

---

## Laboratory tests: automatic systems

Robert N. Oddy

### 9.1 Laboratory experiment in information retrieval

A laboratory is a sheltered place where one constrains the enthusiastic abandon of the real world in order to attempt to lay the blame for one apparently capricious phenomenon upon another. The extraordinary difficulties experienced in all areas of science when designing experiments testify to the complexity of the interactions between natural phenomena. Experiments are concerned with relationships between events or states as reflected by the measurements that we choose, and are able, to make. It is invariably necessary to control, or at least to keep tabs on, factors which are not the immediate concern of the experiment, but are suspected to have some influence on those that are. These comments apply whether an experiment is conducted in laboratory conditions or not. What, then, are the characteristics of a laboratory experiment which distinguish it from a real life experiment? I do not think that there is a clear distinction. However, one might point to a tendency for the experimenter to initiate events at his convenience in the laboratory, whereas he will observe natural occurrences in real life. In a laboratory experiment it will often be possible to *control* subsidiary variables: in a real life experiment one is more often faced with the problem of eliminating them in the analysis. It is easy to find exceptions to these tendencies. In experiments on aspects of human behaviour, for example, it is not always possible to control variables as the researcher would wish, even within the laboratory. Human cognitive activity is a substantial component of information retrieval, and in another chapter of this volume, Keen has discussed the problems of laboratory experimentation on that aspect of the field. In contrast, this chapter will consider laboratory work on the mechanical components of information retrieval. The events that the experimenter observes, in this case, are all executed by a machine whose characteristics can, to a large extent, be determined by the experimenter himself. Thus, in principle, he can exercise a considerable degree of control over variables. Indeed, it is hard to think of a 'purer' laboratory environment in science. The most substantial examples of the type of work to which I refer are the long series of tests with the Smart system (see Chapter 15) and those conducted over a period of several years at Cambridge University by Karen Sparck Jones, C. J. van Rijsbergen and others.

Information retrieval is the instrumentalist wing of information science. Most laboratory work has been explicitly directed toward the establishment of system design principles. Where information retrieval tests have been effected automatically, the theories under test have almost always been prescriptive. This should not be taken for a truism: if a theory is tested by means of a computer program, or other machine, it does not follow that it is simply prescriptive. It is not in practice always true, and certainly not *necessarily* true that a program, constructed as laboratory apparatus, is in some sense a prototype for a real life system. It is quite possible for a program to act, primarily, as a formalism, or detailed interpretation of, say, a descriptive theory. In fact, in the artificial intelligence field, this is frequently the intention of the programmer<sup>1</sup>. However, within the mainstream of research on automated information retrieval, it happens to be the case that theories have been predominantly prescriptive, and laboratory systems have been put up as potential prototypes. Perhaps it would be realistic to view these computer test environments rather as engineering workshops than as laboratories.

Topics that have been investigated in computer laboratories include classification of index terms<sup>2</sup>, document clustering<sup>3-5</sup>, automatic indexing and term weighting<sup>6, 7</sup>, relevance feedback<sup>8-10</sup>, vector space models<sup>8, 11</sup> and probabilistic theories<sup>12, 13</sup>. The usual way of testing the ideas has been to evaluate the ability of retrieval programs based upon them to separate relevant from non-relevant documents. Many aspects of the experimental methodologies used in this type of work derive from those developed by Cleverdon, particularly in the second Cranfield project<sup>14</sup>.

The research methodology which has dominated laboratory work on automated information retrieval can be summarized by *Figure 9.1*. (There are several obvious feedback loops which I have omitted from the picture.) Empirical knowledge (about indexing languages, for instance), combined with the researcher's own intuition, lead him to state some assumptions about the inputs to, and objectives of an information retrieval system. From these he will attempt to derive a system specification perhaps by means of a structure of mathematical deductions. Thus, he can build computer programs which create and organize collections of document descriptions, retrieve references in response to compatibly formulated queries, and monitor their own activities for evaluation purposes. The evaluation uses test data, that is documents and queries chosen by the experimenter and with known characteristics; and it is normally the performance of the system with respect to the objectives that is evaluated, and not the plausibility of the assumptions or theory directly. Over the years the amount of rigour and effort allotted to the various components of the methodology have fluctuated. For example, as experience has been gained with certain classes of retrieval test system, programs have been assembled into flexible packages, so that new mechanisms can more easily be built from the components of old. Thus, the effort required in implementing programs has declined. At the same time, there is a new wave of mathematics in information retrieval research<sup>15, 16</sup>. The assumptions are stated more rigorously than before, and the theoretical development prior to system construction has become a focus of attention<sup>5, 13, 17, 18</sup>.

Another discernible variation in methodology, with time, is that the



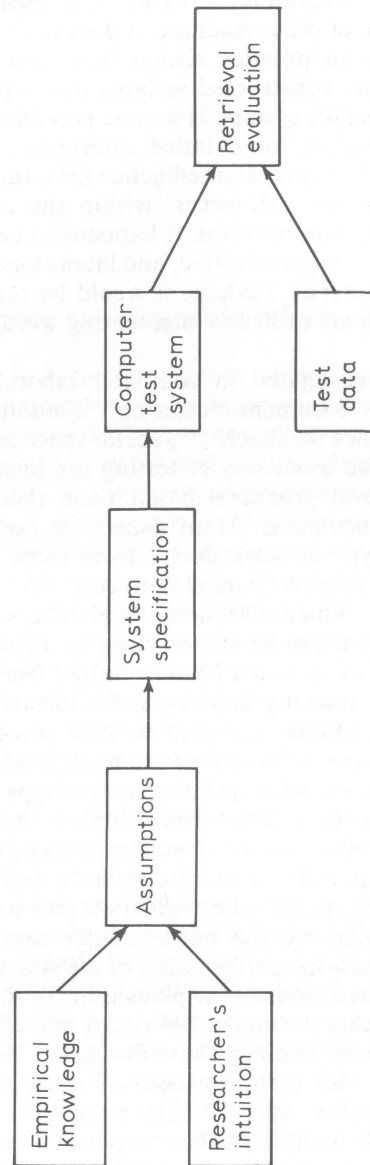


Figure 9.1. Summary of common research methodology

treatment of concepts relevant to information retrieval shifts from one region of the diagram to another. Ideas that previously made their first *well-specified* appearance in the programs now have a place in the mathematical structure of the prescriptive theory from which the programs are derived. An example of this process is evident in Croft's work<sup>5</sup>. Croft proposes a theoretical underpinning of known<sup>3</sup> techniques for searching clusters of document descriptions. Such matters as matching functions and cluster representatives are dealt with in the theory rather than being specified in an *ad hoc* manner in the program. (There are, of course, analogous phenomena in the development of other prescriptive theories: men and women flew with a certain degree of success at a time when they had dangerously little knowledge of aerodynamics.) Theories of information retrieval based on cognitive considerations are in their infancy at the moment, and we find that a number of relatively imprecise assumptions are made; the theory consists largely of non-mathematical arguments leading to a prescription of a general nature only; and a program design (if such exists) in which, consequently, *ad hoc* decisions abound. An example of work at this early stage of development is the attempt of Belkin and Oddy<sup>19</sup> to design an information retrieval system based on a notion of anomalies in the state of knowledge of an enquirer. Perhaps some of the formal structures which evolve in the programming will ultimately find their way into a more formal theory.

It is not my intention in this chapter to survey computer-based laboratory testing of information retrieval techniques. My concern is rather with the role of this type of laboratory work in information retrieval research, present and future, and with its limitations and difficulties. The potential benefits of the methodology can be summarized as follows:

- (1) *Control*. The whole test is performed by a machine and is thus, in principle, entirely manageable. The computer is a perfect laboratory assistant. All experiments are exactly repeatable, and observation (monitoring) is carried out with accuracy and consistency. Components of the system can be isolated and modified or replaced, without affecting the rest of the system. The components can therefore be individually evaluated.
- (2) *Speed*. Needless to say, the execution of searches can be very rapid on a computer, and evaluation measurements can easily be collected and processed, automatically and immediately. In addition, amendments or corrections to a search strategy can often be made by editing small sections of the program, within a matter of a few days.
- (3) *Power of expression*. Programming gives us an additional formal mode of expression for models and theories, and therefore has a useful part to play in theory development.
- (4) *Prototype development*. One can rarely copy program code from a laboratory environment directly into a real life system. However, the laboratory program can be a useful aid to operational system specification.

Some of the limitations of the methodology, and the difficulties associated with it, are:

- (1) *Restricted view*. Human factors in operational information retrieval systems are usually not taken into account in tests of the type which we are presently considering. Some factors, such as command language and display contents and format, could conceivably overshadow the

theoretical considerations which motivated the system design. (I am thinking, particularly, of relevance feedback techniques<sup>8, 9</sup>.)

- (2) *Conceptual simplification*. Very often, extremely complex phenomena, to which the behaviour of real life systems is closely linked, enter the laboratory test as simple abstractions: for example, an information need may become a list of index terms, and a relevance judgement a truth-value. The significance that we place upon test results will depend upon the appropriateness of these abstractions. This problem is not specific to automated laboratory testing, but I wish to mention it here because it is more easily brushed aside when users are not present, reminding one of the simplifications that have been made.
- (3) *Extrapolation problems*. It is usually necessary to work with small samples of document collections and queries in the laboratory, so that it is feasible to assemble complete relevance judgements, and so that the demands on computer resources are not excessive. The statistical problem of extrapolating from laboratory results to real life systems is severe. This chapter is not directly concerned with the problem (see Chapter 2); but it should be borne in mind in the present context.
- (4) *Technical faults*. It is remarkably difficult to ensure the correctness of a computer program of any substantial size, and often it is not at all obvious from its behaviour that a program is faulty. In principle, therefore, we have the problem of demonstrating the credibility of test results.
- (5) *Communication difficulties*. A detailed account of the workings of a program makes for very heavy reading, and is often unnecessary in research papers: a summary, or description by analogy is usually more illuminating. There is, however, a danger of ambiguity if programs so described contain unobvious interpretations of the theory, or *ad hoc* procedural 'theories'.

Before I attempt an assessment of the methodology, I shall give a fairly detailed description of the nature of data and programs used in the mainstream of automatic laboratory information retrieval testing.

## 9.2 The test collection

The data used for information retrieval laboratory testing has been the subject of some discussion recently<sup>20-22</sup>. Retrieval experiments make use of what are commonly referred to as 'test collections'. Such data consists of a static collection of document descriptions, queries and relevance judgements. The numbers of documents and queries are usually small so that the labour required for setting up the test collection (particularly for obtaining complete relevance judgements) is kept within reasonable bounds. This has had serious consequences for the acceptability of the results obtained in laboratory situations, and it is consideration of this problem that has lead Sparck Jones and van Rijsbergen<sup>23</sup> to propose that a large 'ideal' test collection be designed for use by a wide range of experimenters. I shall return to this idea presently, but should like first to deal with the small test collections traditionally used in tests of automatic information retrieval systems.

The raw data for a test collection can take a number of forms. The whole

text of the documents may be available in machine-readable form. More frequently, it will be the abstracts that are available, or perhaps, only the titles. Index terms may have been assigned to the documents manually, using one or more indexing languages. Queries are often expressed as short natural language questions or statements, and may be accompanied by terms chosen manually from an indexing language. The constitution of the raw data will, of course, result from compromises between the experimenter's objectives and data available. Each query must have associated with it a list of documents judged relevant: the judgements are usually recorded as points on a simple ordinal scale of relevance. Before retrieval tests are conducted, the collection is typically transformed into a convenient numerical form.

Some test collections are used by a number of researchers working in different institutions. The collection constructed by Cleverdon in the mid-1960s at Cranfield<sup>14</sup> is one that has seen, perhaps, the heaviest use in computer-based tests, although it was built for a manually executed experiment. An experimenter who chooses to make use of an existing test collection may use the raw data, if he wishes to try a new text processing technique, or he may be able to take advantage of existing indexing as embodied in the numerically coded data.

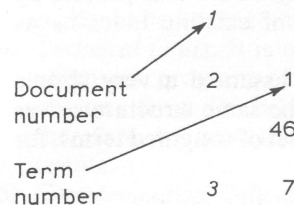
All but a small minority of experiments have assumed a very simple structure for document descriptions and queries; the same structure serves for both. A document (or query) is represented by a set of weighted terms, for example:

cluster (6), file (4), method (4), document (3), single-link (3), hierarchy (3),  
algorithm (2), compare (2), time (2), heuristic (1), theory (1), . . .

The weights are derived from the text and usually indicate the importance, in some sense, of the term to the subject matter of the document (query). The terms can be numbered serially (in an arbitrary order), and if the collection is static, as it invariably is in laboratory tests, the representative can be described as a vector whose elements are term weights. This is the symbolism employed in most of the literature on the SMART experiments<sup>8</sup>. In many tests, a special instance of this representational structure is used, namely a set of *unweighted* terms: if the weights are all the same, they can be dropped from the descriptions. The more complex structures which have appeared in experimental work, such as term classes<sup>2</sup> and document clusters<sup>3, 8, 24</sup>, are derived from simple representatives like these.

In a laboratory situation, it is very unusual for the automatic system to operate in the presence of human enquirers. Consequently, it is not necessary to provide facilities for interpreting and displaying textual data in the search programs. Retrieval and information structuring programs can be simplified considerably if they do not have to handle text. Therefore, documents, queries and terms are given numerical names (serial numbers). To the programs, a document *description* is a document number followed by a list of term numbers, each of which may, in some collections, be accompanied by a numerical weight. A query has the same structure. The relevance judgements pertaining to a query are typically denoted by a query number followed by a list of the serial numbers of documents deemed relevant. From files of records of this sort, other files can be derived by relatively straightforward programs, for the later convenience of retrieval programs.

For instance, the inversion of the document descriptions file contains one record for each term: the term number followed by a list of document numbers. An extremely clear account of the organization of test collection data for experimental information retrieval is to be found in Sparck Jones and Bates' report<sup>6</sup> (p. D1 ff) on their work at Cambridge University. They point out the formal similarity between files of different purport. For instance, a file in what they call 'a b' form consists of a sequence of numbers, *a*, each element of which is followed by a list of numbers, *b*, terminated by a character '/'. A set of document descriptions can be encoded in the 'a b' form (see Figure 9.2), as can a set of queries, and a set of relevance judgements. A single program can be used to invert any of these files, because they are in a standard format, and the same is true of any other process that is required. Having generated the primary files (document descriptions, queries, relevance judgements) from raw data, Sparck Jones and Bates went on to create a standard set of auxiliary files, such as inversions and frequency data, as a matter of course, for all their test collections.



	1	5	57	86	101	110	190	583	/
Document number	2	17	19	157	193	282	291	407	
		466	583	666	702	/			
Term number	3	78	86	96	163	740	/		
	.	.	.						
	.	.							
	.								

Figure 9.2. Document descriptions in 'a b' form

If an experimenter wishes to work from raw data, he must equip himself with programs to derive the numerical representation of the test collection. Typically, the textual material is first processed to form a dictionary of terms, or 'stems', with associated term numbers. Then, the texts are scanned again, their component words matched against the terms in the dictionary, and replaced by the corresponding term numbers. The algorithms employed to construct the dictionary vary from one experimental system to another, usually in minor ways, and may include automatic suffix stripping and allow for the manual inclusion of synonyms. Accounts of the principles underlying these methods can be found in van Rijsbergen<sup>12</sup> and Salton<sup>25</sup>. Note that at the moment I am concerned with the primitive data comprising a test collection: the more sophisticated indexing structures which have been the subject of most recent automatic information retrieval experimentation are derivations or transformations of them. There are a few exceptions. For example, some retrieval methods require a syntactic analysis of the document and query texts<sup>26, 27</sup>. There has been very little evidence of this type of work in information retrieval laboratories for several years now, following generally disappointing results<sup>28, 29</sup>. Other approaches make use of the *positions* of words in the text, so a concordance must be generated from the



raw data of the test collection. Full text retrieval systems, such as STATUS<sup>30</sup> and STAIRS<sup>31</sup> fall into this category, although they do not appear to have been tested in laboratory conditions. Very recently, this sort of data has been used by Belkin and Oddy<sup>19</sup> (who are investigating the computer modelling of anomalous states of knowledge) to generate associative structures from individual texts. Simple files of term postings, weighted or unweighted, are not adequate for this purpose.

For the most part, information retrieval test collections are small: typical numbers of documents are 200, 424, 800, 1400, 11 518; and queries number 42, 24, 63, 221, 193. Robertson has discussed the difficulties of extrapolating results obtained on such small samples in the present volume, and elsewhere<sup>22</sup>. At this point, I shall merely mention some reasons for this state of affairs. First, data collection often involves a great deal of drudgery and cost for the researcher or his assistants. Suitable documents and queries have to be selected, perhaps indexed manually, and prepared in a convenient machine-readable form. Relevance judgements must be made, either by the originator of the query, or by a subject expert. For exhaustive data, the number of decisions required is the product of the number of documents and the number of queries. The largest collection, that I am aware of, for which exhaustive relevance judgements have been made, is the Cranfield 2 test collection assembled by Cleverdon, Mills and Keen<sup>14</sup>. There are 1400 documents and 221 queries; thus 309 400 relevance decisions were made. An experimenter will naturally prefer exhaustive data to simplify the evaluation methodology, and particularly if he is concerned with a relevance feedback mechanism. Other reasons for the use of small test collections are related to the computing aspects. Many processes to which the data are subjected—classification procedures, matching and ranking, for example—consume quantities of computer time which depend on collection size factors (numbers of documents, terms and queries) in a worse than linear fashion<sup>12</sup>. The experimenter is therefore obliged to pay some attention to computational efficiency. The most productive experimenters have made use of large, fast computers. With such equipment it is possible to hold in *core* storage substantial parts (or extensive derived structures) of a test collection of several hundred documents. A program will run very much faster in this circumstance than if it must make frequent reference to several disk files, and so the experimenter will obtain a speedy job turnaround from what is often an over-subscribed university computer service. Paradoxically, an experimenter with a small computer will tend to write programs which can cope with larger files, because for him, even a small test collection is a large data structure. (But his experimental progress will, of course, be slower.)

### 9.3 Laboratory programs

The programs used in experimental work differ in a number of ways from those which would be used in an operational environment to achieve the equivalent processes. Reasons for this are that no interface is required for a human searcher, the goals of experimental programs are not the same as operational ones, and the test collection is assumed to be relatively small and static, with known bounds for all its dimensions. It is common for an

experimenter to disregard the problems of implementing his algorithms as components of a full-scale operational system, and concentrate his programming effort on achieving efficiency in his test programs, and thus a quick turnaround for each stage of the experiment. One often encounters the view (or faith) that, if the merits of the *logic* of the retrieval technique warrant use in a large-scale, real life system, ways will certainly be found to create an acceptable implementation, either through ingenious programming or by devising new hardware. This seems not unreasonable when one considers the history of computing.

I should like to illustrate the nature of information retrieval laboratory programs with a simple example. Although this should not be regarded as a description of an actual program, it does draw upon ideas present in existing programs. Suppose that we wish to model a system that responds to a user's query with a ranked list of references to documents. The document ranking is to be determined by the sum of the weights of terms which are common to both query and document. For my present purpose, it is not important to specify how the weights have been derived; suffice it to say that they are available either in the document descriptions or the queries. One evaluation method, used by Sparck Jones<sup>2</sup> involves aggregating, for all queries in the test, the numbers of relevant and non-relevant documents retrieved at each value of the matching function. (This leads to a type of microevaluation, in Rocchio's terminology<sup>32</sup>.)

An operational system would require the capability of selecting enough of the highest ranking documents to satisfy the demands of the user, and of presenting them in the correct order: a process which must be designed with some subtlety if it is not to use excessive computing resources. A laboratory model could do the task in a very different manner: ranking documents is unnecessary because there is no user to view them. A straightforward computer program can be designed along the following lines:

- (1) Set up a two-column table in core storage to be used for counting relevant and non-relevant documents (see *Figure 9.3*).
- (2) Place all the relevance judgements in core storage.
- (3) Place all the (numerically coded) document descriptions in core storage.
- (4) Proceed through the file of queries, performing the following operations for each query: Compare each document with the query to compute a matching value (which should be a positive whole number); this and the relevance relationship between the document and the query determine a position in the table, to which 1 is added.
- (5) When the last query has been processed, the completed table can be used to calculate figures for a recall/precision plot. Note that this, and *not* retrieved references, is the primary output of the program.

On most computers, the number of documents that can be handled in this way is rather limited, because all their descriptions are held in core storage while queries are processed. However, test collections usually have far fewer queries than documents: one does not often encounter experiments in the literature which use more than 250 queries. Thus, a more useful experimental computer program, which could deal with indefinitely large sets of documents, can be built by simply substituting 'query' for 'document', and vice versa, in steps (3), (4) and (5) above. We now have a program which looks structurally

	Relevant	Non-relevant
Matching value 1		
2		
3		
		•
		•
		•
m		

Figure 9.3. Table for counting relevant and non-relevant documents at each matching value

very different from modern online systems; although it resembles the early magnetic tape-based retrieval systems which processed queries in batches.

If the researcher is concerned with aspects of the interaction between the user and the computer, then the additional complexity of the retrieval process will usually oblige him to adhere more closely to the structure of real life systems. An experiment to evaluate the effect of relevance feedback<sup>8-10</sup>, for instance, would require the program to select a retrieved set of documents, so that their indexing can be used together with relevance decisions to compute a modified query. No doubt, more subtle and efficient ways could be devised, but this may be self-defeating, in that one's confidence in the correctness of the programs may be endangered.

Programs which model some aspect of the user's cognitive behaviour may have no formal specification: the details of the model are worked out in the process of writing, and trying the program. These programs will almost certainly be prototypes (see, for instance, Oddy<sup>33</sup>), inasmuch as they proceed as a real life system would, simply because you cannot design algorithms which efficiently capture the essence of a technique for test purposes until you know precisely what that technique is.

Those who have been engaged in laboratory work in information retrieval over several years now have extensive suites of programs which can be applied in a wide variety of combinations to test collections. The well-known Smart system is described in Salton<sup>8</sup>. Programs are included for deriving the numerically coded form of a collection from natural language text, for clustering documents, for selection of clusters appropriate to a query (the whole collection may be selected), for searching the selected clusters, and for evaluation. All of these programs have a number of optional facilities. The program suites used at Cambridge University are described and discussed by Sparck Jones<sup>2</sup> and Sparck Jones and Bates<sup>6</sup>.

A major advantage of an automatic information retrieval laboratory, equipped with a flexible program suite and a number of conveniently formatted test collections, is that it is possible to try to accumulate convincing

evidence for general hypotheses by conducting series of experiments. A comparative test which indicates, on the basis of one test collection, that one setting, A, of certain factors gives better performance than another setting, B, can be repeated on a number of other collections. In fact, both Sparck Jones and Bates<sup>6</sup> and Salton<sup>8</sup> have reported that a number of results hold for several of their test collections. A comparison<sup>6</sup> of the results of similar experiments by different groups of researchers, however, shows that there is often broad agreement, but that the situation is confused by variations in the evaluation techniques: the various performance averaging methods give materially different figures<sup>34</sup>.

## 9.4 Experimental objectives

What questions are tests of the type I am discussing designed to answer? What are the strengths and limitations of the methodology? Criticisms of the methodology, usually pointing out lack of realism, are so common as to be a part of the information retrieval folklore. Experimenters often acknowledge the problem by qualifying their results appropriately. So, how successful has the methodology turned out to be?

In his review of theoretical work in information retrieval, Robertson<sup>15</sup> discusses the role of experimental work, and distinguishes between experiments which test the assumptions on which a theory is based, and those which test theories by evaluating the retrieval effectiveness of systems based upon them. There have been very few experiments fulfilling the former role—I shall have more to say about this presently. Laboratory experiments are usually intended to determine the effect of some input parameter or system design feature on retrieval effectiveness, that is, on the system's ability to retrieve relevant documents. If the researcher views his tests as a series of engineering trials, this is the obvious approach: he is simply determining whether he has achieved his objective. It is not so obvious that it is the right approach if the researcher's objectives are scientific, in other words, if he wishes to test a theory. Recently, as Robertson<sup>15</sup> points out, theories have arisen which explicitly relate retrieval effectiveness to system parameters. (For instance, one first shows that ranking documents according to probability of relevance gives optimal retrieval performance, in some sense<sup>35</sup>, and then proceeds to devise ways of estimating that probability<sup>12</sup>.) Even in such cases, an experimenter might be accused of impatience if he moves directly to a test of retrieval performance. There are assumptions to test: if the document collection, or the system's users do not conform to the assumptions, what can the experimental result tell us about the theory? In general, nothing! If the result of the test is good, he may have an engineering success, but it is not a scientific one, because he still does not know why the system works as well as it does.

It is not within the scope of this chapter to survey the *results* of laboratory experiments in information retrieval over the past years (see those of Sparck Jones and Salton). I shall therefore confine my account to what is relevant to methodological issues. The effects of various factors on retrieval performance have been studied with the aid of test collections. The factors can be regarded as falling into two broad categories, although the boundary is indistinct.

First, there are collection parameters: of what does the raw data consist, and how is it initially processed to form the document descriptions, queries, and relevance judgements? Indexing can be manual or automatic, and based on titles or abstracts; exhaustivity and specificity can vary; thesauri and stemming procedures can be used to normalize vocabulary; weights can be assigned to index terms; various degrees of relevance may be taken into account. Second, there are variations in retrieval system features. Queries and document descriptions can be matched in a number of ways; relevance judgements relating to retrieved documents can be used in a variety of ways to improve queries; complex structures derived from the simple collection, such as term classes and document clusters, can be exploited in retrieval. All of these factors can, in principle, be controlled by the experimenter, and another way to categorize them is according to the practical difficulties of controlling them. In an automatic laboratory information retrieval system, any single parameter or feature, or any combination of parameters and features, may be varied independently of all other features, whether it makes sense to do so, or not! Factors which can be changed by small to moderate amounts of computer programming, of course, are the ones presenting least difficulty. These include retrieval system features and some of the indexing parameters. Not only is it practically straightforward to vary these factors, but it can also be done very precisely; in fact, it must be so done because the 'values' are coded into programs. Difficulties arise when controlled variability is desired in the characteristics of any data which is generated intellectually. Instances are the use of (conventional) thesaural relations in manual indexing, and the judges and scales used for relevance data. Precise control of these factors is not possible in the same way as for computational factors, because they are not so easily quantifiable or, in the case of procedural factors, specifiable. In addition, alternative forms of the test collection are required for different values of these variables, involving the experimenter in considerable effort and expense. Consequently, very little experimentation has been done with such variables.

It is the hope of the experimenter, whether engineer or theoretician, that results obtained in the laboratory would also be obtained in real life, should an equivalent experiment be conducted. If that were so, he would be in a position to make very precise recommendations concerning the organization of information within the system and the retrieval algorithms which would optimize performance. Unfortunately, this extrapolation is highly problematic. In the first place, the scale of most test collections is very different from that of operational databases. I shall not dwell on the very difficult statistical problem of extrapolation here, but refer the reader to Robertson's chapter (2) in the present volume, and to the illuminating discussion in his thesis<sup>22</sup>. It is true that some laboratory tests have used collections of the order of 10 000 documents<sup>5, 6, 36</sup>, but these are deficient in relevance information, and therefore difficult to use for retrieval tests.

## 9.5 Realism

Other problems have to do with realism: there are aspects of real life information retrieval activities, mostly to do with user behaviour, which



cannot easily be mirrored in laboratory systems, but which may have considerable impact on perceived performance. I should like to point out that these problems fall into two categories, although the categories are subtly interrelated and I shall be forced to discuss them together: some relate to what may be called *parameters* (environmental factors, system design features, charging algorithms, for instance) and their effect upon the retrieval effectiveness obtainable, and others relate to the *goals* of the user and the system and how effectiveness should be measured. The debate on what comprises the effectiveness of an information retrieval system is long and involved. Notable contributions have been made by Cleverdon<sup>14, 37</sup>, Lancaster<sup>38</sup>, Cooper<sup>39</sup> and a number of others. Van Rijsbergen<sup>12</sup> restricts the term 'effectiveness' to refer to 'the ability of the system to retrieve relevant documents while at the same time holding back non-relevant ones' (p. 145), and it is this type of effectiveness, and only this type, that is measured by very nearly all laboratory tests of automatic systems (one exception is a test by Oddy<sup>33</sup> of a browsing mechanism in which measurements related to user effort were made). Relevance-based effectiveness measures are also used *inter alia* in real life experiments. Now, in order to establish a fruitful relationship between the laboratory tests and their hypothetical real life analogues, we must ask two questions:

- (1) Is relevance-based effectiveness safely separable from other performance characteristics for experimental purposes?
- (2) Is relevance in real life the same as relevance in laboratory tests?

Aspects of performance which may be regarded as important by users include the effort that they must expend, the response speed of the system, and the cost-effectiveness<sup>14, 40, 41</sup>. If a system is poor in any of these respects then, clearly, its achievements in the recall/precision domain may simply not be appreciated by the users. However, I think the connection between the different components of performance is deeper than that. System parameters such as the types and powers of storage devices, computer processors, and communication equipment, the complexity of algorithms, the ergonomics of terminal design, and the user interface facilities<sup>42</sup> are all factors which strongly influence performance and which are not usually investigated in information retrieval tests. The assumption made is that the relevance of a document to a query does not depend on such aspects of performance. Relevance in tests is a simple abstract entity, a relation between queries and documents: any links between its real life correlate and characteristics like user effort and response time are disregarded. Of course, such links do exist and they are complex, and have yet to be investigated properly. They arise out of the cognitive activity of the user during the searching process. The user will normally be trying to fulfil some purpose, which will determine the use he makes of the system's output. His progress towards his objective, and thus his attitudes towards the search output will vary as the search itself proceeds (of which, more will be said presently). Therefore, we must expect every apparent aspect of system behaviour to have some influence on relevance-based effectiveness measurements. I am aware of no experiment which attempts to quantify any of this class of effects, although the effects are widely acknowledged<sup>43</sup>, so I am unable to answer question (1), above.

I have said that in laboratory tests, simple abstractions of the phenomenon

of relevance are adopted for evaluation. Recently, theories for information retrieval have emerged out of the background of experimental work and they are founded upon the same abstractions. Robertson and Belkin<sup>44</sup> have made a distinction between two principles for ranking documents in response to a query: one can rank according to *probability* of relevance or *degree* of relevance. Probabilistic theories assume that relevance is a boolean variable, that is it can take on one of two values, denoted: relevant and non-relevant. Systems based upon the use of matching functions or similarity measures, e.g. co-ordination level and cosine correlation, would appear to be estimating the degree of relevance, although evaluation is usually done with dichotomous relevance judgements. Other assumptions typically made about relevance are that, for any document-query pair, the relevance judgement is independent of time, and of the other relevance judgements.

The idea of relevance in the context of real information needs is complex and poorly understood—information retrieval research can be viewed as our attempt to understand it—and has been the subject of a substantial literature, to which I refer the reader through Saracevic's excellent review article<sup>45</sup>. A document retrieval system user generally makes a series of decisions about documents. First, he may make a note (perhaps mental) of the existence of the document; then he may decide to look at the document's contents; finally, he may decide to make use of those contents in his own work. All of these decisions can be regarded as relevance judgements, and the outcome of each obviously depends upon the enquirer's perception of the document, the purposes of the enquirer, and his existing knowledge. By his perception of the document, I mean what aspects of its description or content the enquirer sees (title, abstract, index terms, for example), and in what circumstances he sees them (online or in a batch printout). The 'cognitive view' of perception<sup>46</sup> is that perceived objects are interpreted through the knowledge, or world model, of the perceiver. Online systems are often provided with so-called browsing facilities, presumably to encourage the interleaving of mechanical and intellectual effort (recommended by Doyle<sup>47</sup>, for instance). Unfortunately, the high cost of using today's online services discourages many users from taking the time to contribute significant intellectual effort *during the search*. (Let us hope that this is a temporary situation.) Nevertheless, even under these circumstances, a user's state of knowledge relating to his purpose changes during a search. The purpose itself may also undergo change if Belkin's<sup>48</sup> analysis of the information retrieval situation is to be accepted. A user comes to an information retrieval system because his state of knowledge is, in some way, anomalous; that is, he has recognized that his mental world model cannot cope with his problem in hand. It must be assumed that he may not be able to specify what information is needed to resolve the anomaly. So, his conceptualization of his purpose in searching the literature is subject to modification as his knowledge, and thus the anomaly in his knowledge, changes. The consequence of all this is that relevance is dependent upon three factors related to the user—perception, purpose and knowledge—which are causally closely related to each other, and subject to variation in the course of an interactive search. The picture of relevance decisions that we are obtaining is very different in nature from the relevance judgements included in test collections. Thus my answer to question (2) is clearly 'No'.

What implication does this argument have for the results of laboratory

tests? It is that, in principle, they are inconclusive. An example should illustrate the point.

Experiments on relevance feedback<sup>8-10</sup>, which exhibit relatively outstanding effectiveness, make use of the same set of relevance judgements for two separate purposes. First, they are used to simulate the user's feedback, that is his reactions to documents retrieved, and thus they determine the query modification. Second, they are used to evaluate the effectiveness of the technique. In real life, two *distinct* sets of relevance judgements would be used for a corresponding experiment. As he sits at the terminal, the enquirer would make instant judgements, according to his perception of the documents during the search session. His evaluation of the search would be made at a later time, on reflection. Theories are unrealistic in this respect, and experimental arrangements fulfil their unrealistic assumptions, and are thus inconclusive in relation to real life systems.

What laboratories tests do is to isolate small portions of large, complex systems for independent study. This is a procedure which, when applied to systems with human components, has been strongly criticized<sup>49, 50</sup>. General systems theorists point out that the interactions between the components of a system are profound and cannot be ignored or artificially controlled if the system's behaviour is to be understood—'the whole is greater than the sum of its parts', they are wont to say. Simon<sup>51</sup> describes artifacts (information retrieval programs, for instance) as relatively simple organisms whose behaviour is nevertheless complex because they react to a complex environment. Knowing how they react to a simple or controlled environment does not necessarily help us very much. Modern mathematical theories of information retrieval have arisen out of extensive experimental experience mainly with laboratory programs. (That is perfectly reasonable, and represents a worthy benefit of the earlier laboratory tests.) Unfortunately, tests of the assumptions and predictions of the theories have been mostly confined to the same laboratory environment. Partridge's<sup>52</sup> comment on difficult software engineering tasks applies to information retrieval: 'A more or less "wicked" problem is often the initial "given" and, although it is top priority to transform this problem into a formal analogue before proceeding further, the subsequent implementation swims or sinks in the light of real world application: that is, back in the domain of "wicked" problems' (p. 244). My view is that in this field, the type of laboratories I have described are not the best places in which to put theories to the test—unless we are only interested in theories of test collections! They do, however, have a valuable role as information retrieval engineers' and theoreticians' workshops, where ideas can develop and tentative or exploratory tests can be made. Coupled with this, I would advocate the development of methodologies which facilitate real life testing of the assumptions and consequences of promising theories<sup>53</sup>. I feel uneasy about solutions to the realism problem which involve larger and better planned test collections<sup>23</sup> and repetition of tests on a number of different test collections<sup>6, 8</sup>.

## 9.6 Model building

The automatic information retrieval laboratory environment is used for another type of computer-based activity—that of model building. Because

this is a less common activity, the terms 'theory' and 'model' are used almost interchangeably in the information retrieval field. A proper discussion of the differences between the usages of these words in science would be extensive and out of place here<sup>54, 55</sup>. I should like to use the term 'model' to refer to a particular type of theory: one that attempts explanation of a phenomenon by describing a *process* and exploring its effects. This gives us an approach to the problem of relating the outputs to the inputs of our systems which is distinct from mathematical argument. Recently, mathematical theories have been used to suggest retrieval algorithms<sup>16</sup>: I would not call such an algorithm a 'model' while it is still entirely derivative. One might ask why we should wish to build models without a sound mathematical theory. In information retrieval, we are trying to reproduce automatically a cognitive act: the decision as to whether a document, or 'item' of information, is relevant. It is notoriously difficult to formulate mathematical theories which account for, and enable us to handle the *variability* in human behaviour<sup>56, 57</sup>. According to Farrell<sup>50</sup>, many psychologists believe that outputs cannot be related to inputs in any simple (mathematical) way, and that internal states must be taken into account. This is best done by modelling.

If we are looking for retrieval systems which are more responsive and adaptable to the individual user we could do worse than model the behaviour of good *human* information providers—subject experts, librarians and information scientists. Following a 'systems' approach, one can formulate a high-level structural model of their behaviour, illustrated by a block diagram showing the relationships between components, for instance, and experimentally determine whether the human behaviour fits the model (see, for example, Olney<sup>58</sup>, Ingwersen and Kaae<sup>59</sup>, Bivins<sup>60</sup> and, for methodology, the chapter by Keen (8)). Alternatively, one can attempt to build a computer program with the hypothetical structure, run it, and observe its behaviour. In the latter approach, the modeller must elaborate the meaning of all his high-level components in a very precise way.

Before making more general points, I should like to illustrate the process, briefly, from my own experience, by explaining how certain essential parts of a particular program, called Thomas<sup>61, 62</sup>, were written. The program is an interactive system which provides a browsing facility for the user. He is not required to formulate a query, and as the dialogue progresses, his reactions to the indexed references are used by the program to build a picture of his area of concern. At the highest level, a 'cognitive' model of dialogue, essentially like Hollnagel's<sup>63</sup>, is assumed. Each participant has his own image of the world, which includes an image of the other's world-image: let me call this included image a 'meta-image'. Communication becomes more effective as these meta-images more accurately portray the current concerns of the participants. To implement this model in a man-machine dialogue, the computer program must have a fund of knowledge about the world, and a means of representing its image of the man. It must be able to improve that meta-image as the dialogue progresses, that is, in response to the man's utterances, and in displaying information to him it must aim not only to give him relevant references, but also to help him form his image of the program's world-image. (We are beginning to see some guidelines for a program design.) Thomas' world-image is a graph in which the nodes represent documents, subject terms and authors, and the arcs associations between them, derived

from conventional indexed documents and a thesaurus. The form that its image of the man takes is a subgraph (from its own 'knowledge') together with a few notes about his reactions to information displayed during the dialogue. Now, we turn to the program. The processes in the program were refined in a hierarchical fashion as the meanings of the program's goals were gradually elaborated. The description of one cycle in the dialogue (i.e. at a very high-level in the model) is as follows:

- (1) read a message from the man;
- (2) use it to influence the state of Thomas' meta-image;
- (3) use the meta-image to respond to the man

(For readers who are programmers, I should say that each of the three steps is coded as a procedure call.) I shall not elaborate the first step here, except to say that it includes an interpretation of the text of the man's message which makes reference to Thomas' images. Thus the communication can be regarded as having a cognitive basis<sup>46</sup>. Let us proceed just one more level down in the (informal) program definition with the description of step 2:

- (1) update Thomas' self-assessment according to the man's reaction to the last display;
- (2) 'prune' Thomas' meta-image in the region of nodes which the man does not appear to like;
- (3) 'enrich' the meta-image in the region of nodes in which the man shows interest;
- (4) incorporate new material in the meta-image if the man has explicitly mentioned new words;
- (5) make sure the meta-image is not fragmented

I am not attempting to make this specific program fully comprehensible to the reader (the details can be found elsewhere<sup>33</sup>), but to give him or her a feel for the way a model may be elaborated through programming. For this purpose, I hope that I have now carried the illustration far enough.

The process of elaboration of the model is by no means automatic, and the result is not unique. Some decisions must be made by the programmer in an *ad hoc* way: he uses intuition, introspection and, when he can, the dictates of theories which, he feels, are applicable. This is a well-known method in Artificial Intelligence work, and has been defended ably by Lindsay<sup>56</sup> and Schank<sup>64</sup>. A computer model rarely achieves its final form at the first attempt. Indeed, if it does, one can conclude that it has been abandoned, due to failure or lack of interest. The modeller will try to correct defects in its performance by making what he hopes are appropriate modifications to the program, and hence the model. This is a type of theory which can be very readily changed, and re-tested<sup>65, 66</sup>. The advantage of this facility becomes clear when we consider that, even in the case of a fairly simple procedural model, as incorporated into Thomas, it is extremely difficult to account for its behaviour (i.e. relate output to input) mathematically. The computer modelling



methodology is also a ruthless critic of ill-conceived or vacuous theories, of which I suspect Information Science could boast a number.

I should draw attention at this point to the fact that there are some information scientists who do *not* view computer modelling as a helpful means to understanding, and regard it as somewhat unhealthy. Rosenberg<sup>67</sup> writes: 'If, as I believe, the nature of human information processing is fundamentally different from machine information processing, then the development of digital computer systems becomes an obstacle to the understanding of information and its use' (p. 266).

As with the more conventional automatic information retrieval laboratory testing, computer modelling has its difficulties. First, programs can become very complex, to the point when they are incomprehensible, as patches are made to take care of unwanted behaviour. Simplification is then necessary if the model is to have any scientific value, and this may be dependent upon having a new insight. Second, evaluation of the effectiveness of the retrieval program is, as usual in information retrieval, fraught with difficulty. If we substitute a complex notion of relevance for a simple one in the theory, then we may even deprive ourselves of the easy way out—evaluating the system on its own terms. Finally, computer models can be criticized in the same way as other laboratory systems, for unrealistic isolation of portions of the system for study<sup>50</sup>.

What role can computer modelling in the laboratory play in the development of information retrieval, and what is its relationship to laboratory systems derived from mathematical theories? Clearly, modelling offers us a mode of expression which is distinct from mathematics, and is capable of coping with situations which do not appear to lend themselves to mathematical treatment. Should we regard a model as a stop-gap: a means of getting our ideas in order so that a mathematical theory can be evolved? Or should we accept the belief expressed by Sloman<sup>68</sup> that viewing complex phenomena as computational processes 'should supersede older paradigms, such as the paradigm which represents processes in terms of equations or correlations between numerical variables' (p. 3)? I do not think we have enough experience of computer models to know the answer. In the meantime, the two can fruitfully coexist. The mathematical theories that exist at present refer to limited regions in the information retrieval domain. Perhaps models can be developed to provide the appropriate contexts for the application of the theories.

## 9.7 Program correctness

It is well known that a fault-free program is a rare thing. Some faults cause the program to break down or behave in an obviously erroneous way; others lurk in the program unnoticed for a considerable length of time. These latter faults are responsible for deviations from correct behaviour which are sufficiently small that the results still appear reasonable to the experimenter. How 'small' that is depends upon the expectations of the experimenter. I shall not dwell on the obvious type of fault, but make a few comments about the more subtle ones. It is clear that an experimenter must make every effort to ensure that the results obtained from his computer programs can be relied

upon. In recent years a considerable amount of research has been done on the problems of reliability in computer systems in general. A very useful collection of papers is to be found in Anderson and Randell<sup>69</sup>. Among the techniques for improving program reliability, those for fault avoidance, as opposed to tolerance, and those not requiring a special programming system are of most interest to the experimental information retrieval programmer. Disciplined use of a good programming methodology and a language which allows for a reasonably natural expression of the process and data structures are the most significant techniques.

Correctness of a program is judged with respect to a specification of the required function and behaviour of the program. The judgement can be made in two ways:

- (1) The program can be *tested*: Gerhart<sup>70</sup> discusses the principles of testing. A program must always be tested. The testing of an information retrieval laboratory program is quite conventional, although constructing test data can be tedious. However, testing can never be exhaustive. One may try to test program modules exhaustively, but modules have a habit of interacting with each other in unexpected ways when they are run together, so ultimately one is faced with the prospect of checking every conceivable output of the complete program! One compensates for the inevitably partial testing by constantly keeping an eye on the reasonableness of all output produced by the program, and by combining testing with the second method of judging correctness: reasoning about the program.
- (2) The program may be *proved* to be consistent with the specification. This is extremely difficult and the proofs tend to be unwieldy, but very informal proofs can often be done for parts of the program, if it is well structured, which are convincing enough for most purposes.

With information retrieval programs, we must be clear what we mean by the specification. I am concerned at the moment with the *technical* problem of obtaining a correct program, and not the *research* problem of obtaining an ideal program design. Thus correctness is to be judged against the researcher's design, rather than against the system user's requirement. For this concept of correctness to have a straightforward meaning, the semantics of the researcher's system specification must be quite clear, that is it must be possible to express it formally. There is no problem, in principle, if the system is a consequence of a mathematical theory. If, however, the program *is* the model, in the sense discussed in the previous section, then we are reduced to talking about validating the program against itself! The researcher will probably have a detailed description of the program (perhaps similar in appearance to the extract given above of the description of Thomas), but this is not formal. The meaning of the description is worked out in the program. It is therefore possible that the researcher will be experimenting with a model which differs from his original intention in some unknown way, and which he does not fully understand. (He will have the program text, in some form, but it does not follow that he understands the model.) Of course, many programs which have independent formal specifications for part of them also incorporate heuristics, and that fact, strictly speaking, puts them into the same category. The researcher would normally make the assumption that he

does understand his program. He may make statements such as 'I did  $x$  and observed effect  $y$ ', in which  $x$  is his less-than-formal intention. What I wish to emphasize here is the obligation of the researcher to set himself high programming standards, so that he has a high degree of knowledge about his programs.

If it is appropriate to do so, the experimenter should make use of existing, tested software. His laboratory may collect a subroutine library containing tried program modules to perform certain specific tasks. If he can incorporate some of these into his own program, he will not only speed its development and testing, but he will also, incidentally, assist in trying the modules as he runs his experiment. An important reason for trusting the numerical accuracy of the results which emerge from established information retrieval laboratories such as Smart<sup>8</sup> and Cambridge University<sup>6</sup> is that 'standard' programs and packages are used whenever possible. Frequent use over several years should have revealed most faults.

All experimental sciences suffer to some extent from the inevitable fallibility of apparatus. In addition to doing all it can to ensure that the apparatus is working correctly, the research community must maintain a moderate scepticism in its reception of results: they should be checked by repetition. In information retrieval research, there is very little repetition of tests for this purpose, perhaps because the computer-orientated research community is so small. A useful repetition experiment would be expensive and time-consuming. It is not adequate to simply obtain a copy of the original programs, edit them if necessary, and run them again on another machine. Any faults in the programs would also be copied (and it is in the program that faults are most likely to be). The programs must be written again from an independent specification. Exact agreement between the two sets of results can rarely be expected. Differences in programming language facilities, and hardware characteristics, such as computer word-size will affect the respective programmers' interpretation of the specification in minor ways, which may in turn affect results. Whether a difference in results can be disregarded is a question that must be answered in the light of the nature of the computations.

## 9.8 Conclusion

The interaction between computer-based laboratory experimentation and information retrieval theory development has been very fruitful during the last decade or so. The skills that the experimenters have learned enable them to test (some) new ideas within days, or even hours. I am sure that the well-developed situation in the laboratories has made a significant contribution to the considerable theoretical progress in the field. Some researchers appear to believe that their understanding of the information retrieval problem, through mathematical theories, has attained a plateau—quite a lofty plateau! In this work, there has been too little reference to the real world to justify this optimism. There is no denying the elegance and the power of the current probabilistic and term discrimination theories (to pick the most prominent examples) to prescribe retrieval programs which are very effective under the conditions obtaining in the laboratory. I have tried to point out reasons why

we should not be satisfied with this. What I think we need is an equally fruitful interaction between theory development and a new type of automated laboratory, closer to real life information retrieval services, where we can perhaps test models of human users in conjunction with theories of document collections.

## References

1. BODEN, M. A. *Artificial Intelligence and Natural Man*, Harvester Press, Hassocks, Sussex (1977)
2. SPARCK JONES, K. *Automatic Keyword Classification for Information Retrieval*, Butterworths, London (1971)
3. JARDINE, N. and VAN RIJSBERGEN, C. J. The use of hierarchic clustering in information retrieval, *Information Storage and Retrieval* **7**, 217–240 (1971)
4. ROCCHIO, J. J. *Document Retrieval Systems—Optimization and Evaluation*, Ph.D. Thesis, Report ISR-10, Computation Laboratory, Harvard University (1966)
5. CROFT, W. B. *Organizing and Searching Large Files of Document Descriptions*, Ph.D. Thesis, Computer Laboratory, University of Cambridge (1978)
6. SPARCK JONES, K. and BATES, R. G. *Research on Automatic Indexing 1974–1976*, 2 Vols, British Library Research and Development Report 5464, Computer Laboratory, University of Cambridge (1977)
7. SALTON, G., WONG, A. and YU, C. T. Automatic indexing using term discrimination and term precision measurements, *Information Processing and Management* **12**, 43–51 (1976)
8. SALTON, G. *The SMART Retrieval System—Experiments in Automatic Document Processing*, Prentice-Hall, Englewood Cliffs, N.J. (1971)
9. HARPER, D. J. and VAN RIJSBERGEN, C. J. An evaluation of feedback in document retrieval using co-occurrence data, *Journal of Documentation* **34**, 189–216 (1978)
10. SPARCK JONES, K. Search term relevance weighting given little relevance information, *Journal of Documentation* **35**, 30–48 (1979)
11. MCGILL, M. J. Knowledge and information spaces: implications for retrieval systems, *Journal of the American Society for Information Science* **27**, 205–210 (1976)
12. VAN RIJSBERGEN, C. J. *Information Retrieval*, 2nd edn, Butterworths, London (1979)
13. ROBERTSON, S. E. and SPARCK JONES, K. Relevance weighting of search terms, *Journal of the American Society for Information Science* **27**, 129–146 (1976)
14. CLEVERDON, C. W., MILLS, J. and KEEN, E. M. *Factors Determining the Performance of Indexing Systems*, Vol. I, *Design*, Vol. II, *Test Results*, Aslib Cranfield Project, College of Aeronautics, Cranfield (1966)
15. ROBERTSON, S. E. Theories and models in information retrieval, *Journal of Documentation* **33**, 126–148 (1977)
16. SALTON, G. Mathematics and information retrieval, *Journal of Documentation* **35**, 1–29 (1979)
17. VAN RIJSBERGEN, C. J. A theoretical basis for the use of co-occurrence data in information retrieval, *Journal of Documentation* **33**, 106–119 (1977)
18. SALTON, G. *A Theory of Indexing*, Regional Conference Series in Applied Mathematics No 18, Society for Industrial and Applied Mathematics, Philadelphia (1975)
19. BELKIN, N. J. and ODDY, R. N. *Design Study for an Anomalous State of Knowledge Based Information Retrieval System*, British Library Research and Development Report 5547 (1979)
20. SPARCK JONES, K. and VAN RIJSBERGEN, C. J. Information retrieval test collections, *Journal of Documentation* **32**, 59–75 (1976)
21. GRIFFITHS, J.-M. *The Computer Simulation of Information Retrieval Systems*, Ph.D. Thesis, School of Library, Archive and Information Studies, University of London (1978)
22. ROBERTSON, S. E. *A Theoretical Model of the Retrieval Characteristics of Information Retrieval Systems*, Ph.D. Thesis, University of London (1976)
23. SPARCK JONES, K. and VAN RIJSBERGEN, C. J. *Report on the Need for and Provision of an 'Ideal' Information Retrieval Test Collection*, British Library Research and Development Report 5266, Computer Laboratory, University of Cambridge (1975)
24. AUGUSTSON, J. G. and MINKER, J. An analysis of some graph-theoretic cluster techniques, *Journal of the ACM* **17**, 571–588 (1970)

25. SALTON, G. *Dynamic Information and Library Processing*, Prentice-Hall, Englewood Cliffs, N.J. (1975)
26. BAXENDALE, P. B. An empirical model for machine indexing. In: *Machine Indexing—Progress and Problems*, Proceedings of the Third Institute on Information Storage and Retrieval, pp. 207–218, The American University, Washington D.C. (1962)
27. HILLMAN, D. J. Negotiation of inquiries in an on-line retrieval system, *Information Storage and Retrieval* **4**, 219–238 (1968)
28. SPARCK JONES, K. and KAY, M. *Linguistics and Information Science*, Academic Press, London (1973)
29. MONTGOMERY, C. A. Linguistics and information science, *Journal of the American Society for Information Science* **23**, 195–219 (1972)
30. CROALL, I. F. *An Introduction to the STATUS II Information Retrieval System* (Version 2), United Kingdom Atomic Energy Authority, Harwell (1977)
31. INTERNATIONAL BUSINESS MACHINES. *Program Product. Storage and Information Retrieval System/Virtual Storage (STAIRS/VS)*, Program Reference Manual, SH12–5400–1, 2nd edn IBM (1974)
32. ROCCHIO, J. J. Evaluation viewpoints in document retrieval. In: *The SMART Retrieval System—Experiments in Automatic Document Processing* (Ed. G. Salton), Prentice-Hall, Englewood Cliffs, N.J. (1971)
33. ODDY, R. N. *Reference Retrieval Based on User Induced Dynamic Clustering*, Ph.D. Thesis, Computing Laboratory, University of Newcastle upon Tyne (1975)
34. SPARCK JONES, K. Performance averaging for recall and precision, *Journal of Informatics* **2**, 95–105 (1978)
35. ROBERTSON, S. E. The probability ranking principle in IR, *Journal of Documentation* **33**, 294–304 (1977)
36. VASWANI, P. K. T. and CAMERON, J. B. *The National Physical Laboratory Experiments in Statistical Word Associations and their Use in Document Indexing and Retrieval*, Publication 42, Division of Computer Science, National Physical Laboratory, Teddington (1970)
37. CLEVERDON, C. W. On the inverse relationship of recall and precision, *Journal of Documentation* **28**, 195–201 (1972)
38. LANCASTER, F. W. *Information Retrieval Systems: Characteristics, Testing, and Evaluation*, Wiley, New York (1968)
39. COOPER, W. S. On selecting a measure of retrieval effectiveness, Part 1: The 'subjective' philosophy of evaluation, Part 2: Implementation of the philosophy, *Journal of the American Society for Information Science* **24**, 87–100 and 413–424 (1973)
40. HALL, J. L. *On-line Information Retrieval Sourcebook*, Aslib, London (1977)
41. LANCASTER, F. W. MEDLARS: Report on the evaluation of its operating efficiency, *American Documentation* **20**, 119–142 (1969)
42. LANCASTER, F. W. and FAYEN, E. G. *Information Retrieval On-line*, Melville Publishing Co., Los Angeles (1973)
43. BARRACLOUGH, E. D. On-line searching in information retrieval, *Journal of Documentation* **33**, 220–238 (1977)
44. ROBERTSON, S. E. and BELKIN, N. J. Ranking in principle, *Journal of Documentation* **34**, 93–100 (1978)
45. SARACEVIC, T. Relevance: A review of and a framework for the thinking on the notion in information science, *Journal of the American Society for Information Science* **26**, 321–343 (1975)
46. DE MEY, M. The relevance of the cognitive paradigm for information science. In: *Theory and Application of Information Research*, Proceedings of the Second International Research Forum on Information Science (Ed. O. Harbo and L. Kajberg), pp. 48–61, Mansell, London (1980)
47. DOYLE, L. B. Expanding the editing function in language data processing, *Communications of the ACM* **8**, 238–243 (1965)
48. BELKIN, N. J. The problem of 'matching' in information retrieval. In: *Theory and Application of Information Research*, Proceedings of the Second International Research Forum on Information Science (Ed. O. Harbo and L. Kajberg), pp. 187–197, Mansell, London (1980)
49. BEISHON, J. and PETERS, G. *Systems Behaviour*, 2nd edn, Harper & Row, London (1976)
50. FARRELL, B. A. Some thoughts on the use of models in psychology. In: *Proceedings of the Third International Congress for Logic, Methodology and Philosophy of Science* (Ed. B. van Rootselaar and J. F. Staal), pp. 415–430, North-Holland, Amsterdam (1968)
51. SIMON, H. A. *The Sciences of the Artificial*, MIT Press, Cambridge, Mass. (1969)



52. PARTRIDGE, D. A philosophy of 'wicked' problem implementation. In: *Proceedings of AISB/GI Conference*, Society for the Study of Artificial Intelligence and Simulation of Behaviour, University of Hamburg, July 1978, pp. 238-247 (1978)
53. JAMIESON, S. H. The economic implementation of experimental retrieval techniques on a very large scale using an intelligent terminal. In: *Proceedings of the Second International Conference on Information Storage and Retrieval*, pp. 45-51, ACM-SIGIR, Association for Computing Machinery, New York (1979)
54. HESSE, M. B. *Models and Analogies in Science*, University of Notre Dame Press, Notre Dame, Ind. (1966)
55. WEIZENBAUM, J. *Computer Power and Human Reason*, Freeman, San Francisco (1976)
56. LINDSAY, R. K. In defense of ad hoc systems. In: *Computer Models of Thought and Language* (Ed. R. C. Schank and K. M. Colby), Freeman, San Francisco (1973)
57. TRAVIS, I. L. Design equations for citation retrieval systems: their role in research and analysis, *Information Processing & Management* **13**, 49-56 (1977)
58. OLNEY, J. C. *Building a Concept Network for Retrieving Information from Large Libraries. Part I*, SDC Report TM-634/001/00, System Development Corporation, Santa Monica, Calif. (1962)
59. INGWERSEN, P. and KAAE, S. User-librarian negotiations and information search procedures in public libraries: analysis of verbal protocols. In: *IRFIS 3*, Proceedings of the Third International Research Forum in Information Science (Ed. T. Henriksen), pp. 71-106, Statens Bibliotekskole, Oslo (1979)
60. BIVINS, K. T. Users of information retrieval systems: a frame theoretical approach. In: *Human Aspects of Information Science*, Proceedings of the Third International Research Forum in Information Science (Ed. T. Henriksen), pp. 135-145, Statens Bibliotekskole, Oslo (1979)
61. ODDY, R. N. Information retrieval through man-machine dialogue, *Journal of Documentation* **33**, 1-14 (1977)
62. ODDY, R. N. Retrieving references by dialogue rather than by query formulation, *Journal of Informatics* **1**, 37-53 (1977)
63. HOLLNAGEL, E. The relation between intention, meaning and action. In: *The Analysis of Meaning: Informatics 5* (Ed. M. MacCafferty and K. Gray), pp. 135-147, Aslib, London (1979)
64. SCHANK, R. C. What makes something 'ad hoc'. In: *Proceedings of TINLAP-2, Theoretical Issues in Natural Language Processing-2*, pp. 8-13, Association for Computing Machinery, New York (1978)
65. HAYES, P. J. Computer programming as a cognitive paradigm, *Nature, London* **254**, 563-566 (1975)
66. SUSSMAN, G. J. The virtuous nature of bugs. In: *Proceedings of the AISB Summer Conference*, University of Sussex, July 1974, pp. 224-237, Society for The Study of Artificial Intelligence and Simulation of Behaviour (1974)
67. ROSENBERG, V. The scientific premises of information science, *Journal of the American Society for Information Science* **25**, 263-269 (1974)
68. SLOMAN, A. *The Computer Revolution in Philosophy: Philosophy, Science and Models of Mind*, Harvester Press, Hassocks, Sussex (1978)
69. ANDERSON, T. and RANDELL, B. (Eds.) *Computing Systems Reliability: An Advanced Course*, Cambridge University Press, Cambridge (1979)
70. GERHART, S. L. Program validation. In: *Computing Systems Reliability: An Advanced Course* (Ed. T. Anderson and B. Randell), pp. 66-106, Cambridge University Press, Cambridge (1979)