(90) Sabljic, A.; Trinajstic, N. "Quantitative Structure–Activity Relationships: The Role of Topological Indices". *Acta Pharm. Jugosl.* **1981**, *31*, 189–214.
(91) Kubinyi, H. "Quantitative Structure–Activity Relationships. 2. A Mixed Approach, Based on Hansch and Free-Wilson Analysis". *J. Med. Chem.* **1976**, *19*, 587–600.
(92) Rispin, A. "Introduction to Symposium on the Development and Use of Reliable Data Bases for Quantitative Structure–Activity Relationships". *J. Chem. Inf. Comput. Sci.* **1981**, *21*, 1, and references therein.
(93) Tinker, J. "Relating Mutagenicity to Chemical Structure". *J. Chem. Inf. Comput. Sci.* **1981**, *21*, 3–7.

Some Heuristics for Nearest-Neighbor Searching in Chemical Structure Files

PETER WILLETT

Department of Information Studies, University of Sheffield, Western Bank, Sheffield S10 2TN, England

Received August 17, 1982

Nearest-neighbor searching usually involves inspecting all of the records in a file for the record which best matches an input query. Three heuristics are described for nearest-neighbor searching in chemical structure files where molecules are represented by fragment bit strings. The procedures can reduce the number of compounds inspected by between 81% and 97%, depending upon the heuristic and the matching function used, while still ensuring the identification of the nearest neighbor.

INTRODUCTION

Exact match and partial, or inclusive, match searching algorithms are widely used in computer-based chemical information systems for the purpose of registration and substructure search, respectively. A less common facility is provision for best-match, or nearest-neighbor, searches in which the structure or structures most similar to an input query structure are retrieved, similarity being defined on the basis of some similarity coefficient or distance function[1] which reflects the number of fragments common to the query and to a molecule in the file. Best-match searching forms the basis for the k nearest-neighbor classification[2] and plays an important role in the use of spanning trees[3,4] and of automatic classification techniques.[5-9]

The general problem of finding best matches is defined by Friedman et al.[10] as "...given a file of N records (each of which is described by k real valued attributes) and a dissimilarity measure D, find the m records closest to a query record (possibly not in the file) with specified attribute values". The obvious, brute-force algorithm for best-match searching is to compute the distance between the query and each of the records in the file and then to select the m shortest distances; this algorithm has a file size dependency of $O(N)$ and is much too time-consuming for all but the smallest files.

This paper describes the use of several heuristics which, although not reducing the complexity of the search below $O(N)$, are sufficiently powerful to allow nearest-neighbor searches of chemical structure files to be carried out at reasonable computational cost. All of the experiments consider the retrieval only of the nearest neighbor, i.e., $m = 1$, but the procedures outlined may be generalized to a range of other closest point problems for which $m > 1$.

NEAREST-NEIGHBOR SEARCHING

An efficient nearest-neighbor algorithm will be one which avoids the calculation of most of the distances while still calculating the distances for those few records which are, in fact, near the query structure. Several types of criteria have been suggested to reduce the number of calculations required, including the projection of the d-dimensional records onto a lower dimensional space where most of the distance calculations are performed[11,12] and grouping records into clusters so that several records may be searched, or eliminated from the search, simultaneously.[10,13-16] Many of the cited algorithms may not be directly applicable to best-match searching in a chemical context since they assume that the attributes are continuous variables, whereas chemical structures are usually characterized by a binary fragment description. In this, each of the structures in a file is represented by a bit string in which the i'th bit is set if the corresponding fragment is present in the structure. Also, it is often assumed that the records lie in a d-dimensional space where d is small, typically 2 or 3, so that multiplicative terms in d in the equation describing the number of matches required may be neglected; in a chemical structure system, d may be of the order of 10^2 or 10^3 (the number of bits in the bit string), and such algorithms are, accordingly, quite impracticable. Thus the $O(\log N)$ procedure due to Friedmam et al.[10] involves a constant of proportionality of about 1.6^d while the search method of Bentley et al.[16] involves the inspection of all of the $3^d - 1$ cells adjacent to a given cell in a d-dimensional space. Marimont and Shapiro[17] discuss the dimensionality problem, but their experiments were still restricted to spaces with $d \leq 40$.

van Marlen and van den Hende[18] and Rasmussen et al.[19] have described best-match retrieval algorithms for use with machine-readable mass spectra files, where a structure is characterized by a bit string corresponding to the peaks observed in the molecular mass spectrum, while Smeaton and van Rijsbergen,[20] Murtagh,[21] and Perry[22] have studied best-match searching in the context of document retrieval systems. Smeaton and van Rijsbergen note that an inverted file may be used to increase search efficiency since a query needs to be matched only against those documents with which it has at least one term in common. They then describe experiments with an upper-bound procedure which enables a best match search to be teminated before all of the documents in the inverted file lists corresponding to a query have been inspected. Murtagh and Perry describe an extension of this algorithm in which additional upper bounds are calculated, this resulting in a further reduction in the number of documents that need to be matched against a query.

EXPERIMENTAL DETAILS AND SIMILARITY MEASURES

The experiments used a set of 2335 structurally disparate compounds from the Index Chemicus Registry System. Each

0095-2338/83/1623-0022$01.50/0

of these molecules was characterized by a bit string describing the augmented atom fragments contained within it, and an additional 100 heterogeneous compounds were similarly represented to act as queries for which the best matching structure was required.

A variety of measures has been suggested for the evaluation of interobject similarity.[1] The simplest measure is the number of fragments, c, common to a pair of structures. However, this means that large structures are likely to appear more similar to a query simply because there is a greater probability of them containing the required fragments than a small molecule. Accordingly, a range of similarity coefficients has been suggested[20] in which the size of a structure and of the query are taken into account by using some normalizing factor. Those used here are the Dice coefficient

$$2c/(q + s)$$

and the Overlap coefficient

$$c/\min(q,s)$$

where c is the number of fragments common to a query and to a structure containing q and s nonzero bits, respectively. The fourth function considered is the Hamming distance which is merely the number of positions at which a pair of bit strings differ

$$q + s - 2c$$

The final function is derived from the observation that a pair of structures having a very frequently occurring substructure in common should be regarded as less similar that a pair which share a very infrequently occurring feature. Harrison[6] suggested an inverse frequency weighting scheme in which the contribution to the total similarity between a pair of structures arising from a fragment of collection frequency f_i should be the negative logarithm of its probability of occurrence in a given structure

$$\log_e (N/f_i)$$

SERIAL SEARCHING USING BLOCKED RECORDS

A simple heuristic for best-match searching is described by van Marlen and van den Hende for use when the Hamming distance is chosen as the matching function.[18] Given a query with q fragments assigned, the file is partitioned into a series of mutually exclusive groups, in each of which all of the compounds have the same number of fragments. The groups are searched in increasing order of the value of

$$|q - s|$$

which is the minimum possible number of bit mismatches between the query structure and a compound with s nonzero bits in its bit string. The search is terminated as soon as this value becomes greater than the distance to the current nearest neighbor since no subsequent molecule can possibly be closer.

The procedure is basically the same as the reference point algorithm originally described by Burkhard and Keller[13] and by Shapiro.[14] The analysis presented in these two papers involves the use of the triangle inequality and and is applicable only to metrics such as the Hamming distance. However, the approach may be generalized to nonmetric matching functions without invoking the inequality as will be illustrated by reference to the Dice coefficient. The maximum possible number of common fragments is

$$\min(q,s)$$

from which an upper bound to the value of the matching function is

$$2\min(q,s)/(q + s)$$

This upper bound is calculated for each distinct value of s which is present in the file, and the structures are inspected in decreasing order (increasing order in the case of a distance function) of their upper-bound value until the current nearest neighbor has a higher match value than the upper bound for the next uninspected block of compounds.

The efficiency of the search may be further increased if a mechanism is available for the grouping of individual compounds into blocks, each of which is characterized by a bit string obtained from the logical union of the bit strings of all of the molecules in that block. This idea has been discussed by several workers[23-25] as a means of reducing disk accesses, but always in the context of Boolean retrieval, where an exact or partial match is required, rather than for best-match searching as here. A query is matched initially against such a block bit string, which shall be referred to as a *representative*, to identify the number of fragments, c_{max}, common to the query and to the representative. Since c_{max} is an upper bound to the number of fragments common to the query and any of the molecules in the block, it may be used to calculate an upper bound for the similarity function: if the calculated value is less than the current largest similarity coefficient, the structures within the block may be ignored since none of them can possibly be the best match for the query. In fact, c_{max} can be replaced by $\min(c_{max},s)$ since the number of common fragments cannot possibly be greater than s, and the upper bound may then be reduced accordingly.

The extension of these ideas to encompass inverse frequency weighting is slightly more complex since the calculated upper bound for a block is the sum of the weights for the $\min(c_{max},s)$ most highly weighted matching fragments. Similar modifications for inverse frequency weighting are needed for the inverted file searches described in the next section.

As the representative for a block is the union of all of the compound bit strings within a block, the number of nonzero bits in the representative will depend in large part upon the number of compounds in the block and thus, since the total number of structures in the collection, N, is fixed, upon the number of blocks n_b. When n_b is small, there will be many nonzero elements in the representative, and the calculated upper bound will be high: accordingly, many or most of the blocks will need to be inspected in detail. Conversely, when n_b is large, the number of query-representative coefficients evaluated will be greater, but fewer query-molecule comparisons will need to be made. Thus, for a given set of structures and queries, there will be some number of blocks which results in a minimal number of coefficients being evaluated.

USE OF AN INVERTED FILE

An inverted file provides a simple means of reducing the number of coefficients evaluated in best-match searching. This arises because coefficients need only be calculated between the query and those structures which occur in the inverted file lists corresponding to the nonzero bits in the query bit string: molecules which do not occur in any of these lists will not have any fragments in common with the query and will thus give rise to a zero-valued coefficient were it to be evaluated. It should, however, be noted that it is *possible* for the nearest neighbor to have no terms in common with the query if the Hamming distance is used as the similarity function. A simple search algorithm hence consists of isolating the lists of molecules from the inverted file which correspond to the fragments in the query, and matching the query against each of the isolated structures. Since a compound having m fragments in common will occur in m of the selected lists, a further reduction in computation may be obtained by ensuring that each structure is compared only once with the query: this may be achieved either by "ORing" the lists before any of the

coefficients are evaluated or by maintaining a record of the structures which have been inspected. A potential limitation of this approach in a chemical context is that the Zipfian distribution of fragment occurrences[26] is so skewed that some fragment types will occur in very many of the molecules in a file, thus ensuring that the majority of the structures will need to be inspected for each query.

Smeaton and van Rijsbergen[20] have described an upper-bound strategy which can reduce the computation further by eliminating the need to inspect some of the longer inverted file lists corresponding to query fragments. The upper bound is calculated after the processing of the lists of compounds corresponding to a particular query fragment has been completed: if this upper bound is less than the current maximum similarity, the algorithm terminates.

The algorithm involves two preprocessing steps. First, the list of query fragment is sorted into increasing order of their frequency of occurrence in the file of compounds which is to be searched: this implies that if any of the inverted file lists do not in fact need to be inspected, it is the longer ones that will be omitted, thus increasing the efficiency of the procedure. Second, when the inverted file is created, a note is made of the maximum and minimum values for the number of nonzero bits within the bit strings of all molecules which have been assigned a given fragment.

Let the maximum and minimum values for the m'th query fragment be s_m^{max} and s_m^{min}. Assume that $m - 1$ of the q inverted file lists corresponding to query fragments have been processed at some point in the search. Then the maximum possible number of fragments in common between the query and a previously uninspected compound occurring in the list corresponding to the m'th fragment is

$$q - m + 1$$

By use of the Dice coefficient as an example, the corresponding upper bound to the similarity is

$$2(q - m + 1)/(q + s_m^{min})$$

It is possible that

$$q - m + 1 > s_m^{max}$$

and in this case, the maximum possible number of common fragments is

$$\min((q - m + 1), s_m^{max})$$

giving

$$2\min((q - m + 1), s_m^{max})/(q + \min((q - m + 1), s_m^{max}))$$

as the upper bound. If u_m is the upper bound for the m'th query fragment calculated by using either of the above formulas, the maximum possible upper bound for all uninspected structures after $m - 1$ lists have been processed is

$$\max\{u_j\},\ m \leq j \leq q$$

If this upper bound is less than the match value for the current nearest neighbor, none of the uninspected documents can possibly be a better match, and processing can be terminated at that point. The analysis presented here is an improved version of that given by Smeaton and van Rijsbergen in the sense that the upper bounds calculated here are smaller in value than those in their original paper: the reduction in the number of matches required is correspondingly greater.

While the upper bound removes the need to inspect entire inverted file lists, Murtagh[21] and Perry[22] calculate additional upper bounds while a list is being processed to remove the need to inspect some of the individual compounds within that list. Their procedure requires random access to an N-element table which contains the value of s for each of the compounds.

Table I. Fraction of the File Inspected

matching function	method[a] (a)	(b)	(c)
simple	0.14	0.05	0.03
dice	0.09	0.11	0.04
overlap	0.06	0.12	0.03
hamming	0.10	0.12	0.03
inverse frequency	0.19	0.04	0.04

[a] (a) The blocked serial search, (b) the upper-bound inverted file search, and (c) the extended upper-bound inverted file search.

When a previously uninspected structure, i, is encountered during the processing of the inverted list corresponding to the m'th query term, the maximum possible number of terms common to the structure and the query is calculated by using

$$\min((q - m + 1), s_i)$$

from which an upper bound for the similarity measure may be calculated. The compound's bit string is then matched against the query only if this new upper bound is greater than the similarity for the current nearest neighbor. This is obviously a much more precise upper bound than that calculated for the entire set of structures in the uninspected inverted file lists, and it may hence be expected to result in further substantial reductions in the number of coefficients that are evaluated.

RESULTS AND DISCUSSION

The results obtained for each of the three heuristics, using each of the matching functions, are shown in Table I. In each case, the figure quoted is the mean fraction of the file which must be searched, the average being calculated over the entire set of 100 query structures.

The analysis in the Serial Searching Using Blocked Records section suggested that there will be some block size, N/n_b, which will minimize the number of coefficients calculated in the blocked serial search. For the set of structures and queries used here, the optimal value for N/n_b was found to be 8, and the figures listed in Table I for the blocked serial search are those obtained with this block size. The results for the inverted file searches may be compared with the fraction of the file that would need to be inspected in a full inverted file search, i.e., one in which no upper bounds were calculated: this fraction was found to be 0.78.

The figures in Table I show clearly the efficiency of the search heuristics in reducing the number of compounds that must be matched against a query, with the two-stage inverted file search in particular giving a substantial increase in efficiency. However, the overall computational costs will also depend upon other factors such as the amount of additional storage and preprocessing needed and on the extent to which accesses must be made to backing storage. The preprocessing requirements have been described in the previous two sections, and discussion will hence be restricted to the latter two factors.

The only additional storage needed for the blocked serial search is that required for the representatives. Since there is one representative per block, storage equivalent to n_b molecular bit strings will be needed, together with space for a table giving the value of s for each of the blocks. The inverted file searches need to store the maximum and minimum s values for each inverted file list and the s values for each compound. The inverted file lists may be extracted from the basic data matrix merely by isolating the appropriate bit position in the N bit strings, so that the inverted file itself need require no additional storage at all.

It has been assumed so far that all of the bit strings can be held in the main store of the computer, but this is unlikely to

be possible if very large files are to be searched. Consideration must hence be given to the effect of disk accesses on the search algorithms. In the case of the blocked serial search, a considerable amount of seek time will be needed to allow the scanning of the file in the order dictated by the upper bounds for the blocks. The most efficient approach is probably to choose a block size such that the set of representatives will fit into the main store and then to access only those blocks for which c_{max} is such as to suggest the presence in a block of a nearest neighbor; the chosen block size is likely to be substantially greater than the optimal value found in these experiments where the entire file was held in core. The inverted file searches will require accesses for the inverted file lists, however stored, corresponding to the query fragments, and, more importantly, accesses will also be needed for each of the molecular bit strings which must be compared with the query.

If a very large file is to be searched, the best approach is probably to use the inverted file algorithm described by Willett[9] for the calculation of intermolecular similarity coefficients in automatic classification experiments. This algorithm requires a fair amount of working storage and also results in the same number of coefficients being evaluated as in the full inverted file search. However, it requires no more than q accesses per query, one for each inverted file list, and will thus be considerably faster in operation despite the greater number of coefficients that must be calculated.[27]

CONCLUSIONS

Nearest-neighbor searching normally involves the inspection of all of the compounds in a machine-readable structure file. The heuristics described in this paper permit a substantial reduction in the number of molecules that need to be matched against a query structure, while still ensuring the identification of the best match.

ACKNOWLEDGMENT

My thanks are due to Shirley Perry for the use of her programs and to Michael Lynch and Steven Welford for helpful comments.

REFERENCES AND NOTES

(1) Adamson, G. W.; Bush, J. A. "A Comparison of the Performance of some Similarity and Dissimilarity Measures in the Automatic Classification of Chemical Structures" *J. Chem. Inf. Comput. Sci.* **1975**, *15*, 55–58.
(2) Kowalski, B. R.; Bender, C. F. "The K-Nearest Neighbor Classification Rule. Pattern Recognition Applied to Nuclear Magnetic Resonance Spectral Interpretation" *Anal. Chem.* **1972**, *44*, 1405–1411.
(3) Ritter, G. L.; Isenhour, T. L. "Minimal Spanning Tree Clustering of Gas Chromatographic Liquid Phases" *Comput. Chem.* **1977**, *1*, 145–153.
(4) Miyashita, Y.; Takahashi, Y.; Yotsui, Y.; Abe, H.; Sasaki, S. I. "Application of Pattern Recognition to Structure–Activity Problems. Use of Minimal Spanning Tree" *Anal. Chim. Acta* **1981**, *133*, 615–620.
(5) Adamson, G. W.; Bush, J. A. "A Method for the Automatic Classification of Chemical Structures" *Inform. Storage Retr.* **1973**, *9*, 561–568.
(6) Harrison, P. J. "A Method of Cluster Analysis and some Applications" *Appl. Stat.* **1968**, *17*, 226–236.
(7) Adamson, G. W.; Bawden, D. "Comparison of Hierarchical Cluster Analysis Techniques for the Automatic Classification of Chemical Structures" *J. Chem. Inf. Comput. Sci.* **1981**, *21*, 204–209.
(8) Willett, P. "A Comparison of some Hierarchal Agglomerative Clustering Algorithms for Structure Property Correlation" *Anal. Chim. Acta* **1982**, *136*, 29–37.
(9) Willett, P. "The Calculation of Inter-Molecular Similarity Coefficients Using an Inverted File Algorithm" *Anal. Chim. Acta* **1982**, *138*, 339–342.
(10) Friedman, J. H.; Bentley, J. L.; Finkel, R. A. "An Algorithm for Finding Best Matches in Logarithmic Expected Time" *ACM Trans. Math. Soft.* **1977**, *3*, 209–226.
(11) Friedman, J. H.; Baskett, F.; Shustek, L. J. "An Algorithm for Finding Nearest Neighbors" *IEEE Trans. Comput.* **1975**, *C-24*, 1000–1006.
(12) Lee, R. C. T.; Chin, Y. H.; Chang, S. C. "Application of Principal Component Analysis to Multikey Searching" *IEEE Trans. Soft. Eng.* **1976**, *SE-2*, 185–193.
(13) Burkhard, W. A.; Keller, R. M. "Some Approaches to Best-Match File Searching" *Commun. ACM* **1973**, *16*, 230–236.
(14) Shapiro, M. "The Choice of Reference Points in Best-Match File Searching" *Commun. ACM* **1977**, *20*, 339–343.
(15) Yuval, G. "Finding Nearest Neighbours" *Inf. Proc. Lett.* **1976**, *5*, 63–65.
(16) Bentley, J. L.; Weide, B.W.; Yao, A. C. "Optimal Expected Time Algorithms for Closest Point Problems" *ACM Trans. Math. Soft.* **1980**, *6*, 563–580.
(17) Marimont, R. B.; Shapiro, M. B. "Nearest Neighbor Searches and the Curse of Dimensionality" *J. Inst. Math. Its. Appl.* **1979**, *24*, 59–70.
(18) van Marlen, G.; van den Hende, J. H. "Search Strategy and Data Compression for a Retrieval System with Binary-Coded Mass Spectra" *Anal. Chim. Acta* **1979**, *112*, 143–150.
(19) Rasmussen, G. T.; Isenhour, T. L.; Marshall, J. C. "Mass Spectral Library Searches Using Ion Series Data Compression" *J. Chem. Inf. Comput. Sci.* **1979**, *19*, 98–104.
(20) Smeaton, A. F.; van Rijsbergen, C. J. "The Nearest Neighbour in Information Retrieval. An Algorithm Using Upperbounds" *ACM SIGIR Forum* **1981**, *16*, 83–87.
(21) Murtagh, F. "A Very Fast, Exact Nearest Neighbour Algorithm for Use in Information Retrieval". *Inf. Technol.: Res. Dev.* **1982**, *1*, 275–283.
(22) Perry, S. A., unpublished MSc Thesis, Univeristy of Sheffield, 1982.
(23) Burke, J. M.; Rickman, J. T. "Bitmaps and Filters for Attribute-Oriented Searches" *Int. J. Comput. Inf. Sci.* **1973**, *2*, 187–200.
(24) Vallarino, O. "On the Use of Bitmaps for Multiple Key Retrieval" *ACM SIGPLAN Notices* **1976**, *11*, 108–114.
(25) Pfalz, J. L.; Berman, W. J.; Cagley, E. M. "Partial-Match Retrieval Using Indexed Descriptor Files" *Commun. ACM* **1980**, *23*, 522–528.
(26) Adamson, G. W.; Cowell, J.; Lynch, M. F.; McLure, A. H. W.; Town, W. G.; Yapp, A. M. "Strategic Considerations in the Design of a Screening System for Substructure Searches of Chemical Structure Files" *J. Chem. Doc.* **1973**, *13*, 153, 157.
(27) Perry, S. A.; Willett, P. "The Use of Inverted Files for Best Match Searching in Information Retrieval Systems", submitted for publication in *J. Inf. Sci.*