ELSEVIER

Use of Markush structure analysis techniques for descriptor generation and clustering of large combinatorial libraries

John M. Barnard,* Geoff M. Downs,* Annette von Scholley-Pfab,* and Robert D. Brown†

*Barnard Chemical Information Ltd., Stannington, Sheffield, United Kingdom
†Molecular Simulations Inc., San Diego, California, USA

This article outlines a system in which a Markush structure representation is used for analysis of a combinatorial library, giving savings in terms of storage and processing requirements when compared with an enumerated library. The Markush representation used is described, along with the way in which it can be built from a reaction- and precursor-based description of a library. The process used to generate SMILES, and structure fingerprint and calculated property descriptors for each specific molecule in the library is discussed. Comparative performance figures are given for the Markush approach and various standard fingerprint generation programs based on the enumerated members of the library; the former shows time savings of a couple of orders of magnitude. The use of these rapidly generated fingerprints for clustering of large combinatorial libraries is described. © 2000 by Elsevier Science Inc.

Keywords: clustering, combinatorial library, descriptor generation, fingerprint, Markush structure

INTRODUCTION

Methods for representing libraries[1] can be divided into two groups: those that are based on the reactions and precursors used to synthesize the library members, and those that explicitly describe the reaction products, which are members of the library. In the latter case, either all the products can be individually enumerated, or a Markush (generic) structure can be used as a compact representation of them. All these methods have advantages and disadvantages in particular situations. Leland et al.[2] have drawn attention to one useful distinction between them, namely, that reaction-and-precursor and Markush-based approaches have space requirements based on the sum of the numbers of building blocks in each reactant pool (Σn_i), whereas storage of enumerated products has space requirements based on the product of these numbers (Πn_i). The former thus avoid the combinatorial explosion inherent in combinatorial libraries, which can be of considerable importance in reducing computational requirements.

A reaction-based approach represents the product set implicitly rather than explicitly, but has the advantage of specifying how the library was actually made, rather than what it was intended to contain (but might not do, because some of the reactions did not work as intended). It is a sum-based representation and thus fairly compact, but sophisticated software may be required to identify the products themselves. This will involve using the reaction description to clip off the leaving groups from the reactants and to connect them together in the appropriate orientations.

Interconversion of different library representations is an essential feature of most software systems. Until recently, the predominant approach has been to use Markush structures for input, with automatic generation of the enumerated products. More recently, there has been a move toward reaction-based input, again with automatic enumeration of the products. Reaction-based input is arguably more natural and convenient, because software can be linked directly to reagent databases. The convenience of Markush structures for display, and their potential advantages for library analysis and design, mean that conversion of reaction-based input to Markush storage is likely to become increasingly important and is discussed later. Conversion from Markush structures or enumerated products to reaction-based representations has attracted little attention to date, although there has been some interest in building

Corresponding author: Geoff M. Downs, Barnard Chemical Information Ltd., 46 Uppergate Road, Stannington, Sheffield S6 6BX, United Kingdom. Tel.: 44(0)1909-515-444; fax: 44(0)1909-774-074. *E-mail address*: downs@bci12.demon.co.uk (G.M. Downs).

Journal of Molecular Graphics and Modelling 18, 452–463, 2000
© 2000 by Elsevier Science Inc.
655 Avenue of the Americas, New York, NY 10010

1093-3263/00/$–see front matter
PII S1093-3263(00)00067-X

Markush display representations from sets of individual molecules.[3]

When analyzing a combinatorial library, perhaps to design an appropriately diverse one, or to ensure that its members meet various predefined constraints, it may be useful to work either in reactant space or in product space. Some considerations obviously are based purely on reactant properties, such as their chemical compatibility, or availability/price. Many groups have found it important to analyze the products also,[4] for instance, to obtain suitable subsets based on diversity or properties (such as logP and molecular weight). A number of papers have been published recently[5-9] in which Genetic Algorithms or Simulated Annealing techniques have been used to optimize reactant pool subsets on the basis of the resulting product set diversity, among other possible factors.

Work involving analysis of library products thus far has been largely based on enumeration of the full set of products, with property values for the individual molecules being calculated separately. This is the simplest and most obvious approach if relatively small libraries are involved, and it is appropriate for one-compound-per-bead type synthesis, where the library members are individually characterized. It has the additional advantage of giving a direct interface to existing software for compound registration, QSAR analyses, etc. However, the storage, input/output (I/O), and processing requirements of the enumerated library members depend on Πn_i, and may become impractical for large virtual libraries, such as those used during computer-assisted library design. A modification of this approach, in which the individual molecule representations are built "on the fly," analyzed immediately and discarded, has the advantage of saving on I/O and storage of the enumerated products, but offers little or no reduction in processing time. A third approach, involving direct analysis of the Markush representation, saves on storage, I/O, and processing, and is the main subject of this article. A less rigorous version of this approach, in which property values for individual molecules are approximated by summing contributions from the relevant precursor molecules, with a "correction term" taking account of the library scaffold and the changes involved in the synthesis steps, has been used by some other groups.

Following initial research, which demonstrated the effectiveness of the approach,[10,11] we have implemented a software system for Markush-based analysis.[12] Figure 1 shows its main components. The system currently includes features to build an internal Markush representation from a variety of input formats, which can be used to enumerate members of the library and/or for rapid generation of their structure fingerprints and calculated physicochemical property values and topological indices. The descriptors also can be used to cluster the members of the library and to generate certain composite descriptors of the library.

MARKUSH REPRESENTATIONS

Markush structures are sum-based representations and thus are very compact. They comprise a core to which the R-groups (substituent groups) are attached. In a Markush structure with three R-groups, each with 100 possible values, attached to the core, the Markush structure stores only 1 + 100 + 100 + 100 = 301 fragments to represent 100 × 100 × 100 = 1 million molecules. However, in their simplest form, Markush structures assume that each R-group is independent of the others. There may be situations in which, for synthetic feasibility or other reasons, some alternatives for R1 cannot occur with some alternatives for R2—in other words, the library is not strictly combinatorial, but is "nonregular." Simple Markush structures may therefore give misleading or inaccurate representations of product sets, although more sophisticated Markush representations can avoid most of these problems. One way to deal with the problem of nonregular libraries is to tabulate the permissible combinations of values for each R-group, although a possible disadvantage of this solution is that the size of the table will depend directly on the number of products (but the individual building blocks need only be stored once); listing forbidden combinations might be more compact.

A related problem occurs where the values taken by one R-group are related directly to those taken by another. An example of this[13] is where two multiply substituted dienes undergo a Diels-Alder cyclization with a single asymmetric dienophile, as shown in Figure 2.

Because each diene can react in two orientations, there are four possible products. However, the simple Markush structure shown, with all the R-groups independent, implies 3 × 3 × 4 × 4 = 144 different products. A way around this problem is to make a distinction in the Markush structure between substituent groups attached to the core and the actual structural variables (which correspond to the reactant pools used in the synthesis). Several substituents then can be defined together as a single structural variable, using separate ("dot-disconnected") fragments for each separate substituent. In Figure 3, all four substituent groups in the Diels-Alder product Markush are defined together, thus covering the actual set of products correctly.

This solution goes a considerable way toward actually enumerating all the alternative products and thus has failed to save much storage space. For example, the groups U, V, W, X, Y, and Z are being stored twice each. However, the point is that this enumeration is only being done when absolutely necessary to represent the product set accurately, and then only to the extent necessary. If there was another R-group, independent of the four shown, this still could be shown separately in the Markush, without the need to enumerate all of its combinations with the other four.

The same approach, along with a doubly connected R-group, also can be used where cyclic dienes are used in the reaction, resulting in a variably bridged ring system, and this is shown in Figure 4. In some situations, it may be useful to define R-groups with two disconnected components in some alternatives and as a linking group in other alternatives.

More complex Markush structures are possible, involving nested R-groups and variation in attachment position, or in the number of head-to-tail repetitions of a linking group, or even in the use of "homologous series" expressions (e.g., R1=alkyl C1–4) to define parts of the structure. All these occur in chemical patent claims[1,14] and are handled by Markush storage and retrieval systems for patents, such as MARPAT and Markush DARC. At present, combinatorial libraries are largely confined to variation in substituent groups, and in any case the other types of variation usually can be shown (perhaps with less than optimum efficiency) by enumerating alternative substituent groups. However, as more complicated reactions are used in library synthesis, informatics systems may be needed that can handle them

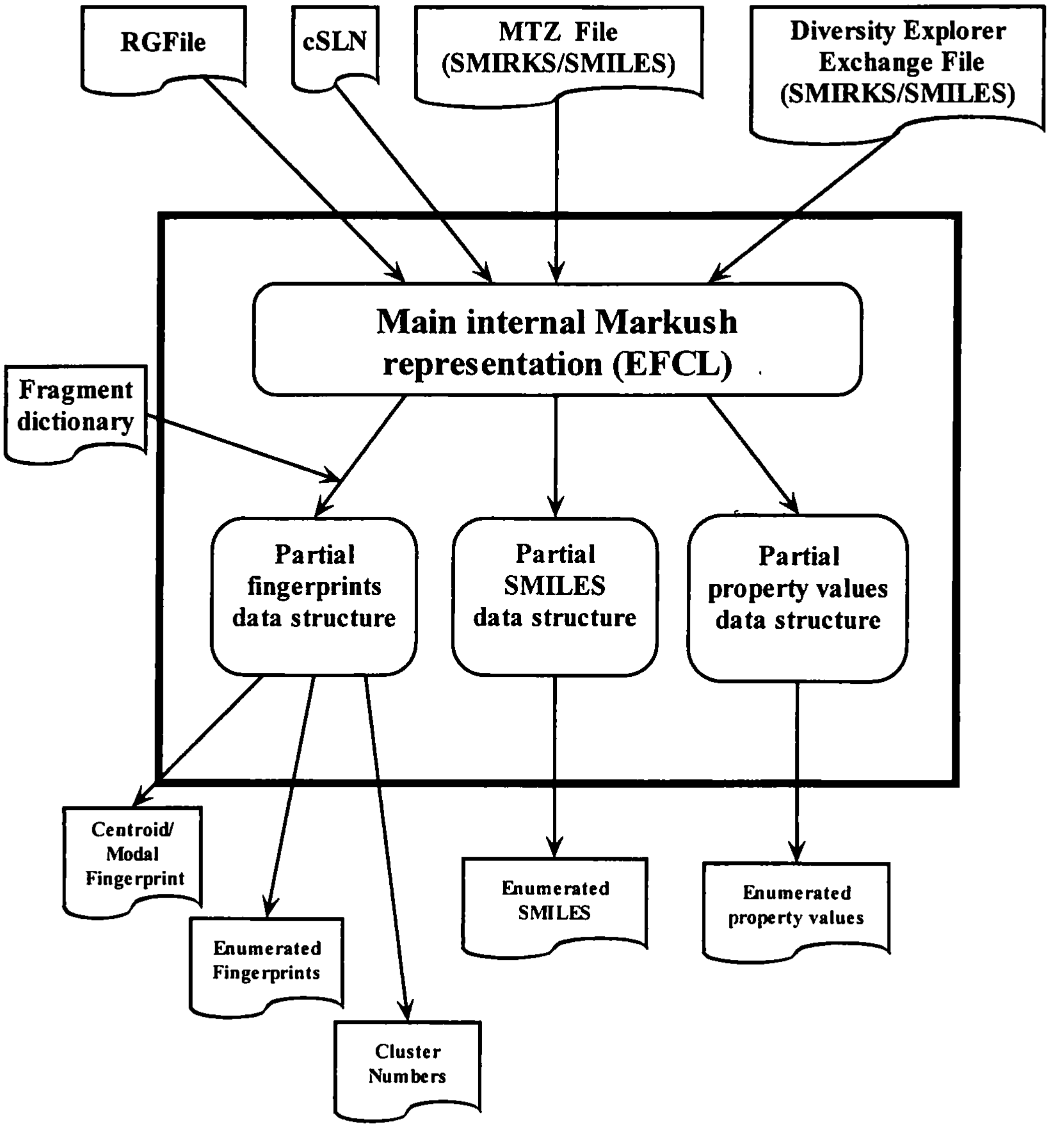

RGFile
cSLN
MTZ File (SMIRKS/SMILES)
Diversity Explorer Exchange File (SMIRKS/SMILES)
Main internal Markush representation (EFCL)
Fragment dictionary
Partial fingerprints data structure
Partial SMILES data structure
Partial property values data structure
Centroid/ Modal Fingerprint
Enumerated Fingerprints
Cluster Numbers
Enumerated SMILES
Enumerated property values

Figure 1. Main components of the Markush-based combinatorial library analysis system.

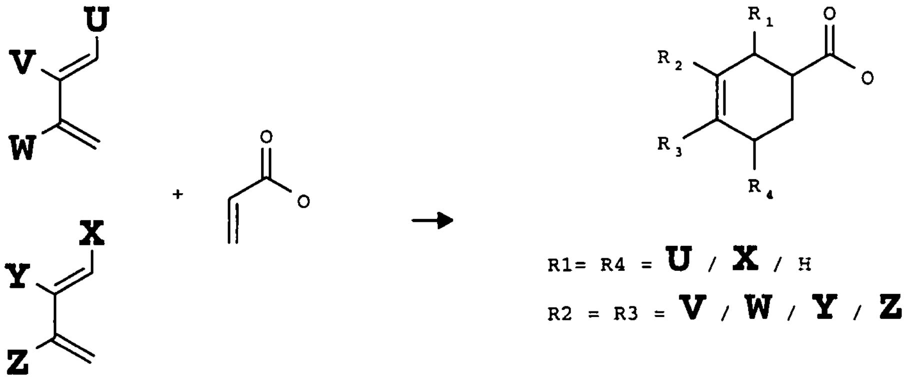


Figure 2. Schematic Diels-Alder cyclization reaction, in which a simplistic Markush representation with independent R-groups implies more products than are actually generated.

effectively and efficiently. Although combinatorial libraries may not add any fundamentally novel means of expression to the Markush structure types found in patents, Markush representations for libraries may introduce complexities not normally encountered in patents. This can apply particularly where the library involves diversity in the scaffold, such as can be obtained using multicomponent syntheses. In these cases, the Markush core may be quite small and may involve

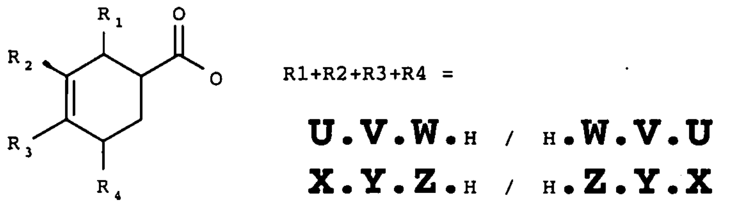


Figure 3. Markush representation of the Diels-Alder cyclization products from Figure 2, which uses combined R-group definitions to indicate the correct number of products. See text for discussion.

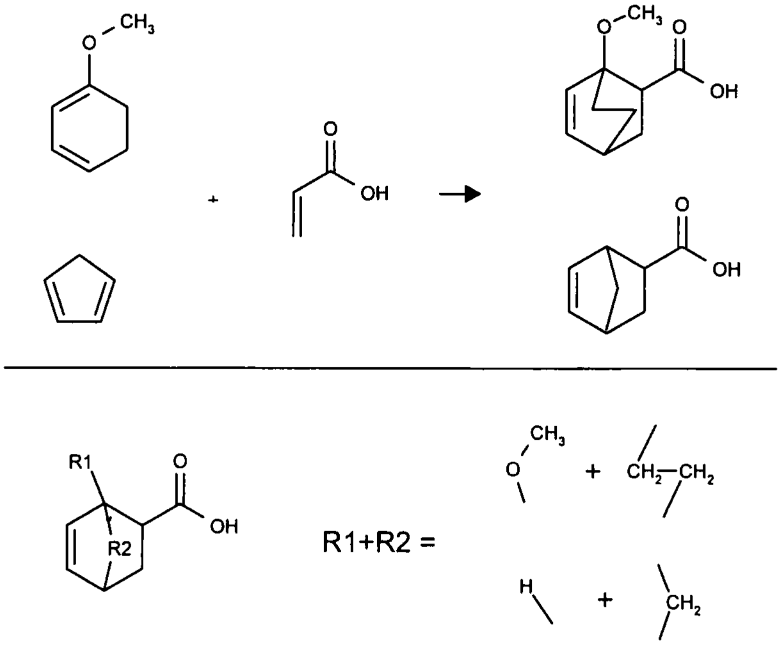


Figure 4. Diels-Alder cyclization reaction involving cyclic dienes and a Markush representation of the product set, which uses combined R-group definitions to generate the correct products.

several disconnected segments linked in various ways by multiply connected R-groups (possibly directly connected to each other). The definitions of these may either "link through" to form rings (in some library members) or remain as separate acyclic substituent groups (in others). Ultimately, a Markush structure can be made up of a "vestigial" core, consisting of a single unattached R-group, which is defined by the enumerated specific molecules comprising the library members. Of course, such a "Markush" representation (for what would be, effectively, a noncombinatorial library) would offer no advantages over the simple enumeration approach.

Some additional complications can arise with the representation of stereochemistry in Markush structures, as the values taken by otherwise unrelated R-groups can determine whether or not a particular atom is chiral or not (particularly where hydrogen is an alternative for an R-group), and combinations of R-groups can affect the possibility of diastereomer formation. Because the descriptors we currently are generating are not affected by stereochemistry, we have not thus far incorporated stereochemical features into our Markush representation; however, suitable extensions will clearly be necessary if structure searching capabilities are added or if 3D descriptors are to be generated. In principle, such extensions should not present particular problems, although the representation will need to indicate the particular R-group combinations that do or do not give rise to chiral centers in the specific molecules in the library.

BUILDING THE INTERNAL MARKUSH REPRESENTATION

The internal Markush representation used for combinatorial libraries is based closely on the Extended Connection Table Representation (ECTR) developed at Sheffield University for representation of Markush structures from chemical patents,[15] and for this reason is called the "ECTR for Combinatorial Libraries" (EFCL). It has the structure of an AND/OR logic tree, in which the nodes (called Partial Structures) represent the building blocks of the library, and the edges (called "Gates") indicate the logical and positional ways in which the Partial Structures can be connected together in the specific molecules that constitute the library. Each Partial Structure consists of a simple atom-bond connection table, with special atom types used to indicate R-groups, and the ends of "apical bonds" (bonds to atoms in other Partial Structures). The Gates contain information listing the alternative Partial Structures used to define each R-group and the relevant atom numbers for bonds between Partial Structures. The Gates allow algorithms to be used, which can visit each Partial Structure in turn, and which can trace atom paths from one Partial Structure to another [both from a "parent" into the alternative Partial Structures ("children") for its R-groups, and from a child into its parent]. The data structure for the EFCL is defined in the C programming language using a series of dynamic arrays and pointers, and it is held in memory only while needed for analysis. The logical structure of the EFCL tree can be arbitrarily complex, involving unlimited connections between Partial Structures and unlimited nesting of R-groups. Figure 5 shows a somewhat enhanced version of the bridged-ring Diels-Alder library from Figure 4, and Figure 6 shows the structure of the corresponding EFCL.

Identical EFCL representations can be built from a variety of input formats. The software currently is able to read input from MDL's RGfiles,[16] Tripos's Combinatorial Sybyl Line Notation (cSLN),[17] and two reaction-based file formats discussed later. RGfiles and cSLNs present no particular problems because the input file is already a Markush representation and its logical organization is essentially the same as that of the EFCL. The input procedure is one of reformatting connection tables and building certain housekeeping data useful for subsequent analysis.

Reaction-based input is more complex. Several companies[18] now provide software that allows input of combinatorial libraries described by means of generic reactions and sets of precursor molecules, which in principle could be used to build Markush representations in our system. Our current input module utilizes the Daylight Reaction Toolkit. Leach et al.[19] recently described a language called MTZ ("Molecule—Transform—Zap") that can specify a library by means of sets of reagent molecules (SMILES), generic reaction transforms (SMIRKS), and by-product substructures (SMARTS) to be eliminated (zapped). In the ADEPT system developed by Leach et al., a Daylight Toolkit program is used to enumerate the individual members of a library that has been input using the MTZ language. A similar approach is taken in the Library Specification Module of MSI's WebLab Diversity Explorer program.[20] Our software is able to use either the MTZ language or a related format (also based on SMILES and SMIRKS notations) derived from WebLab Diversity Explorer to build the EFCL directly.

SMIRKS is a linear notation developed by Daylight Chemical Information Systems Inc., which provides a very sophisticated means of specifying a generic reaction, in which the required environment of the reacting group can be defined in some detail. It is an extension of Daylight's SMARTS substructure pattern language and specifies the substructures required on the reactant side and the substructures formed from the same atoms on the product side. Unique atom class designators are used to map the atoms between the two sides. A disadvantage is that SMIRKS is difficult to encode manually, although a number of Daylight users have written software with varying degrees of sophistication to generate it from graphical input.[19,21] The Reaction Toolkit allows a SMIRKS transform to be applied to a set of specific reactant molecules, generating the specific products that they form. This can be done for every possible combination of reactants from different pools to generate a library of specific products, the products at each reaction step forming one of the reactant pools at the next step. However, because of the combinatorial explosion involved, this process can be very slow.

In building the EFCL from reaction-based input, we have adapted this process, incrementally building the Markush representation by adding one R-group to the core at each reaction step (or rather, for each independent pool of reactants). At each step, atoms included in the SMIRKS generic reaction pattern (which, by definition, must occur in all reagent molecules in the pool) are added to the Markush core. Other atoms are replaced by an R-group node, and a series of new Partial Structures is created to define the R-group, each containing the atoms from one of the reagent molecules, excluding those atoms that are part of the required substructure pattern. Where a reactant pool contains only a single molecule, all its relevant atoms can be added to the Markush core, and no R-group is necessary. In some cases, the required substructure may occur in more than one place in a single reactant molecule, and this may result in more than one Partial Structure being created from that molecule, although special provision is needed to exclude symmetrically equivalent occurrences. In practical implementation, a number of complications can arise, for example, where the reacting group in the "set of products formed so far" (i.e., in the Markush structure) is not in the Markush core, or where an intramolecular reaction takes place between different Partial Structures in the EFCL.

After the EFCL initially has been built from whatever input format is being used, it is "optimized" to make subsequent

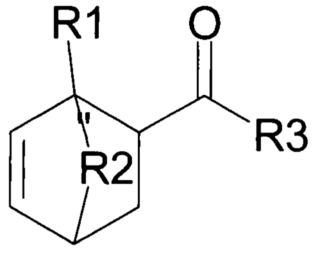


Figure 5. Depiction of a Markush structure, whose corresponding EFCL is shown in Figure 6. Note that R1 and R2 are defined together as a single variable, with two alternatives, one of which involves a nested R4 on the acyclic side-chain (R1). R3 is independent of the other groups and has three alternatives; R4 has two alternatives. The library thus includes nine separate compounds ([1 + (1 × 2)] × 3).

R1 + R2 = –H . –CH$_2$– / –O–R4 . –CH$_2$–CH$_2$–

R3 = –OH / –OCH$_3$ / –OCH$_2$CH$_3$

R4 = –H / –CH$_3$

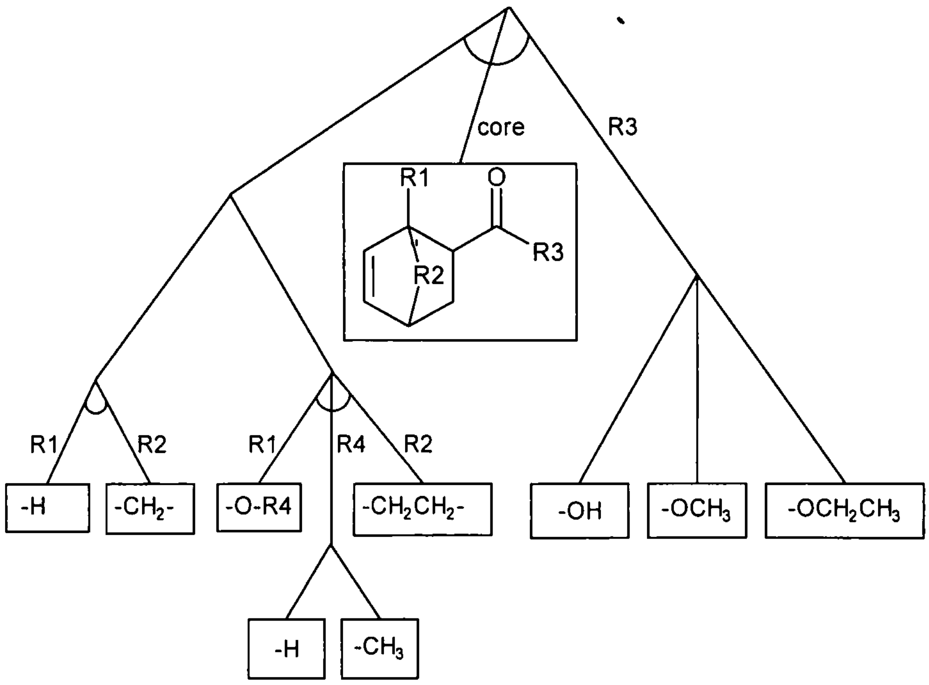
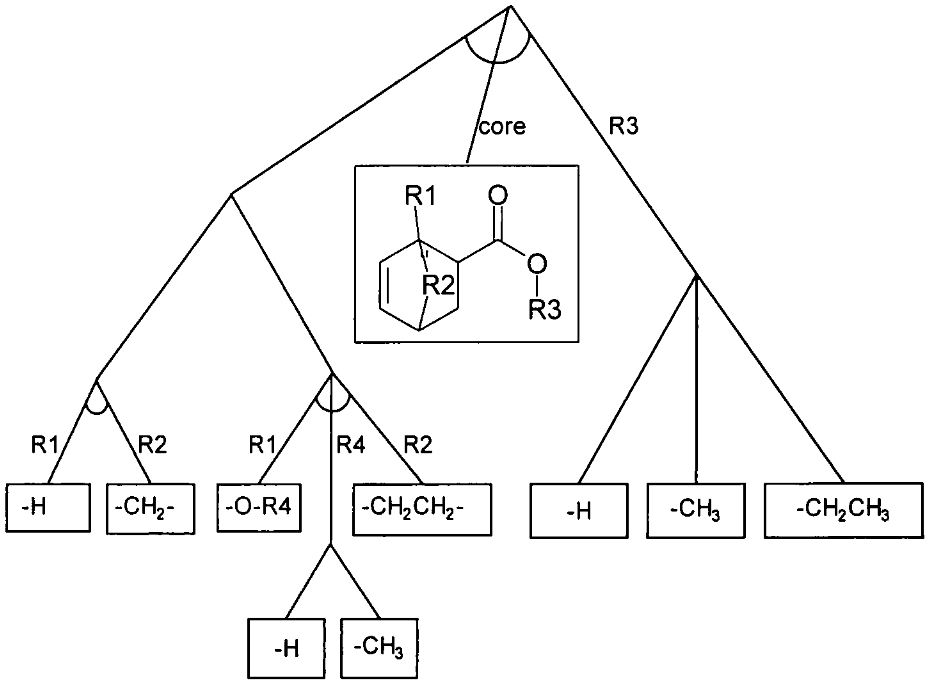


Figure 6. Schematic illustration of the EFCL corresponding to the Markush structure depicted in Figure 5. The boxes correspond to Partial Structures, and the lines linking them correspond to Gates. Lines joining at a point with a small arc link Partial Structures in AND relationship (i.e., occur together in members of the library). Lines without the arc link Partial Structures in OR relationship (i.e., occur in different members of the library). The Markush core is shown in the middle, in AND relationship with the three possibilities for R3 (on the right) and the two possibilities for R1 and R2 (defined together) on the left. The second of these alternatives involves the nested R4, whose two alternatives (in OR relationship with each other) are thus in AND relationship with the definitions given for R1 and R2.

processing faster. One aim of this is to make the Markush core as large as possible. Thus, if all the alternatives for an R-group include a common substructure, this is identified, using a subgraph isomorphism algorithm. The atoms involved are deleted from each R-group alternative and added to the core, a process referred to as "bubbling up." Subsequent analysis of the structure will be more efficient because the common part will be analyzed only once, instead of once for each alternative for the R-group. Some restrictions are imposed on the subgraph isomorphism detection to avoid algorithmic problems in identifying multiple maximum common substructures. Figure 7 illustrates the optimization of the EFCL in Figure 6. The common oxygen atom in all the alternatives for R3 has been bubbled up to the core, which makes identification of fragments such as —C(=O)O more efficient, as the whole fragment now is contained in the core. Although this is a rather

Figure 7. Optimized version of the EFCL in Figure 6, in which the common oxygen atom for the R3 alternatives has been bubbled up to the core.



trivial example in which the processing efficiency savings are not likely to be that great, there are situations where it can make a much bigger difference, as illustrated by the tripeptide library in Figure 8. The first representation of this obscures the fact that there is actually a substantial constant backbone to the structure. The atoms common to all the alternatives in each R-group can be identified and bubbled up to the core. In identifying substructural fragments (e.g., for fingerprint generation), the peptide backbone now needs only to be traced once, rather than laboriously identified multiple times by combining part fragments from all possible combinations of three amino acid residues.

ANALYSIS OF THE MARKUSH STRUCTURE

The basic idea of directly analyzing the Markush structure is to avoid multiple analyses of common/repeated parts with respect to the different products, with a consequential saving in storage, I/O, and processing. Clearly, if some structural descriptor (e.g., a 2D fingerprint) is to be generated for all compounds in a library, at least some operations will have time requirements that depend on the number of products (Πn_i). Our approach to the analysis is therefore a two-stage one, in which as much of the analysis as possible is carried out using the Markush representation itself [$O(\Sigma_i)$], creating a secondary intermediate data structure containing "partial descriptors" derived from the individual Partial Structures and the overlaps between them. The second stage [$O(\Pi_i)$] then rapidly assembles these partial descriptors in appropriate logical combinations to form a full descriptor for each molecule.

An underlying requirement for the success of this approach is that the descriptors in question should be additive in nature, i.e., the descriptor for the whole molecule should be obtainable by adding together contributions from individual parts. The current system is able to use the approach to generate enumerated SMILES, fragment-based structure fingerprints (presence/absence of substructure keys), and certain property descriptors (the "rule of 5" descriptors used by Lipinski et al.,[22] along with a count of aromatic rings). We currently are extending the approach in order to generate descriptors based on selected topological indices. Figure 1 shows how the two-stage analysis processes are integrated into the system as a whole.

The generation of individual SMILES strings for the molecules covered by a Markush provides a good illustration of the way the analysis works. The process takes advantage of a "trick" in the definition of SMILES,[23] involving ring closure numerals. In the SMILES linear notation, matching ring closure numerals are used to indicate bonds between nonadjacent atoms in the string. Such numerals can, in fact, be used even when no ring is involved. Thus, in the valid SMILES "C1.C1," the numerals indicate a bond between the two carbon atoms, which are otherwise disconnected by the dot between them. Therefore, the molecule represented is ethane, more usually shown by the SMILES notation "CC."

In the initial EFCL analysis phase, a "partial SMILES" is constructed for each Partial Structure. Where there is a bond to an atom in a different Partial Structure, this is shown by an unsatisfied ring closure numeral. To avoid potential clashes with "real" ring closure numerals, numerals from 10 upward (which in SMILES syntax must be preceded by a % symbol) are used for this purpose. For the optimized Markush core shown in Figure 7, the partial SMILES is:

C1=CC%10%11C(C(=O)O%13)CC1%12

where %10 indicates the connection to R1, %11 the connection from the same atom to R2, %12 the other connection to R2, and %13 the connection to R3. The numeral 1 indicates the "real" ring closure for the cyclohexene ring. For the three Partial Structures for R3 (shown in the bottom right of Figure 7), the partial SMILES are:

H%13

C%13

and C%13C

where the %13 indicates the connection to the R3 position in the core. Similar partial SMILES expressions are constructed for the various R1, R2, and R4 Partial Structures. Appropriate housekeeping, to ensure that matching ring closure numerals are used only where partial structures are attached together, means that assembly of complete SMILES for the enumerated molecules is simply a question of concatenating the appropriate partial SMILES strings together, separated by dots. One of the enumerated molecules covered by this library is shown by:

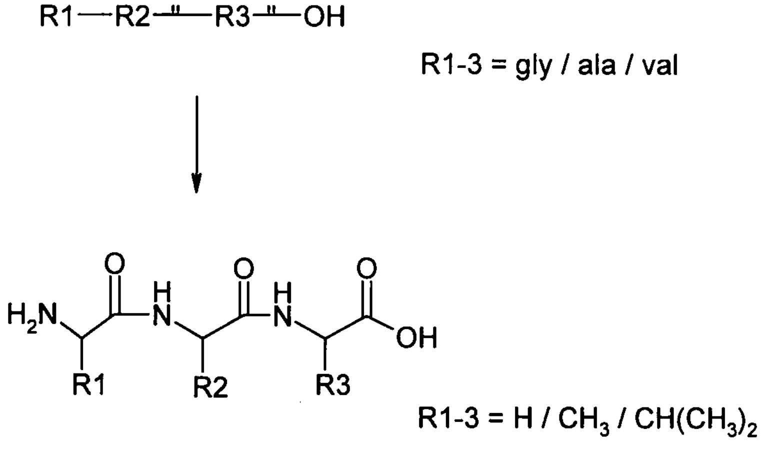


Figure 8. Tripeptide library in which optimization of the initial representation reveals a substantial common core.

C1=CC%10%11C(C(=O)O%13)CC1%12.

O%14%10 . C%11C%12 . C%13C . C%14

in which all the unsatisfied ring closures match up across the dots, forming an unconventional, although perfectly valid, SMILES. Note that the third component of this complete SMILES includes both %11 and %12 ring closure numerals, thus closing the R2 bridge.

Simple string concatenation is an extremely rapid operation; thus, enumeration of the specific SMILES for the molecules in a library is very fast. The system contains an option to call Daylight's own canonicalization routine if a canonical SMILES[24] is required. Because this must be done separately for each individual molecule, it can slow the enumeration process down very substantially.

STRUCTURE DESCRIPTOR GENERATION

Structural fingerprint descriptors generated by the system are based on the presence or absence of 2D structure fragments, which are traced within (and, where appropriate, between) the Partial Structures in the EFCL. Each fragment traced is looked up in a customizable dictionary of fragments (identical to that used with our specific-structure fingerprint generation software[25]), which specifies a fingerprint bit position to be set. Like MDL's ISIS keys therefore (and unlike Daylight fingerprints), each fingerprint bit represents the presence or absence of a specific fragment specified in the dictionary. Fragment types that are traced include augmented atoms, atom sequences (equivalent to the Daylight atom paths), atom pairs, and ring descriptors.

All the fragment tracing and dictionary lookup is done during the initial analysis phase; each fragment found is added to the intermediate data structure, in which a "partial fingerprint" is associated with each Partial Structure or in some cases combination of Partial Structures. In the analysis for SMILES generation, each partial SMILES is confined to a single Partial Structure, which considerably simplifies the problem. Because some of the fragments traced for fingerprint generation (e.g., sequences of connected atoms) are split over two or more Partial Structures, some additional complexity is needed in the intermediate data structure. Also, the speed advantages of the Markush approach diminish with an increase in the number of different R-groups involved in a single fragment (because, for example, all the pairwise combinations of alternatives for two R-groups need to be enumerated). The EFCL optimization procedure described earlier helps to minimize this problem, which will be discussed further in the performance evaluation.

In the second "fingerprint assembly" phase, appropriate logical combinations of these partial fingerprints can be used to assemble the full fingerprint for each individual molecule, a relatively rapid process. Output options include the full set of enumerated fingerprints (in various formats), and centroid and modal fingerprints (which summarize features present in the whole library). The individual fingerprints, which can be assembled "on the fly" from the intermediate data structure as needed, also can be used as a basis for clustering the library members, as discussed later.

The property descriptors used by Lipinski et al.[22] to estimate solubility and permeability comprise feature counts (molecular weight, rotatable bonds, hydrogen-bond acceptors, and hydrogen-bond donors) and logP. These, along with a count of aromatic rings, can be generated using an additive approach and so are amenable to generation using the Markush method. Molecular weight is simple to process because it is additive for each atom, and so the summed molecular weight for atoms within a given Partial Structure can be associated with that Partial Structure. The same applies to the aromatic ring count, because the current version of the program is implemented so that the EFCL does not contain aromatic rings that are split across more than one Partial Structure.

The environment defining hydrogen-bond acceptors, hydrogen-bond donors and, in particular, rotatable bonds can span more than one Partial Structure. As with fragment generation for fingerprints, there is additional complexity in the detection and subsequent storage in the intermediate data structure. Some customization also is possible in the definitions of the required environments for these features. For rotatable bonds, if an atom has a single bond and attachments to atoms in more than one other Partial Structure, then whether or not it is rotatable can depend on the combination of alternatives for several different R-groups. This occurs if certain combinations of the R-groups lead to situations in which the atom at one end of the possibly rotatable bond has neighbors that are identical terminal atoms (in which case the bond is not considered rotatable, as in a —CF_3 group), while other combinations lead to situations in which these neighbors are different or nonterminal (in which case the bond is considered rotatable). In most cases, however, its rotatable state depends only on the value taken by a single R-group, and its contribution to the overall count can be stored in association with the relevant Partial Structure. This can be true even where the bond in question lies between Partial Structures.

The most complex Lipinski descriptor to calculate is the logP value. The standard ClogP[26] fragment-based method for calculating logP values defines a wide range of fragments in fairly broad environments, making it unsuitable for use in a Markush context. The alternative atom-additive methods tend to define fragments (atom types) in groups, each within a fairly narrow environment. Study of the sets of atom types used in the AlogP,[27] XlogP,[28] and SlogP[29] atom-additive methods suggested that the SlogP method would be the most appropriate for implementation in a Markush context and therefore has been our choice. The published method defines 68 atom types as Daylight SMARTS patterns, with another four "supplemental" atom types for C, H, N, and O atoms not classified by the other types. In our implementation, however, we have defined them using a modified form of the fragment dictionary used for fingerprint generation, which has allowed us to adapt the Markush fragment generation software (described earlier) to generate logP values. The modified dictionary contains 797 fragments comprising 120 single atoms, 644 augmented atoms, 26 atom sequences, three "special cases" (which need to be generated separately), and the four "supplemental" atom types. SlogP fragment tracing, dictionary lookup, storage of "partial SlogP values" in the intermediate data structure. and the assembly of full SlogP values are conducted in a similar manner to the handling of fragments for fingerprints outlined earlier.

Topological indices are another category of descriptor that can usefully be generated for combinatorial libraries. From the large number of published indices,[30] those requiring analysis of a complete molecule, such as the Hosoya Index, are not suitable for generation in a Markush context. Descriptor generation in our system currently is being extended to include topological indices that are defined additively from simpler and more limited environments, i.e., those that can be generated from feature counts or from limited-environment fragments. Subgraph Index Counts (order 0 to 3) are simple counts of features that can be handled in a similar manner to the Lipinski counts. The Kappa Shape Indices (order 1 to 3), and related Phi Flexibility Indices (order 1 and 2), require counts of the numbers of vertices and edges, with the calculation performed at the final assembly stage. The Information Atomic Content Indices similarly require just the counts of vertices and edges, but it still has to be determined whether the more complex Information Content indices can be generated. The Zagreb Index is the sum of the squared number of connections to nonhydrogen atoms and so can be generated from simplified augmented atoms. The lower order (0 to 3) Chi Connectivity Indices correspond to single atoms, augmented atoms, and linear sequences. The wider environment covered by the Wiener Index and Balaban Indices, which require the distances between all pairs of atoms, increases the complexity of generation and expected generation times, and so might not be suited to efficient generation in a Markush context.

DESCRIPTOR GENERATION PERFORMANCE

The performance of the descriptor generation routines was tested using an artificial library consisting of a benzodiazepine core, with three R-groups, each having 100 alternatives, thus giving one million molecules. Table 1 shows the times required to generate the SMILES, Lipinski properties, and fingerprints. It can be seen that the method will allow large libraries to be analyzed in acceptable times.

For comparison of fingerprint generation times, the individual library members were enumerated and passed to three standard commercially available fingerprint generation programs. The specific-structure version of the BCI fingerprint generator produced the same fingerprints as the Markush version (2,045 fragments assigned to 1,052 bits), the Daylight Fingerprint program produced 1,024-bit hashed fingerprints, and the MDL generator produced ISIS 960-key fingerprints. Figure 9 compares the times needed to generate all one million fingerprints for the three programs; for all except the Markush method, there is also a component for the enumeration of the individual molecules (which was done with MSI's Cerius^2 enumeration routine). The figures in parentheses give the overall number of fingerprints generated per second (including structure enumeration time where appropriate). The results show clearly that the Markush method is a couple of orders of magnitude faster than the other methods.

In order to look in more detail at some of the factors influencing the speed of the Markush method, a second library was created in which the benzodiazepine core was replaced by a single carbon atom core, to which all three R-groups were attached directly. The results are summarized in Figure 10. For each library, times are shown for the fragment generation and fingerprint assembly stages (as well as the total), plotted against library size for a variety of subsets of the full library.

In the case of the benzodiazepine core, the fragment generation times (crosses) are very small, and virtually independent of the library size; the plots for fingerprint assembly time (plus signs) and total time (stars) are almost on top of each other and show a more-or-less linear increase with library size. In contrast, for the vestigial single carbon atom core, the fragment generation times (triangles) are much higher. This is because many more fragments spanning Partial Structures from different R-groups must be traced. The fingerprint assembly times (circles) also are higher because there are more partial fingerprints (from different combinations of R-group alternatives) to be assembled, and the total times (squares) thus are higher still.

The conclusion from this is that, as expected, the Markush approach offers greater savings when the R-groups are well separated on a large core (hence the value of EFCL optimisation, which aims to maximize the size of the core). However, even when they are not, the total times still offer a substantial saving on the enumeration approach. This is because the nature of the fragments being generated is such that they are very unlikely to involve more than two R-groups, and thus (in this three R-group system) one component of the combinatorial explosion is removed.

CLUSTERING

The chemical informatics community has given much attention over recent years to the Jarvis-Patrick nonhierarchical clustering method, due to its speed, and to the Ward hierarchical agglomerative clustering method, due to the quality of the

Table 1. Times required to generate SMILES, Lipinksi properties, and fingerprints for a one million compound benzodiazepine library

	Partial descriptor generation (s)	Descriptor assembly (s)	Total time (s)	Structures per second
Noncanonical SMILES		25.79	25.83	38,715
Canonical Smiles	0.04	1169.64	1169.68	855
Lipinski properties		95.59	95.63	10,457
Fingerprints	2.03	363.61	365.64	2,735

Run on a Silicon Graphics single processor 195-MHz Octane R10000 workstation with 128 MB of RAM.
Separate figures are given for the partial descriptor generation and descriptor assembly phases; the figure of 0.04s shown for partial descriptor generation includes partial SMILES and partial property values for all the Lipinski properties. SMILES canonicalization was done with the appropriate Daylight SMILES Toolkit routine on the noncanonical SMILES. The fingerprints were generated using a dictionary of 2,045 augmented atom, atom sequence, and ring fragments assigned to 1,052 bits.

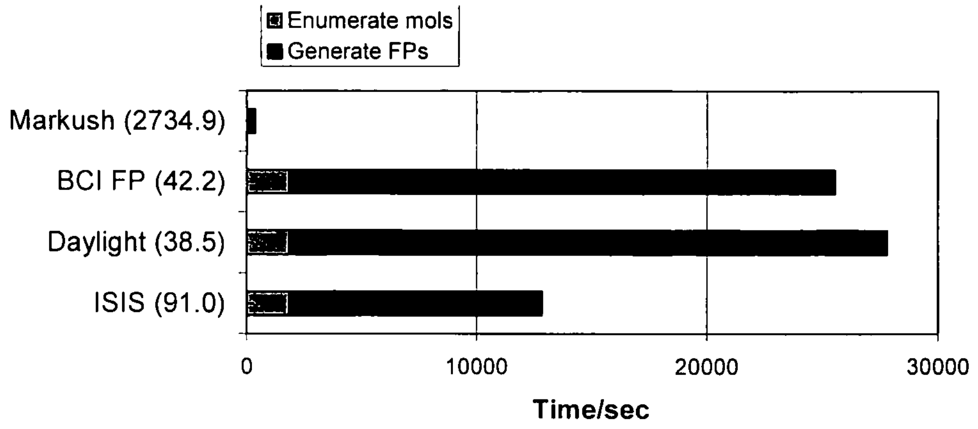


Figure 9. Comparison of generation times for one million fingerprints from a benzodiazepine library using the Markush method and three standard specific-structure methods. The lighter area at the left end of all but the Markush bar represents the time required to enumerate the individual molecules before fingerprint generation. Values in parentheses show the number of fingerprints generated per second.

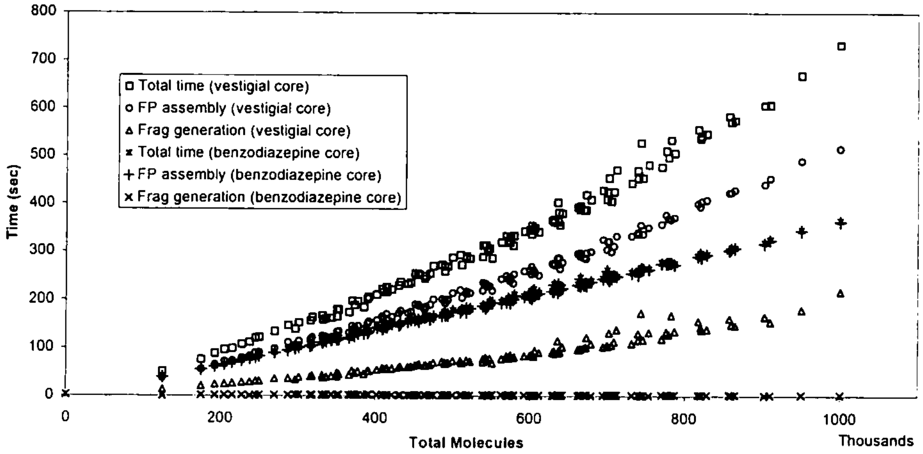


Figure 10. Comparison of fragment generation and fingerprint assembly times for various subsets of one million compound libraries, with a benzodiazepine core and a vestigial single carbon atom core, respectively. See text for discussion.

resultant clusters.[31] Because both of these methods have time requirements that rise with the square of the number of compounds to be clustered, neither is particularly suited to processing very large datasets. However, the K-means nonhierarchical clustering method is faster than either Ward or Jarvis-Patrick and, if suitable starting points are chosen, can produce clusters of similar quality to those produced by Ward.[32] Its time requirements are linear in the number of compounds, and its space requirements are linear in the number of clusters. In our system, a user-specified number of clusters is initialized with a random selection of library members. Each molecule then is assigned to the nearest cluster centroid, the centroids recalculated, and the molecules reassigned until no further changes take place. Convergence usually occurs in 10 to 30 iterations, depending on a variety of factors, including the initial choice of seeds, number of clusters required, and how naturally well clustered the data is. Fingerprints or property data are required for each individual molecule, and these are reassembled from the intermediate data structures when required, thus avoiding the overheads of storing and retrieving them.

Figure 11 shows the number of iterations and the total time required to cluster the full library of one million benzodiazepines into between 100 and 10,000 clusters, based on their fingerprints. There. is a falloff in the number of iterations required to reach convergence for larger numbers of clusters; thus, although the average time required for each iteration increases with the number of clusters (because each molecule must be compared to more cluster centroids), the total time required actually plateaus off with more clusters (at least over the range of values tested). This can be explained by the fact that when a larger number of clusters is requested, those clusters tend to be smaller and tighter, and thus less relocation of molecules between clusters is needed. The necessary relocations can be achieved in fewer iterations. However, it should be noted that this type of behavior can be heavily influenced by the number of "natural" clusters in the dataset and by the initial choice of cluster seeds. Nevertheless, the absolute time values shown demonstrate that libraries of this size can conveniently be clustered on a standard workstation in an overnight or weekend run.

CONCLUSIONS AND FUTURE DIRECTIONS

The system described in this article shows how Markush structure representations for combinatorial libraries can be built from reaction- and precursor-based descriptions, and that direct analysis of such Markush structures can allow generation of descriptors, such as SMILES strings, physicochemical properties, and fragment-based fingerprints, for the individual molecules covered by the library, with substantial time savings over

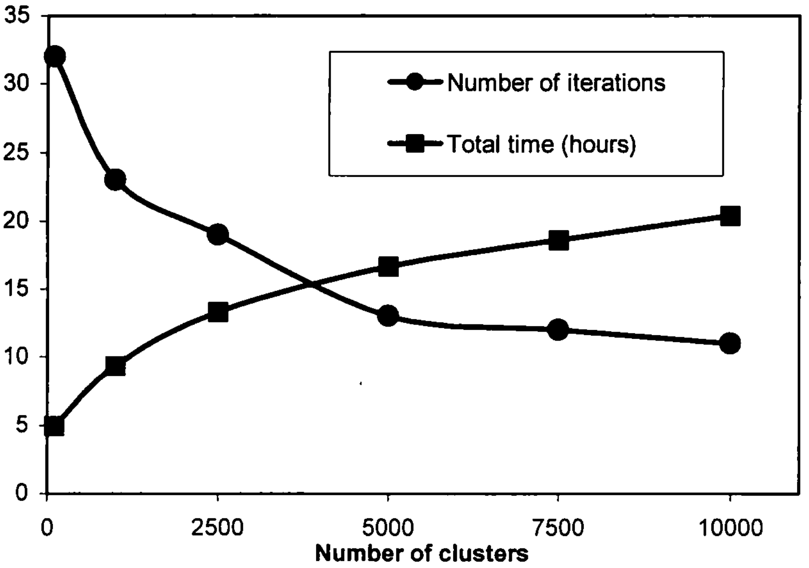


Figure 11. Plots of the number of K-means iterations and the time required to produce various numbers of clusters from a one million compound benzodiazepine library (run on a Silicon Graphics single processor 195-MHz Octane R10000 workstation with 128 MB of RAM).

approaches based on initial enumeration of the molecules themselves and their subsequent individual analysis.

A useful extension to the current system would allow output of a display depiction for a Markush structure. This could provide a convenient user interface for Markush structures derived from reaction- and precursor-based input. We currently are working on a Markush output format based on the "partial SMILES" strings described earlier, in which the unsatisfied ring closures are replaced by dummy atoms of the form "[n*]," where n is formally an "atomic mass number" for a "*" atom (defined in SMILES as an atom of atomic number zero). In practice, it would be the same number as is used for the ring closure. Such modified Partial SMILES strings conform fully to SMILES syntax, and sets of them describing complete Markush structures could be depicted by any software capable of depicting valid SMILES strings. Given the different expressive capabilities of various existing Markush structure representations (and in the light of the continuing failure of the chemical informatics community to agree on a single exchange format for specific structure representations), it seems unlikely that such a format could provide a "universal" exchange format for libraries, but it might have some value in this context.

An obvious extension to the descriptor types available in the current system is the generation of some form of 3D descriptor. We currently are examining approaches to the problem of assembling conformations stored for the individual building blocks of the Markush structure. We also plan to use ideas already implemented in Markush-based chemical patent information systems to provide approaches to library registration and searching, including features such as the identification of the overlap between different libraries.

REFERENCES

1 Barnard, J.M., and Downs, G.M. Computer representation and manipulation of combinatorial libraries. *Perspect. Drug Discovery Design* 1997, **7/8, 13–30**

2 Leland, B.A., Christie, B.D., Nourse, J.G., Grier, D.L., Carhart, R.E., Maffett, T., Welford, S.M., and Smith, D.H. Managing the combinatorial explosion. *J. Chem. Inf. Comput. Sci.* 1997, **37, 62–70**

3 Bayada, D.M., Cho, W., Marshall, C., and Johnson, A.P. C3: Clustering based on common cores. Presented at the 210th ACS National Meeting, Chicago, IL, August 20–24, 1995

4 Gillet, V.J., Willett, P., and Bradshaw, J. The effectiveness of reactant pools for generating structurally-diverse combinatorial libraries. *J. Chem. Inf. Comput. Sci.* 1997, **37, 731–740**

5 Brown, R.D., and Martin, Y.C. Designing combinatorial library mixtures using a genetic algorithm. *J. Med. Chem.* 1997, **40, 2304–2313**

6 Good, A.C., and Lewis, R.A. New methodology for profiling combinatorial libraries and screening sets: Cleaning up the design process with HARPick. *J. Med. Chem.* 1997, **40, 3926–3936**

7 Brown, R.D., and Clark, D.E. Genetic diversity: Applications of evolutionary algorithms to combinatorial library design. *Expert Opin. Ther. Patents* 1998, **8, 1447–1460**

8 Zheng, W., Cho, S.J., Waller, C.L., and Tropsha, A. Rational combinatorial library design. 3. Simulated Annealing Guided Evaluation (SAGE) of molecular diversity: a novel computational tool for universal library design and database mining. *J. Chem. Inf. Comput. Sci.* 1999, **39, 738–746**

9 Gillet, V.J., Willett, P., Bradshaw, J., and Green, D.V. Selecting combinatorial libraries to optimise diversity and physical properties. *J. Chem. Inf. Comput. Sci.* 1999, **39, 169–177**

10 Downs, G.M., and Barnard, J. M. Techniques for generating descriptive fingerprints in combinatorial libraries. *J. Chem. Inf. Comput. Sci.* 1997, **37, 59–61**

11 Barnard, J.M., and Downs, G.M. Use of Markush structure techniques to avoid enumeration in diversity analysis of large combinatorial libraries. Presented at the Daylight Chemical Information Systems Inc. MUG 97 meeting, Laguna Beach, CA, February 27, 1997. http://www.daylight.com/meetings/mug97/Barnard/970227JB.html

12 The software system described in this article is available commercially as the LibEngine module of Cerius^2 (version 4.5), and the reaction-based input has been incorporated into version 1.7 of WebLab Diversity Explorer, both available from Molecular Simulations Inc., 9685 Scranton Road, San Diego, CA 92121, USA

13 http://www.afferent.com /generic-structure.html

14 Barnard, J.M. A comparison of different approaches to Markush structure handling. *J. Chem. Inf. Comput. Sci.* 1991, **31, 64–68**

15 Barnard, J.M., Lynch, M.F., and Welford, S.M. Computer storage and retrieval of generic chemical structures in patents. 4. An extended connection table representation for generic structures. *J. Chem. Inf. Comput. Sci.* 1982, **22, 160–164**

16 Dalby, A., Nourse, J.G., Hounshell, W.D., Gushurst, A.K., Grier, D.L., Leland, B.A., and Laufer, J. Description of several chemical structure file formats used by computer programs developed at Molecular Design Limited. *J. Chem. Inf. Comput. Sci.* 1992, **32, 244–255**

17 Ash, S., Cline, M.A., Homer, R.W., Hurst, T., and Smith, G.B. SYBYL Line Notation (SLN): A versatile language for chemical structure representation. *J. Chem. Inf. Comput. Sci.* 1997, **37, 71–79**

18 Examples include Afferent Systems Inc. (1550 Bryant Street, Suite 760, San Francisco, CA 94103, USA; http://www.afferent.com), Cambridgesoft Corp. (10 Cambridge Park Drive, Cambridge, MA 02140, USA; http://www.camsoft.com), Daylight Chemical Information Systems Inc. (27401 Los Altos Suite 360, Mission Viejo, CA 92691, USA; http://www.daylight.com), Oxford Molecular Ltd. (The Medawar Centre, Oxford Science Park, Oxford OX4 4GA, UK; http://www.oxmol.co.uk), Synopsys Scientific Systems Ltd. (5 North Hill Road, Leeds LS6 2EN, UK; http://www.synopsys.co.uk), and Tripos Inc. (1699 South Hanley Road, St Louis, MO 63144, USA). MDL Information Systems (14600 Catalina Street, San Leandro, CA 94577, USA; http://www.mdli.com) also has announced plans for products in this area

19 Leach, A.R., Bradshaw, J., Green, D.V., and Delaney, J.J. Implementation of a system for reagent selection, library enumeration, profiling and design. *J. Chem. Inf. Comput. Sci.* 1999, **39, 1161–1172**

20 WebLab Diversity Explorer is available from Molecular Simulations Inc., 9685 Scranton Road, San Diego, CA 92121, USA

21 Barnard, J.M., and von Scholley-Pfab, A. Conversion of MDL query formats to SMARTS and SMIRKS. Presented at the Daylight Chemical Information Systems Inc. MUG 99 meeting, Santa Fe, NM, February 1999. http://www.daylight.com/meetings/mug99/Barnard/index.html

22 Lipinski, C.A., Lombardo, F., Dominy, B.W., and Feeney, P.J. Experimental and computational approaches to estimate solubility and permeability in drug discovery and development settings. *Adv. Drug Delivery Rev.* 1997, **23, 3–25**

23 Weininger, D. SMILES: A chemical language and information system. 1. Introduction to methodology and encoding rules. *J. Chem. Inf. Comput. Sci.* 1988, **28, 31–36**

24 Weininger, D., Weininger, A., and Weininger, J.L. SMILES. 2. Algorithm for generation of unique SMILES notation. *J. Chem. Inf. Comput. Sci.* 1989, **29, 97–101**

25 Barnard, J.M., and Downs, G.M. Chemical fragment generation and clustering software. *J. Chem. Inf. Comput. Sci.* 1997, **37, 141–142**

26 Leo, A.J. Calculating log P(oct) from structures. *Chem. Rev.* 1993, **93,** 1281–1306

27 Ghose, A.K., Viswanadhan, V.N., and Wendoloski, J.J. Prediction of hydrophobic (lipophilic) properties of small molecules using fragmental methods: An analysis of AlogP and ClogP methods. *J. Phys. Chem. A* 1998, **102, 3762–3772**

28 Wang, R., Fu, Y., and Lai L. A new atom-additive method for calculating partition coefficients. *J. Chem. Inf. Comput. Sci.* 1997, **37,** 615–621

29 Wildman, S.A., and Crippen, G.M. Prediction of physicochemical parameters by atomic contributions. *J. Chem. Inf. Comput. Sci.* 1999, **39,** 868–873 [Note: the publication does not name the method, but in a personal communicaton Wildman refers to it as SlogP (for SMARTS logP).]

30 Balaban A.T. Applications of graph theory in chemistry. *J. Chem. Inf. Comput. Sci.* 1985, **25, 334–343**

31 Brown, R.D., and 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, 572–584**

32 Higgs, R.E., Bemis, K.G., Watson, I.A., and Wikel, J.H. Experimental designs for selecting molecules from large chemical databases. *J. Chem. Inf. Comput. Sci.* 1997, **37, 861–870**