CLUSTERED FILE GENERATION AND ITS APPLICATION TO COMPUTER SCIENCE TAXONOMIES

TR 76-295

D. Bergmark and G. Salton

Department of Computer Science

Cornell University

Ithaca, N.Y. 14853

Clustered File Generation and its Application to Computer Science Taxonomies

Gerard Salton and Donna Bergmark *

ABSTRACT

A clustered file organization is one where related, or similar records are grouped into classes, or clusters of items in such a way that all items within a cluster are jointly retrievable. Such a file organization is advantageous for interactive searching where tentative query formulations may be used and the records may be specified incompletely or approximately.

An inexpensive file clustering method applicable to large files is given together with an appropriate file search method. The method is used to cluster a file of research articles in computer science based on citation similarities between the papers; this leads to the identification of groups of active computer science research topics and of productive computer scientists.

*Department of Computer Science, Cornell University, Ithaca, NY 14853.
This study was supported in part by the National Science Foundation under grant GJ 43505.

1. RETRIEVAL ENVIRONMENT

Most modern information retrieval systems currently in use operate interactively in real-time under user control. Typically, a search request is introduced using a console entry device, and a reformulation of the query is attempted on the spot whenever the search results appear unsatisfactory. Since the search output must be produced rapidly while the customer waits at the console a sequential search of the stored records cannot be used. Instead, an inverted file system is normally utilized, consisting of two files: a file directory listing for each allowable search term the set of all records (or record numbers) pertaining to that term, and a main data file which stores the records themselves arranged in order by record numbers (rather than by subject).

In an inverted file system, a search is conducted in several steps [1,2]:

a) for each term included in the search request the corresponding list of record numbers is obtained from the directory;

b) these record lists are combined, or merged to produce all record numbers that satisfy the query specification;

c) the actual data records corresponding to the previously determined record numbers are retrieved from the main file.

An inverted file system leads to rapid retrieval because the list processing operations involving the directory (steps (a) and (b) above) are performed in fast storage, whereas retrieval from the main data file is restricted to those records only that actually correspond to the query statement.

Unfortunately, inverted file systems exhibit substantial disadvantages:

a) the search vocabulary (index terms, keywords, etc.) used to formulate the search requests is closed and prespecified because a term for which no advance provision is made in the file directory produces no search output;

b) query statements must be completely and precisely formulated using the same vocabulary also used for record identification;

c) retrieval is an "all or nothing" operation in that items that completely match the search specification are retrieved, while everything else is rejected;

d) the retrieved items are not ranked in decreasing relevance order, and the user has little control over the size of the output set.

In actual fact the conditions under which retrieval activities are ideally carried out are quite different: much of the time, the user does not know how to formulate a precise, unambiguous query, but can instead suggest tentative, incomplete statements; in the same way, the stored records might not be completely specifiable because the actual content of an item may be in doubt, or because ambiguous record attributes may in fact be appropriate in some cases. Finally, a complete unambiguous content specification using closed and controlled indexing vocabularies is generally difficult to generate for normal retrieval system users.

What is wanted instead is a system where approximate searches can be conducted between partially formulated queries and possibly incompletely specified information items to identify potentially relevant items. These potentially important items can then be scrutinized further before actual retrieval takes place. Such a retrieval environment appears to lead directly to the use of clustered file organizations.

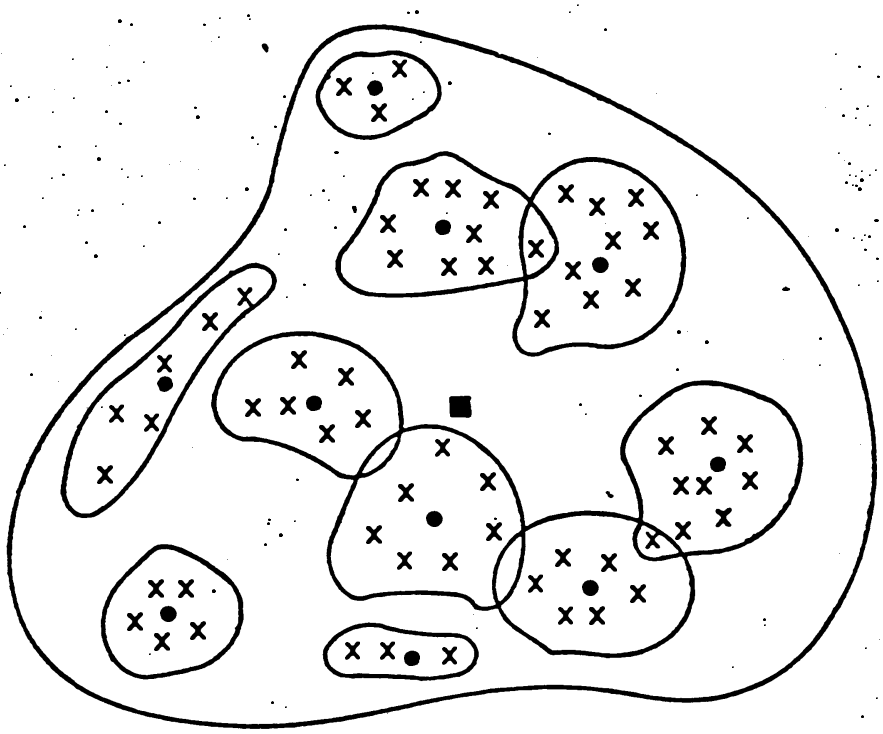
2. CLUSTERED FILE ORGANIZATION

A clustered file organization is one where related, or similar records are grouped into classes, or clusters of items in such a way that all items within a cluster are jointly accessible without excessive delay. A

typical example is given in Fig. 1, where each X represents a record and the circular configurations are the clusters. In the representation of Fig. 1, each cluster of records is identified by a dummy, central item, known as the centroid, and the distance in the picture between a centroid and an individual record (or between pairs of records is inversely related to the similarity between the corresponding items. To construct an organization of the type shown in Fig. 1, the computation of a similarity coefficient s(X,Y) between items X and Y is then assumed possible. The size of the coefficient s is normally made to depend on the similarities between the index terms assigned to the corresponding information items.

It may be seen in Fig. 1 that overlap occurs between the clusters in the sense that certain items may appear in more than one cluster; furthermore, the size of the various groups may vary somewhat from one cluster to the next. [3,4]

In a clustered file organization, a search is carried out by first comparing a query formulation with the cluster centroids; this is followed by a matching of the query with only those records whose corresponding query-centroid similarity was found to be sufficiently large in the earlier comparison. It is obvious from this description, that clustered file searches can be conducted rapidly, since only small portions of the files are actually processed. Furthermore, records that are similar to each other are easily identified because a complete cluster of records is normally stored in adjacent storage locations (for example, on the same track of a disc file), and hence becomes available simultaneously. Since the retrieval process depends on a global match between the complete query formulation and the content identifiers attached to the records, the



x stored record

● cluster centroid

Typical Clustered File Organization

Fig. 1

operations will carry through also for partially, or incompletely specified items.

The major problem relating to clustered files arises not in connection with the search operation but rather with the file generation process. Most classical clustering methods are based on the availability of a complete similarity matrix giving a similarity measure between all pairs of records. Since there are n(n-1)/2 pairs of records, the computation of this similarity matrix is already of order n^2 for files of n records, and the clustering operation becomes too expensive for practical use.

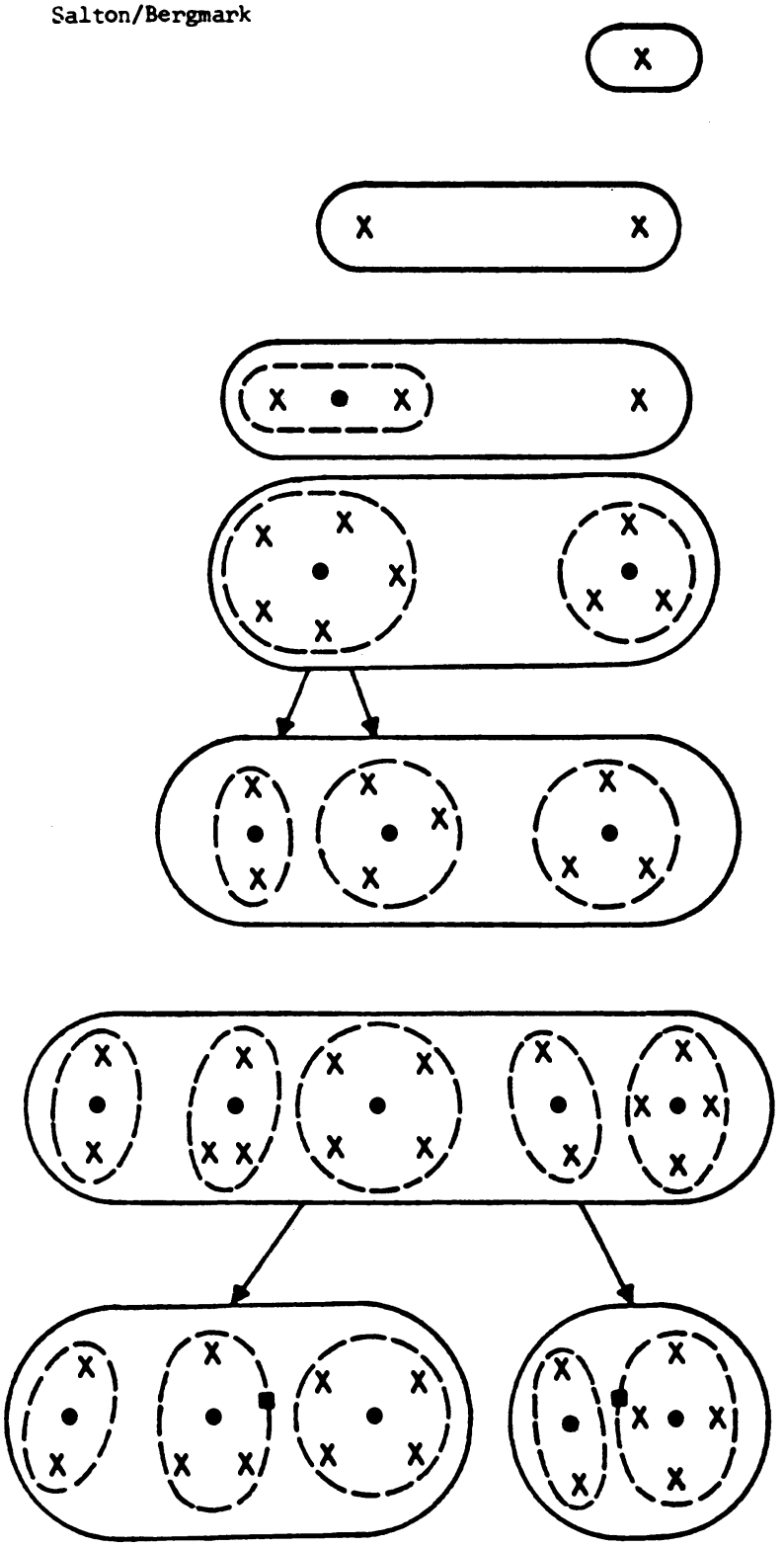
Fortunately, there exist single-pass clustering schemes in which each item is processed only once and the cost of generation is of order n log n for n items. [4,5,6] Typically the construction process is bottom-up. The first item is initially identified with cluster one. The next item is compared with cluster one and merged with it if found to be sufficiently similar. If the new item is not similar to any already existing cluster, a new cluster is generated. Obviously, the cluster centroid is redefined whenever a new item is entered into a cluster.

The process must include controls over cluster size, cluster overlap, and number of clusters to be generated. When a cluster becomes too big, that is, when it contains too many individual records, its centroid may be split, and two new clusters may emerge from a single original one. The same applies to the cluster centroids themselves which are assumed to be contained in a larger supercluster. Whenever the number of centroids contained in the supercluster becomes too large, a second supercentroid may be created by splitting the original one. This transforms the single original supercluster into two new ones.[6] The upward splitting process

just described is identical in concept to the updating procedure normally used for B-tree processing. [7]

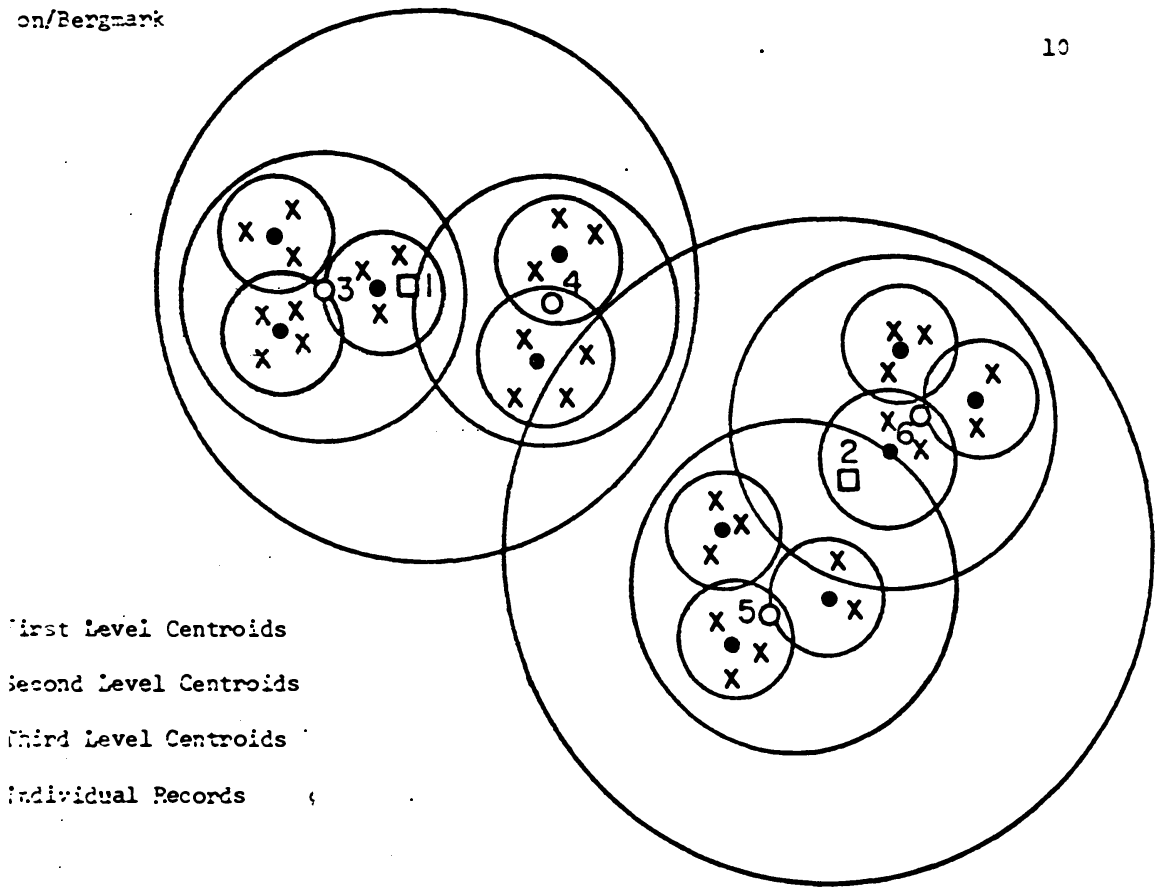
A typical cluster generation process is illustrated in Fig. 2. The assumption is that whenever five items (records, clusters, superclusters, etc.) occur in a group, two new groups must be created using the splitting process. In the example of Fig. 2, two clusters are first created from the incoming items. One cluster eventually contains three records and the other five. The latter is split to form clusters of two and three items, respectively. When the single original supercluster reaches size 5, two superclusters are created as shown at the bottom of Fig. 2.

The cluster search process is illustrated in Fig. 3. At the top of the Figure, a typical multilevel cluster tree is shown consisting of individual records, regular (third-level) clusters, super (second-level) clusters, and finally super-super (first-level) clusters. The corresponding search tree is presented at the bottom of Fig. 3. An incoming query is first compared with the first-level centroids. Each first-level centroid leads to a number of second-level centroids, and these in turn lead to the third level centroids and eventually to the records themselves. It is clear from the picture that narrow (depth-first) searches may be conducted by considering at each level only the highest matching item. Alternatively, broader searches may be performed by developing (that is, by matching against the query) several highly correlated nodes. It is also easy to show that the number of comparison operations needed for cluster searching, cluster generation, or cluster updating is of order n log n, where n is the number of records on the lowest level. [6]

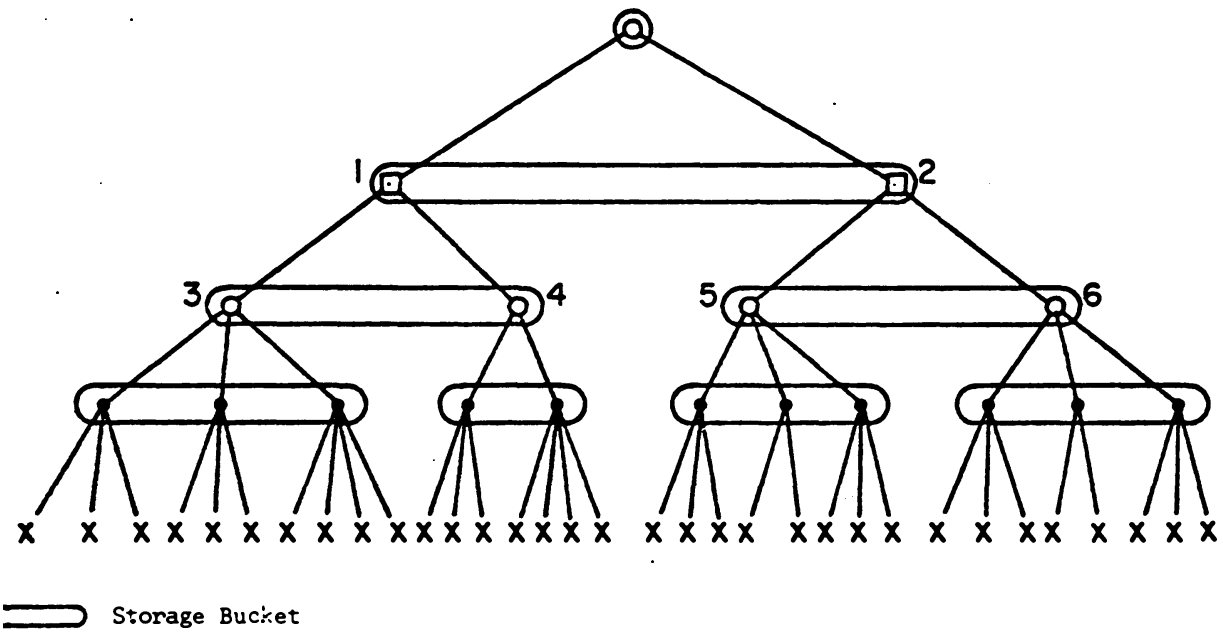


Single-Pass Bottom-Up Cluster Generation

Fig. 2



a) Clustered File Organization



b) Search Strategy for Clustered File

Fig. 3

Clustered file organizations are designed for library (reference) retrieval purposes. In addition they may also be useful for business files. In the latter case, it is possible to implement both the usual exact search capability as well as the partial cluster search by adding an inverted directory constructed from the cluster centroid terms. [3,4]

An illustration of file clustering applied to computer science research papers is contained in the next section.

3. CITATION CLUSTERS IN COMPUTER SCIENCE

A clustered file organization may be utilized for document collections, in which case the clustering concept is used to group the items according to affinities in subject matter. Under normal circumstances, the subject headings or other content identifiers that are attached to the various documents might constitute the principal clustering criterion; that is, items exhibiting similar subject descriptions would be grouped into clusters.

In the present study, the content identifiers or index terms are replaced by references and citations. A reference is a bibliographic item appearing in the bibliography of a document (A refers to B); a citation, on the other hand, is a reference made by an outside item to a given document (B cites A). The clustering criterion to be used operates in such a way that the similarity coefficient between two items will depend on the number of references shared by their bibliographies, and on the number of common citations from the outside.

Consider, as an example, a given document $D=(c_1,c_2,..,c_n,r_1,r_2,..,r_m)$ where c_i represents the ith citation, and r_j the jth reference. When two documents share many citations, the c-terms in the document vectors exhibit

many common elements; hence the corresponding similarity coefficient, s, between these documents will be large. The same is true for documents sharing many common references (many common r-terms).

As an example of the clustering process, a multi-level clustered file was generated for a collection of research articles in computer science. The basic clustering criteria and collection coverage are described in Fig. 4. The base collection consisted of 334 articles appearing either in various ACM publications during 1974, or in Computer and Control Abstracts (the Inspec data base) in 1968 and 1971. The complete set of references -- an average of 11 for each of the base papers -- was then added to the collection to produce a total of 3,520 distinct articles.

The bottom-up clustering process illustrated in Fig. 2 could then be used to generate citation clusters for these documents. One supplementary step not previously mentioned consisted in absorbing into the citing item (or citing cluster) all singly cited items without references of their own. This absorption operation was performed by merging the respective citation vectors, and could be utilized at each level of the cluster tree to delete items whose position in the citation network was predetermined. An example of the absorption operation is shown in part 3 of Fig. 4.

Sixty lowest level clusters (corresponding to level three in the cluster tree of Fig. 3) were generated for the sample collection, the cluster size varying between 10 and 60. Each of these low level clusters exhibited strong citation links and could be associated with an active, homogeneous research area.

Two typical clusters are shown in Fig. 5, identified as "Knapsack

1. Clustering Criterion

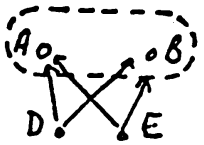
Similarities between articles are based on outgoing references and incoming citations

A refers to B,C

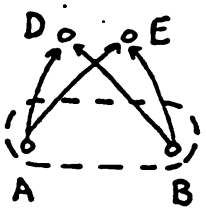
B,C cite A

2. Clusters result from citation and reference similarities between items

cocitations



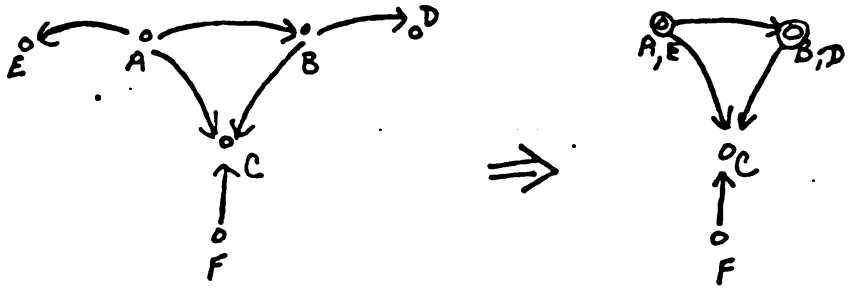
common references



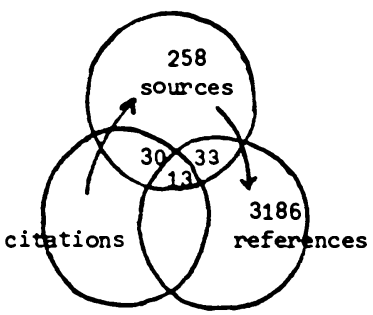
$A=(c_1,..,D,E,..,c_n,r_1,r_2,..,r_m)$ $A=(c_1,c_2,..,c_n,r_1,..,D,E,..,r_m)$

$B=(c_1',..,D,E,..,c_n',r_1',r_2',..,r_m')$ $B=(c_1,c_2,..,c_n,r_1,..,D,E,..,r_m)$

3. Items without references that are cited by one other item only are absorbed by the citing items



4. Initial Collection



334 source items and all their references
3232 distinct references
3893 distinct links
3520 distinct items

1974 ACM Communications
1974 ACM Journal
1974 ACM Computing Surveys
1968 and 1971 Inspec Computer Abstracts

Collection Clustering Criteria

Fig. 4

3 4699 FAST APPROXIMATION ALGORITHMS FOR THE KNAPSACK AND SUM OF SUBSET PROBLEMS
4 4866 A POLYNOMIAL-TIME ALGORITHM FOR THE KNAPSACK PROBLEM WITH TWO VARIABLES
(R) 4701 COMPUTING PARTITIONS WITH APPLICATIONS TO THE KNAPSACK PROBLEM
(R) 4706 APPROXIMATE ALGORITHMS FOR THE 0/1 KNAPSACK PROBLEM

CLUSTER 6009 NO. OF ITEMS 11 TOPIC: KNAPSACK PROBLEM

2573 3 SAHNI, SARTAJ K

CLUSTER 6024 NO. OF REFERENCES: 0 CITED BY: 1141 1199 1354 6009

7 1001 AN EFFICIENT ALGORITHM FOR FINDING AN IRREDUNDANT SET COVER#
12 1014 AN APPROACH FOR SIMPLIFYING SWITCHING FUNCTIONS BY UTILIZING THE COVER
2 3799 ON SIMPLIFYING TRUTH FUNCTIONS: A PRELIMINARY REDUCTION OF CORELESS
8 4810 THE FANOUT STRUCTURE OF SWITCHING FUNCTIONS
(R) 1011 INTEGER PROGRAMMING: METHODS, USES, COMPUTATION#
(R) 1323 THE PROBLEM OF SIMPLIFYING TRUTH FUNCTIONS#
(R) 1324 A WAY TO SIMPLIFY TRUTH FUNCTIONS#
(R) 3854 INTRODUCTION TO THE THEORY OF SWITCHING CIRCUITS#
3851 MINIMIZATION OF BOOLEAN FUNCTIONS#
3855 PRECONJUNCTIVE AND DISJUNCTIVE AND CONJUNCTIVE FORMS OF A BOOLEAN FUNCTION#

CLUSTER 6024 NO. OF ITEMS 35 TOPIC: SWITCHING FUNCTIONS

56 2 QUINE, W B
1764 2 MOTT, T H
115 4 DAS, S R
1767 3 CHOUDHURY, A K
1761 3 MCCLUSKEY, E J
1697 2 PINTER, CHARLES
2631 3 HAYES, J P

ID NUMBER — NUMBER OF PAPERS

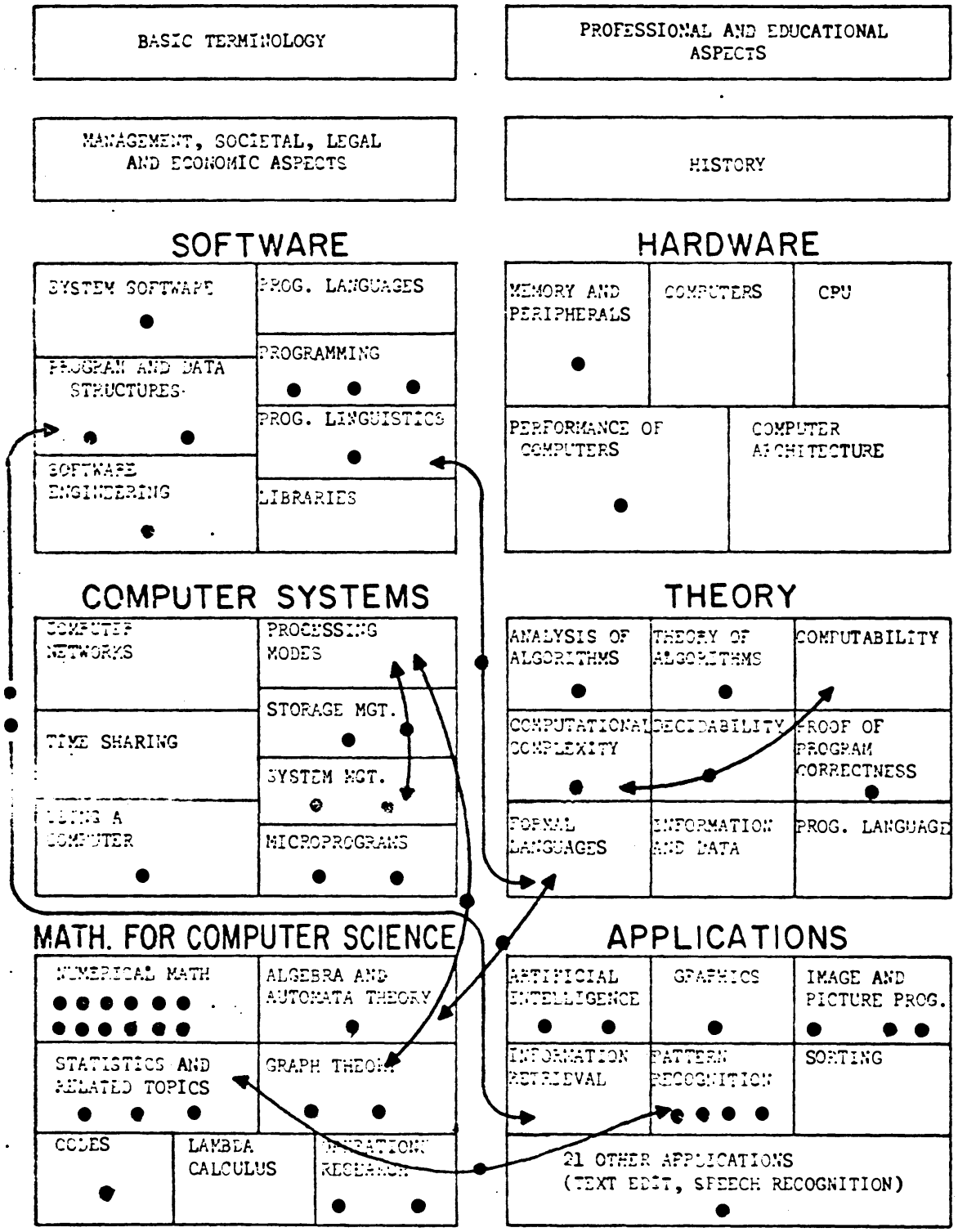
Clusters 6009 and 6024 Knapsack Problem - Switching Functions

Fig. 5

Problem" and "Switching Functions" respectively. Each line in the Figure represents either a single document, or a group of documents consisting of an item together with various documents cited by that item. Also shown in Fig. 5 are the author clusters produced by the respective document clusters. An author appears in the Figure only when associated with at least two cluster items.

A clustered collection such as the one generated in the present study can be used for rapid search and retrieval, as previously explained. In practice, a network consisting of ten to fifteen thousand research articles in a given subject field may offer sufficiently extensive coverage to become interesting for operational retrieval purposes. In the current study, retrieval is not of primary interest. Instead, an attempt is made to identify clusters of research topics in computer science -- either documents or authors -- as reflected by the publication activities between 1969 and 1974. This latter aim can be attained only imperfectly because the available ACM and Inspec articles do not by themselves cover all of computer science. The coverage of the 3,500 available papers may however be sufficiently extensive to provide an indication of what may be obtainable by a clustering process for a more comprehensive document collection.

In Fig. 6 an attempt is made to determine which computer science topic areas were sufficiently active to be represented by homogeneous research clusters in the early nineteen-seventies. This is done by matching the sixty low-level citation clusters with the computer science taxonomy used to classify articles in the recently published Encyclopedia of Computer Science [8]. This taxonomy consists of ten principal areas including terminology, professional aspects, history, software, hardware, computer



First Level Clusters Using Encyclopedia Taxonomy

Fig. 6

systems, theory, mathematics for computer science, and applications. In Fig. 6, each subdivision within a box represents a subtopic of the taxonomy, and each dot stands for a research cluster.

It is not surprising that the four relatively nontechnical encyclopedia areas shown at the top of Fig. 6 turned out to be empty of clusters. The best correspondence between encyclopedia areas and research clusters is obtained for the mathematics area. Surprisingly few homogeneous research topics were found in the software sections, and even fewer for hardware. Moreover, as might be expected, some existing research clusters were difficult to fit precisely into any of the encyclopedia areas. This was the case notably for clusters in formal language theory, complexity analysis, pattern recognition and information retrieval. For these cases arrows straddling several boxes are used in Fig. 6.

It appears from the output of Fig. 6 that a substantial part of the encyclopedia taxonomy matches the research clusters reasonably well. Furthermore the correspondence might be more pronounced if a larger, more diversified document collection had been used for clustering purposes. Some of the classification difficulties may, however, be expected to persist across all document collections, including the artificial separation between various applications areas and the algorithms used to implement the applications, and between the theory of computation and the system and software areas reflecting the theoretical advances.

As an additional application of the clustering work, it may be of interest briefly to consider the author clusters derived from the clusters of research papers. In particular, one may ask whether inclusion of a name within the structure of author clusters correlates with other marks of

distinction within the field. For this purpose it is convenient to study the citation properties of various recognizable subgroups of computer scientists. Various citation parameters are listed in Fig. 7 for several computer science groups or committees in the United States, including sixty chairmen of Ph.D. granting computer science departments; the twenty-four members of the Computer Science Board, two-thirds of whom are also department chairmen; the ten members of the Advisory Panel of the Division of Computer Research of the National Science Foundation; a thirteen member Computer Science Planning Group organized by the National Academy of Sciences; and finally a group of panelists conducting the so-called Computer Science and Engineering Research Study (Cosers) under the sponsorship of the National Science Foundation. For control purposes, a random sample of sixty members of the Association for Computing Machinery (ACM) is added at the bottom of Fig. 7.

It is clear from the Figure, that three recognizable groupings are included: the group of randomly chosen ACM members with citation properties close to zero; the department chairmen and computer science board members, over fifty percent of whom have not authored any documents among the 3,500 in the collection, and most of whom are not included in the research clusters; and finally the remaining panel and committee members, over sixty percent of whom are collection authors, with a substantial twenty percent also included in the research clusters.

The two right-most columns in Fig. 7 give the proportion of people in the various groups with either zero citations, or with over thirty citations, in the 1975 issue of the Science Citation Index (SCI). The data from the SCI are not completely comparable with the remaining figures in

Group	Number of Persons	Papers per Person in Collection	Percent Authors in Collection	Percent Authors in Clusters	Science Citation Index 1975 — Percent Persons with Count 0	over 30
Chairmen Ph.D. granting CS departments	60	1.50	47%	5% (3)	22%	15%
Computer Science Board	24	1.29	46%	8% (2)	32%	14%
NSF-DCR Advisory Panel	10	4.90	60%	20% (2)	10%	0%
NAS Computer Science Planning Group	13	2.38	62%	8% (1)	20%	40%
Cosers Steering Committee	14	4.43	64%	21% (3)	8%	23%
Cosers Panels	82	3.68	68%	26% (21)	13%	18%
Random Sample of 60 ACM Members	60	0.12	5%	0		

Citation Properties of Computer Science Groups

Fig. 7

the Table, because for papers of joint authorship the citations in the index are restricted to the first author only. Nevertheless, it is obvious that the group of chairmen includes a larger proportion of "zeros", and a smaller one among the "over 30" group, than the other panels and committees.

Various comments can be offered concerning the data of Fig. 7. To explain the citation and clustering properties for department chairmen, one could mention, for example, that a substantial number are people who are not primarily computer scientists who cannot therefore be expected to function as authors of computer science research papers. For present purposes, it is sufficient to note the obvious correlation between the clustering properties of research authors and their inclusion among recognizable subgroups in the computer scince community.

The construction and use of clustered files is becoming easier and better understood. One may hope that before long their usefulness for search and retrieval purposes may become clear not only in the laboratory, but also in practical file processing environments.

References

[1] I. Flores, Data Structure and Management, Prentice Hall Inc., Englewood Cliffs, NJ, 1970.

[2] D. Lefkovitz, File Structures for On-Line Systems, Spartan Books, New York, 1969.

[3] G. Salton, editor, The SMART System — Experiments in Automatic Document Processing, Prentice Hall Inc., Englewood Cliffs, NJ, 1971.

[4] G. Salton, Dynamic Information and Library Processing, Prentice Hall Inc., Englewood Cliffs, NJ, 1975.

[5] R.E. Williamson, Real Time Document Retrieval, Cornell University Doctoral Thesis, Ithaca, NY, 1974.

[6] J.A. Hartigan, Clustering Algorithms, John Wiley and Sons, New York, 1973.

[7] D.E. Knuth, The Art of Computer Programming, Vol. 3, Sorting and Searching, Addison Wesley Publishing Co., Reading Mass., 1973.

[8] A. Ralston and C.L. Meek, Encyclopedia of Computer Science, Petrocelli/Charter Publishing Co., New York, 1976.