A COMPUTER SCIENCE VIEW OBTAINED BY AUTOMATIC DOCUMENT PROCESSING

G. Salton and D. Bergmark

TR 77-309

Computer Science Dept.
Cornell University
Ithaca, N.U. 14853

A Computer Science View Obtained by Automatic
Document Processing

G. Salton and D. Bergmark*

Abstract

Automatic document classification techniques have been widely advocated for the study of various fields of learning, the identification of individual research topics and of influential contributors in a given field, and for information storage and retrieval purposes.

Several thousand research documents in computer science are automatically classified in the present study, leading to the generation of a taxonomy which reflects the state of computer science as of 1974. The popularity of various subject areas in the field is assessed, and the clustering characteristics of particular document classes and authors is given.

1. Clustered Files

A clustered file is a structure in which the stored records are grouped into classes, or <u>clusters</u> of records according to affinities that may exist between them. Typically each file item, or record may be characterized by a set of properties, or attributes reflecting the contents of the record. A similarity coefficient between pairs of records may then be computed as a function of the number of common attributes between them, or alternatively as a function of the weights

* Computer Science Department, Cornell University, Ithaca, New York, 14853.

of the common attributes. Whenever the similarity coefficient between items exceeds a given threshold the tendency is to assign them to a common cluster.

It is convenient to identify each cluster by a set of cluster attributes consisting of a combination of the attribute sets for the individual items in that class. The set, or vector, of cluster attributes is known as the class profile, or centroid vector. A rapid file search and retrieval process suggests itself based on a comparison of the information requests with some of the centroid and record vectors: initially, the requests are compared against the centroids only; this initial query-centroid comparison is then followed by a selective query-record comparison for the records only whose centroids are known to be similar to the queries. If too many clusters are defined in a given environment, it may be useful to group the corresponding cluster centroids into superclusters, identified by supercentroids. The latter, in turn, may be grouped into still larger aggregates until eventually a few large subsections of the file are left to constitute the complete file.

Such an arrangement produces a hierarchical search system in which the search requests are first compared against the identifiers attached to the largest subsections of the file; this comparison in turn gives access to smaller clusters included in the larger ones; by proceeding down into the cluster hierarchy one is able eventually to reach the individual records included in some of the lowest-level clusters. [1,2]

The main problem with the practical use of clustered files arises not in connection with file searching but rather with the initial cluster generation. The best known clustering methods are all based on the availability of complete similarity information specifying the similarity between pairs of file items. When the file is large the computation of the similarity matrix (of order n^2 for n items) becomes too expensive to be carried out in practice. In recent years, cheaper clustering methods have been proposed which are theoretically less appealing because the final cluster structure depends on the order in which the file items enter into the clustering process; these inexpensive procedures do however generate, and update the file in a time proportional to n log n for n items. [1,2] One may hope that clustered files may thus be rendered practical for many applications.

File clustering methodologies are used in many fields of learning. When the objects to be classified are written documents or books, file clustering techniques based on bibliographic citation data have been used to construct networks of paper [3], to evaluate the significance of individual research papers or topics [4,5,6], to predict the importance and influence of research workers [7], to assign content identifiers to bibliographic data [8,9,10], to evaluate the importance of certain journals in the literature [11], and to control file search and retrieval procedures.

In the remainder of this study a clustered collection of 3,520 documents in the computer science literature is examined and conclusions are drawn regarding the scope of the field, the importance of various subareas, and the characteristics of certain documents and authors:

2. Automatic Clustering of Computer Science Documents

A computer science collection was initially generated by taking 334 base articles published in the ACM literature (ACM Communications, ACM Journal, ACM Computing Surveys) during 1974, or listed in Computer and Control Abstracts during 1968 and 1971. All the references included in the bibliographies of the base articles were added to the original 334 items to form a complete collection of 3,520 distinct articles. Ideally a document collection might be clustered by comparing keywords or content identifiers assigned to the various items. In the present case, keywords were not available for most items; attribute vectors were therefore constructed by using bibliographic references and citations. For each paper, the references are defined as the items included in the bibliography attached to that paper; the citations, on the other hand, are references made by outside documents to each given paper. For each paper, an attribute vector can then be defined as

$$D = (r_1, r_2, \ldots, r_n, c_1, c_2, \ldots, c_m)$$

where r_i is an identifier representing the ith reference, and c_j the jth citation for document D.

Citation vectors such as the one previously given make it possible to group pairs of documents which share many references and/or many citations. Since a direct link between two documents (A refers to B, or B cites A) also provides information that can be used in the clustering process, a self-reference indicator is added to each document vector to about 270 documents for the P4 clusters.

The cluster tree obtained for the computer science collection is presented in Fig. 2. Each path in the tree from bottom (level 0) to top (level 4) represents the clustering characteristics for a certain set of documents. Thus the left-most path, designated as (P1 P2 P3 P4) represents 1125 items that are first entered into P1 clusters; these P1 clusters are then merged into P2 clusters, which themselves are transformed into P3 and eventually into P4 clusters. The 1125 items described by the left-most path in Fig. 2 appear to fit easily into the hierarchical clustering schema, since they could be successively entered into larger and larger cluster structures. On the other hand, the right-most path in Fig. 2 identifies 12 documents that could not be incorporated into any cluster after four clustering passes. These documents remain "loose" at the end of the process.

The document sets of Fig. 2 are separated somewhat arbitrarily into three categories, identified as "easy", "medium", and "hard" depending on the ease with which they adapt themselves to the clustering process. The clustering paths corresponding to the three categories are shown in the Figure together with the document percentage represented by each path. It may be seen that approximately 50 percent of the items are easy to cluster; 20 percent have medium clustering characteristcs, and the remaining 30 percent are hard to cluster. The document properties which render an item easy or hard to cluster are examined in more detail in the next section. For the moment it may suffice to note that a surprisingly large number of items (almost 1000 out of 3,500) are difficult to incorporate into the identifying the document itself. When item B cites item A, the A identifier in vector B will then match with the A identifier included in A to provide an appropriate term match. The document grouping process can then be based on common references, common citations and direct citations between documents.

The make-up of the computer science collection is shown schematically in Fig. 1. It may be seen that the collection consists of 334 base articles plus 3,186 references not previously included among the base articles. 13 of the base articles were referred to by other base articles and did themselves also cite other base articles. 33 base articles not included among the references cited other base articles, and 33 base articles were included among the references but did no citing of their own. A total of 3,893 distinct links (either references or citations between items) were found for the 3,520 articles.

A multilevel clustering process was used to classify the computer science collection. The process is bottom-up starting with the individual documents and creating increasingly larger clusters on successive (higher) levels of the cluster hierarchy. In pass 1 (P1), all articles that were referred to by one other document only and that themselves did not cite other items in the collection were incorporated into a common cluster with the citing item. This produced 327 pass-1 clusters. The remaining pass-2, pass-3, and pass-4 clusters (P2, P3, and P4) were formed by vector matching operations as previously explained. As one proceeds upward in the cluster hierarchy, the cluster size increases and the amount of similarity (the number of matching citation links) between an item and its peers in the same cluster decreases. The clustering statistics of Table 1 show that cluster size increases from an average of about 9 documents per P1 cluster clustering process. Over 100 documents do not fit into any P1 or P2 cluster, and an additional 830 items are incorporated into P1 clusters that cannot be merged further until at least level 4 of the tree.

The unbalanced tree of Fig. 2 is difficult to utilize for search purposes because an incoming query must apparently be compared initially with 9 level 4 centroids, 2 level 3 centroids, 8 level 2 centroids, 91 level 1 centroids, and finally 12 loose documents at level 0. Dummy nodes, and hence dummy centroid vectors may however be introduced, represented by heavy dots in Fig. 2. The use of dummy vectors increases the storage overhead but simplifies the search: for the structure of Fig. 2, each query must effectively be compared with 13 level 4 centroids only, of which 4 are dummy vectors; these in in turn give access to 28 level 3 centroids including 6 dummies, and so on down to the 3520 documents on level 0 of the tree.

The actual clustering results for computer science are examined in the remaining sections of this study.

3. Computer Science Taxonomy

Before building a taxonomy for a given field, it is important to define the elements to be included in the structure. In the present case, the individual documents do not by themselves constitute complete topic classes; on the other hand, the pass-3 and pass-4 clusters, each comprising an average of nearly 100 and 300 items respectively, appear too large for the representation of homogeneous subject areas. The taxonomy is therefore defined in terms of pass-2 clusters,and those pass-1 clusters that are not themselves incorporated in any pass-2 cluster. The corresponding taxonomy is reproduced as Fig. 3.

Each P1 or P2 cluster included in Fig. 3 is listed under one or more topic headings suggested by the cluster contents. The overall structure follows the model used to classify articles in the Encyclopedia of Computer Science. Six main headings are used including mathematics for computing, hardware, theory, software, computer systems, and methodology. [12] Two additional headings are included although the subjects are not well represented by the collection under study: applications and societal aspects, respectively.

The following coding system is used in Fig. 3 to identify the various research clusters:

a) three digit identifiers starting with a 6 are used for the 41 P2 clusters that appear in the easy clustering category;

b) four digit identifiers starting with a 6 represent the 20 P2 clusters with medium clustering characteristics;

c) four digits starting with digits 1, 2, 3, 4 or 5 represent the 57 P1 clusters with medium clustering properties;

d) the remaining P1 clusters which are hard to cluster are identified by 5 digit codes starting with digits 1 to 5 and ending in a 0.

It may be noticed that a number of clusters particularly in the easy clustering category are listed several times in the chart, the various occurrences being identified by subsripts. These clusters generally represent popular topics that cannot easily be identified by a single description. Thus, cluster 641 is described by "search systems" and "hashing" under Information Storage and Retrieval (Methodology), by "trees" and "decision trees" under Program and Data Structures (Software), and finally by "access methods"

The taxonomy of Fig. 3 may be further examined from three viewpoints: active vs. inactive topic classes; subjects that lend themselves to the formation of homogeneous research clusters, and others that do not; and finally, topics that relate well to other taxonomic areas, and others that remain isolated.

A number of topic classes proved surprisingly active in the computer science literature. Among them are

a) Mathematical computations (random number generation, contingency tables, etc). under Mathematics for Computing;

b) Storage management (segmentation, partitioning, virtual systems) under Computer Systems;

c) Coding systems (error correcting codes, convolution codes, etc.) under Computer Systems;

d) Pattern recognition (line drawings, curve detection feature analysis) under Methodology;

e) Computational models (mathematical programming, simulation, decision theory, etc.) under Methodology;

f) Combinatorial mathematics (game theory, clsssification, flow and network theory, scheduling theory, etc.) under Methodology; and

g) Graph theory (shortest path, connectivity, planarity, etc.) listed under combinatorial mathematics in the Methodology section.

On the other hand, certain other topic classes are poorly represented by the literature sample under study. Among them are not surprisingly Applications, and Societal Aspects, these topics being poorly covered in the ACM literature at large. Additional topics, such as data base management, were largely absent under Storage Management (Computer Systems). Similarily, cluster 610 appears as "splines" (Approximation Theory under Mathematics for Computing) and "moment method" (Pattern Recognition under Methodology).

The only documents not taken into account in the taxonomy of Fig. 3 are 57 hard to cluster single items that did not fit into any P1 or P2 cluster. These documents are discussed separately in the next section.

Before examining the taxonomy in more detail, it may be useful to say a few words about the main heading labelled Methodology, which incidentally is not used in the Encyclopedia of Computer Science. The methodology section appears to be somewhat of a catch-all, since it encompasses a variety of computer science topics such as artificial intelligence, graphics, information storage and retrieval, and data base systems. In addition, the section also includes a variety of techniques, such as pattern matching and combinatorial mathematics; a number of subjects tangential to computing such as graph theory and network theory; and mathematical or computational models such as mathematical programming, simulation, decision theory, information theory, and queueing theory.

All of these topics are covered quite extensively by the literature sample under study, and a case can be made for breaking up these subjects and assigning separate main beadings. The point of view taken here asserts that from a computer science viewpoint the techniques and methodologies developed for the automatic treatment of these topics is of principal interest rather than the particular application. The Methodology section must then be distinguished from the various applications areas (computers in business, computers in education, etc.), where the papers specifically appeal to readers that are not primarily computer scientists.

from the literature of 1974, and are therefore not represented by the research clusters of Fig. 3.

Another important characteristic of the taxonomy is the ease with which homogeneous research clusters are produced for certain subject areas, but not for some others. The most representative clusters from the point of view of subject identification are the P2 clusters containing an average of about 30 documents each. It turns out that the P2 clusters which are directly relatable to recognizable subtopics are unevenly distributed among the various subject headings. A number of areas are listed in Table 2 according to the ease with which they lend themselves to the construction of P2 clusters.

It may be seen from the Table that the topic classes generally recognized as the core of computer science fit easily into the structure; they are represented by P2 classes and exhibit many relations with other subject areas. Among these is the complete Software section, including programming and programming languages, and the systems topics (operating systems, etc.) included under Computer Systems. Also listed among the "easy" clustering topics is the section on Mathematics for Computing and the various Methodology sections that were mentioned earlier.

The topics that fit less well into the taxonomy are Hardware and Applications, and a number of theoretical topics such as automata theory, formal languages, and theorem proving that generate many small P1 clusters, but are difficult to merge into larger substructures.

Consider now the actual relationships that exist between the various subjects included in the classification of Fig. 3. The following topic similarities are easily recognized by examining the research clusters assigned to more than one area:

a) Computer Systems vs. Hardware

storage management - CPU logic design
access methods, scheduling - storage allocation
microprogramming - terminal equipment
networks - computers

b) Software vs. Theory

software engineering - program correctness
program linguistics - formal languages

c) Methodology (Pattern Matching) vs. Mathematics for Computing

pattern analysis - splines
character recognition - approximation theory
iterative analysis - matrix computation

d) Computer Systems vs. Methodology

storage management - queueing theory
networks - flow analysis.

The existence of these relationships is easy to verify. In each case, the boundary line between topics is difficult to draw, for example between Systems and Hardware, or between Software and Theory; and the connection between mathematical methodology and the Mathematics section is also obvious. Indeed the Methodology section relates directly to most other portions of the taxonomy.

A general summary of the taxonomy is not easy to draw: the topics of computer science are all represented and recognizable in the classification. The wide open Methodology section which impinges on so many related topic classes demonstrates, however, that the boundaries of computer science are not well-defined. This is, of course, confirmed by the fact that computing is a

Table 4(b) contains surveys and historical articles where the situation is the dual of the one previously described: instead of heterogeneous citations, there are many heterogeneous references to other items. Tables 4(c) and 4(d) contain programming language descriptions and various interdisciplinary items that are in part tangential to computing. Here both the references as well as the citations may relate to several different topic areas.

The last class of items in Table 4 covers well-known research articles. Several different reasons may be given in this case for the clustering failure. Some of the articles are seminal works starting new endeavors in the field for which citations had not accumulated by 1976; some of the other papers contain interdisciplinary references; and a few finally exhibit no references at all.

One may conclude from the foregoing exposition that the clustering characteristics of a given item have nothing to do with the importance of the item in the field. Completely novel items that open up new areas are difficult to classify, as are basic reference works and interdisciplinary items. In these circumstances it is difficult to predict in advance where a given item, or a given author, will fit into the cluster organization for a given field.

This is confirmed by examining the list of 25 "prolific" authors shown in Table 4. These authors are represented at least 10 times in the data base of 3520 articles; they are entered in the table under the "easy", "medium", or "hard" label using the criteria of Fig. 2. That is, if a substantial number of articles by a given author follow the clustering paths described as "easy" in Fig. 2, the author is listed in the "easy clustering" category. The same applies to the medium and hard categories. Once again, it is clear that clustering ease does not seem to be correlated with eminence in the field.

popular topic in many areas of the sciences and humanities in addition to computer science itself.

4. Authors and Papers

It is not possible in the present context to give a detailed analysis of individual documents and authors. It may, however, be of interest to inquire briefly into the nature of those documents that did not fit into any of the P1 and P2 clusters [paths (— — — —), (— — P3 —), and (— — — P4) of Fig. 2]. In so doing, one must remember that an item is incorporated into a cluster whenever its similarity with a given cluster centroid is substantially higher than its similarity with the other centroids. Items that present clustering difficulties are therefore those that exhibit no substantial similarities with any of the centroids, or those on the contrary that present affinities with a large number of centroids. In the latter case, the choice is between entering the items into all of these clusters simultaneously, or alternatively to keep them unclustered. The former possibility produces a great deal of storage overhead, and should therefore be avoided whenever possible.

A list of representative unclustered items is contained in Table 3 arranged in several different categories. Table 3(a) contains basic reference works that are cited by a large class of heterogeneous items. Included in this class are the famous volumes by Knuth on "The Art of Programming". A clustering diagram for a typical basic reference work is shown in Fig. 4, indicating citations originating in many different clusters. The item illustrated in Fig. 4 is eventually included in a cluster at level 4 of the cluster tree.

It does appear, however, that authors in the software area are generally included in the "easy" class, whereas many theoreticians exhibit "medium" or "hard" clustering characteristics.

In principle, it seems desirable to draw some overall conclusions about computer science as a scientific discipline, or at least about the state of computer science around 1974. However such conclusions might be shallow and emphemeral. It seems safest then to let the automatic taxonomy speak for itself. In a few years, the exercise might be repeated, possibly with a larger cross-section of the literature to determine whether the core has remained identical and to examine the extent to which the field has evolved.

REFERENCES

(1) G. Salton, D. Bergmark, and A. Wong, Generation and Search of Clustered Files, Computer Science Technical Report 77-299, Cornell University, Ithaca, New York, January 1977.

(2) G. Salton, Dynamic Information and Library Processing, Prentice Hall Inc., Englewood Cliffs, New Jersey, 1975, Chapter 8.

(3) D. J. deSolla Price, Networks of Scientific Papers, Science, Vol. 149, No. 3683, 1965, p. 510-515.

(4) J. H. Westbrook, Identifying Significant Research, Science, Vol. 132, No. 3435, 1960, p. 1229-1234.

(5) J. Margolis, Citation Indexing and Evaluation of Scientific Papers, Science, Vol. 155, No. 3767, 1967, p. 1213-1219.

(6) M. M. Kessler, Bibliographic Coupling Between Scientific Papers, Am.Documentation, Vol. 14, 1963, p. 10-25.

(7) E. Garfield, Citation Indexing, Historio Bibliography and the Sociology of Science, Third Int. Congress of Medical Librarianship, Amsterdam, May 1969, p. 187-206.

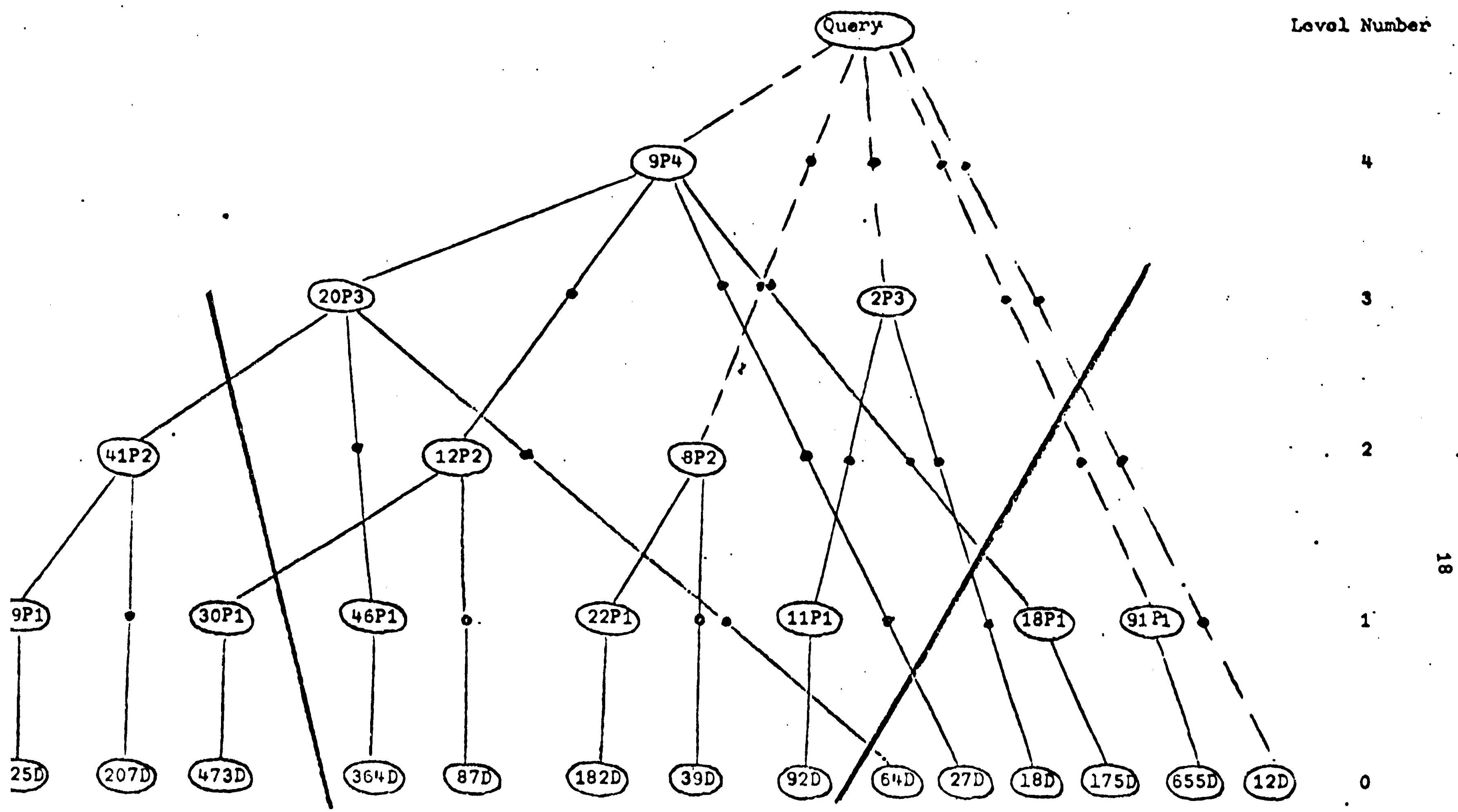
(8) E. Garfield, Science Citation Index - A New Dimension in Indexing, Science, Vol. 144, No. 3619, May 1964, p. 649-654.

(9) M. M. Kessler, Comparison of Results of Bibliographic Coupling and Analytic Subject Indexing, Am. Documentation, Vol. 16, No. 3, 1965, p. 223-233.

(10) G. Salton, Automatic Indexing Using Bibliographic Citations, Journal of Documentation, Vol. 27, No. 2, June 1971, p. 98-110.

(11) E. Garfield, Citation Analysis as a Tool in Journal Evaluation, Science, Vol. 178, No. 4060, November 1972, p. 471-479.

(12) A. Ralston and C. L. Meek, Encyclopedia of Computer Science, Petrocelli Charter, New York, 1976.



Easy clustering 51%

P1 P2 P3 P4	32%
- P2 P3 P4	6%
P1 P2 - P4	13%

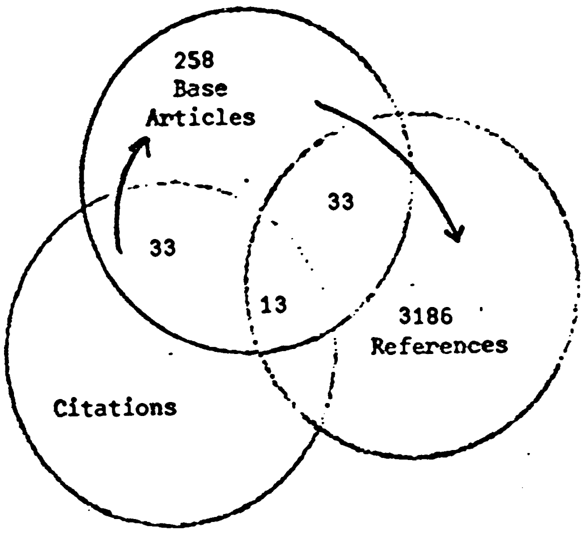
Medium clustering 21%

P1 - P3 P4	10%
- P2 - P4	2%
P1 - P3 -	3%
P1 P2 - -	5%
- P2 - -	1%

Hard clustering 28%

- - P3 P4	2%
P1 - - P4	5%
- - - P4	1%
- - P3 -	0.5%
P1 - - -	19%
- - - -	0.5%

Cluster Tree for 3520 Computer Science Documents
(with clustering characteristics identified)



334 base articles

3,232 distinct refcrences from base articles

3,520 distinct collection items

3,893 distinct links

Make-up of Computer Science Collection

(1974 ACM literature plus 1968 and 1971 articles from

Computer and Control Abstracts)

Fig. 1

3520 original documents	1.0 document/cluster
327 pass-1 clusters	9.5 document/cluster
61 pass-2 clusters	33.0 document/cluster
22 pass-3 clusters	95.0 document/cluster
9 pass-4 clusters	274.3 documents/cluster

Clustering Statistics

Table 1

division	616_1
Approximation Theory	617_1 651_2 21920
Interpolation	25660
splines	610_1 6034 25120
least squares	13470
Chebyshev approximation	6033_1
partial fractions	3829
Ordinary Differential Equations	
initial value problems	6008
boundary value problems	6026
Partial Differential Equations	640_2
Numerical Integration	616_2 21950
Newton Cotes	2798
Fast Fourier Transform	614 621_2 3764
cyclic convolution	3771
signal theory	35470 37340
Matrix Computations	619_1 621_3 3946
eigenvalues	619_3
root finding	644_1
clustering	620_3
determinants	3765
Error Analysis, Stability	623_3 651_3
Statistical Computations	620_2
Kolmogorov Smirnov distr.	2931
random number generation	2932 6033_2
contingency tables	22000
classification error	28232
digital filtering	3737

__Hardware__

Performance reliability	6364
Computers	
super computers	636_2
microprogrammed	3784_1
Computer Design	37530
virtual machine design	27580_2
control structures	40510
parallel microcomputer	3785_3
Memory Systems	
failure tolerant design	3797
storage allocation	600_3
Peripheral Equipment	
terminals	627_2
Central Processing Unit	
logic design	654_2

Computer Science T[illegible]

Fig. 2

finite state automata	[illegible]
pushdown automata	1394
sequential machines	1319 3644
mixture automata	3018
stochastic automata	3800 3830 37570
Formal Languages	648_2 656_3 47550
context free grammar	1010
parsing	3664_2
right linear grammars	3756
closure preservation	3539
ambiguity	37470_2
production systems	25710
Theorem Proving	645 647_1
undecidability	13480
refutation	13490
mathematical logic	33580_3
Theory of Algorithms	609_2 630_3 659_3
Complexity, Computability Unsolvability	607_3 6057
Program Correctness	630_3 643_1 656_3 25
loop predicates	2572
Program Optimization	643_2 659_2 10090
Boolean Algebra	624_3 13550

__Software__

__Programming__	
structured	601_1 620_1 629_3 630
debugging	33470
practice	629_2 16710
concurrent	23500
style	628_3
System Software	
opcode interpretation	37320
opcode utilization	37690
self stabilizing systems	16670
compilers	650
Programming Languages	
Pascal	601_2
high-level	6304
Cobol evaluation	51190
Simula and Fortran	33290
AI languages	1558_1
Math. logic languages	33580_2
Software Engineering	638_2
portability	
chief programmer team	628_2
Program and Data Structure	632 5538
decision tables	649_1
strings	638_1
trees, decision trees	641_2 649_2
stacks	
Programming Linguistics	643_3
state transition techniques	3031
parsing	648_1 648_3 658_3
string processing	640_2

error correcting codes	6570 60950 21690
convolution codes	3507
burst errors	4764
radix transformation	616 3768

Storage Management

extendible arrays	13540
dynamic allocation	11390
register allocation	1666 5253
paging	600_2 602 603_3 653_3 642_2 7500 7502
partitioning	4649
virtual system	27500_3
file organization	647_1
storage structures	654_3
access methods	600_1 641_4
segmentation	637_4

Time Sharing

scheduling methods	

Networks	636_3
computer utility	639_3

320

Processing Mode

parallel processing	607_1 636_1 658_1
multiprocessing	627_2 653_2 658_2 3785_2

Microprogramming	612_1 622_1 3700_2

System [illegible]ement

security	627_3 637_1
cryptography	637_2
protection	627_3
data compression	23530

Applications

Music	16450
Social Science Applications	77030
Computing in Business	[illegible]
Education	25070
teaching machines	[illegible]
time table construction	635_1

Societal Aspects

privacy	637_3
[illegible] systems	639_3

6xx	77 clusters easy clustering
1xxx to 5xxx	71 clusters medium clustering
10xx	77 clusters medium clustering
1xxx0 to 7xxx0	71 clusters hard clustering

621_1 646_1 [illegible]

moment method	610_2
line drawings	605_2
image processing	611_2 3826
character recognition	617_2 620_4 646_2
iterative analysis	619_2
bubble chamber scan	6011
curve detection	1762
chromosome recognition	[illegible]
mechanical diagnostics	3015

Artificial Intelligence

heuristic processing	647_2
speech recognition	1664_1 3803
fact retrieval	
AI languages	1558_2

Computational Models

mathematical programming	605_3 653_3 6052_1
integer [illegible]amming	11410
dynamic programming	609
nonlinear algebra	48510
simulation	
decision theory	623_2
information theory	24559
[illegible]ing theory	653_1 11360

Combinatorial Mathematics

switching theory	624_2
game t[illegible]	35810
classification	670_3 660_3 3623_3
flow analysis	601_3 603_2 659_3
sc[illegible]ing theory	23610 37420_1
network theory	
sorting network	618_1 [illegible]
graph theory	
graph coloring	657_2 635_2
connectivity	3345 37510
partitioning	636_3
chromatic number	660_2
shortest path	2705 4857 6852_2 34970_2
Hamiltonian path	13460 34400
spanning tree	2194 3730
graph matching	4215
planarity	1344
transitivity sets	23620

Information Storage and Retrieval	29362
automatic indexing	40600
search systems	640 603_2 34420 38700
document retrieval	16700 27040
retrieval evaluation	36160
hashing	643_3

Data Base Systems

Computational Linguistics

content analysis	3440

Symbol Manipulating Systems

algebraic computations	

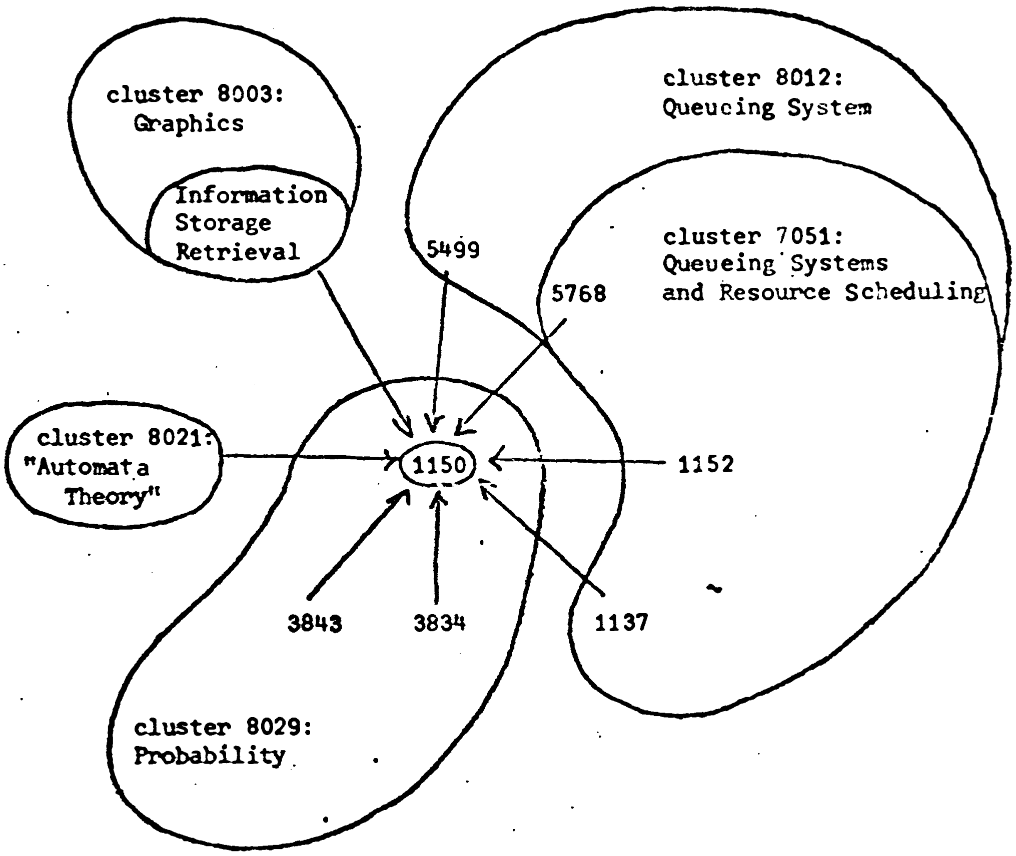
Graphics

hidden surface	633_2
raster display	638 30340
scanning	
half tone	633_1

Homogeneous Research Areas Represented by P2 clusters	Nonhomogeneous Areas Not Represented by P2 clusters
Software (complete section)	Hardware
Mathematics for Compiling	Composite Design
(except for statistics)	Theory
Computer Systems	Automata Theory
Storage Management	Formal Languages
Processing Modes	Theorem Proving
Systems Management	Computer Systems
Methodology	Coding Systems
Pattern Recognition	Applications
Computational Models	(complete section)
Combinatorial Math.	Methodology
	Information Storage and Retrieval

Homogeneity of Clusters of Topic Classes in Taxonomy

Table 2



8xxx level 4 clusters

7xxx level 3 clusters

A → B

A refers to B

B is cited by A

Clustering Diagram for a Basic Reference Work

(Feller, An Introduction to Probability Theory)

Fig. 4

Ginsburg	An Introduction to Mathematical Machine Theory
Gries	Compiler Construction for Digital Computers
Harrison	Introduction to Switching and Automata Theory
Hartmanis and Stearns	Algebraic Structure Theory of Sequential Machines
Husson	Microprogramming Principles and Practice
Knuth	The Art of Programming, Volumes 1 and 3
Nilsson	Problem Solving Methods in Artificial Intelligence
Rabin	Probabilistic Automata

a) Basic Reference Works

Bobrow and Raphael	Comparison of List Processing Computer Languages
Dreyfus	Shortest Path Algorithms
Feldman and Gries	Tanslator Writing Systems
Geoffrion and Marston	Integer Programming: A Framework and State-of-the-Art Survey
London	Bibliography on Proving Program Correctness
Martin	Sorting

b) Survey and Tutorial Articles
(many references)

Falkoff and Iverson	APL/360 - User's Manual
Naur	Revised Report on the Algorithmic Language Algol 60
Van Wijngaarden	Report on the Algorithmic Language Algol 68

c) Programming Language Descriptions

Sample Items with Hard Clustering Characteristics

Table 3

Berlekamp	Algebraic Coding Theory
Berge	Theory of Graphs and its Applications
Birkhoff and Maclane	Survey of Modern Algebra
Chomsky	Aspects of the Theory of Syntax
Peterson	Error Correcting Codes
Salton	Automatic Information Organization and Retrieval
Shannon and Weaver	Mathematical Theory of Communications
Weinberg	Psychology of Computer Programs

Tangential or Interdisciplinary Items
(references to and citations from several areas)

Cline	Curve Fitting Using Splines Under Tension
Dahl, Dijkstra, and Hoare	Structured Programming
Dijkstra	Structures of the THE Multiprogramming System
Floyd	Assigning Meaning to Programs
Greibach	Hardest Context Free Language
Hopcroft and Tarjan	Efficient Algorithms for Graph Manipulation
Huffman	A Method for the Construction of Minimum Redundancy Codes
Karp	Reducibility among Combinatorial Problems
Knuth	Empirical Study of Fortran Programs
Landin	Correspondence between Algol 60 and Church's Lambda Notati
McCarthy	A Basis for a Mathematical Theory of Computation
McNaughton and Anderson	Regular Expressions and Static Graphs for Automata
Nievergelt and Wong	Dynamic Storage Aloocation for Binary Search Trees
Ross	The AED Approach to Generalized Computer Coded Design

Research Articles
(Starting new areas, no references,
interdisciplinary references)

Sample Items with Hard Clustering Characteristics

Table 3 (cont.)

	Author Name	Total Pages for Author	Number of Pages with Clustering Characteristics: Easy	Medium	Hard
1.	A. V. Aho	17	16	0	1
2.	E. G. Coffman	21	16	2	3
3.	P. J. Denning	11	10	0	1
4.	H. Freeman	14	14	0	0
5.	K. S. Fu	11	11	0	0
6.	I. J. Good	11	10	0	1
7.	M. A. Harrison	12	7	4	1
8.	C. A. R. Hoare	13	11	1	1
9.	S. L. Pollack	16	13	0	3
10.	C. V. Ramamoorthy	19	17	0	2
11.	J. D. Ullman	34	28	3	3
12.	N. Wirth	11	11	0	0
13.	C. K. Wong	12	10	0	2

a) Authors with Easy Clustering Characteristics

14.	E. W. Dijkstra	20	13	2	5
15.	J. Hartmanis	10	4	4	2
16.	IBM	20	11	3	6
17.	D. E. Knuth	23	14	3	0
18.	A. Rosenfeld	19	8	10	1
19.	J. R. Slagle	11	6	2	3

b) Authors with Medium Clustering Characteristics

20.	D. G. Bobrow	12	1	8	3
21.	T. Winograd	11	2	5	4
22.	J. E. Hopcroft	12	2	2	8
23.	A. L. Rosenberg	13	0	4	9
24.	G. Salton	13	1	1	11
25.	R. E. Tarjan	13	3	0	10

c) Authors with Hard Clustering Characteristics

Total for 25 Authors	379	63%	14%	23%

Clustering Characteristics for Prolific Authors
(at least 10 papers per author)

Table 4