ARTICLES

Efficient Exploration of Large Combinatorial Chemistry Spaces by Monomer-Based Similarity Searching

Ning Yu* and Gregory A. Bakken

Pfizer Global Research and Development, Eastern Point Road, Groton, Connecticut 06340

Received October 23, 2008

In modern drug discovery, 2-D similarity searching is widely employed as a cost-effective way to screen large compound collections and select subsets of molecules that may have interesting biological activity prior to experimental screening. Nowadays, there is a growing interest in applying the existing 2-D similarity searching methods to combinatorial chemistry libraries to search for novel hits or to evolve lead series. A dilemma thus arises when many identical substructures recur in library products and they have to be considered repeatedly in descriptor calculations. The dilemma is exacerbated by the astronomical number of combinatorial products. This problem imposes a major barrier to similarity searching of large combinatorial chemistry spaces. An efficient approach, termed Monomer-based Similarity Searching (MoBSS), is proposed to remedy the problem. MoBSS calculates atom pair (AP) descriptors based on interatomic topological distances, which lend themselves to pair additivity. A fast algorithm is employed in MoBSS to rapidly compute product atom pairs from those of the constituent fragments. The details of the algorithm are presented along with a series of proof-of-concept studies, which demonstrate the speed, accuracy, and utility of the MoBSS approach.

INTRODUCTION

The fundamental premise of similarity searching is the *neighborhood principle*, which states that structurally similar molecules often have similar properties.[1,2] Under this principle, if subsets of molecules within a compound collection similar to an active molecule are screened, a higher likelihood of finding new actives should be observed than by screening the entire collection at random.[3] Molecular similarity can generally be assessed on two levels depending on the way molecules are represented: on the 2-D level when topological descriptors are used and on the 3-D level when conformations are used. Correspondingly, similarity searching comprises 2-D and 3-D approaches.[4] 2-D approaches consider the topological arrangements of atoms and bonds and model molecular structures as 2-D graphs, whereas 3-D approaches consider the geometric features of molecular shapes and model molecular structures as atomic spheres packed together according to low energy conformations. Despite the expectation that 3-D approaches should better represent the actual modes of interaction between small molecules and their protein targets, studies have found that 2-D methods are often more effective at retrieving active molecules.[5−7] Furthermore, it has been suggested that a carefully constructed set of 2-D topological and property descriptors can implicitly take into account some of the 3-D features.[8] These findings combined with the speed advantage of 2-D descriptors have resulted in a renewed interest in 2-D similarity searching as a major tool in virtual screening, which defines the scope of the present work.

Traditionally, similarity-based virtual screens have focused on large collections of druglike compounds whose numbers are typically in the few hundreds of thousands to millions. Lately, virtual combinatorial libraries are increasingly becoming the targets of these efforts.[9−11] A typical virtual reaction repository in a large pharmaceutical company contains hundreds of registered reactions and tens of thousands of building materials also known as monomers, which can, in theory, encode trillions of reaction products. Such a virtual library defines a vast synthetically accessible chemical space that drug discovery projects often wish to exploit to search for novel hits or to evolve their existing leads into more favorable patent and/or property space.[12] In the mean time, the sheer size of these large combinatorial libraries has created a tremendous challenge for traditional product-based similarity searching:[13] assuming the chemical structures of reaction products can be enumerated at a throughput of a few thousand molecules per second, the calculation of the descriptors needed to measure the similarity between an enumerated product and the query typically proceeds at a few hundred molecules per second.[14] This implies that before a similarity search against the full product space covered by a trillion virtual molecules can be carried out, the calculation of structural descriptors alone will likely require dozens of years of CPU time. Notwithstanding that descriptor calculation is a one-time cost and notwithstanding that computing powers continue to surge, the current gap that is several orders of magnitude in size is probably still beyond a remedy.

On the other hand, a virtual library covering a product space of a trillion molecules is not expected to contain as

* Corresponding author e-mail: ning.yu@pfizer.com.

10.1021/ci800392z CCC: $40.75
Published on Web 03/24/2009

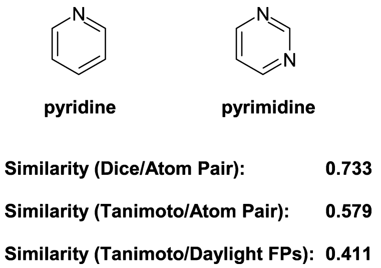

pyridine
pyrimidine
Similarity (Dice/Atom Pair): 0.733
Similarity (Tanimoto/Atom Pair): 0.579
Similarity (Tanimoto/Daylight FPs): 0.411

Figure 1. Chemical structures of pyridine and pyrimidine and the similarity between them calculated using AP descriptors and Daylight fingerprints.

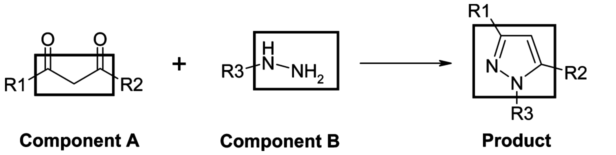
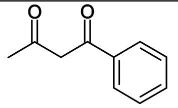
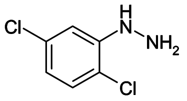
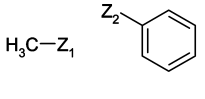
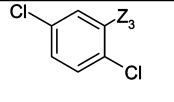


Figure 2. An example of a 2-component virtual reaction using diketones and hydrazines to synthesize products containing a pyrazole core. The boxes enclose the pyrazole core and its constituent parts in the reactants.

Table 1. Illustration of the Calculation of the AP Similarity Measure between Pyridine and Pyrimidine

atom pair	count in pyridine	count in pyrimidine	shared pairs
C•X2-1-C•X2	4	2	2
C•X2-2-C•X2	4	3	3
C•X2-3-C•X2	2	1	1
N•X2-1-C•X2	2	4	2
N•X2-2-C•X2	2	2	2
N•X2-3-C•X2	1	2	1
N•X2-2-N•X2	0	1	0
subtotal	15	15	11

Table 2. Sample Monomer Structures for the 2-Component Reaction Shown in Figure 2 Containing Diketones and Hydrazines

	Component A	Component B
Sample Monomer		
Clipped Monomer		

many diverse substructures as a real library of a trillion druglike molecules.[15] This is because combinatorial chemistry reactions merely generate various combinations of monomers and tend not to produce many new substructures. Thus, when similarity searching virtual libraries, it would be inefficient to enumerate the structures of all the virtual products, compute their descriptors, and evaluate the similarity between each product and the query, since this approach requires repeated calculations over the same substructure features. In fact, utilization of exactly or approximately pairwise additive molecular properties to filter combinatorial libraries has been undertaken and shown to be successful for simple properties such as molecular weight, for feature counts such as numbers of hydrogen bond donors/acceptors, and for physicochemical properties calculated using reasonably additive empirical formulas such as SlogP and Solvent Accessible Volume.[16] An alternative approach has also been attempted where a small subset of virtual compounds is selected and their descriptors calculated, and then nonlinear models are trained to predict these descriptor values based on those of the monomers.[17] It was suggested that such models could be used to predict the descriptors for all the virtual compounds without the necessity of enumerating their structures. Nevertheless, both of these methods only address molecular property descriptors as opposed to substructure descriptors, which remain a challenge for structure-based similarity searching of combinatorial libraries.

Consequently, substructure descriptors that can be rapidly calculated for library products from their constituent monomers have the potential of radically eliminating the barrier for structure-based similarity searching of combinatorial libraries. With a judicious choice of the substructure descriptor, the actual computational cost may even scale like similarity searches in monomer space. Examples of the commonly used 2-D substructure descriptors include Daylight fingerprints,[18] Scitegic ECFP/FCFP fingerprints,[19] MDL MACCSS keys,[20,21] BCI fingerprints,[22] etc., none of which can be easily adapted to an additive procedure. Barnard et al. proposed the use of Markush notation to represent virtual products and to generate fingerprints, but their fingerprints were still path-dependent and would be prohibitive for extremely large libraries.[23-25] Recently, however, Boehm et al. reported a technique to search combinatorial chemistry spaces using a special type of substructure descriptor - Feature Trees, which is a reduced graph representation of chemical structures.[26-28] In addition, Ivancius et al. noted that descriptors computed from interatomic distances lend themselves to simple additivity. These latter authors also showed an efficient procedure to compute Wiener-type indices for virtual libraries without enumerating product structures.[29]

Indeed, a type of substructure descriptor that has been largely overlooked for this purpose is atom pair (AP), the subject of the current paper. AP descriptors represent substructure features using both atom types and the shortest topological bond-by-bond distance separating the pair. Herein, we present an efficient approach, named Monomer-Based Similarity Searching (MoBSS), to rapidly calculate AP descriptors for virtual products based on those of the monomers. As will be shown, MoBSS attains the speed improvement by replacing the expensive and repetitive graph traversals with an arithmetic manipulation of fragment atom pairs, allowing the cost of similarity searching combinatorial product space to scale nearly proportionally to the size of monomer space and allowing for the detection of any similar compounds out of a virtual library encompassing trillions of products.

This paper is organized as follows: first, the details of the MoBSS approach are explained; second, the speed and accuracy of MoBSS is examined by comparing the MoBSS results to those of a conventional product-based approach for a set of small libraries; third, two applications of MoBSS to drug molecules are presented to demonstrate how MoBSS can be used to efficiently search combinatorial chemistry spaces for structural analogs as well as finding multiple routes of synthesis; finally, the conclusion and our outlook for the future of this method can be found in the last section.

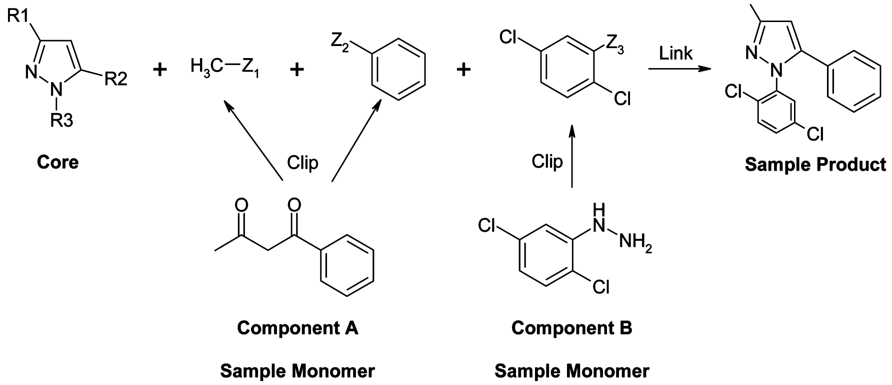


Figure 3. Illustration of the clipping and linking processes to enumerate a product structure where the sample monomers listed in Table 2 are clipped into the corresponding clipped monomers, which are linked to the core to form the product.

Table 3. Complete Listings of the Partial Atom Pairs in the Core (a) and Sample Clipped Monomer A (b) of the Example Reaction in Figure 4

atom pair	count
(a)	
C•X3-1-R1	1
C•X2-2-R1	1
C•X3-3-R1	1
N•X2-2-R1	1
N•X3-3-R1	1
C•X3-1-R2	1
C•X2-2-R2	1
C•X3-3-R2	1
N•X3-2-R2	1
N•X2-3-R2	1
C•X3-2-R3	1
C•X2-3-R3	1
C•X3-3-R3	1
N•X3-1-R3	1
N•X2-2-R3	1
R1-4-R2	1
R1-4-R3	1
R2-3-R3	1
(b)	
CX1-1-Z1	1
C•X3-1-Z2	1
C•X2-2-Z2	2
C•X2-3-Z2	2
C•X2-4-Z2	1

COMPUTATIONAL METHODOLOGY

Atom Pair Descriptors. The atom pair descriptors employed in this work are calculated using an in-house program implementing the original work by Carhart et al.[30] extended by Bush and Sheridan[31] among others.[32] We define 47 atom types which include multiple types of C, O, N, S, and P atoms depending on the number of multiple bonds and number of non-hydrogen neighbors, a single type of each of the halogens and B, Si, and Se atoms, and a default type to capture the remaining elements. For each pair of atoms in a molecule, we calculate the shortest topological bond-by-bond distances and set the distances for all the pairs longer than 20 bonds apart as 20 bonds. Stereochemical information is not included in AP descriptors, and thus we will not be concerned with molecular chirality throughout the paper.

An example of AP descriptor calculation is shown in Figure 1 and Table 1. Figure 1 shows the chemical structures of pyridine and pyrimidine. An example of an AP feature in both molecules is C•X2-1-C•X2, which represents a pair of carbon atoms both with one π electron (denoted by symbol "•") and two non-hydrogen neighbors (denoted by "X2"). With the AP descriptors generated, Carhart et al. suggested the Dice coefficient as a measure of the similarity between two molecules[30]

$$S_{AB} = \frac{2c}{a+b} \tag{1}$$

where a is the sum of counts of features present in molecule A, b is the sum of counts of features present in molecule B, and c is the sum of counts of features common to both molecules. It should also be noted that the metric that Carhart et al. suggested is different from the somewhat more familiar Tanimoto coefficient:[4]

$$S_{AB} = \frac{c}{a+b-c} \tag{2}$$

The choice of the expression has a substantial impact on the calculated similarity values. As Figure 1 shows, the AP similarity between pyridine and pyrimidine is 0.733 using the Dice definition and 0.579 using the Tanimoto definition. The latter is obviously closer to the similarity value of 0.441 calculated using the same definition but with Daylight fingerprints. However, both definitions seem to enjoy a similar level of popularity among users, and we have decided to report only the results calculated using the Dice expression for the sake of maintaining consistency with the original definition by Carhart et al.

Library Enumeration. Figure 2 shows a simple virtual reaction using diketones and hydrazines to synthesize products containing a common pyrazole core. In this paper, a set of monomers bearing the same reactive substructure is collectively called a component. The reaction in this example has two components: the first component is characterized by the diketone substructure and the second the hydrazine substructure. The product structure for this two-component reaction is represented using the "core-plus-R-groups" notation on the right side of the reaction arrow in Figure 2, which stipulates that all the product molecules will have the same pyrazole core but varying R1, R2, and R3 groups according to the corresponding monomer. The process through which the R-groups are derived from the monomers is called "clipping". Algorithmically, this involves searching the connection table of a monomer to locate the reactive substructure, deleting it, and adding an R-group atom to the other end for each of the dangling bonds. The bulk of the computational cost is in the substructure searching and manipulation steps. Table 2 contains an example of the monomers for components A and B as well as the resulting clipped monomers.

The process through which the clipped monomers are linked to the core to form the product is called "assembly".

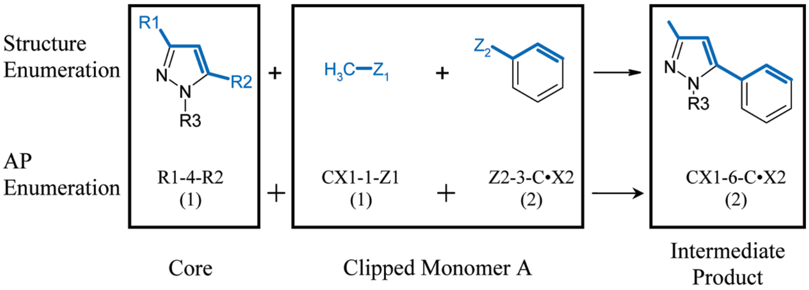


Figure 4. Comparison between structure enumeration and AP enumeration when the core is joined with clipped monomer A. The substructures highlighted in blue correspond to the AP features listed below them. The numbers in parentheses are feature counts.

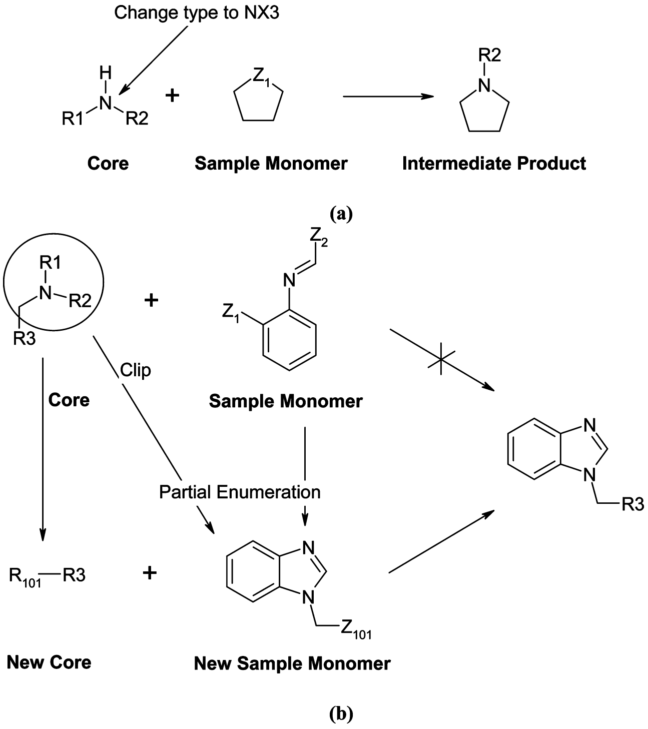


Figure 5. Examples of special procedures applied to virtual reactions during preprocessing. In panel (a), the type of the nitrogen atom is changed from NX2 to NX3 to account for its different number of neighbors upon enumeration. In panel (b), the reaction definition is changed to avoid the formation of a new ring during enumeration. The circle denotes the boundary of the clipped piece in the core. See text for more information.

Algorithmically, this involves creating bonds between the matching R-atoms of the core and the clipped monomers using the so-called Compatibility Rule (R1 to Z1, R2 to Z2, etc.) to construct the connection table for the product. This process by itself is not particularly time-consuming. However, the subsequent calculation of substructure descriptors for all the products, which can number in the millions or even billions, is prohibitive. Figure 3 summarizes the procedure for enumerating the product structure from the sample monomers for components A and B listed in Table 2.

MoBSS Algorithm. In this paper, we refer to cores and clipped monomers as fragments. MoBSS avoids full product structure enumeration by working directly with AP descriptors of fragments. Some fragment atom pairs have real atoms on both ends and are termed complete pairs; some have a real atom on one end and an R-group on the other and are termed one-R pairs; some have R-groups on both ends and are termed two-R pairs. The latter two are both called partial atom pairs. It should be immediately clear to the reader that all the complete pairs in fragments are also present in enumerated products. We now need to extend the AP definition and introduce new types to map to R_1, Z_1, R2, Z_2,..., R_N, and Z_N. The complete listings of the partial atom pair features in the core and clipped monomer A in the preceding example are shown in Table 3.

The approach MoBSS uses to calculate AP descriptors for products closely mimics the process of product structure enumeration, which can be carried out in a stepwise fashion. In each step, a core is joined with a clipped monomer, and the bonds of R-atoms with matching indices are fused to form an intermediate product. The resulting intermediate product can be treated as a new core and combined with the next clipped monomer. This process is repeated until each component has been considered and there are no unmatched R-atoms left. The top half of Figure 4 shows an example of such a partial structure enumeration where the core in our example is combined with the sample clipped monomer of component A. In the same way, a partial atom pair from a core and another from a monomer can be joined together using the Compatibility Rule, where, instead of connecting bonds, the following expressions

$$Pathlength_{AB} = Pathlength_{A} + Pathlength_{B} - 1 \qquad (3)$$

and

$$Count_{AB} = Count_{A} \times Count_{B} \qquad (4)$$

can be used to determine the path length and count for the synthesized feature. The right-hand side of the bottom half of Figure 4 shows the resulting atom pair feature in the intermediate product after two successive applications of eqs 3 and 4 on the compatible features in the core and the clipped monomer. In essence, at the heart of MoBSS is a simple arithmetic manipulation of compatible partial atom pairs to rapidly synthesize resultant AP features. Since the operation defined by eqs 3 and 4 is computationally much more efficient than graph searches on connection tables, it allows for on-the-fly enumeration of product AP descriptors during similarity searches. The complete details of the MoBSS algorithm are not further elaborated in this section but can be found in the Appendix.

In order to further speed up searching, at the onset of each MoBSS run we prescreen the pool of clipped monomers against the query structure to filter out those monomers whose structures are too dissimilar to the query. This is implemented by requiring the Tversky similarity, as defined below

$$S_{AB} = \frac{c}{\alpha(a-c) + \beta(b-c) + c} \qquad (5)$$

between the clipped monomers and the query to be greater than a certain threshold value. In eq 5, A denotes the

Table 4. Speed and Accuracy Information of the MoBSS Approach[a]

		product-based time (seconds)		MoBSS time (seconds)		
reaction no.	product space size	product structure enumeration	product AP calculation	monomer AP calculation	MoBSS AP enumeration	average similarity
1	7.1E+3	3	49	12	1	0.96
2	7.7E+3	4	58	1	1	1.00
3	7.9E+3	4	89	4	2	1.00
4	1.1E+4	7	108	2	2	1.00
5	1.2E+4	10	194	1	2	1.00
6	1.5E+4	9	145	1	2	0.97
7	1.6E+4	8	166	0	3	1.00
8	1.7E+4	18	219	14	3	0.97
9	2.0E+4	22	165	1	2	1.00
10	2.7E+4	17	140	1	3	1.00
11	4.7E+4	34	264	1	5	1.00
12	5.2E+4	61	652	0	9	1.00
13	5.4E+4	89	853	5	9	1.00
14	8.2E+4	193	1304	3	15	0.95
15	9.1E+4	150	1240	3	16	1.00
16	1.0E+5	248	648	1	11	1.00
17	1.0E+5	242	1059	1	15	1.00
18	1.3E+5	198	1426	2	21	1.00
19	1.3E+5	278	1549	18	22	0.96
20	1.4E+5	313	1907	3	20	1.00
21	1.5E+5	606	835	4	13	0.96
22	2.2E+5	959	4513	7	42	1.00
23	2.5E+5	449	3634	2	40	1.00
24	3.8E+5	1540	6354	4	60	1.00

[a] Recorded are the approximate library size, time spent in product structure enumeration, time spent in product AP calculation, time spent in monomer AP descriptor calculation, and time spent in MoBSS AP enumeration. All times are in seconds. The similarity values were evaluated using the Dice coefficient as defined in eq 1.

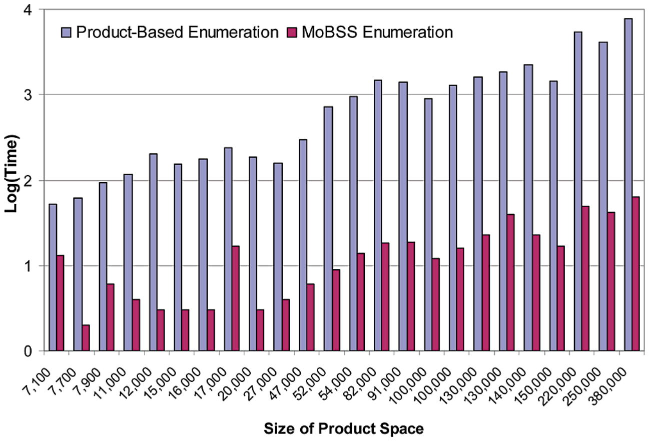


Figure 6. Plot of total CPU time in seconds for product AP descriptor enumeration using the product-based approach and MoBSS against the library size for 24 selected virtual reactions. The *y*-axis is on logarithmic scale.

monomer and B the query. Only the complete pairs in A are considered, and we have chosen $\alpha=1$ and $\beta=0$. Our particular choice for α and β parameters is used to emphasize the fraction of the common features in the monomer. The threshold value for Tversky similarity should be set to be slightly lower than the threshold value for the overall similarity. In our current implementation, it is taken to be 0.1 less than the overall similarity threshold. As the benchmarking test in the Results section suggests, we have found this prescreening step to generate considerable computational savings without compromising the quality of the results.

The MoBSS approximation is exact under two assumptions: 1) atoms in fragments retain their original types after fragments are joined to form products; and 2) two fragments are always joined at a single R-atom so that eq 3 holds to compute the shortest distance between two atoms from two separate fragments. In reality, though, neither assumption is true for all the reactions in our in-house repository. Figure 5 shows two examples that require some preprocessing in order for MoBSS to work. In Figure 5(a), the type of the nitrogen atom in the core is NX2. However, when the core is combined with the sample monomer, where the dummy R-atom is on a ring, the number of non-hydrogen neighbors becomes three. In Figure 5(b), the core and sample monomer are joined at two R-atoms: R1 and R2. The reaction was probably registered in this way so that monomers with different substitution patterns on the phenyl ring can be accommodated. However, for every pair of atoms between the core and the sample monomer, there exist two alternative paths, both of which constitute a valid atom pair in MoBSS. These examples, if left untreated, would cause the results of MoBSS calculations to be off by a considerable amount. Fortunately, most of these issues can be detected and addressed in the preprocessing stage at a low computational cost. For example, the problem in Figure 5(a) can be prevented by forcing a type change of the nitrogen atom to NX3 and saving the new type in the database. Similarly, the second problem can be addressed by finding all the ring formation reactions, clipping out the substructure in the core that is part of the new ring in the product structure, and performing a partial enumeration between the clipped substructure and the corresponding monomer to complete the ring. These procedures are similar to the ones that Boehm et al. employed in their work and have been shown to correct most of the problems.[26] However, there are still some reactions in our repository that were registered in ways that make them less amenable to corrective procedures. For example, some monomers might have only a hydrogen atom

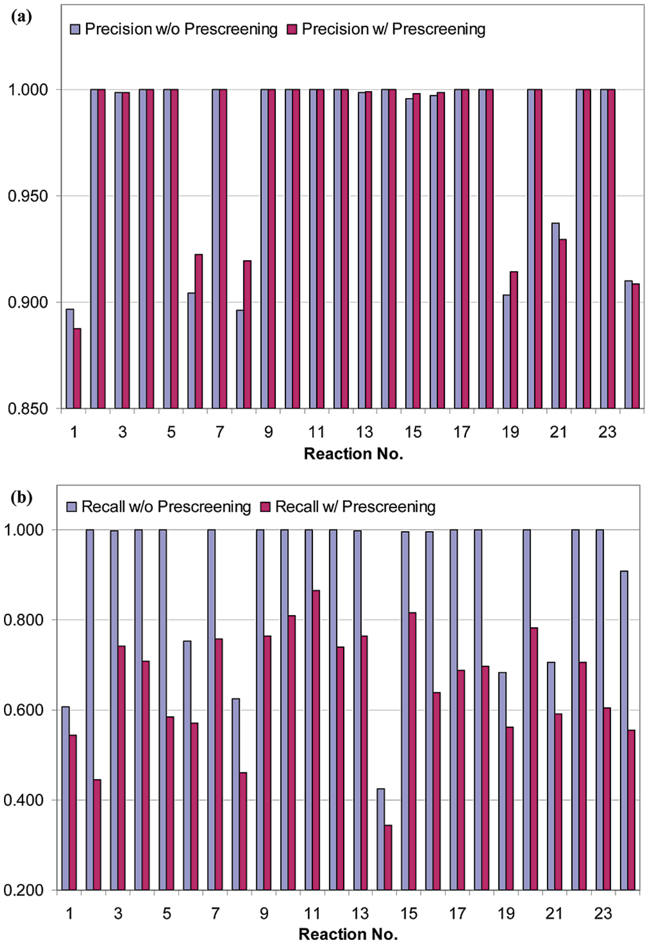


Figure 7. Average Precision (a) and Recall (b) statistics of the hits retrieved by MoBSS without and with prescreening for 24 selected virtual reactions.

in a certain R-group, which means the corresponding linkage atom in the core is effectively mistyped for these particular monomers because their number of heavy-atom neighbors is overcounted by one. Although solving issues like this example should not be an insurmountable obstacle, addressing every problem would require individual solutions, the implementations of which would complicate the code so much that they would make future maintenance much more difficult. Moreover, as we will demonstrate in the Results section, while failures of the MoBSS approximation cause some of the enumerated AP descriptors to be slightly different from those of the product-based approach, the effects might often not be as significant as those introduced by the prescreening step. Consequently, we made a decision of not pursuing a solution for every problematic reaction at this time.

Programming Details. The MoBSS program was written in ANSI C++ and compiled using the g++ compiler from GNU. The OEChem library from OpenEye Scientific Software, Inc.[33] provides a platform for most of the basic cheminformatics operations such as file I/O, connection table traversal, and substructure searching, which are utilized extensively throughout the program. MoBSS works with virtual reactions defined in the “core-plus-R-group” format, where each reaction is split into a core and two, three, or four components, each of which contains a set of clipped monomers. A separate utility program was written to preprocess the reactions, where some of the reactions that would pose difficulties for MoBSS are modified as outlined above, and calculate the AP descriptors for the cores and clipped monomers and store them in databases. This preprocessing step is utilized so that MoBSS can skip the generation of atom pairs for fragments which results in significant computational savings during similarity searches. After the preprocessing step is completed, MoBSS takes as input the structure of a query molecule, enumerates the atom pairs for all the virtual products on the fly, and returns the hit molecules whose similarity to the query are above the user-specified threshold as well as their associated reaction ID and constituent monomer IDs.

RESULTS AND DISCUSSION

Our in-house virtual reaction repository contains several hundred reactions and tens of thousands of unique monomers, defining a chemical space of several trillion virtual molecules. We have taken this collection and generated AP descriptor databases from the structures of the fragments. The utility program took about an hour of CPU time to preprocess 441 virtual reactions, which encodes a theoretical virtual product space of 5.6 trillion molecules. Care was taken to ensure the modifications we introduced in the preprocessing stage did actually produce the desired results. This was accomplished by selecting a few product structures for each reaction, calculating their AP descriptors, and attempting to recover them from the database with a similarity threshold value of 0.99. All the reactions that will be discussed later in this section passed this self-consistency test.

Method Validation and Performance Assessment. To validate MoBSS and assess its performance, a set of 24 small virtual reactions was selected from our in-house repository to assemble a test set. These virtual reactions span a variety of reaction types, and the size of the product spaces ranges between 7000 and 380,000. For each of the 24 reactions, we enumerated all the product structures, calculated the AP descriptors for all the products, and evaluated the similarity of the AP descriptors between the product-based approach and the MoBSS approach averaged over the whole product space for that reaction. It is worthwhile to point out that the computational test here only examines the enumeration aspect of MoBSS. Nevertheless, such a test is still expected to be an accurate performance benchmark for MoBSS because the computational cost of the similarity searching part should be small and equal for both approaches. The results of this test are collected in Table 4. Comparing the time spent in product structure enumeration and in MoBSS AP enumeration, it appears MoBSS does provide substantial time savings in the enumeration step. The core algorithm in both processes involves finding the compatible pairs and linking them, but MoBSS bypasses the generation of connection tables and the subsequent processing steps such as ring detection, aromaticity detection, etc., and replaces them with an arithmetic manipulation of the constituent partial atom pairs, which explains its speed improvement. However, the difference between these two processes is dwarfed when the time spent in calculating the atom pairs for the enumerated products is compared to that for clipped monomers. As explained earlier, this is because calculating the atom pairs for library products incurs repeated traversal of the connection tables of the recurring substructures, which is extremely inefficient. The small inaccuracies of MoBSS are attributed to the issues with the virtual reactions in our in-house repository discussed in the previous section.

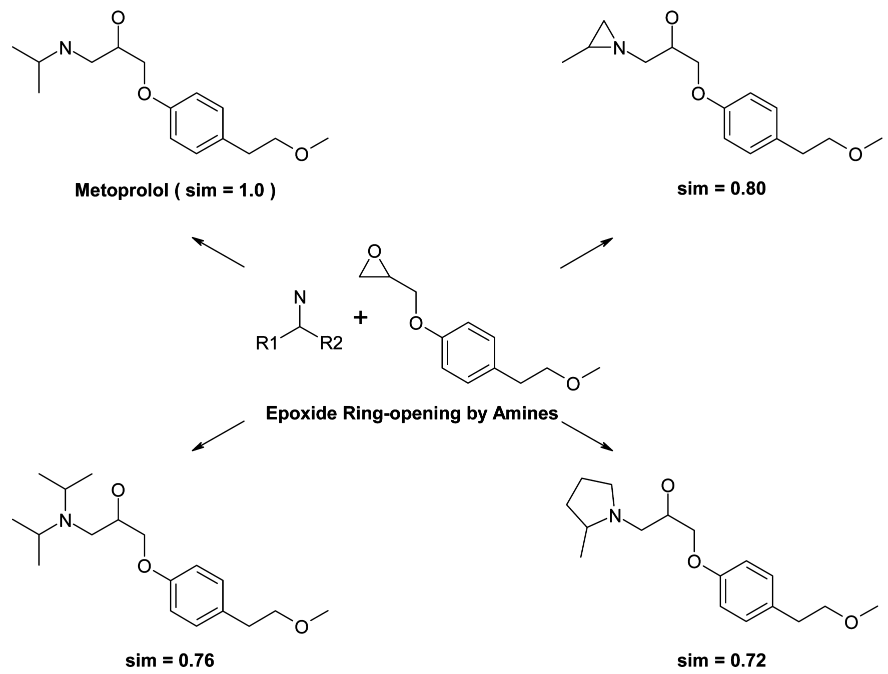


Figure 8. Analogs of metoprolol that could be synthesized using an epoxide ring-opening reaction by amines as suggested by MoBSS.

To further demonstrate the speed advantage of MoBSS, we plot the sum of the total CPU time in seconds required by the product-based approach as a function of the size of the product space for each of the 24 reactions in Figure 6 and contrast it to the sum of the total CPU time required by MoBSS. The CPU times are displayed on logarithmic scale so that the CPU times for both approaches can fit on the same graph. As Figure 6 shows, as the product space size increases the total computational cost for the product-based approach scales up almost exponentially, corroborating our view that even with the ever-increasing computing power the product-based approach will probably never be realistic for extremely large combinatorial chemistry spaces. On the other hand, the MoBSS CPU time shows a much more favorable profile in that for most of the reactions in this test MoBSS generated computational savings of a few orders of magnitude. However, the CPU time for MoBSS also shows steady growth as the size of the product space increases, albeit at a lower rate than for the product-based approach, necessitating the adoption of the prescreening step in practical applications of MoBSS to virtual reaction repositories with trillions of products.

Next, we decided to test the similarity searching aspect of MoBSS on the same set of 24 small virtual reactions. For each reaction, 20 molecules were randomly selected from its product space, and independent product-based AP similarity searches were performed using a threshold value of 0.7 against the precomputed AP descriptors obtained in the previous step. This was followed by 20 MoBSS runs using the same query molecules and the same threshold value without and with prescreening. A threshold value of 0.6 was used for the prescreening step. This test allowed us to compare the hits found by MoBSS to those of product-based searches and assess their accuracy with the Precision and Recall statistics as defined below

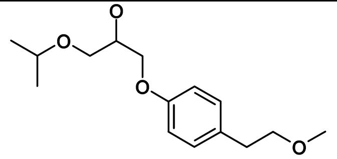
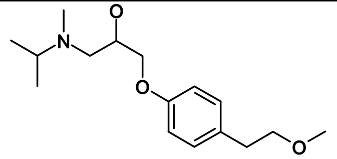
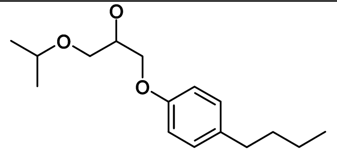
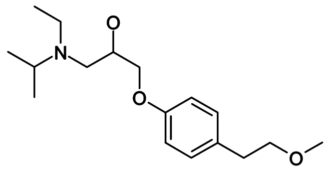
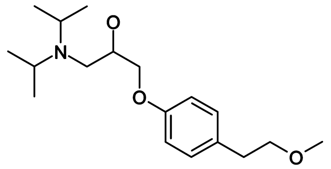
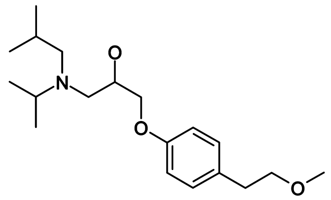
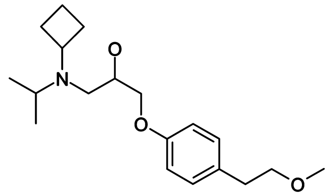
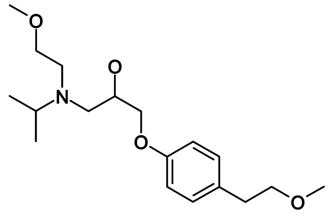
$$Precision = \frac{number\ of\ common\ hits}{number\ of\ MoBSS\ hits} \tag{6}$$

$$Recall = \frac{number\ of\ common\ hits}{number\ of\ product - based\ hits} \tag{7}$$

where *number of product-based hits* is the number of hits retrieved by the product-based search, *number of MoBSS hits* is the number of hits retrieved by the MoBSS search, and *number of common hits* is the number of hits common between the two searches. It also allows us to estimate the impact of prescreening on the accuracy of the similarity searches. Figure 7 reports the Precision and Recall values averaged over the 20 independent similarity searches for our test reactions. Figure 7(a) demonstrates the remarkable precision of MoBSS, which is mostly close to one with a minimum of 0.88. It is also interesting to note for reactions 6, 8, and 19 the precision of MoBSS with prescreening is slightly better than MoBSS without prescreening, suggesting that prescreening can sometimes partially offset the errors introduced by the MoBSS approximation. Figure 7(b) shows the MoBSS approximation without prescreening gave a recall rate of close to one for 19 out of 24 reactions. It also indicates the prescreening step with a threshold used in this test can result in a significant loss of true hits due to the constituent monomers being filtered out, which is obviously the case for the second reaction. On inspection, it turned out that the products missed by prescreening for this reaction all have a very large monomer for the B component and a very small monomer for the A component. In such cases, the smaller monomer has very little net effect on the overall similarity value when compared to full product structures, but on its own is dissimilar enough that it fails qualification. This result is not unexpected and can be mitigated by adjusting the threshold value used in prescreening in order to achieve an optimal balance between efficiency and accuracy. Lastly, it is important to note that the reactions for which MoBSS both without and with prescreening performed poorly in this test are mostly the ones that have shown problems in Table 4, namely reactions 1, 6, 8, 14, 19, and 21, which explains why they earned relatively low scores on Precision and Recall as well.

Application to Metoprolol. To demonstrate how MoBSS may be used in real-world drug discovery projects, we took the drug metoprolol, a selective β_1 receptor blocker used in the treatment of several diseases of the cardiovascular system, as the query and ran a MoBSS search against an in-house

Table 5. Analogs of Metoprolol Listed in the Order of Decreasing Similarity to Metoprolol That Could Potentially Be Synthesized Using Four Other Types of Reactions than the Synthetic Route in Figure 8

Analog	AP Similarity	Reaction Description
	0.89	Epoxide ring-opening by alchols
	0.85	N-alkylation of secondary amines with alkyl halides
	0.81	Epoxide ring-opening by alchols
	0.80	Reductive amination; Amide bond formation followed by amide reduction
	0.76	Reductive amination
	0.72	Reductive amination
	0.72	Reductive amination
	0.72	Reductive amination

collection of 441 reactions using a similarity threshold value of 0.7. The calculation took less than two minutes on a Linux PC with a 3.0 GHz CPU and 4 GB of memory running Red Hat Enterprise Edition 5.0. As Figure 8 shows, one of the hits that MoBSS found has an identical structure to metoprolol, which MoBSS suggested could be synthesized using epoxide ring-opening reaction by amines. Within the product space covered by the same reaction, MoBSS also suggested a few other analogs that can be synthesized by varying the amine component; the three most similar analogs are also shown in Figure 8. Not only was MoBSS able to find the epoxide ring-opening reaction as a route to synthesize these analogs but it found four other types of reactions as potential alternative reactions to make closely related analogs. These alternative reactions include epoxide ring-opening by alcohols, reductive amination, N-alkylation of secondary amines with alkyl halides, and amide bond formation followed by amide reduction. The products of these reactions most similar to metoprolol are listed in Table 5.

Application to Sorafenib. Sorafenib, our second example, is a multikinase inhibitor for treating cancer. We ran a MoBSS search against our virtual library using sorafenib as the query with an AP similarity cutoff of 0.7. The search took less than 8 min of CPU time and returned 205 hits. We grouped the hits according to their associated reactions, sorted them by decreasing similarity, and show the four top-ranking hits from four different reactions in Figure 9. As MoBSS suggests, three of the analogs could be synthesized using two-component reactions: analog 1 could be synthesized by nucleophilic substitution of a heteroaromatic halide with a phenol; analog 3 could be synthesized as a urea from an amine and an isocyanate; and analog 4 could be synthesized as a sulfonylurea from an isocyanate and a sulfonyl chloride. Analog 2 has to be synthesized with a three-component reaction by aryl ether formation followed by BOC deprotection and urea formation, but this reaction can be used to make 92 analogs above the similarity cutoff of 0.7. The other three reactions only provide 33, 43, and 4 hits, respectively.

CONCLUSIONS

We have presented the Monomer-Based Similarity Searching algorithm to rapidly calculate AP descriptors for library products which enables efficient similarity searches of large combinatorial product spaces. This algorithm derives its speed by performing descriptor calculations in the monomer space as opposed to the product space and obtains product descriptors through an arithmetic manipulation of partial atom pairs of the constituent fragments. A validation study on a set of 24 small libraries suggested the MoBSS approximation can yield computational savings of a few orders of magnitude compared to the product-based approach. It further demonstrates the AP descriptors calculated by the MoBSS approximation are close to exact. Analysis of the results of similarity searches by MoBSS using the Precision and Recall statistics shows the accuracy of MoBSS without applying the prescreening step is very good and the effects of prescreening are generally small, which can be further reduced by fine-tuning the threshold value.

Applications of MoBSS to search our in-house virtual reaction repository using two drug molecules as queries have also been conducted and showed MoBSS can find several synthetic routes for preparing the queries and their chemical analogs. These results establish the utility of MoBSS as a similarity searching and lead hopping tool to efficiently navigate combinatorial chemistry spaces with sizes on the order of trillions of molecules. A notable change in chemical motif can be seen in the first and third molecules in Table 5 where the amino nitrogen of metoprolol is replaced with an oxygen atom, and the resulting molecules are calculated to be more similar to metoprolol than many other analogs

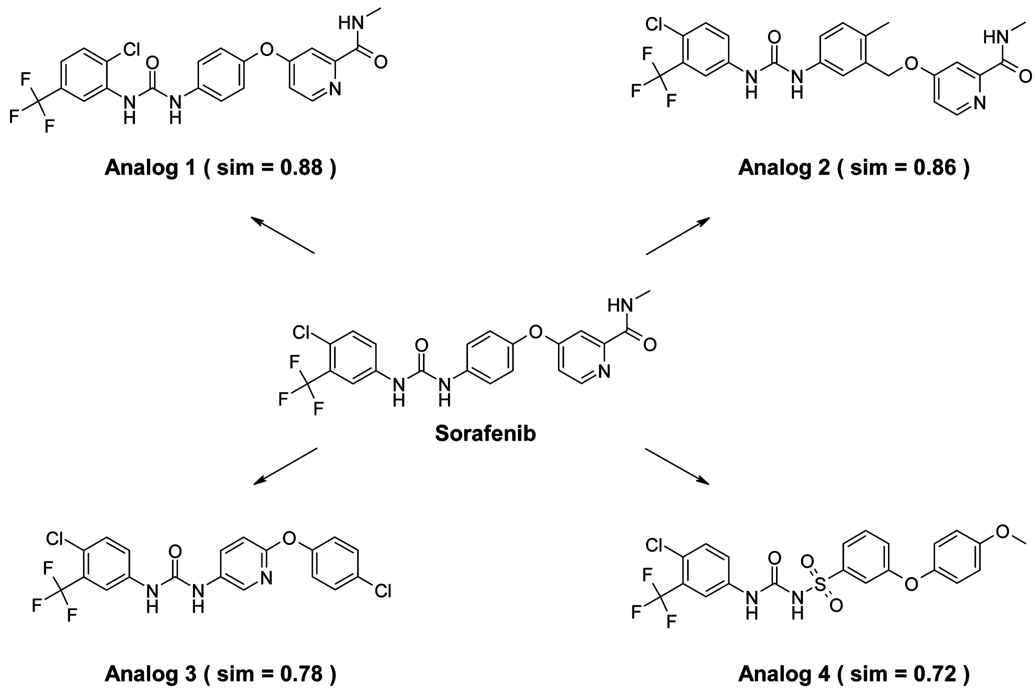


Figure 9. Four analogs of sorafenib ranked on top by AP similarity in the product spaces of their respective virtual reactions.

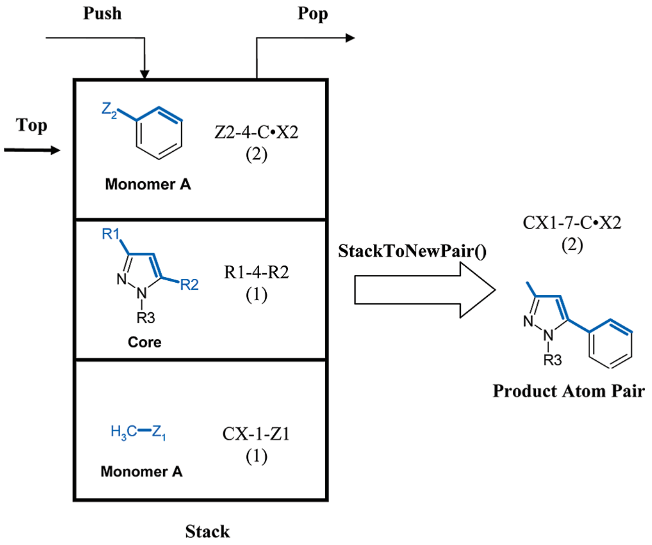


Figure 10. Snapshot of the stack data structure used by the MoBSS algorithm to accumulate the compatible partial atom pairs which are joined together to form a new atom pair in the intermediate product. The substructures highlighted in blue correspond to the AP features listed next to them. The numbers in parentheses are feature counts.

preserving the nitrogen. This brings up the possibility of using MoBSS as a lead hopping tool in addition to a similarity searching tool.[34−39] Experience has shown that AP descriptors have the capability of relating seemingly remote chemical analogs in a way not seen with other types of substructure descriptors. Notwithstanding the few rare anomalies that may arise in AP-based similarity searches, the pairwise similarity values evaluated with AP descriptors have been shown to correlate quite well with those computed with path-dependent descriptors.[40] Thus the differences that MoBSS accommodates are still very conservative, in contrasts to the lead hopping approach based on Feature Trees descriptors employed by Boehm et al., which can introduce considerably larger structural perturbations to query molecules such as different rings, regio-products, etc.[26] In the context of lead hopping in synthetically accessible combinatorial chemistry spaces, it seems that each method has its unique characteristics and applicability. Therefore, it is our position that the user of these methods bears the important responsibility of choosing the appropriate tool for the specific drug discovery project at hand based on a sound understanding of the underpinnings of each method.

Finally, it is also conceivable that MoBSS may be extended beyond its currently intended area of application and utilized as a library profiling tool. Indeed, QSAR models that predict molecular properties based on AP descriptors alone or combinations of AP and other descriptors can all benefit from the speed of MoBSS. This will undoubtedly lead to many interesting new possibilities in the related areas such as combinatorial library design and optimization.

ACKNOWLEDGMENT

We thank our Pfizer colleagues Meihua Tu for writing the initial version of the atom pair code and Scot Mente, Shaughn Robinson, Markus Boehm, and Christopher Keefer for critical reading of the manuscript and helpful suggestions.

APPENDIX

MoBSS Algorithmic Design. We start with the recognition that enumeration of AP descriptors for a combinatorial product from those of the constituent core and clipped monomers can be performed in the same stepwise fashion as structure enumeration. In each step, atom pairs from a core are combined with those from a clipped monomer using the Compatibility Rule. At the beginning of such a step, MoBSS first collects the complete atom pairs in each fragment and deposits them in the product list. It then invokes a recursive algorithm to form new product atom pairs from compatible partial atom pairs in the fragments. This algorithm is necessarily recursive in order to handle partial atom pairs that are spread across several disconnected fragments, such as those in clipped monomer A in our example in Figure

Scheme 1. Pseudocode in a C++-like Syntax for Calculating AP Descriptors for an Intermediate Product by Combining the Pairs from a Core and a Clipped Monomer[a]

```
function EnumerateAtomPairDescriptors()
{
    core = 0;
    monomer = 1;
    for pPair <- partialPairs.oneRs[monomer, 1:n]
    {
        seekMol = core;
        seekType = pPair.rGroupTypes[0];
        partialPairsInit[core, 1:n] = partialPairs[core,1:n];
        partialPairsInit[monomer, 1:n] =
            partialPairs[monomer, 1:n].IncompatibleSubSet(seekType);
        stack.push(pPair);
        DepthFirstTraversal(seekMol, seekType, partialPairsInit, stack);
        stack.pop(pPair);
    }

    if partialPairs.twoRs[monomer].HasNoElement()
        return;

    for pPair <- partialPairs[core, 1:n]
    {
        seekMol = monomer;
        if pPair.HasOneRGroup() AND
            partialPairs.twoRs[monomer].IsCompatible(pPair.rGroupTypes[0])
        {
            seekType = pPair.rGroupTypes[0];
        }
        else if pPair.HasTwoRGroups() AND
            partialPairs.twoRs[monomer].IsCompatible(pPair.rGroupTypes[0])
        {
            seekType = pPair.rGroupTypes[0];
        }
        else if pPair.HasTwoRGroups() AND
            partialPairs.twoRs[monomer].IsCompatible(pPair.rGroupTypes[1])
        {
            seekType = pPair.rGroupTypes[1];
        }
        else
            continue;
        partialPairsInit[core, 1:n] =
            partialPairs[core,1:n].IncompatibleSubSet(seekType);
        partialPairsInit[monomer, 1:n] = partialPairs.twoRs[monomer, 1:n];
        stack.push(pPair);
        DepthFirstTraversal(seekMol, seekType, partialPairsInit, stack);
        Stack.pop(pPair);
    }
    return;
function DepthFirstTraversal(seekMol, seekType, partialPairs, stack)
{
    for pPair <- partialPairs[seekMol, 1:n]
    {
        if  NOT pPair.IsCompatible(seekType)
            continue;
        stack.push(pPair);

        if  NOT pPair.otherEnd.IsRGroup() OR
            NOT partialPairs[1-seekMol,1:n].IsCompatible(pPair.otherEndType)
        {
            StackToNewPair(stack);
        }
        else {
            seekMolNext = 1 - seekMol;
            seekTypeNext = pPair.otherEndType;
            partialPairsNext[seekMol,1:n] =
                partialPairs[seekMol,1:n].IncompatibleSubSet(seekTypeNext);
            partialPairsNext[seekMolNext,1:n] = partialPairs[seekMolNext,1:n];
            DepthFirstTraversal(seekMolNext, seekTypeNext,
                                 partialPairsNext, stack);
        }
        stack.pop(pPair);
    }
    return;
}
```

[a] The italicized words are function names. The subprocedures *IncompatibleSubSet(), HasNoElement(), HasOneRGroup(), HasTwoRGroups(), Compatible(),* and *StackToNewPair()* are straightforward to implement, whose pseudocode is thus omitted from presentation.

4. A depth-first traversal is utilized to accumulate compatible pairs on a stack data structure. At the bottom of the stack is a partial atom pair with exactly one compatible end. Examples of the bottom of the stack are CX1-1-Z1 and R1-4-R3 in Table 3, in which Z1 and R1 are the outstanding compatible groups, respectively. The algorithm then alternates between the core and the monomer and tries to find the next compatible atom pair and push it into the stack. The accumulation terminates when no compatible atom pair from the other fragment can be found for the partial atom pair on the top of the stack, at which point the content of the stack is taken to enumerate a new atom pair for the intermediate product. After this, back-tracking of the stack follows. Figure 10 shows a snapshot of the stack containing three partial atom pairs at a certain point during recursion and illustrates how function *StackToNewPair* may combine the partial atom pairs into a complete atom pair using eqs 3 and 4. Scheme 1 shows the pseudocode of the key functions utilizing a recursive algorithm to accumulate compatible atom pairs from a core and a clipped monomer in a stack data structure.

REFERENCES AND NOTES

(1) Johnson, M.; Lajiness, M.; Maggiora, G. Molecular similarity: a basis for designing drug screening programs. *Prog. Clin. Biol. Res.* **1989**, *291*, 167–71.

(2) Lajiness, M. S.; Johnson, M. A.; Maggiora, G. M. Implementing drug screening programs using molecular similarity methods. *Prog. Clin. Biol. Res.* **1989**, *291*, 173–6.

(3) Patterson, D. E.; Cramer, R. D.; Ferguson, A. M.; Clark, R. D.; Weinberger, L. E. Neighborhood Behavior: A Useful Concept for Validation of "Molecular Diversity" Descriptors. *J. Med. Chem.* **1996**, *39* (16), 3049–3059.

(4) Willett, P.; Barnard, J. M.; Downs, G. M. Chemical Similarity Searching. *J. Chem. Inf. Comput. Sci.* **1998**, *38* (6), 983–996.

(5) 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.

(6) Brown, R. D.; Martin, Y. C. The Information Content of 2D and 3D Structural Descriptors Relevant to Ligand-Receptor Binding. *J. Chem. Inf. Comput. Sci.* **1997**, *37* (1), 1–9.

(7) Matter, H.; Potter, T. Comparing 3D Pharmacophore Triplets and 2D Fingerprints for Selecting Diverse Compound Subsets. *J. Chem. Inf. Comput. Sci.* **1999**, *39* (6), 1211–1225.

(8) Martin, E. J.; Blaney, J. M.; Siani, M. A.; Spellmeyer, D. C.; Wong, A. K.; Moos, W. H. Measuring Diversity: Experimental Design of Combinatorial Libraries for Drug Discovery. *J. Med. Chem.* **1995**, *38* (9), 1431–1436.

(9) Langer, T.; Wolber, G. Virtual combinatorial chemistry and in silico screening: Efficient tools for lead structure discovery. *Pure Appl. Chem.* **2004**, *76* (5), 991–996.

(10) Schuster, D.; Nashev, L. G.; Kirchmair, J.; Laggner, C.; Wolber, G.; Langer, T.; Odermatt, A. Discovery of nonsteroidal 17beta-hydroxysteroid dehydrogenase 1 inhibitors by pharmacophore-based screening of virtual compound libraries. *J. Med. Chem.* **2008**, *51* (14), 4188–99.

(11) Schneider, G.; Lee, M. L.; Stahl, M.; Schneider, P. De novo design of molecular architectures by evolutionary assembly of drug-derived building blocks. *J. Comput.-Aided Mol. Des.* **2000**, *14* (5), 487–94.

(12) Nikitin, S.; Zaitseva, N.; Demina, O.; Solovieva, V.; Mazin, E.; Mikhalev, S.; Smolov, M.; Rubinov, A.; Vlasov, P.; Lepikhin, D.; Khachko, D.; Fokin, V.; Queen, C.; Zosimov, V. A very large diversity space of synthetically accessible compounds for use with drug design programs. *J. Comput.-Aided Mol. Des.* **2005**, *19* (1), 47–63.

(13) Leland, B. A.; Christie, B. D.; Nourse, J. G.; Grier, D. L.; Carhart, R. E.; Maffett, T.; Welford, S. M.; Smith, D. H. Managing the Combinatorial Explosion. *J. Chem. Inf. Comput. Sci.* **1997**, *37* (1), 62–70.

(14) Agrafiotis, D. K.; Lobanov, V. S.; Salemme, F. R. Combinatorial informatics in the post-genomics ERA. *Nat. Rev. Drug Discovery* **2002**, *1* (5), 337–46.

(15) Rogers, D.; Brown, R. D.; Hahn, M. Using extended-connectivity fingerprints with Laplacian-modified Bayesian analysis in high-throughput screening follow-up. *J. Biomol. Screening* **2005**, *10* (7), 682–6.

(16) Shi, S.; Peng, Z.; Kostrowicki, J.; Paderes, G.; Kuki, A. Efficient combinatorial filtering for desired molecular properties of reaction products. *J. Mol. Graphics Modell.* **2000**, *18* (4−5), 478–496.

(17) Lobanov, V. S.; Agrafiotis, D. K. Combinatorial networks. *J. Mol. Graphics Modell.* **2001**, *19* (6), 571−8610−3.

(18) *Daylight Theory Manual, 4.9*; Daylight Chemical Information, Inc.: Aliso Viejo, CA.

(19) *Pipeline Pilot, 6.1*; Accelrys: San Diego, CA.

(20) *Symyx, 2.5*; Symyx Technologies: San Ramon, CA.

(21) McGregor, M. J.; Pallai, P. V. Clustering of Large Databases of Compounds: Using the MDL "Keys" as Structural Descriptors. *J. Chem. Inf. Comput. Sci.* **1997**, *37* (3), 443–448.

(22) *BCI, 1.0*; Barnard Chemical Information Ltd.: Stannington, Sheffield S6 6BX, U.K.

(23) Downs, G. M.; Barnard, J. M. Techniques for Generating Descriptive Fingerprints in Combinatorial Libraries. *J. Chem. Inf. Comput. Sci.* **1997**, *37* (1), 59–61.

(24) Barnard, J. M.; Downs, G. M.; von Scholley-Pfab, A.; Brown, R. D. Use of Markush structure analysis techniques for descriptor generation and clustering of large combinatorial libraries. *J. Mol. Graphics Modell.* **2000**, *18* (4−5), 452–63.

(25) Downs, G. M.; Gillet, V. J.; Holliday, J. D.; Lynch, M. F. Computer storage and retrieval of generic chemical structures in patents. 10. Assignment and logical bubble-up of ring screens for structurally explicit generics. *J. Chem. Inf. Comput. Sci.* **1989**, *29* (3), 215–224.

(26) Boehm, M.; Wu, T. Y.; Claussen, H.; Lemmen, C. Similarity searching and scaffold hopping in synthetically accessible combinatorial chemistry spaces. *J. Med. Chem.* **2008**, *51* (8), 2468–80.

(27) Rarey, M.; Dixon, J. S. Feature trees: a new molecular similarity measure based on tree matching. *J. Comput.-Aided Mol. Des.* **1998**, *12* (5), 471–90.

(28) Rarey, M.; Stahl, M. Similarity searching in large combinatorial chemistry spaces. *J. Comput.-Aided Mol. Des.* **2001**, *15* (6), 497–520.

(29) Ivanciuc, O.; Klein, D. J. Computing Wiener-Type Indices for Virtual Combinatorial Libraries Generated from Heteroatom-Containing Building Blocks. *J. Chem. Inf. Comput. Sci.* **2002**, *42* (1), 8–22.

(30) Carhart, R. E.; Smith, D. H.; Venkataraghavan, R. Atom pairs as molecular features in structure-activity studies: definition and applications. *J. Chem. Inf. Comput. Sci.* **1985**, *25* (2), 64–73.

(31) Bush, B. L.; Sheridan, R. P. PATTY: A programmable atom type and language for automatic classification of atoms in molecular databases. *J. Chem. Inf. Comput. Sci.* **1993**, *33* (5), 756–762.

(32) Kearsley, S. K.; Sallamack, S.; Fluder, E. M.; Andose, J. D.; Mosley, R. T.; Sheridan, R. P. Chemical Similarity Using Physiochemical Property Descriptors. *J. Chem. Inf. Comput. Sci.* **1996**, *36* (1), 118–127.

(33) *OEChem, 1.6.1*; OpenEye Scientific Software, Inc.: Santa Fe, NM, 2008.

(34) Cramer, R. D.; Jilek, R. J.; Guessregen, S.; Clark, S. J.; Wendt, B.; Clark, R. D. ”Lead hopping. Validation of topomer similarity as a superior predictor of similar biological activities. *J. Med. Chem.* **2004**, *47* (27), 6777–91.

(35) Barker, E. J.; Buttar, D.; Cosgrove, D. A.; Gardiner, E. J.; Kitts, P.; Willett, P.; Gillet, V. J. Scaffold hopping using clique detection applied to reduced graphs. *J. Chem. Inf. Model.* **2006**, *46* (2), 503–11.

(36) Renner, S.; Schneider, G. Scaffold-hopping potential of ligand-based similarity concepts. *ChemMedChem* **2006**, *1* (2), 181–5.

(37) Zhang, Q.; Muegge, I. Scaffold hopping through virtual screening using 2D and 3D similarity descriptors: ranking, voting, and consensus scoring. *J. Med. Chem.* **2006**, *49* (5), 1536–48.

(38) Bergmann, R.; Linusson, A.; Zamora, I. SHOP: scaffold HOPping by GRID-based similarity searches. *J. Med. Chem.* **2007**, *50* (11), 2708–17.

(39) Zhao, H. Scaffold selection and scaffold hopping in lead generation: a medicinal chemistry perspective. *Drug Discovery Today* **2007**, *12* (3−4), 149–55.

(40) Chen, X.; Reynolds, C. H. Performance of similarity measures in 2D fragment-based similarity searching: comparison of structural descriptors and similarity coefficients. *J. Chem. Inf. Comput. Sci.* **2002**, *42* (6), 1407–14.

CI800392Z