Algorithm5: A Technique for Fuzzy Similarity Clustering of Chemical Inventories

Thompson N. Doman,* John M. Cibulskis, Michael J. Cibulskis, Patrick Dale McCray, and Dale P. Spangler

G. D. Searle, 4901 Searle Parkway, Skokie, Illinois 60077

Received April 4, 1996[⊗]

Clustering of chemical inventories on the basis of structural similarity has been shown to be useful in a number of applications related to the utilization and enhancement of those inventories. However, the widely-used Jarvis−Patrick clustering algorithm displays a number of weaknesses which make it difficult to cluster large databases in a consistent, satisfactory, and timely manner. Jarvis−Patrick clusters tend to be either too large and heterogeneous (i.e., "chained") or too small and exclusive (i.e., under-clustered), and the algorithm requires time-consuming manual tuning. This paper describes a computer algorithm which is nondirective, in that it performs the clustering without manual tuning yet generates useful clustering results.

INTRODUCTION

"Clustering" may be generally defined as the partitioning of a set of objects into subsets on the basis of similarity. The technique finds broad applicability in many fields of study, as a means for partitioning large datasets into more tractable pieces. Biological taxonomy is a well-known example, in which there is a careful hierarchy of similarities, from the very broad similarity represented by Kingdoms, through the increasingly restrictive distinctions of Phylum, Class, Order, and Family, down to Genus and Species. This classification scheme is extremely important, since understanding the relationships between the millions of known living things would be nearly impossible without it.

In the field of chemistry, the datasets most commonly encountered are collections of chemical compounds. In some cases these collections are quite large. An example is the Chemical Abstracts Service Chemical Registry, which as of 1994 consisted of over 13 million substance records and was growing at the rate of 750 000 new substances per year.[1] With the advent of techniques such as combinatorial synthesis, these collections are expected to grow at unprecedented rates.

The proprietary chemical databases owned by pharmaceutical companies are rich sources of novel leads for new therapeutic agents. These databases often represent the labors of chemists over many years in diverse therapeutic areas. As such, these databases tend to be large and diverse collections of chemical compounds, sometimes numbering in the hundreds of thousands. There are a number of challenging tasks associated with the utilization and enhancement of these databases. First, the sheer size of the database may make pharmacological screening of the entire inventory impractical. In this case, it is desirable to generate a subset of the inventory which is (hopefully) representative of the inventory as a whole. Second, when evaluating chemical species for acquisition or synthesis, it is useful to know whether or not these new chemical entities overlap with chemistry already in the database. Finally, there are instances when it may be desirable to characterize the composition of these databases in terms of chemical classes which are represented therein. For each of these tasks, clustering by *structural* similarity may furnish a convenient solution.

The clustering algorithm proposed by Jarvis and Patrick[2] is widely used for chemical similarity clustering. While this method is quite robust and has a number of strong points, the Jarvis−Patrick method is also plagued by several shortcomings. In this paper a new algorithm for clustering, known as **algorithm5**, will be introduced. On the basis of clustering results from three example inventories it will be shown that algorithm5 obviates many of the shortcomings of the Jarvis−Patrick algorithm.

BASIC CONCEPTS

The basic principles involved in clustering chemical data by structural similarity have been discussed by Willett.[3] Initial clustering work at Searle was performed using the popular Daylight Clustering Package.[4] The Daylight Clustering Package consists of several programs representing steps in the clustering process:

Step 1. The molecules are "fingerprinted." This consists of exhaustively fragmenting the molecule, looking for all paths up to eight atoms (or seven bonds) in length. If a particular fragment is found, then a bit is set in a binary fingerprint. These fingerprints are "folded" or "hashed" in order to generate a fingerprint with a reasonable size and information density (typically 512−2048 bits).

Step 2. Nearest neighbors are calculated for each molecule. The similarity between molecules is calculated using the Tanimoto similarity metric: $T_{AB} = N_{A\&B}/(N_A + N_B - N_{A\&B})$ where N_A, N_B, and $N_{A\&B}$ are the number of bits set in fingerprints A, B, and A-and-B.

Step 3. The Jarvis−Patrick algorithm is used for clustering. In this scheme, two structures cluster together if (a) they are in each other's list of **J** nearest neighbors and (b) they share **K** of their **J** neighbors, where **K** <= **J**.

SHORTCOMINGS OF THE JARVIS−PATRICK ALGORITHM

The Jarvis−Patrick algorithm has a number of attractive features which make it very useful in clustering chemical structures. It is straightforward and very efficient. Implicit in the procedure is an automatic scaling based on compound

[⊗] Abstract published in *Advance ACS Abstracts,* October 15, 1996.

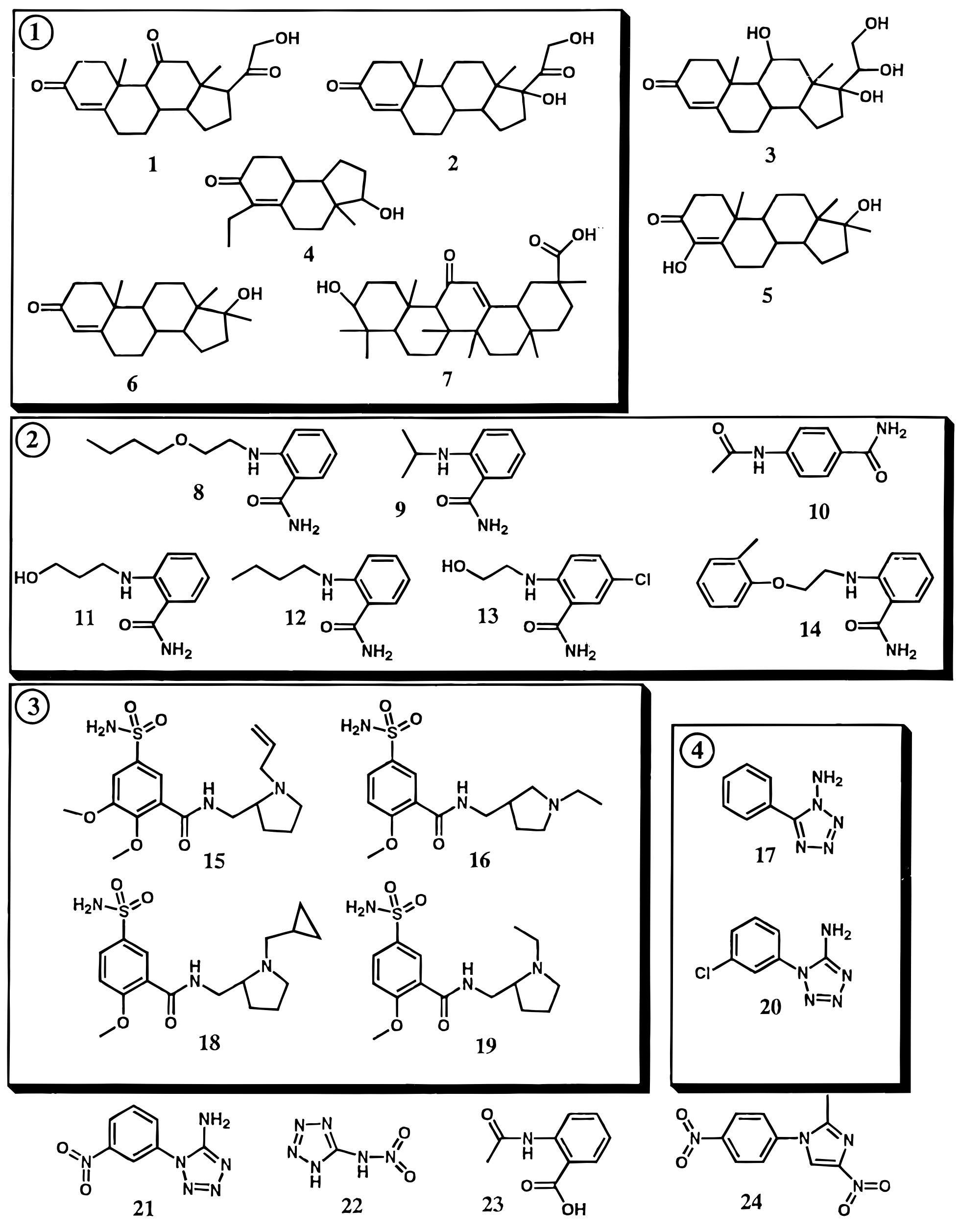


Figure 1. Jarvis–Patrick 8/9 clustering of inventory 1.

size. The cluster resolution may be adjusted by variation of **J** and **K**. However, this latter “advantage” may turn into a disadvantage for two reasons. First, the parameters **J** and **K** must be optimized. This optimization is often painstaking, especially for larger databases. Second, the requirement of these parameters introduces human bias into the clustering. Jarvis–Patrick requires the *user* to decide what clusters are present. A clustering algorithm requiring no user-intervention in the form of adjustable parameters would greatly enhance the utility of clustering. Such an algorithm would decide on its own what clusters are present. This would facilitate the straightforward comparison of clustering results on different databases.

In order to demonstrate clustering results, three inventories of chemical compounds have been chosen from the “Master File” collected by Hansch and Leo.[5] Inventory 1 (24 compounds) shown in Figure 1 was chosen by careful visual inspection of the Master File in order to generate a small group of compounds in which several obvious clusters are present. Inventory 2 (21 compounds) was chosen according to the following procedure. Several compounds were randomly chosen from the Master File. The 20 nearest neighbors for each of these compounds were then calculated using the Daylight Fingerprint and Tanimoto coefficients. The groups of 21 compounds (randomly chosen compound plus 20 neighbors) were then visually inspected, and the group which demonstrated the most significant change in structure from neighbor 1 to neighbor 20 was chosen. Inventory 2 is that group and is shown in Figure 2. This inventory consists of compound **25** and its 20 nearest neighbors. Note that the two most distant neighbors to **25**, compounds **45** and **35**, are structurally very different than **25**.

Inventory 2 was constructed in the expectation that it would challenge the Jarvis–Patrick algorithm, which sometimes tends to produce “chaining”, a condition in which two or more clusters join to form an unsatisfactorily heterogeneous cluster. Chaining results when a compound which is a hybrid of two or more clusters causes these clusters to coalesce. Jarvis–Patrick does not allow compounds to be

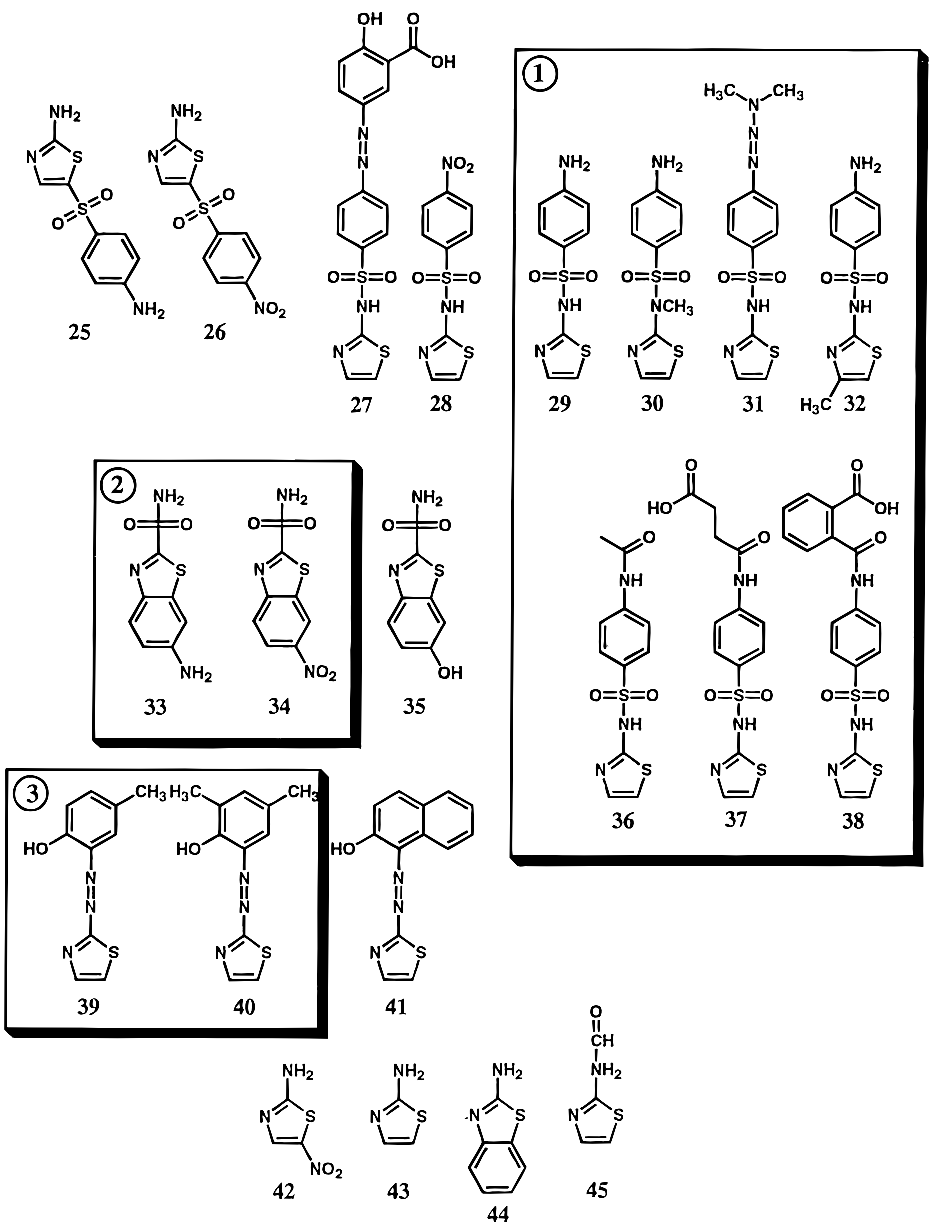


Figure 2. Jarvis–Patrick 8/9 clustering of inventory 2.

in more than one cluster, so the only alternative to chaining is underclustering.

Inventory 3, shown in Figure 7, was constructed by randomly selecting 24 compounds from the Master File. Visual inspection of this collection shows no clusters. This inventory was prepared in order to test whether or not one of the clustering algorithms would generate clusters where none actually existed.

Jarvis–Patrick was used to cluster inventory 1, and satisfactory clustering was found with **J** = 9, **K** = 8. These results are shown in Figure 1. Four clusters are obtained, while six compounds are found to be singletons. When these same **J**,**K** parameters are used in clustering inventory 2, there is noticeable underclustering. As shown in Figure 2, compounds **25** and **26**, which are extremely similar, do not cluster together. Compound **28** probably belongs to cluster 1, compound **35** should probably be included in cluster 2, and compound **41** might be included in cluster 3.

With these results in mind, the Jarvis–Patrick clustering was relaxed just slightly. The **K** parameter, 8, was held fixed, while the **J** parameter was ratcheted up a notch from 9 to 10. This change is among the most subtle perturbations which could be made. The results are shown in Figure 3. Now compound **28** is incorporated into cluster 1, compound **35** goes into cluster 2, and compound **41** enters cluster 3. These are all desirable results. However, while compounds **25** and **26** now cluster, they form a chained cluster with the other members of cluster 1. In searching for the root cause of this chaining, it is instructive to note that compounds **25** and **29** have identical empirical formulas and thus are comprised of the same types and numbers of atoms. In addition, they share a number of fragments in common and thus will have particular regions of their fingerprints which match. However, compound **25** is a thiazole with a sulfone moiety at the 5-position, while the thiazole in **29** is substituted at the 2-position with a sulfonamide group. These are significant changes which in the opinion of the authors suggests that **25** and **29** belong in different clusters, at least in the context of inventory 2. Thus, in the two experiments shown in Figures 2 and 3 we see the extremes of Jarvis–Patrick clustering: underclustering and chaining.

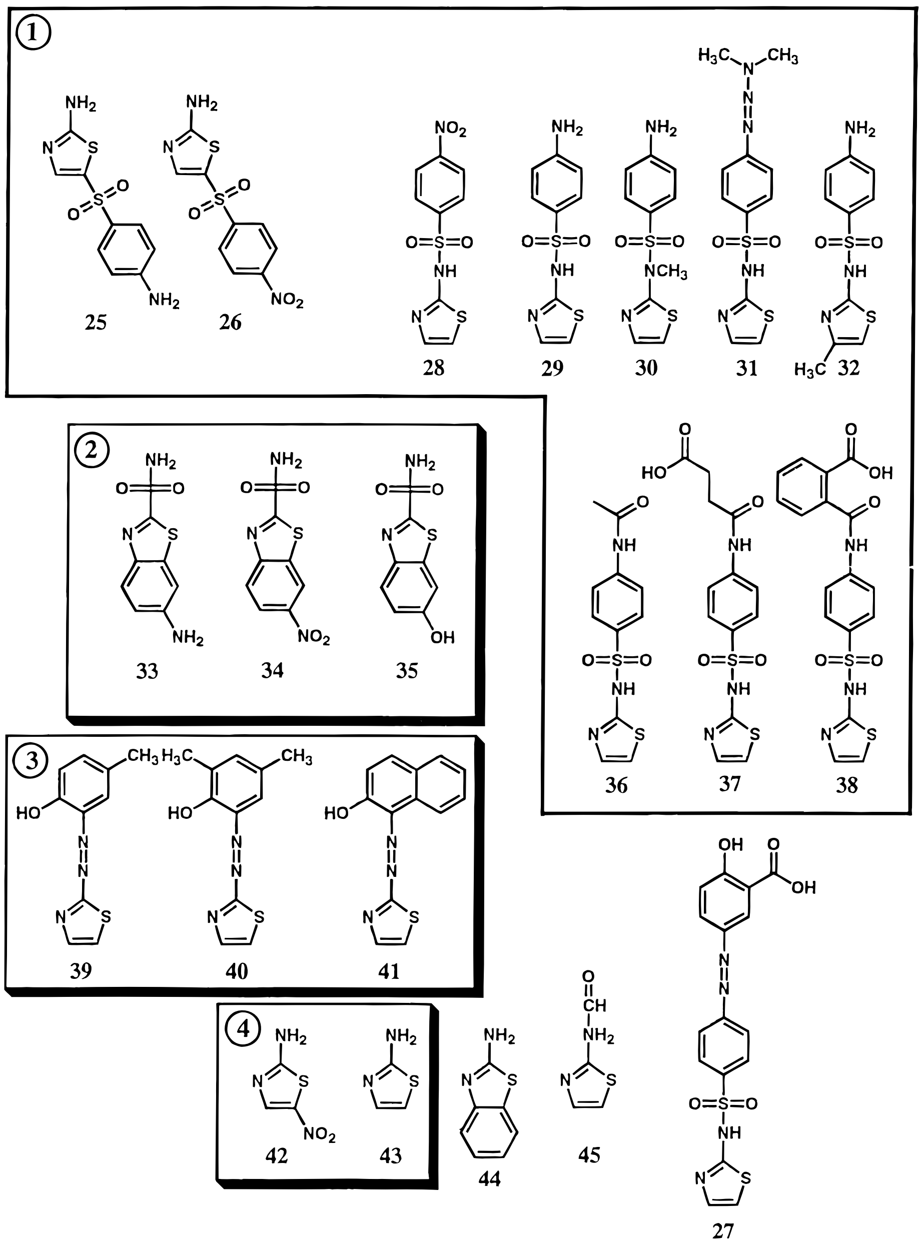


Figure 3. Jarvis–Patrick 8/10 clustering of inventory 2.

OBJECTIVES AND OVERALL APPROACH

When we began our work, we sought an algorithm which would circumvent the problems associated with Jarvis–Patrick clustering. In particular, we sought an algorithm which would

(1) generate reasonable clusters
(2) require no user-supplied parameters
(3) allow compounds to be in more than one cluster

In our hands, we found that Jarvis–Patrick performed admirably on areas of our inventories which were sufficiently "dense". For example, collections of steroids, which tend to have uniformly high Tanimoto coefficients, clustered together quite nicely. Considering also the speed of Jarvis–Patrick, we reasoned that it made sense to put Jarvis–Patrick to work on those areas of the chemical inventory with which it was most proficient. Then we planned to perform "fuzzy" clustering around the dense Jarvis–Patrick clusters. An introduction to fuzzy clustering may be found in the review by Dubes and Jain.[6] In the following sections we will introduce some terminology and then describe the new algorithm, which we call **algorithm5**. Then we will briefly describe how the algorithm has been implemented and used at Searle. Finally, we will present several examples of the performance of algorithm5 versus traditional Jarvis–Patrick clustering.

TERMINOLOGY

Although one thinks of clustering as acting upon the actual objects of interest (in this case, chemical compounds) the reality is that clustering actually takes place in a more abstract setting. The objects of interest are first subject to some sort of measurement or analysis which yields one or many numerical values. This collection of values (for a single object) is usually grouped into a vector referred to as the **feature vector** for that object. The collection of all these feature vectors is referred to as the **feature space**. This mapping from the objects to the features is usually not one-to-one; many objects may be mapped to the same feature vector. Clustering algorithms always do their work on the feature space, and as a final step the investigator converts these abstract clusters back to the concrete objects which they represent. For the purposes of this paper, we shall largely ignore this distinction. We shall allow ourselves to speak as if we are clustering the objects themselves and only draw the distinction when some point needs to be made.

Traditional clustering research has centered upon the notion of dividing the feature space into mutually disjoint sets with the property that each object belongs to some such set (cluster). From this point of view, a clustering of a feature space X is identical to what is known in set theory as a partition of the set X. That is, $\{\mathbf{C_1},...,\mathbf{C_k}\}$ is a **clustering** of **X** provided that

1. For each i, $\mathbf{C_i} \neq \varnothing$
2. If $\mathbf{i} \neq \mathbf{j}$, then $\mathbf{C_i} \cap \mathbf{C_j} = \varnothing$
3. $\cup \{\mathbf{C_1},...,\mathbf{C_k}\} = \mathbf{X}$

The disadvantages of using such a restrictive notion of clustering becomes clear when one returns to thinking about the objects being clustered. The reality is that when one is presented with a collection of chemical compounds a specific compound may not clearly fall into one specific cluster. Even if you have identified several clearly defined clusters of compounds, the compound in question may have similarities with compounds in several of them. It may not be easily decided which of two or more clusters in which to include the compound. Traditional clustering forces us to make that decision. We are faced with three alternatives: (1) choose one of the clusters to be the (unique) recipient of the compound, (2) create a new cluster to contain the compound, or (3) merge the clusters likely to contain the compound into a single cluster. (Traditional Jarvis–Patrick clustering chooses the third alternative. The result is that clusters have a tendency to end up too large and heterogeneous.) On closer examination, one comes to the realization that the problem lies with the dichotomous nature of set containment: an object either belongs to a set, or it does not. We would like to be able to say that the errant compound belongs (to some extent) to each of the clusters—perhaps more to one than to the others. The notion of such a weakening of the notion of set containment has been the subject of a considerable amount of research by Zadeh and co-workers.[7] The area of study is called **Fuzzy Set Theory**, and the sets are called **fuzzy sets**.

With each traditional set A (a subset of some underlying set **X**) there is associated a function referred to as the **characteristic function** of the set. This is defined to be that function $\chi_A : \mathbf{X} \rightarrow \{0,1\}$ which assigns the value 1 to an element of **X** provided that it belongs to the set A and the value 0 otherwise. Thus, knowing that $\chi_A(x) = 1$ tells us that $x \in A$; $\chi_A(x) = 0$ tells us $x \notin A$. In other words, knowing the characteristic function of the set allows us to know the set itself. We therefore shift our way of thinking about sets and alternatively define a **crisp subset of X** to be any function $\mathbf{c} : \mathbf{X} \rightarrow \{0,1\}$. Fuzzy subsets of **X** extend this idea in a natural way: rather than restricting ourselves to only taking the values 0 and 1, we allow the function to take on any value between 0 and 1. Thus, we define a **fuzzy subset of X** to be any function $\mathbf{c} : \mathbf{X} \rightarrow [0,1]$.[8] The value taken on by the function is thought of as the "degree of membership" or "certainty of membership" in the set. If $\mathbf{c}(x) = 1.0$, then it is "absolutely certain" that x belongs to the subset; if $\mathbf{c}(x) = 0.0$, then it is "absolutely certain" that it does not.

With this shift in paradigm, we may define the notion of a **crisp clustering** of **X**: $\{\mathbf{c_1},...,\mathbf{c_k}\}$ is a **crisp clustering of X** provided the following:

1. For each **i**, $\mathbf{c_i}$ is a crisp subset of **X**.
2. For each **i**, there is at least one $x \in \mathbf{X}$ with $\mathbf{c_i}(x) = 1$.
3. For each $x \in \mathbf{X}$, there is at least one **i** with $\mathbf{c_i}(x) = 1$.
4. If $\mathbf{i} \neq \mathbf{j}$, then for each $x \in \mathbf{X}$, either $\mathbf{c_i}(x) \neq 1$ or $\mathbf{c_j}(x) \neq 1$.

To extend this notion to fuzzy sets, we select a real number M with $0 < M < 1$. $\{\mathbf{c_1},...,\mathbf{c_k}\}$ is a **fuzzy clustering** of **X** provided the following:

1. For each **i**, $\mathbf{c_i}$ is a fuzzy subset of **X**.
2. For each **i**, there is at least one $x \in \mathbf{X}$ with $\mathbf{c_i}(x) = 1$. (sets in the clustering are nonempty)
3. For each $\mathbf{x} \in \mathbf{X}$, there is at least one **i** with $\mathbf{c_i}(x) \geq M$. (every element of **X** weakly belongs to some set in the clustering)
4. If $\mathbf{i} \neq \mathbf{j}$, then for each $x \in \mathbf{X}$, either $\mathbf{c_i}(x) \neq 1$ or $\mathbf{c_j}(x) \neq 1$.

By a **metric** (distance function) **on a set X** we mean a function $\mathbf{d}:\mathbf{X} \times \mathbf{X} \rightarrow$ R (the real numbers) which satisfies the following requirements:

1. For each $x, y \in \mathbf{X}$, $\mathbf{d}(x,y) \geq 0$. (positive)
2. $\mathbf{d}(x,y) = 0$, precisely when $x = y$. (definite)
3. For each $x, y \in \mathbf{X}$, $\mathbf{d}(x,y) = \mathbf{d}(y,x)$. (symmetric)
4. For all $x, y, z \in \mathbf{X}$, $\mathbf{d}(x,z) \leq \mathbf{d}(x,y) + \mathbf{d}(y,z)$. (triangle inequality)

In a sense, the Tanimoto similarity coefficient $\mathbf{T}:\mathbf{X} \times \mathbf{X} \rightarrow$ R is just the opposite of a metric. Indeed, if we define $\mathbf{d}(x,y) = 1 - \mathbf{T}(x,y)$ for all $x, y \in \mathbf{X}$, it may be shown that **d** is a metric on **X**. Since "nearness" and clustering concepts are usually described in terms of distance rather than in terms of similarity, we shall primarily use metric concepts in the remainder of this paper. The reader should keep in mind, however, that the distance we shall be using is just 1 minus the Tanimoto similarity coefficient.

We conclude with a few definitions that we shall require in the next section. Given $x_0 \in \mathbf{X}$ and $d_0 > 0$ define $\mathbf{N}(x_0,d_0) = \{x \in \mathbf{X} \mid \mathbf{d}(x,x_0) \leq d_0\}$. This is just the set of all elements of **X** which lie a distance less than or equal to d_0 from x_0. If $A \subseteq \mathbf{X}$, we define $\mathbf{d}(x,A) = \min\{\mathbf{d}(x,a) \mid a \in A\}$. This is the smallest distance from x to an element of the set A and is considered to be the distance from x to the set A. Finally, if $Y \subseteq \mathbf{X}$, let **card(**Y**)** (the cardinality of Y) denote the number of elements in the set Y.

THE ALGORITHM

In the description of the algorithm that follows we assume familiarity by the user with the Jarvis–Patrick clustering algorithm applied to chemical compound databases. The essence of the algorithm may be described as follows. Rather than trying to cluster the entire database at one time, we begin by restricting our attention to the "densest" part of the space. First we apply the Jarvis–Patrick clustering algorithm to this dense portion and obtain conventional (crisp) clusters. Then we perform fuzzy clustering by inspecting the remaining unclustered compounds and assigning them to any (one or more) crisp clusters which they may be near. Finally we identify those members of X which have little likelihood of belonging to any cluster defined thus far. From this restricted collection, we again choose the "densest" subset to cluster and proceed as before. We perform this process iteratively until no more clusters can be identified.

Preparatory Step. The algorithm relies upon two constants: a value $d_0 > 0$ and a compound count N_0. Once these constants were determined, they were used for all clustering activities and may be considered to be part of the definition of the algorithm. The constants must satisfy the following conditions:

1. It is considered to be allowable that two compounds lying a distance less than or equal to d_0 from each other be allowed to belong to the same cluster. In other words, two such compounds should be expected to exhibit similar chemical properties.
2. The algorithm requires[9] that $d_0 < 1/2$.
3. N_0 should be small enough so that it can be used for the maximum number of nearest neighbors to include in the nearest neighbor list for a Jarvis−Patrick clustering.
4. For the compound collections of interest, there should be a sufficient number of points x with the property that $\mathbf{card}(\mathbf{N}(x, d_0)) \geq N_0$. It is this collection of points which will be used for the initial clustering pass. Thus, the choice may also depend upon the available memory and computer resources.

Values of $d_0 = 0.15$[10] and $N_0 = 20$[11] were selected.

Make one pass through all compounds constructing the nearest neighbor table (with N_0 nearest neighbors) and recording $\mathbf{card}(\mathbf{N}(x,d_0))$ for each $x \in \mathbf{X}$.

The nearest neighbor table should be augmented with the following information for each compound:

1. A field which will hold the normal Jarvis−Patrick index of the cluster to which the compound might be assigned.
2. $\mathbf{card}(\mathbf{N}(x,d_0))$

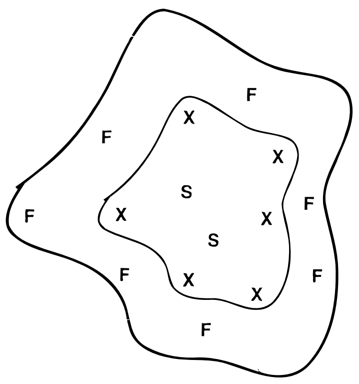
Initialization. Set $i = 0$ and let $\mathbf{Y_0} = \mathbf{X}$, $d_0 = 0.15$, and $N_0 = 20$.

Step 1. Choose a subset of the compound space which consists only of those compounds which have a sufficient number of "close" neighbors: Let $\mathbf{X_i} = \{x \in \mathbf{Y_i} \mid \mathbf{card}(\mathbf{N}(x,d_0)) \geq N_i\}$.

Step 2. Apply the Jarvis−Patrick algorithm to this subset using a number of nearest neighbors which would insure that the nearest neighbor list for each compound consisted only of "close" neighbors:

J−P parameters: Number of nearest neighbors to consider: N_i
Number of mutual nearest neighbors required: (3/4) N_i
(In implementation, this last value is replaced by the smallest integer greater than or equal to the computed value.)

No clusters containing only one compound should be retained. Let the resulting clusters be denoted by $\{\mathbf{C_0},...,\mathbf{C_n}\}$. The points in any of these sets are considered to be **crisply clustered**.



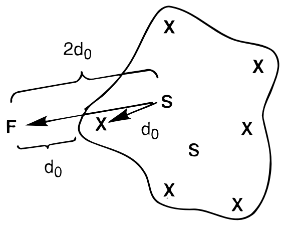


Figure 4. An idealized algorithm5 cluster shown in two dimensions.The symbol "S" represents a skeleton compound, "X" is a crisply-clustered but nonskeleton compound, and "F" represents fuzzily-clustered compounds. In the upper cartoon, the inner ring represents a crisp cluster, while the outer ring contains the entire fuzzy cluster. In the lower cartoon, the distance from an "S" compound to an "X" compound is 0.15 (in Tanimoto space), the distance from that same "X" to an "F" is also 0.15, and so the distance from the "S" to the "F" is (at most) 0.30 or $2d_0$. Thus, fuzzy compounds are approximately d_0 from the "edge" of crisp clusters.

Step 3. Create a fuzzy clustering of the entire database by assigning p's to each compound which describes the "degree of membership" to each of the clusters determined in step 2.

In other words, for each $\mathbf{j} = 0,...,\mathbf{n}$, we need to define a function $\xi_j : \mathbf{X} \rightarrow \mathrm{R}$ which satisfies

1. $\xi_j(x) \geq 0$
2. if x should be a member of $\mathbf{C_j}$, then $\xi_j(x)$ is "large".
3. $\xi_j(x)$ decreases toward 0.0 as the "distance" from x to $\mathbf{C_j}$ increases.

To achieve this, we proceed as follows:

For each cluster $\mathbf{C_j}$, determine a collection of "representative" points. The set of these points should be thought of as the **skeleton of** $\mathbf{C_j}$ and will be denoted by $\mathbf{S_j}$.

Let $T(x,y)$ denote the Tanimoto similarity coefficient.

To select this skeleton proceed as follows:

Let $\mathbf{L}$ denote a temporary list of the compounds in $\mathbf{C_j}$. This list $\mathbf{L}$ will be reduced as points are removed from it.

Choose $s_1 \in \mathbf{L}$ so as to maximize the number of compounds in $\mathbf{L}$ which are in s_1's nearest neighbor list.

Remove from $\mathbf{L}$ those points which are in s_1's nearest neighbor list.

Of these remaining points in $\mathbf{L}$ choose s_2 exactly as was done for s_1.

Continue until all points are gone.

Finally, if $x \in \mathbf{X_i}$, define $\xi_j(x) = \max\{T(x,s_k) \mid s_k \in \mathbf{S_j}\}$.

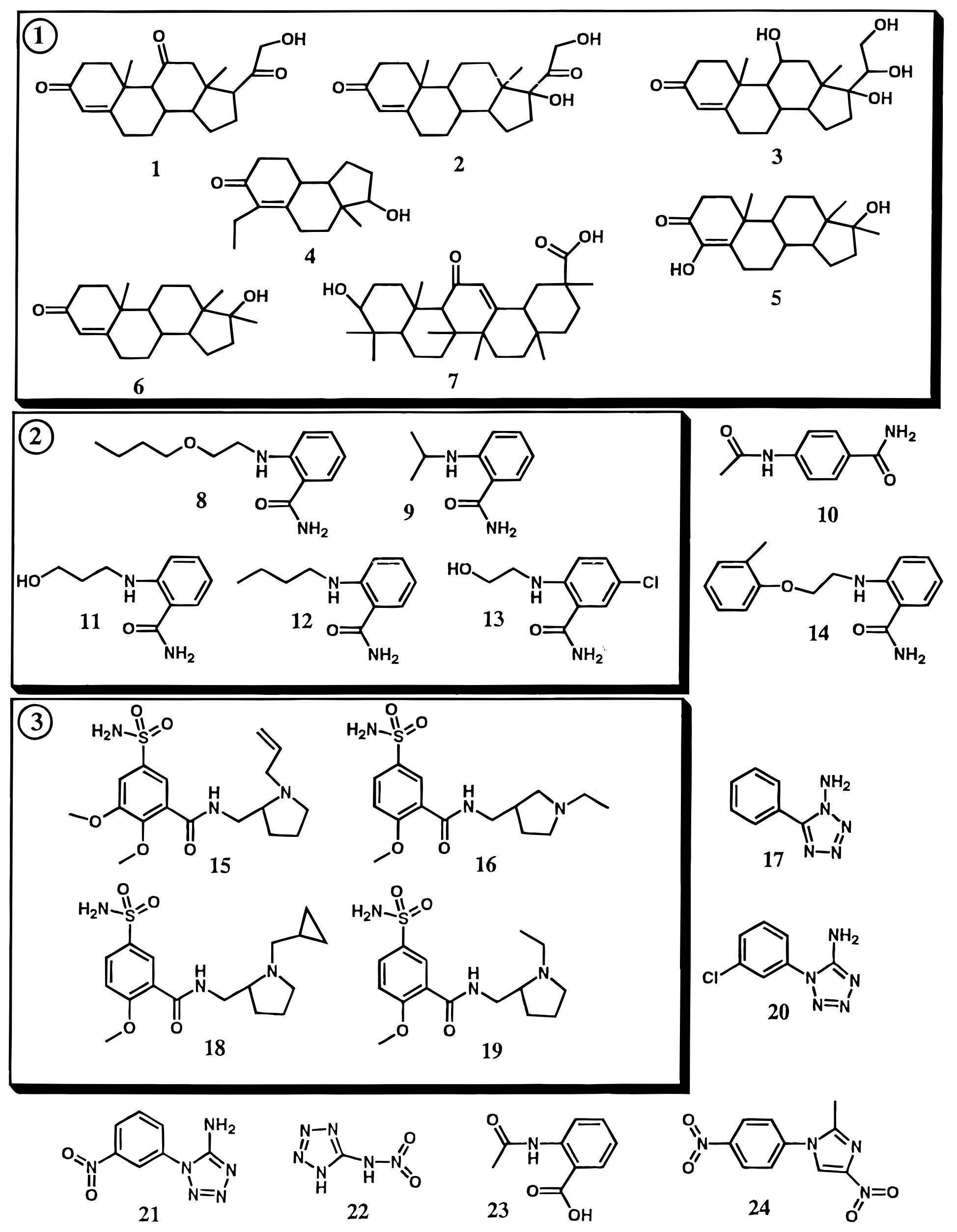


Figure 5. Algorithm5 clustering of inventory 1.

Note: If $x \in \mathbf{C_j}$, then x was removed from $\mathbf{L}$ at some step in the construction of the skeleton $\mathbf{S_j}$; say it was removed when s_k was selected. Then, since x is in the nearest neighbor list of s_k, it follows that $d(x,s_k) \leq d_0$ so that $T(x,s_k) \geq 1.0 - d_0$.

For each $\mathbf{j}$, save the skeleton $\mathbf{S_j}$ so that $\xi_j(x)$ can be computed in the future.

Step 4. Determine which compounds do not belong to any of the existing clusters on the basis of their p-values.

Set $p_0 = 1.0 - 2d_0$. (If a point lies a distance $2d_0$ from a point in the skeleton, then d_0 may be considered to be an approximation to the distance from the point to the cluster itself. This is shown by the diagram in Figure 4.) Those points which are not crisply clustered are said to be **fuzzily clustered** provided that there is some j for which $\xi_j(x) \geq p_0$.

Let $\mathbf{Y_{i+1}} = \{x \mid \text{for all j}, \xi_j(x) < p_0\}$. These are the points which have little likelihood of belonging to any of the clusters found thus far. They are the points which up till now are neither crisply nor fuzzily clustered.

By the note in step 3 above, it follows that if x is in some cluster $\mathbf{C_j}$, then $x_j(x) \geq 1.0 - d_0 > 1.0 - 2d_0 = p_0$, and so x will not belong to the set $\mathbf{Y_{i+1}}$.

Step 5. Determine a new (smaller) value for N_i to be used in step 1.

Set $N_{i+1} = N_i - 1$.

1. Scan through the augmented nearest neighbor table counting those compounds x that

   a. belong to $\mathbf{Y_{i+1}}$;

   b. satisfy $\mathbf{card}(\mathbf{N}(x,d_0)) \geq N_{i+1}$.

2. If (count $\geq$ 3% of $\mathbf{card}(\mathbf{Y_{i+1}})$) STOP. The value for N_{i+1} is determined. Increment i so that N_{i+1} becomes the new N_i and $\mathbf{Y_{i+1}}$ becomes the new $\mathbf{Y_i}$. Return to step 1.

3. Otherwise, set $N_{i+1} = N_{i+1} - 1$.

4. If ($N_{i+1} = 0$) perform one final pass through steps 1−4, then STOP. The clustering is concluded. All remaining points are singletons, not included in any clusters.

5. Otherwise, repeat starting at step 5.1.

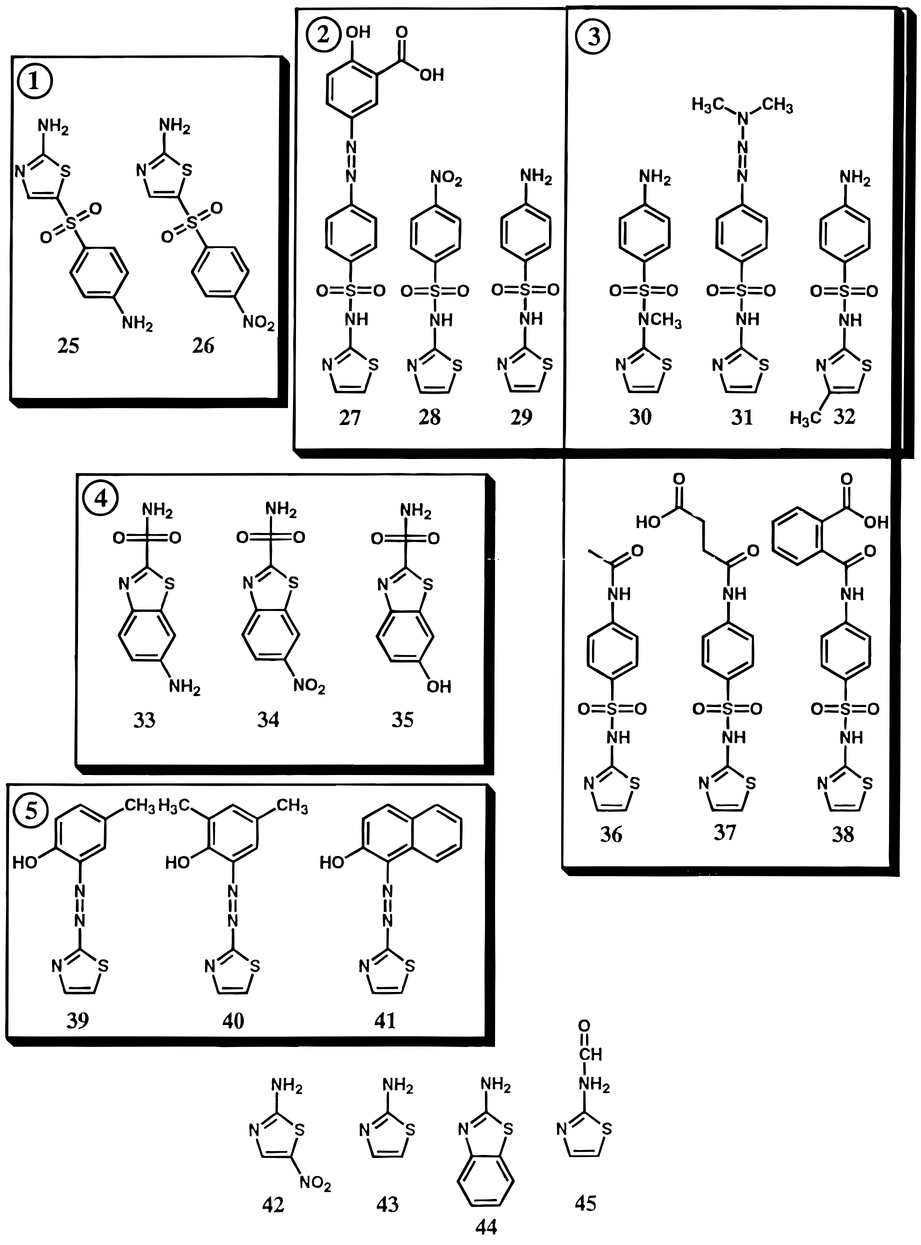


Figure 6. Algorithm5 clustering of inventory 2.

Let **Z** ={$x \in$ **X** | x is either crisply or fuzzily clustered}. It can easily be seen that with $M = p_0$, the set of ξ_j form a fuzzy clustering of **Z**.

IMPLEMENTATION AND USAGE OF THE ALGORITHM

Algorithm5 has been implemented as a series of "C" programs which run on Silicon Graphics unix machines. The algorithm is presently used at Searle as the method of choice in clustering chemical inventories. Inventories up to 300 000 compounds have been successfully clustered. An inventory of that size requires approximately 140 h of CPU time on a 75 MHz R8000 Power Indigo2machine. However, the slowest step, which is the creation of the nearest neighbor list, may be conveniently split and run in parallel on several machines. In our case we split this step up into five chunks of 60 000 compounds each, and, in so doing, we were able to reduce the elapsed time for the clustering to just over 40 h. We estimate that chemical inventories containing up to 500 000 compounds may be clustered with our present computer memory capacity of 128 megabytes per machine.

RESULTS AND DISCUSSION

Figure 5 shows the results of algorithm5 clustering of inventory 1. While this differs in certain minor respects from the Jarvis−Patrick clustering of this inventory shown in Figure 1, we do not deem the differences to be especially significant. However, the results of algorithm5 clustering of inventory 2, as shown in Figure 6, differ remarkably from the results obtained by Jarvis−Patrick clustering and shown in Figures 2 and 3. Figure 6 clearly demonstrates the advantages of algorithm5 clustering. In this case, compounds **25** and **26** cluster only with themselves. The benzenesulfonamides **27**−**32** and **36**−**38** now form two overlapping clusters, 2 and 3, instead of just one. This is a pleasing solution for a series of compounds with relatively smooth gradations in structural difference. Compounds **30**−**32** are fuzzily clustered into *both* clusters 2 and 3. The ability of

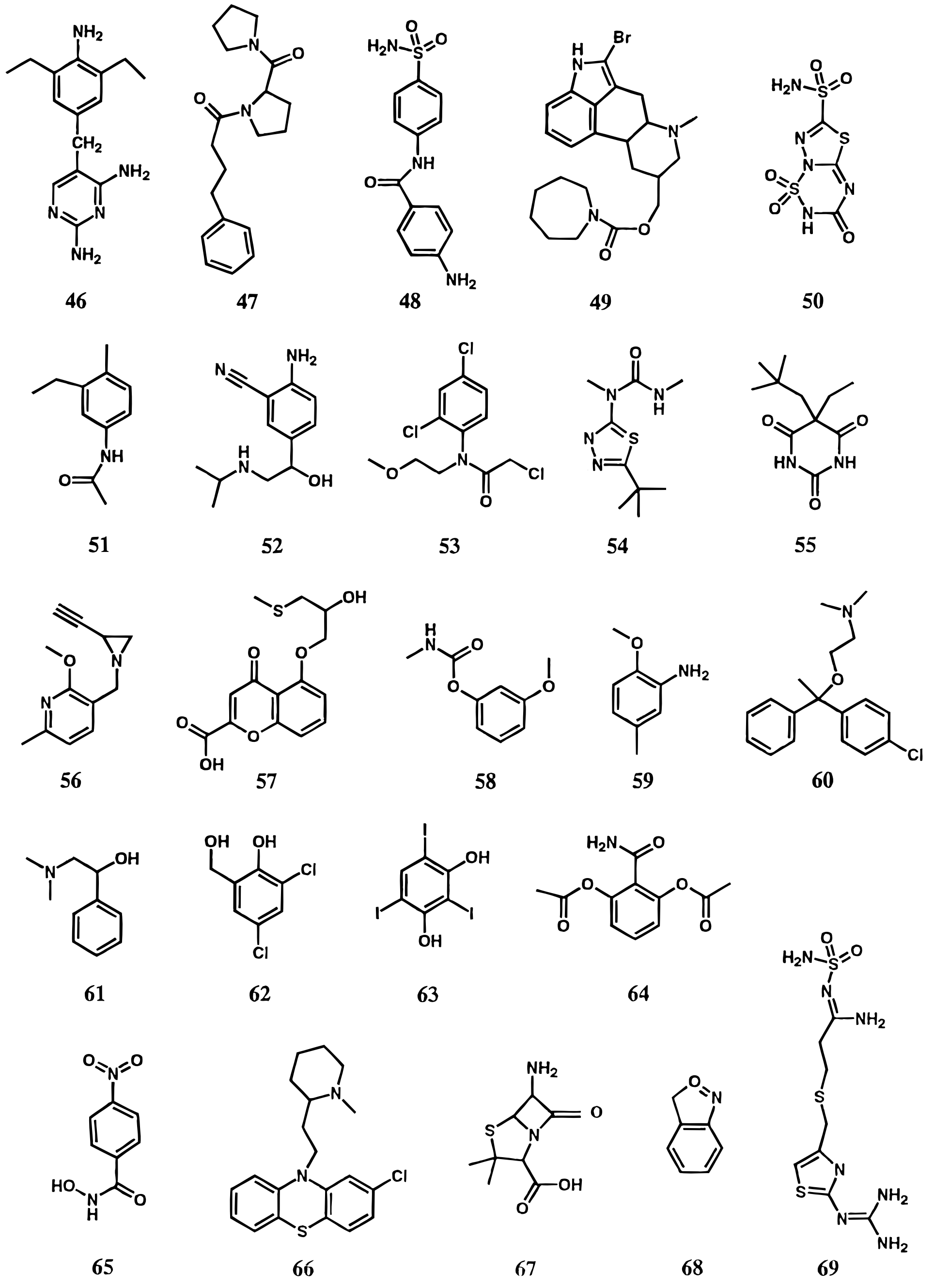


Figure 7. Inventory 3.

algorithm5 to place compounds into more than one cluster is a distinct advantage over Jarvis–Patrick.

One further comment about the differences between Jarvis–Patrick and algorithm5 is in order. In Figures 1 and 3 Jarvis–Patrick is shown to generate small (two-member) clusters, each denoted cluster 4, which are not generated by the analogous algorithm5 experiments. Jarvis–Patrick clustering, using the Tanimoto metric, scales automatically for the size of molecules. This is best understood by returning to the definition of the Tanimoto coefficient. This coefficient essentially corresponds to the number of "on" bits in common divided by the total number of unique (i.e., nonduplicated) "on" bits which are set. For smaller molecules, fewer fragments are present, and correspondingly fewer bits are set "on" in the fingerprint. This means that the denominator in the Tanimoto coefficient becomes smaller for smaller molecules, and thus a minor change in structure between two small compounds leads to a more dramatic change in the Tanimoto coefficient than a minor change between two larger compounds. This is completely intuitive. Returning to the two small clusters mentioned earlier, inspection of the Tanimoto coefficients for **17** and **20** ($T_C = 0.326$) and **42** and **43** ($T_C = 0.739$) shows that each member of the pair is relatively "distant" in Tanimoto space, especially in the case of the former pair. So from an algorithm5 perspective they are unlikely to cluster, and in this case they do not. This is not the case with Jarvis–Patrick, however. Jarvis–Patrick uses the Tanimoto value in constructing the nearest neighbor list, but after that step does not explicitly consider the "distance" between molecules. Thus, even quite small molecules such as **17** and **20**, and **42** and **43**, are clustered together by Jarvis–Patrick. So the question becomes, which approach is correct? Clearly, each of the compounds, **17**, **20**, **42**, and **43** is a small molecule (molecular weight less than 200) with correspondingly minimal functionality. In permuting compound **17** into compound **20**, the substitution position on the tetrazole is changed and a chloro substituent is added to the benzene ring. In **42** and **43**, the molecules consist essentially of a single heterocycle, with the latter containing an additional nitro group. Unlike Jarvis–Patrick, algorithm5 considers these changes to be too drastic for clustering. It is left to the reader to decide which method is more appropriate for his or her applications involving smaller molecules. On a final note, it is curious that Jarvis–Patrick clusters **20** with **17** instead of with **21**. Even the Tanimoto coefficient for **20** and **21** ($T_C = 0.726$) suggests **21** is much "closer" to **20** than is **17**. Again, without an explicit distance consideration built into the clustering algorithm, these things happen.

A reviewer has noted that some workers are not interested in the exact makeup of clusters, and thus Jarvis–Patrick is not "painstaking" as mentioned before since the parameters do not have to be carefully optimized in those cases where the exact complexion of clusters is not critical. This reviewer has suggested that we study an inventory with no obvious clusters and compare Jarvis–Patrick with algorithm5. Any clusters generated from such an inventory are obviously "wrong".

Inventory 3, shown in Figure 7, is such a collection. Using Jarvis–Patrick clustering, we found that the parameters $\mathbf{J} = 8$, $\mathbf{K} = 10$ caused 16 compounds to cluster together. Using $\mathbf{J} = 8$, $\mathbf{K} = 9$ was also unsatisfactory because five compounds clustered together. It was only when we went to stringent clustering conditions like $\mathbf{J} = 8$, $\mathbf{K} = 8$ that we obtained the desired result; that is, no clusters. The condition $\mathbf{J} = \mathbf{K}$ means this is the most stringent possible Jarvis–Patrick clustering. In our work, we have never used such stringent clustering conditions for conventional inventories. By contrast, algorithm5 finds the desired result (no clustering) without any user supervision.

In conclusion, the Jarvis–Patrick algorithm, which is a valuable tool for chemical similarity clustering, suffers from several drawbacks. These include the need for (often painstaking) human optimization of parameters and the restriction to only one cluster per compound. A new algorithm, algorithm5, does not require any user-supplied parameters and allows compounds to be clustered into more than one cluster. This eliminates the "chaining" problem common with Jarvis–Patrick clustering without going to the other extreme of under-clustering.

Supporting Information Available: The Tanimoto similarity matrices for each of the inventories 1–3 is available (9 pages). See any current masthead page for ordering and Internet access instructions.

REFERENCES AND NOTES

(1) Ross, L. R. CAS Aims to Keep Customers By Improved Service, Pricing, Management. *Chem. Eng. News* **1994**, *72*, 44, 19–23.

(2) Jarvis, R. A.; Patrick, E. A. Clustering Using a Similarity Measure Based on Shared Near Neighbors. *IEEE Transactions on Computers* **1973**, *C-22*, 1025–1034.

(3) Willett, P. *Similarity and Clustering in Chemical Information Systems*; Research Studies Press:New York, 1987.

(4) Weininger, D.; Delany, J. *Clustering Package User's Guide;* In *Daylight User's Manual Release 4.41*; Daylight Chemical Information Systems, Inc.: Irvine, CA, 1995.

(5) Hansch, C.; Leo, A.; Hoekman, D. *Exploring QSAR. Hydrophobic, Electronic, and Steric Constants*; American Chemical Society: Washington, D.C., 1995.

(6) Dubes, R.; Jain A. K. Clustering Methodologies in Exploratory Data Analysis. *Adv. Comput.* **1980**, *19*, 113–228.

(7) Zadeh, L. A. Fuzzy Sets. *Inf. Control* **1965**, *8*, 338–353.

(8) All of the standard set operations have been defined for fuzzy subsets and have been extensively studied. For a discussion of these concepts see reference 7.

(9) d_0 is required to be $\leq 1/2$ because a later step in the algorithm requires the distance $2d_0$, which is nonsensical for $d_0 > 1/2$.

(10) The value 0.15 used for d_0 was found by a manual optimization process. This consisted simply of adjusting this parameter slightly, clustering a set of small test inventories, then visually inspecting the results, and iterating until these became satisfactory. We have found this value to be optimal for all chemical inventories which we have studied to date, and so we consider d_0 to be an unchanging "implementation parameter". Therefore, in claiming that the algorithm does not require manual tuning, we are referring to manual tuning done by end-users of the algorithm. Coincidentally, Brown and Martin observe that compound pairs related by a Tanimoto coefficient of 0.85 or greater show high probabilities of sharing similar bioactivities. This corresponds to a distance of 0.15, the exact value we use for d_0. See: Brown, R. D.; Martin, Y. C. Use of Structure-Activity Data to Compare Structure-Based Clustering Methods and Descriptors for Use in Compound Selection. *J. Chem. Inf. Comput. Sci.* **1996**, *36*, 3, 572–584.

(11) In our implementation, cardinalities up to 300 have been saved. So for the value N_i in step 2 we calculate min(neighbors saved, **card**).

CI960361R