population (ICPSR 1991 General Social Survey). The chart is produced in real time, from variables selected by the user (Figure 3).

MULTIMEDIA

Of course the newest area of exploration is that of aural and visual media and the combinations of all media into what is known as multimedia. The emphasis in audiovisual media has been mostly on the *delivery* of the "documents"; no small problem in itself, but access to the contents is still largely text based.

For example, in a database of musical sound recordings, one would obviously need to access recordings by name, e.g., *Beethoven's Fifth Symphony* conducted by von Karajan with the Berlin Philharmonic Orchestra. Such indexing is nothing new; the main problem is delivering the sounds over a network and playing them on a workstation with reasonable fidelity. One would like to be able to retrieve songs by indexing the music itself; a user should be able to sing a melody or play it on a keyboard, or enter a harmonic progression and then retrieve the citations for the pieces that match it, and then hear the matching music. This is not an unknown concept; "thematic indexes" that organize musical themes by their melodic intervals have existed in paper for a long time. Indexing of images is even more complicated, since the actual shapes must somehow be encoded into structures that can be referenced.

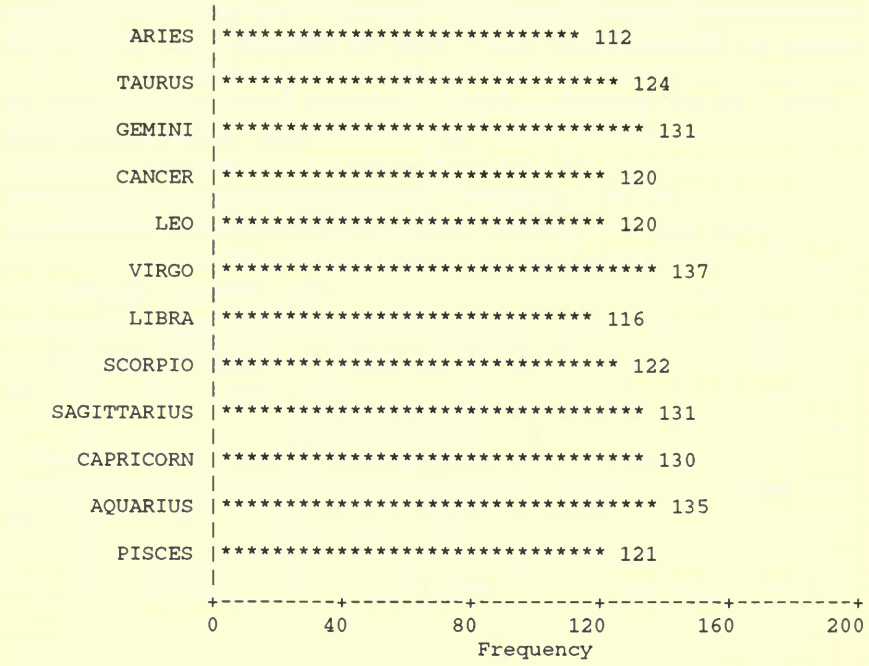

Figure 3. Chart produced from user-selected variables

At this point, the main effort in the multimedia area has been in the use of hypermedia as a presentation mechanism. Here still, the emphasis is on the presentation and delivery of media, with access following a kind of stream-of-consciousness model. Any actual indexing that is done is still textual.

CONCLUSION

The important point to remember, with the variety of information that can now be delivered to the desktops of users anywhere in the world, is that each has different needs regarding access and delivery. New database engines are needed to provide access to these data resources, and new delivery mechanisms will display them. The challenge is to develop the instruments that will accomplish this.