A free-text retrieval system using hash codes

C. E. Goble

INSPEC, The Institution of Electrical Engineers, Savoy Place, London, WC2R 0BL

An algorithm is presented for efficient serial searching of files whose records have arbitrary length free-text retrieval keys. It is most applicable when a batch of enquiries is to search a given file once only, which is an implicit feature of the SDI application for which it was designed. Unlike some other serial systems, an arbitrary number of enquiries can be handled with a single pass of the search file, and the algorithm is simple in concept, and straightforward to implement. Specimen performance figures are quoted in the appendix.

(Received August 1973)

1. Introduction

This paper describes some aspects of an information retrieval system designed primarily to support the operation of an SDI (Selective Dissemination of Information) service covering scientific and technical literature. The background of this topic is not discussed here, but a useful survey appears in (Barlow, 1972). However, the application can essentially be regarded as requiring the selective retrieval of records from an inherently sequential unindexed file. In principle, the only property of the records themselves that is significant to the retrieval process is that each contains a string of characters, and enquiries are stated in terms of the occurrence (or non-occurrence) of specific substrings. For example, in the application for which the system was originally written each record in the file to be searched describes an item of published material. The text of each record includes information such as the name of the author of the original material, the language, a codified version of the journal title, various codes classifying the subject area covered, and a collection of free-language phrases which describe the concepts treated.

To get a feeling for the way in which retrieval requests are presented to the system the reader should refer to **Fig. 1,** which illustrates a much simplified enquiry set up to retrieve documents dealing with cognition, recognition, perception processes or robotics, which do not concern themselves with character recognition or OCR. Moreover, only papers written in English or German are retrieved.

Field 00 provides the enquiry number, complete with check character. Fields 01, 02 and 03 define various search strings, each comprising several alternatives, known as *terms*, only one of which need be present in a document for that field to deliver a value of **true**. Terms delimited by an asterisk will match the corresponding text wherever it appears in a document, but without asterisks that term will match complete words only. Thus *COGNIT* will match RECOGNITION or COGNITIVE but OCR will not match SOCRATES, for example. Finally, field 91 is the master logic statement, and a document is retrieved exactly when this field is **true**. Further details about the INSPEC SDI system from a user point of view can be found in the SDI users manual (1970).

Field number	*Text*
00	0501-X
01	*COGNIT* PERCEPT* ROBOT*
02	CHARACTER* OCR
03	ENGLISH GERMAN
91	(01 AND NOT 02) AND 03

Fig. 1 Example of an enquiry

Conventionally, there have been two alternative software approaches to an implementation of such a system, namely serial and inverted. Briefly, an inverted system involves the creation and maintenance of an index, known as an inverted file, to the text in the master file, and record selection and retrieval is performed using this index. There have been many implementations of inverted file retrieval systems (McCracken and Veal, 1973) mostly designed to support, amongst other things, retrospective or interactive operations, and some of these have used hash-coding in one form or another (Harrison, 1971; Higgins and Smith, 1971).

By contrast, however, the system being described was designed for an SDI application in which a relatively small document file (2000-3000 records), containing only the current week's material, was to be searched once only with a relatively large batch of enquiries (150-400). In this environment it was felt that the usual extensive software development and hardware commitment characteristic of the inverted approach could profitably be avoided, while still yielding a computationally efficient file searching module. This paper is therefore concerned with a serial search algorithm in which the master file is passed once only, and records are processed and searched entirely in core. To facilitate the matching of enquiry terms against each document the text content of the individual document is essentially 'inverted', and a hash-coding technique is used to increase the efficiency of the matching process.

Traditionally, one of the limitations of serial search systems has been the number of enquiries that can be processed on a single pass of the document file. This has been overcome by blocking the enquiries in buckets on disc, and swapping them in and out of core for each document on the file. The present paper concentrates on the search phase of the system only, and ignores the questions of enquiry input and maintenance, and the sorting and printing of the SDI notification cards themselves.

2. Hash-codes

Fundamental to the search system is the concept of a hash-code. These are widely used in computing, but will be described within the present context. A hashing algorithm is a function (implemented as a subroutine) which takes a string of characters as input and produces an integer as output. The range of integers which may be produced depends on the specific function chosen, but the particular algorithm used maps four-character strings onto the range 0-2047. Furthermore, it will be helpful to consider strings of more than four characters as producing the hash-code generated by their first four characters. The following properties of hash-codes should now be clear:

1. Two identical strings generate the same hash-code.
2. Unequal strings may or may not generate the same hash-code.

Additionally, a property of a good hashing algorithm is that it spreads generated hash-codes evenly across the permitted range, thus minimising the chance of two unequal strings generating the same hash-code.

3. Inverting a string of text

The hashing algorithm may now be used in the following way. Suppose we have a string of text. We may take every group of four consecutive characters in that string and compute a hash-code from them. If, for a given group of four characters, the hash-code is *k*, then we can set the *k*th word of a table to point to the first character in the group. In practice, however, property (2) above complicates matters slightly and forces us to make the *k*th entry in the table point to a chained list of pointers to the first character of every sub-string (of four characters) which generates the hash-code *k*. This data structure is known as an inverted string of text. (See **Fig. 2**).

4. Pattern matching

Suppose we have a string of characters A (which represents a term in an enquiry) and we wish to know if it appears as a substring of a string of text B. The approach is as follows:

1. Invert the string of text B, as described earlier.
2. Compute the hash-code of the first four characters of A—*p*, say.
3. Look at the *p*th entry in the hashtable for B. If it is zero then property (1) guarantees that A is not a sub-string of B. Otherwise compare A character by character with those sub-strings of B given by the *p*th entry in the hashtable. If one such sub-string matches A, then A is a sub-string of B—otherwise it is not.

5. Application to file searching

The retrieval system uses the preceding techniques in the following way. A document is read from tape into core store, and inverted. Each term of an enquiry is looked up rapidly, one after the other, via the hashtable, and the boolean expression linking them is evaluated in a conventional manner. The next enquiry is then processed against the same inverted document. When all the enquiries have been compared with the document, the whole process is repeated for the next document, and so on.

6. Features of the system

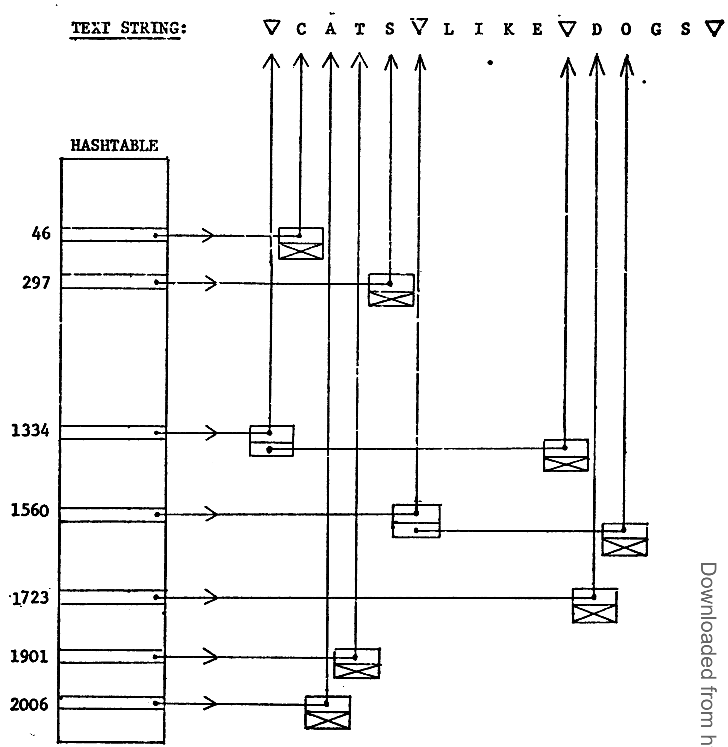
(*a*) Each document is inverted once only during an SDI run. Once this is done, enquiry terms are looked up very rapidly, since most of the time the entries looked up in the hashtable will be zero, indicating that a term is not present, and avoiding an expensive string comparison. If the hashing algorithm is good, we would hope that when an entry is non-zero and a string comparison initiated we would stand a good chance of finding a match of the term with the document.

(*b*) The hash-code for each term of an enquiry can be computed once and for all at the beginning of the SDI run and thereafter stored with the text of the term.

(*c*) From (*a*) and (*b*) above we see that finding out if a term is present in a document is nearly always a matter of one indexed access to a linear table held in core store, and is thus very efficient. Moreover, if the entry is non-zero the associated list points to those sub-strings in the document, which is also held in core, for which there is a possibility of a match.

(*d*) As a result of the sequential processing of enquiries, it is possible to process one bucket at the same time as the next bucket is being read from the disc, and the present system is balanced so that swapping and processing take similar amounts of time. Thus no time penalty is paid for storing the enquiries on disc, and core store requirements are modest.

(*e*) The simplicity of the technique enabled the search part of the system to be developed, using ICL assembler (PLAN), in less than two man-months. Specimen performance figures appear in **Appendix 1.**



This example assumes that the hashing function (H) performs as follows:

H: ∇CAT →1334	H: LIKE	Not shown for clarity
H: CATS → 46	IKE∇	
H: ATS∇ →2006	KE∇D	
H: TS∇L →1901	E∇DO	
H: S∇LI → 297	H: ∇DOG→1334	
H: ∇LIK →1560	H: DOGS→1723	
	H: OGS∇ →1560	

Fig. 2 An inverted string of text

Acknowledgement

I should like to thank my colleagues for their help throughout the project, and during the drafting of this paper. The work, as part of a continuing research programme, was funded by the Office for Scientific and Technical Information (O.S.T.I.).

Appendix 1 Performance figures

Hardware used:	ICL 1902A
Storage:	13*k* × 24-bit words of core 1 × EDS 8 (i.e. 8 megabytes of disc)
Average number of enquiries:	250
Average number of terms per enquiry:	46
Average length of terms:	6 characters
Average number of documents searched:	2,200
Average length of documents:	224 characters
Rate of processing:	1200 documents/hour

References

Barlow, D. H. (1972). Information Retrieval. *The Computer Bulletin*, Vol. 16, No. 5, pp. 250-255.

Harrison, M. C. (1971). Implementation of the substring test by hashing, *CACM*, Vol. 14, pp. 777-779.

Higgins, L. D., Smith, F. J. (1971). Disc access algorithms, *The Computer Journal*, Vol. 14, No. 3, pp. 249-253.

McCracken, I. C., Veal, M. A. (1973). INFIRS—A Generalized Information Retrieval System, *4th International Conference on Mechanized Storage and Retrieval Systems*, Cranfield, July 1973.

SDI user manual. *INSPEC publication.*

Book reviews

Computational Methods for Large Molecules and Localized States in Solids, Edited by F. Herman, A. D. McLean and R. K. Nesbet, 1973; 396 pages. (*Plenum*, 1973.)

This volume contains the proceedings of a symposium held on 15-17 May, 1972, at the IBM Research Laboratory, San Jose, California. To judge from the preface, the (unlisted) participants were nominally international, but in practice predominantly American. Certainly the active contributors were almost all, if not themselves American, then currently working in the USA at the time of the symposium. European work is thus rather under-represented; nevertheless one should recognise that most of the work in this field is indeed done where most of the big machines are—i.e. in the USA! Consequently the contributors to this volume do after all comprise a powerful assembly of the most active and creative workers in the field.

The leading motive of the symposium was the recent convergence of concepts and techniques between quantum chemistry and solid-state physics. Each of these deals with electron energies and wave-functions, satisfying the Schrödinger equation in the presence of an array of atomic nuclei. They have grown apart, over the last 40 years, because solid-state theory copes with infinite lattices by means which rest on their periodicity, while quantum chemistry handles a general atomic geometry at the expense of a restriction to structures of rather small size. Recently, however, solid-state theory has begun to grapple with non-periodic structures (amorphous solids, and localised defects) for which some 'chemical' techniques have been required, while also the simplifying devices developed in solid-state theory, to represent atomic potentials in a more compact way (e.g. the various pseudopotentials) have been tried out in quantum chemistry. The convergence of the two fields is thus a real and fruitful thing. This symposium was designed to help it forward.

Each day of the conference was devoted to a different topic: Scientific challenges, Computational methods, and Localized states and disordered solids. The first day (Scientific challenges) was entirely chemistry, and indeed something of a rag-bag; the general theme was to show the wide variety of problems currently under attack, picking especially those which were still at an early stage of development, computationally at least, if not also conceptually. As a result J. A. Pople's clear survey of computational success on small molecules rather stood out, and would perhaps have fitted better into the second day. So, in another way, did W. A. Little's contribution on 'Molecular Modelling': he was the only speaker dealing with minicomputers, which were all he required (plus a graphic display) for his interactive geometrical modelling programs.

For the would-be computer, day 2 provided the richest meat. Quantum chemistry (and its traditional methods, pushed to their limits) held sway in the morning, while the afternoon was begun with two techniques, originated by solid-state workers, but with molecular problems in mind. (K. H. Johnson: SCF-Xα scattered wave method; W. A. Harrison: orbital correction method.) These were followed by a panel discussion on computational methods. This runs to twenty pages of the book, and makes many useful points. (It goes far to make up for the lack of recorded discussion on any of the individual contributions.) Two extreme examples came up in the discussion, both emanating from the host laboratory. On the one hand was a 500-configuration calculation of the carbon atom, accurate to 3 milli-Hartrees of correlation energy. On the other was a 28-atom ab-initio minimal-basis calculation (on 2,4,7-trinitro-9-fluorenone) which involved 120 basis functions and 11 million integrals. After the discussion of this work K. H. Ruedenberg remarked 'The lesson to be learned is: If even IBM cannot afford hundreds of calculations of this size in one year, then one should think very carefully on which big molecules such a tremendous effort should be expended . . .' The participants could not finally answer this challenge, but they had much of interest to say about it.

Solid-state problems were represented on day 3. The clearest paper here was the excellent review of amorphous semiconductors by D. Weaire and M. F. Thorpe, but though rich in concepts this was poor in computational methods. It raises however (to the reviewer) the prospect of further computer experiments on the structure (especially the ring structure) of the random networks they were discussing, or indeed of any other models of amorphous solids. The paper by J. Keller on cluster scattering was by contrast rich in computable ideas (though one would need to go back to Keller's references to define them fully).

The book closes with a banquet address by G. S. Hammond, (centred on the role of 'cults' and fashions generally, in science) and lastly a useful summary of the conference by R. K. Nesbet.

I. D. C. Gurney (Newcastle upon Tyne)

Management Systems, by Peter P. Schoderbeck, 1971; 561 pages. (*John Wiley & Sons Ltd*, £5·65.)

This book belongs to the Wiley Series in 'Management and Administration' and as such is aimed at the manager interested in computing rather than the computer specialist. I am therefore not completely convinced that the *Computer Journal* is the correct place for this book to be reviewed or that I am completely competent to judge this book in its entirety. However with those reservations in mind let us proceed . . . .

Management Systems attempts to encompass a wide range of material from the highly theoretical 'general systems concept' to the practicalities of PERT and the SABRE system. The book, a compendium of papers, is divided into three major parts: The system concept, Management information systems and Systems applications together with a short section entitled Prologue to the future. The three major parts are divided into sections each of which is composed of an editorial, a collection of papers and a bibliography.

Overall I found the book interesting although the first part (The system concept) was an uphill struggle possibly because of lack of the necessary theoretical background. I had hoped that Management information systems would provide some interesting practical papers but my hopes were dashed with the exception of one paper by Konvalinka and Trentin. When I finally reached the third part of the book (Systems applications) this proved to be the most rewarding. The five papers in the PERT section presented a balanced view of the subject and set a standard which the rest of the book should have achieved.

Each section contained a selection of papers which were chosen so that the subject was viewed from different angles by a variety of authors. For example the section on Industrial Dynamics contained three papers; the first two by J. Forrester and E. D. Roberts (both of MIT) extolling the virtues of ID and the third by Ansoff and Stevin trying to assess the subject impartially and in consequence challenging some of Forrester's assertions. Some of the papers selected seemed to me to be rather too dated to be relevant today although there are always exceptions: Stafford Beer's paper on 'What has Cybernetics to do with Operations Research' is still interesting although first published in 1959. However a section on Information technology and its impact on the organisation consisted of six papers, three of which were very neatly summarised in a fourth within this same section! Design, implementation and control of MIS—a promising title for a section I originally thought—contained no paper published later than 1964 and seemed particularly irrelevant

I cannot recommend this book for the shelf of the computer specialist or as a textbook for computer science. I enjoyed reading it but doubt whether it will prove useful except perhaps for the extensive bibliographies which provide pointers to other background readings in the management and computing area.

R. C. Welland (Reading)