SwiFT: An Index Structure for Reduced Graph Descriptors in Virtual Screening and Clustering

J. Robert Fischer and Matthias Rarey*

Center for Bioinformatics Hamburg, University of Hamburg, Bundesstrasse 43, D-20146 Hamburg, Germany

Received January 22, 2007

A reduced graph descriptor represents molecules by small node-labeled graphs. They allow fast similarity calculation, while retaining the overall arrangement of functional groups. The feature tree as an example of this descriptor type abstracts a molecule by a node-labeled, unrooted tree. One available algorithm for pairwise feature tree comparison is the match-search algorithm, which matches the subtrees of two feature trees on each other and therefore creates an alignment. In this work, we document the extension to reuse partial results on the global level of the whole feature tree data set where a high number of identical subtrees exists. The method is based on indexing all occurring subtrees in a data set. On the basis of this index, the similarity value between every subtree combination has to be computed only once. While calculating identical similarities, this approach leads to a substantial reduction in run time by up to 80% and can be used in a parallel computation environment. The search tree built for indexing can also be used to identify duplicated feature trees.

INTRODUCTION

A major concept in the search for new lead structures in drug design is molecular similarity between small organic compounds. Already known bioactive compounds are used to select a small set of compounds which are likely to be bioactive out of a huge data set with thousands or millions of candidates.[1] This approach is motivated by the similarity principle[2] and the lock and key concept,[3] which says that there is a high chance for molecules with similar physicochemical properties to show a similar binding profile to that of proteins. Though arguable, this concept has proven to be successful in numerous cases.[4−7] With increasing library sizes to screen, computing time becomes an important factor.

Furthermore, similarity-based methods are used as pre- and postprocessing steps in drug design projects. Exemplarily, clustering is used to analyze the diversity of data sets by grouping similar compounds into families. Creating the distance matrix for 10 000 compounds, ∼50 million pair-comparisons have to be made. Again, this makes the computing time of the similarity measure an important factor.

Regarding the increasingly huge molecule data sets, a fast comparison approach for molecules is required. The molecular descriptor should be efficiently comparable while preserving important physicochemical properties. A lot of different descriptors are available, which can be roughly differentiated in three groups: There are descriptors coding macromolecular properties like the molecular weight or descriptors retrieved from topological or 3D information. Examples for topological descriptors are 2D fingerprints storing the occurrence/absence of small fragments, while multiple-point pharmacophores are 3D descriptors storing the spatial arrangement of functional groups which might be relevant for binding.[8,9] Most of the descriptors in use store the molecular properties in a vector format which can be evaluated very quickly by similarity measurements and coefficients.[4]

The feature tree descriptor[10−12] as implemented in the FTrees program[13] overcomes the conformation problem by being a topological descriptor. It represents the molecule by a tree structure conserving the overall topology without considering the conformational space. This idea is based on the reduced representation of molecular graphs proposed earlier.[14−17] In contrast to the comparison of reduced graphs via pseudo-SMILES[15] or an edit distance approach,[18,19] feature trees are compared by matching subtrees on each other. Since this is more complex than using a distance metric or similarity coefficient for vector comparison, substantially longer computing times have to be accepted. When dealing with huge data sets, a speedup is desirable for reduced graph descriptors like feature trees.

Here, we present a new algorithm for multiple feature tree comparisons based on the idea that, in a huge data set, a great amount of fragments, in this case, FTrees−subtrees, occur more than once. We use an indexing method to reuse redundant information and, therefore, avoid the multiple calculation of the similarity of subtree combinations already computed. This results in a substantial reduction in run time. We call this method SwiFT: "searching with indexed feature trees". Additionally, the indexing method can be used to identify identical feature trees without comparing all feature trees. Thus, a redundance-free feature tree data set can be generated very fast which decreases the comparison run time as well. Note that this is a duplicate analysis on the more abstract level of reduced graph descriptors in comparison to the identification of redundant molecules (e.g., Baurin et al.[20]).

In the following, we present the design of SwiFT validated by some experiments using several typical molecule data sets. After a short introduction to feature trees and the comparison

* Corresponding author. Phone: +49 40 428387351. Fax: +49 40 428387352. E-mail: rarey@zbh.uni-hamburg.de.

10.1021/ci700007b CCC: $37.00
Published on Web 06/14/2007

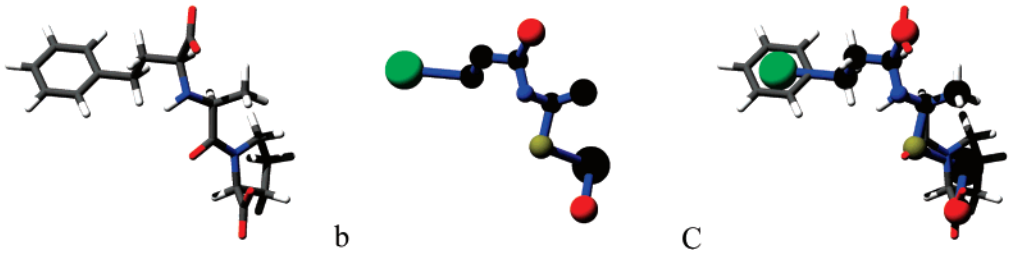

Figure 1. (a) The molecular graph of captopril, an inhibitor of ACE (angiotensin converting enzyme).[21] (b) Illustration of the corresponding feature tree descriptor. For better understanding, the feature tree descriptor is shown with the same 3D coordinates as the molecular graph. (c) Superposition of the molecule and its feature tree. The molecular building blocks are condensed to feature tree nodes (pictures generated with FlexV[22]).

algorithm match-search (for a more detailed description, see Rarey and Dixon[10]), we describe the methods for index creation, duplicate identification, and the extension of the match-search algorithm in detail. In the results section, we analyze the index creation and identification of duplicate feature tree descriptors, followed by a comparison of the computing times using SwiFT and the standard match-search algorithm.

METHODS

Feature Trees. The feature tree descriptor represents molecules in an abstract way by an unrooted tree where the nodes are labeled with the sterical and physicochemical properties of the original molecule building blocks (information is stored with respect to size, number of hydrogen donors, number of hydrogen acceptors, hydrophobicity, etc.). In principle, the edges of a feature tree represent the rotatable bonds of the molecule, preserving the overall topology and connectivity of the molecule, and the feature tree nodes are the connected fragments that remain. As a consequence of the level of abstraction, it can be presumed that scaffold hopping between chemical classes with similar biological activity is also possible. Figure 1 shows the molecular graph of Captopril[21] and its corresponding feature tree.

Match-Search Algorithm. One available comparison algorithm in the FTrees program is the so-called "match-search algorithm".[10,12] Here, the comparison of two feature trees is based on finding the best possible matching by aligning subtrees onto each other. A "matching" is now defined as the enumeration of aligned subtrees ("match"). The input to this algorithm consists of two rooted subtrees ("rooted" means that a node is explicitly marked as a root). To obtain such subtrees, the algorithm needs to start with an initial split of the two feature trees. Since the nondirected feature tree edges are described by a pair of antiparallel edges, a "split" is a pair of directed "cuts"—splitting both feature trees into two rooted subtrees respectively, see Figure 2 for an example. To simplify matters, we will call them "subtrees" from here on. The output of the match-search algorithm is a similarity value and a list of splits and matches defining the subdivision and match of the subtrees. Note that only splittings will be initially done, which result in balanced subtree pairs within a given size threshold. This results in a heuristical restriction of the search space.

In summary, the five computational phases of the match-search algorithm are listed below. Additionally, the match-search algorithm containing small modifications is shown in Figure 3. A detailed description of the algorithm is given by Rarey and Dixon.[10] The computational phases are as follows: (1) search for initial splits in the two feature trees; (2) first recursive call for matching subtrees; (3) extend the matching by a match (a set of nodes) containing both root nodes ("extension match"); due to the fact that the root nodes are strictly included, this leads to a connected set of matches; (4) recursive calls for all combinations of subtrees resulting from separating the extension matches; (5) search for the maximum overall similarity value; this is the highest similarity value between any subtree combinations obtained by the splits.

The fact that every input for the algorithm differs only in the cut which separates the subtrees allows it to reuse already calculated partial similarities by applying a dynamic programming matrix.[23] This means that the match-search algorithm uses the dynamic programming matrix on a "local level". We speak of "local" in the context of a single pair comparison of two feature trees and of "global" in the context of large data sets further on.

The SwiFT Indexing Method. Due to the fact that a high number of identical subtrees exists in a data set, the identification of the redundant subtrees leads to some computational advantages in terms of computing time. To retrieve the information of redundancy, an index is created identifying all identical subtrees and storing a nonredundant set. This is done using a search tree based on a novel linear subtree descriptor which can be easily compared. On the basis of this index, we implemented two different methods. When the index is applied to the match-search algorithm, every subtree combination has to be computed only once. Therefore, all partial results are stored in a global matrix, and indexed access to this matrix is given (Figure 4). The second method identifies duplicate feature tree descriptors in a data set using the search tree, making the pairwise comparison of all compounds unnecessary.

Index Creation. For a better understanding, we outline the mechanism of index creation, before the single steps are described in more detail (see also Figure 2 for an example):

- In an initial step, all subtree descriptors of a feature tree are generated. Due to the fact that every undirected feature tree edge is represented by two antiparallel directed edges, we get two distinct subtrees by a single edge cut (see Figure 5). So in conclusion, there are $2n$ subtrees for a feature tree with n edges.

- Subsequently, we generate the linear subtree descriptor using the cut-directed edge as the root for each subtree. From here, the levels of the subtree are condensed to elements of a list, for example, the properties of a node at a distance i to the root are added to list element i.

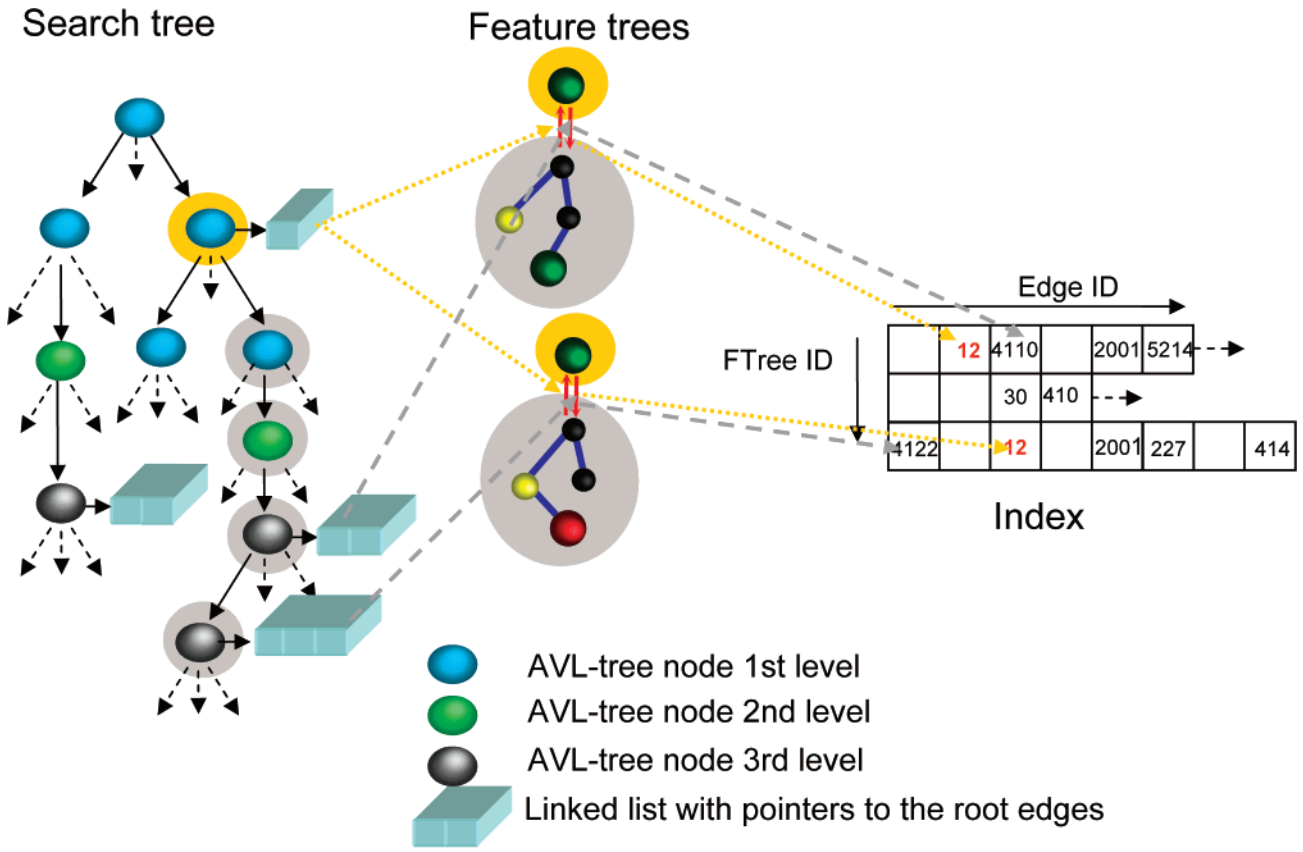

Figure 2. Underlying data structures of SwiFT. On the left, a cutout of a search tree is shown. The dashed arrows indicate hidden tree areas. An additional pointer (e.g., from a blue to a green node) connects to an AVL tree for the next descriptor level. The different levels are displayed in different colors. Nodes which represent the last level of a descriptor possess a list with elements pointing to a representing subtree. For each subtree, the index value can be accessed via the feature tree ID and the ID of the edge cut for separation from the feature tree. Exemplarily, each of the two feature trees shown is split into two subtrees (the pair of antiparallel edges, marked red). The small subtrees (yellow marked) are identical and use, therefore, the same node (yellow marked) in the search tree and get the same index value. The larger subtrees (gray marked) differ in the last level. They use the same (prefix) nodes for the first and second level, but different nodes in the third level (gray marked). They also get different index values.

• We insert the linear subtree descriptor in a search tree. The search tree is nested so that every level of the descriptor is handled separately. This means that each search tree node consists of a link to a search tree for the next level. Therefore, a search tree for level i is only reached by the linear subtree descriptors, which have the identical $i-1$ "prefix" levels using the identical $i-1$ "prefix" nodes. If no corresponding node exists in this search tree, a new one is generated. If the node for the last level of a descriptor is found, we apply the following step.

• To resolve collisions between subtrees with the same linear subtree descriptor, the new inserted subtree is matched on all subtrees in the list appended to this search tree node. If two subtrees are identical, the new subtree gets the same index value. If it is the second identical subtree, a new index value is generated. Otherwise, the new subtree is inserted in the list but does not get an index value.

• When two subtrees are compared with the match-search algorithm, the cell in the global matrix can be accessed by the index values. In turn, these values are obtained by the feature tree IDs and the IDs of the root edge.

Linear Subtree Descriptor. The linear subtree descriptor condenses the levels of a subtree, starting at the root (see Figure 5) and combining the properties of nodes with the same distance from the root. For most properties, this is a simple addition of the property values.[24] Additionally, it stores the number of condensed nodes on this level. The descriptor abstracts a subtree but conserves its overall topological coherence by coding the properties of the nodes in reference to the distance to the root. Descriptors based on the radial distribution of features like it is done here are frequently used (see, for example, Sheridan et al.[25] or Xing and Glen[26]) in the context of similarity searching.

At level 0 of the subtree descriptor (see Figure 5), the properties of the root node are stored. At level 1, the properties of all nodes one edge away from the root node are stored. This principle is repeated until all nodes are processed.

Structuring a subtree with levels leads to the idea of implementing the descriptor as a linked list of levels. The length of this descriptor is therefore equivalent to the height of the subtree. For the generation of the descriptor, every node of the subtree has to be visited only once in a depth-first search.

Search Tree. To create the index, a search tree is used. We applied an AVL tree,[27] which is a binary, balanced tree, resulting in $O(\log(n))$ computing times for inserting and searching objects in a tree with n nodes.

Due to the fact that the descriptor levels are independent from each other, the representation of the subtree descriptor as a linked list of levels can be used to decrease the memory and run-time requirement of indexing. Only the first level

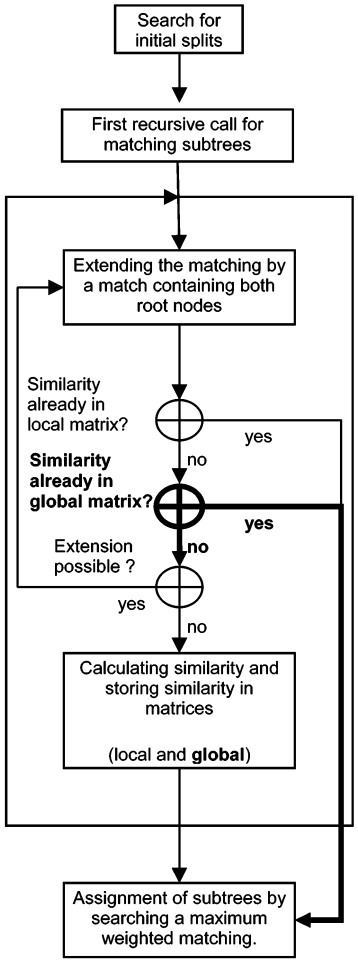

Figure 3. The modified match-search algorithm. The modification of the algorithm is highlighted by bold face text and lines.

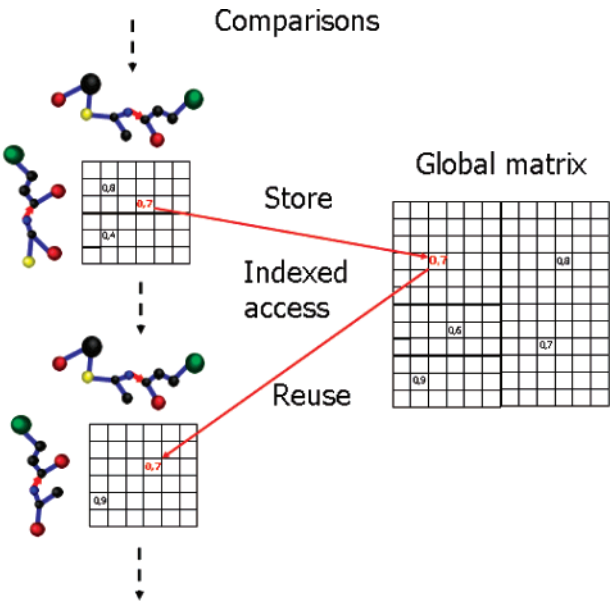

Figure 4. The match-search extension by SwiFT. The principal idea is to reuse already calculated partial results. Therefore, all calculated results are stored in a global matrix. Already computed entries can be accessed via the index, while newly calculated results are stored in the global matrix.

(level 0) will be inserted in the search tree. The corresponding AVL tree node stores the properties of this first descriptor level. An additional link points to the root of a new AVL tree. This AVL tree stores second-level elements of descriptors with an identical first level. Each of these elements stores a link to a third-level AVL tree in turn and so on. This leads to three advantages, even if the overall search tree is not balanced anymore:

1. Descriptors with the same prefix levels use the same prefix nodes, equally to a suffix tree (see, e.g., Kurtz and Giegerich for a review[28]).

2. Inserting a new element, at each level, we still have to compare the properties of the corresponding descriptor level with the node properties. In conclusion, only the properties of one level have to be considered at each comparison step for navigating through the search tree.

3. For a descriptor with e levels, we can reduce the number of steps for insertion from $O(\log(n))$ for n descriptors to $O(\log(n_1) + \log(n_2) + ... + \log(n_e))$, where n_i is the number of nodes in level i for all different subtree descriptors for which the $i-1$ prefix levels are identical.

For the navigation through the search tree, we developed a comparison function. The properties of the descriptor level which is under consideration are compared to the properties which are stored in the search tree node. We compare each of these properties in sequence. If the current property differs from the property stored in the node, for example, if the number of condensed feature tree nodes is greater or lower, we have to navigate to the right or left, respectively. If all properties are identical, the corresponding node in the search tree has been found.

Collision Handling. Some topologically different subtrees produce the same linear rooted feature tree descriptor, which is displayed in Figure 6. Consequently, it is not sufficient for correct indexing to insert a linear rooted feature tree descriptor in the search tree. To address this problem, the search tree has to be extended in the following way: The idea, known from hashing with chaining (see, e.g., Cormen et al.[29]), is to attach a list to every node containing links to the subtrees. By default, this list is empty. If a new AVL-tree node is inserted for the last descriptor level, the sole list element is a pointer to this subtree. If the search tree node already exists, the new subtree will be matched to all contained subtrees. If a perfect matching can be found, the subtree is identical to this other subtree already in the list. Consequently, it receives the same index number. Otherwise, a new pointer to this subtree is appended to the list. Although all subtrees have to be inserted into the search tree, no index value is necessary for a nonredundant subtree. Consequently, a new index value will only be created if a second identical subtree is inserted in the search tree the first time. This new index value will be assigned to both subtrees.

Modified Match-Search Algorithm. Besides the subtrees, the input of the match-search algorithm is extended by another data structure. This structure includes one or two global indices which are created by SwiFT, and the global matrix for the storage of already computed partial results. The number of indices depends on the kind of comparison, which will be discussed in detail later.

In every recursive call of the match-search algorithm, the global matrix is checked. In cases where the actual combination of subtrees was already computed (see Figure 4), the similarity value is returned, and the recursion stops. Otherwise, the algorithm extends the matches and performs a recursive call. The resulting similarity value will be stored

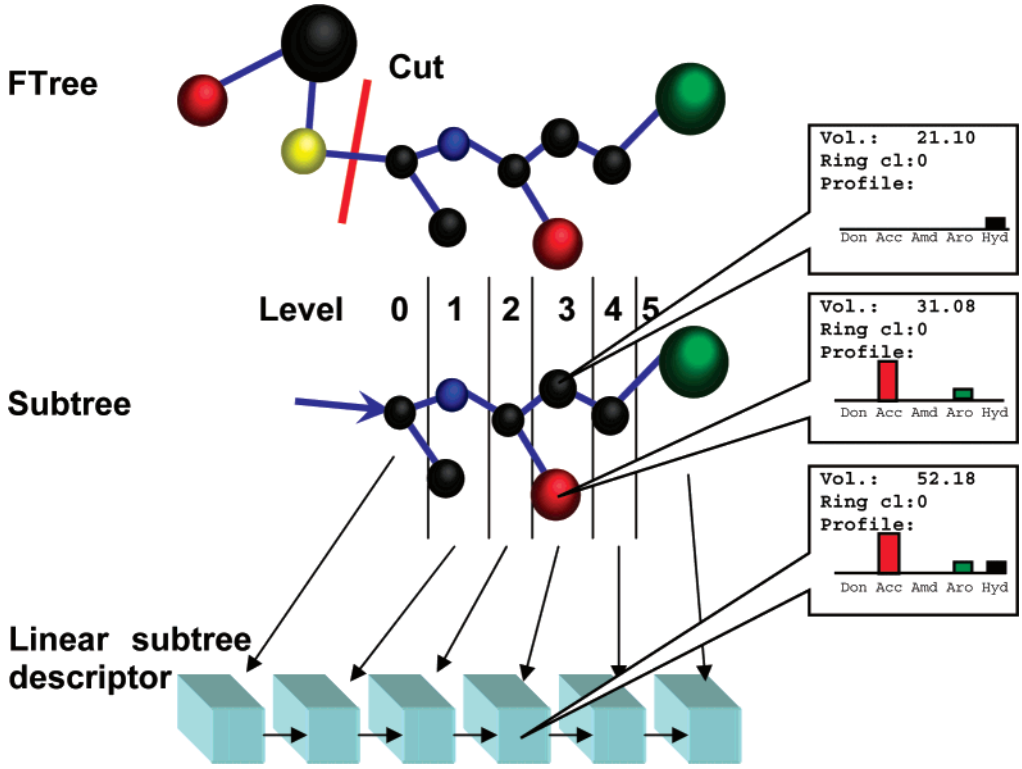

Figure 5. Presentation of the linear rooted feature tree descriptor for a subtree (rooted feature tree), which was separated from the feature tree by a cut (red line). Node properties within the same distance from the root are condensed to one level of the linear subtree descriptor.

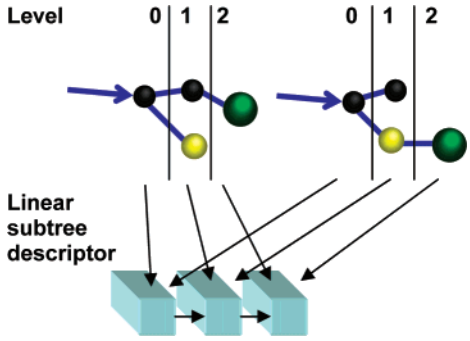

Figure 6. Two topologically different subtrees. These produce the same linear rooted feature tree descriptor.

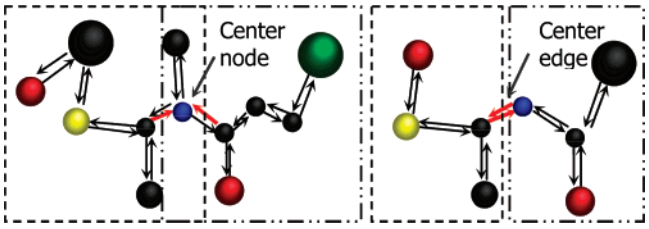

Figure 7. Identifying subtrees. Depending on the number of nodes along the longest path, the center of a feature tree is either a "center node" or a "center edge". When the feature tree is divided at this point in (overlapping) subtrees, only these identifying subtrees have to be found to identify a feature tree correctly. The roots of the so-called "identifying subtrees" are marked in red.

in the local and global matrix. The modified match-search is shown in Figure 3.

Matrix Access and Size. In the following, we speak of different comparison modes. A "1−*n*" comparison refers to a screening of a single compound against a data set. In clustering applications, an "*n*−*n*" comparison comparing all compounds with each other is needed to generate a distance matrix. If the distance matrix is computed in segments, an "*n*−*m*" comparison refers to the comparison of two segments.

The access to the global matrix differs depending on the mode of comparison: For a 1−*n* comparison, the access is given by the edge ID of the query molecule and an index for the screened data set. Computing an *n*−*n* comparison, the access can be done by the same index, and for an *n*−*m* comparison, two indices are used. It is obvious that the size of the global matrix differs depending on the chosen comparison mode. For a 1−*n* comparison, the matrix has a size of $i \cdot j$ entries, wherein *i* is the number of directed edges (the root edges of all possible subtrees) of the query and *j* is the number of unique subtrees occurring at least twice (index size). Furthermore an *n*−*m* comparison deals with a matrix of $i \cdot j$ entries, *i* and *j* representing the different index sizes. An *n*−*n* comparison uses the lower triangular matrix of the size $(i(i-1))/2$, *i* being the index size for the data set. Access is given using the larger index value as a first parameter and the smaller one as a second parameter.

Identification of Duplicate Feature Trees. Although only one feature tree can be generated for each molecule, the same feature tree can correspond to different molecules (for FTrees' current implementation, this would refer to, for example, stereoisomers and some constitutional isomers). This is caused by the fact that cycles are condensed into one node and stereochemistry is not coded as a node or edge property. Computing times are, therefore, further reduced by comparing identical feature trees only once.

The search for duplicate feature trees can be organized by employing the already introduced search tree to find identical subtrees. Two feature trees are identical if they have subtrees at exactly the same unique positions in the search tree. Therefore, only the subtrees which are required to represent the whole feature tree (see Figure 7) have to be

Table 1. Creation of Redundance-Free Feature Tree Files[a]

	no. feature trees	no. duplicates on feature tree level	time for index creation [mm:ss]	time for finding duplicates [mm:ss]	process size [MB]	decrease in comparisons for an $n-n$ comparison
Maybridge	58786	3698	00:25	00:36	420	12%
NCI	247159	38558	02:08	03:06	1,882	30%
Hert	102612	4472	01:22	01:49	1,287	10%
Zinc Leadlike	750219	383035	08:40	13:17	2,458	76%
Bionet	33762	3769	00:13	00:18	239	20%
Maybridge*	30000	1514	00:12	00:15	236	10%
NCI*	30000	899	00:14	00:17	270	6%
zinc*	30000	1077	00:11	00:15	230	7%
active set	9024	426	00:07	00:12	156	10%

[a] The table shows the number of feature trees and duplicates, the run time for index creation and identification of the duplicates (plus writing of a new redundance-free data file), and the process size in the order they are used later on. Randomly selected subsets are annotated with the "*" symbol. The active set consists of the compounds of the active classes as described by Hert et al.,[35] but without the protein kinase C inhibitors. Furthermore, it includes 1183 additional angiotensin II AT1 antagonists.

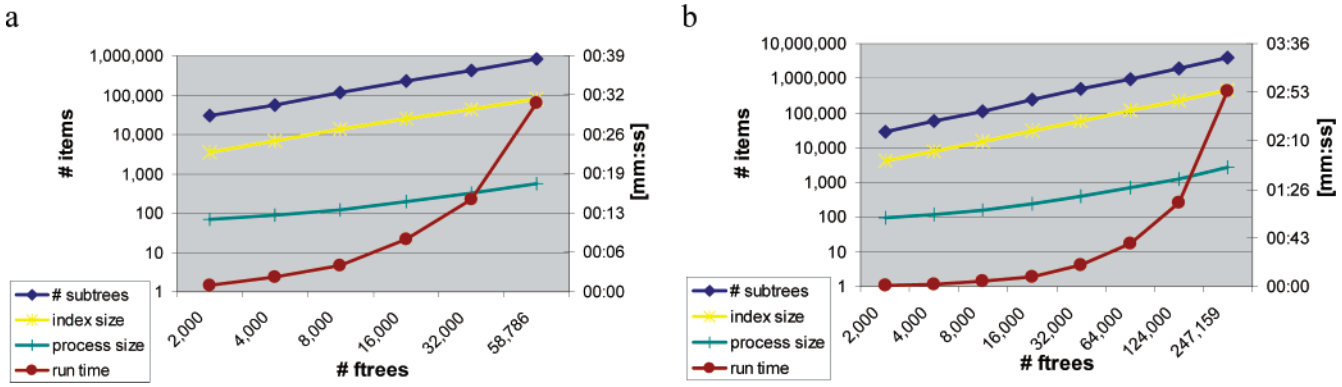

Figure 8. Plots (# items is logarithmically scaled) displaying the properties of the index creation method for different-sized subsets of the Maybridge (a) and NCI (b) data sets. All properties show a linear behavior.

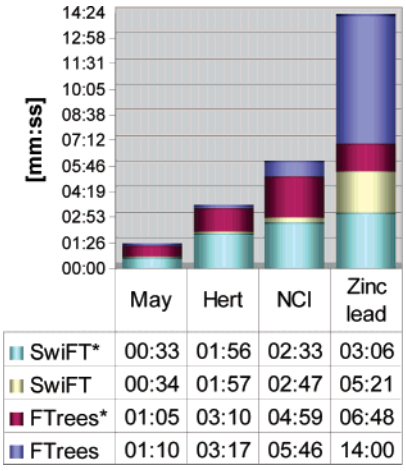

Figure 9. Average run times for $1-n$ comparisons using the standard match-search algorithm and SwiFT. A total of 47 of the 50 top-selling drugs of 2003 were screened against the Maybridge, Hert, NCI, and Zinc Leadlike data sets. The use of nonredundant data sets is marked with the "*" symbol.

searched to identify a feature tree correctly. Hence, we identify the longest path in the feature tree and distinguish between two cases: (1) The number of nodes on the longest path is odd, and a center node exists. (2) The number of nodes on the longest path is even, and a center edge exists. In the first case, we use all directed edges pointing to this center node and lying on the longest path as a root of the representing subtrees. In the second case, we use both directed antiparallel edges representing the nondirected center edge of the feature tree as root edges. These subtrees are then inserted into the search tree. If all identifying subtrees are found, the whole feature tree is identified correctly because all smaller and larger subtrees in the search tree must be identifying subtrees of respectively smaller and larger feature trees strictly.

For this method, the introduced search tree has to be modified. To identify the feature trees which own a certain subtree, we store the ID of the feature tree in a list appended to the element pointing to the corresponding subtree. Due to the fact that feature trees can have identical identifying subtrees (e.g., axis-symmetrical feature trees like a feature tree with two identical nodes connected by one edge), we also store the number of appearances of this subtree for each feature tree.

RESULTS AND DISCUSSION

For validation of SwiFT, the following six data sets were used: the Bionet screening database,[30] including 33 762 synthetic organic compounds; the Maybridge screening collection,[31] consisting of 58 786 organic compounds; the NCI data set,[32] consisting of 247 159 compounds; the Zinc leadlike subset,[33] consisting of 750 219 compounds which were selected from the Zinc database by criteria based on the publication of Teague et al.;[34] the "Hert data set" as described by Hert et al.,[35] consisting of 102 535 compounds taken from the MDDR database.[36] For the Hert data set, a total of 11 classes of active compounds were selected to create a benchmark data set. The feature tree data set we

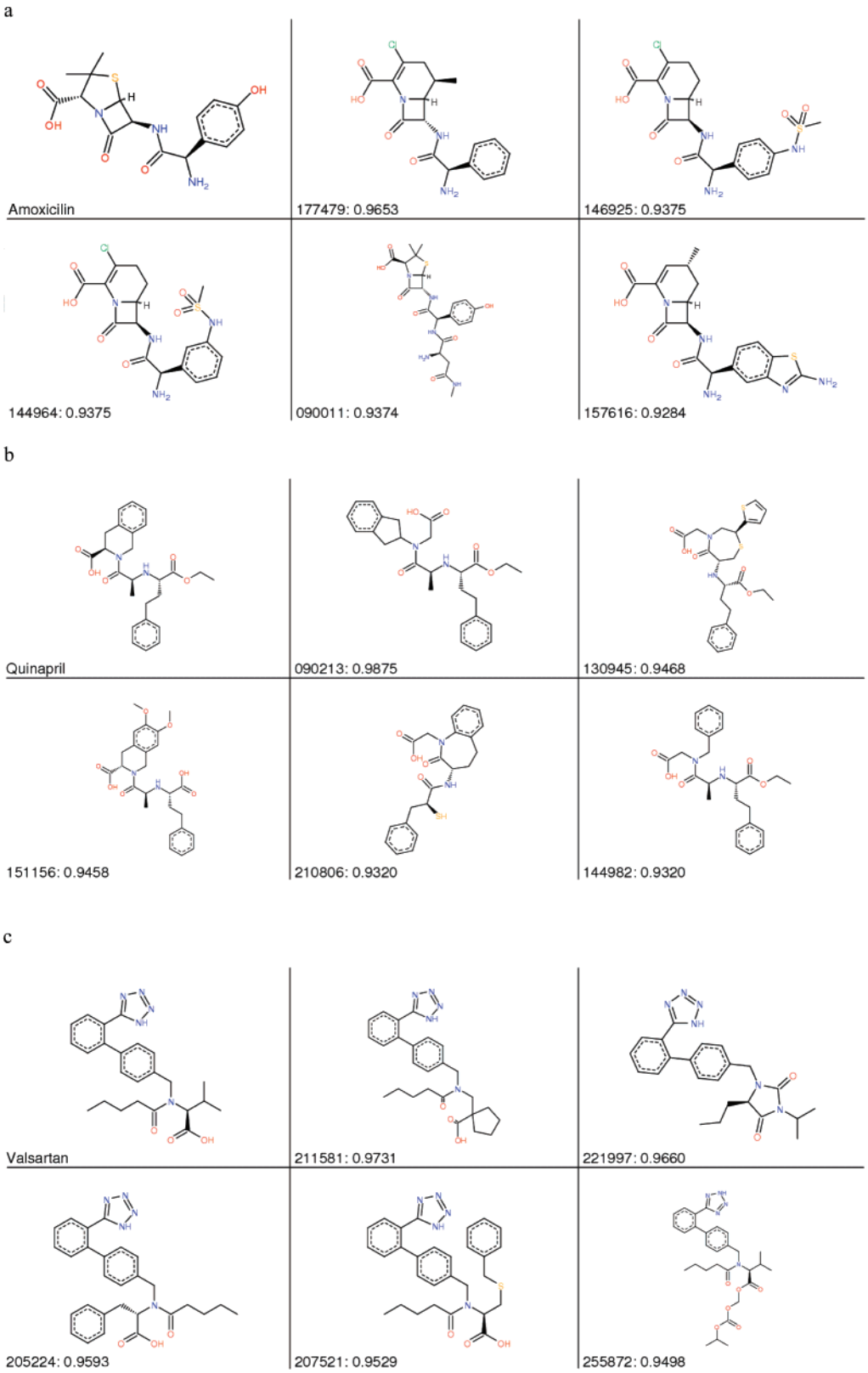

Figure 10. Five preferably diverse compounds of the top 20 results from the Hert data set for the query molecules amoxicillin (a), quinapril (b), and valsartan (c). In the left corner, the MDDR[36] ID and the computed similarity value according to the query are shown (drawings generated with FBrowser[39]).

used consists of 102 612. This difference will be investigated later on.

For the $n-n$ test cases, we chose subsets of 30 000 compounds respectively from the Maybridge, NCI, and Zinc leadlike data sets. Every ith molecule was selected by dividing the size of the data set by 30 000. The molecules for which the feature tree descriptor exceeded the standard recursive comparison thresholds, for example, the feature tree descriptor has too many nodes or a node contains a macro cycle, were discarded, and the next possible molecule was chosen instead. Additionally, we discarded feature tree descriptors representing molecules with complex ring systems

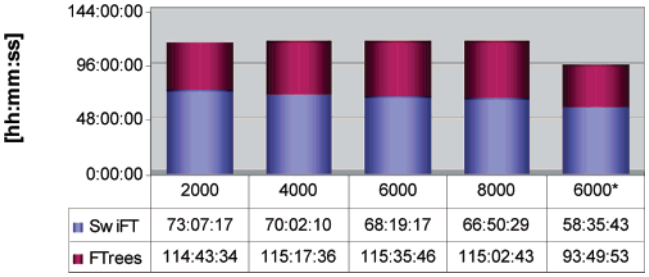

Figure 11. Impact of differently sized segments on the run time. Larger segments lead to a higher decrease in run time for SwiFT in comparison to the standard match-search (labeled as FTrees in this diagram). Usage of the redundance-free data set is marked with a "*".

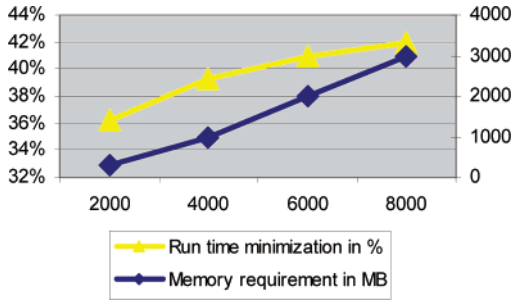

Figure 12. Correlation of run time minimization and memory requirement for the Bionet data set.

which cannot reasonably be represented with feature trees. All computations were done on 64-bit computers with two Intel Xeon 3 GHz processors, 3 GB of RAM, and 1 MB of cache, using a single CPU.

Redundance-Free Feature Tree Files. Finding identical feature trees can be used as a preprocessing step for similarity comparison. Note that this is only a redundancy on the feature tree level. To investigate the influence of this method on the run time, we generated a redundance-free data file for all data sets used in the comparison experiments, which were conducted with the initial data set and the redundance-free data set, respectively. Table 1 shows the number of feature trees and duplicates found as well as the run time in minutes and seconds [mm:ss] and the memory requirement for the data sets.

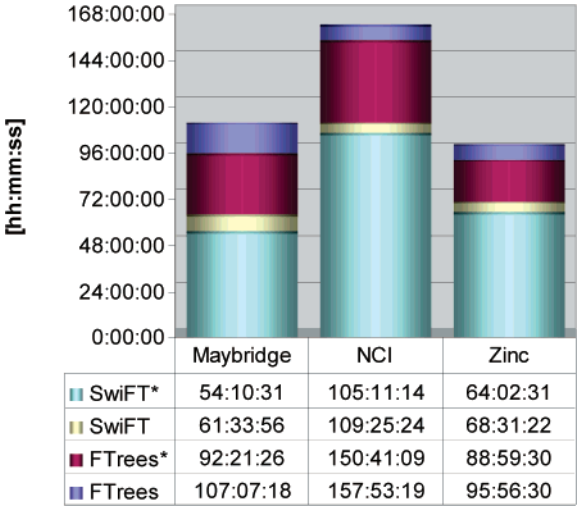

Figure 13. Run times for $n-n$ comparisons using a segment size of 6000 compounds.

The preprocessing step leads to a performance improvement concerning the number of pairwise comparisons, which is exemplarily shown in Table 1 for $n-n$ comparisons. Especially in the case of the Zinc leadlike data set, we can reduce the data set to less than half of its original size in about 20 min. This is due to a high number of isomeric structures which are below the sensitivity of the feature trees measure. It should be noted that this has to be done only once using the SwiFT I/O routines.

Index Creation. In order to analyze the effect of the database size on the computing time, we investigated how the method would behave if it received differently sized data sets as input. We used the Maybridge and the NCI data sets, starting with the first 2000 compounds and doubling the number of compounds every time until we reached the size of the data set. The properties which are shown in the plots (Figure 8) are the number of feature trees (# ftrees); the number of inserted subtrees (# subtrees); the size of the index, which is the number of the unique subtrees which occur at least twice (the subtrees, which exist only once, do not have to be indexed) and which also describes one dimension of the global matrix (index size); the process size which is given in megabytes (process size); and the run time of the index creation process in [mm:ss] (run time).

Table 2. Compounds Which Are Assigned to Two or More Activity Classes[a]

activity class	number	MDDR IDs
5HT1A agonists, 5HT reuptake inhibitors	21	145764, 223486, 226940, 226941, 226942, 226943, 243411, 243930, 243931, 243932, 263840, 266016, 266017, 266018, 266019, 266020, 272723, 272724, 272726, 272727, 281698
5HT1A agonists, 5HT3 antagonists	5	189753, 193313, 193314, 193315, 193316
5HT1A agonists, D2 antagonists	46	156610, 159334, 159810, 159811, 159812, 159813, 161228, 161229, 161230, 165810, 167877, 167878, 172429, 173745, 173746, 173747, 173748, 174295, 178583, 178584, 178585, 178586, 178587, 190248, 190249, 190250, 190251, 190252, 190253, 190255, 200881, 200919, 203430, 203431, 203432, 203433, 203434, 203435, 203605, 225400, 227496, 231699, 233376, 233377, 257398, 269638
5HT3 antagonists, D2 antagonists	5	248820, 250981, 251588, 251589, 251590
5HT reuptake inhibitors, D2 antagonists	1	269463
5HT1A agonists, 5HT3 antagonists and D2 antagonists	3	206309, 208035, 227533

[a] This table shows the activity classes, the number of compounds and the MDDR ID of these compounds.

Table 3. Additional Actives of the Hert Data Set

activity class	no. new actives	MDDR IDs
5 HT reuptake inhibitors	12	091079, 172645, 234459, 258659, 258662, 258663, 258664, 258665, 258666, 258667, 258745, 281789
5 HT1A agonists	21	162295, 193864, 209946, 241748, 242106, 242107, 242108, 243362, 243566, 243567, 243568, 246112, 246114, 246115, 272566, 275032, 276141, 277810, 281694, 281695, 281696
5 HT3 antagonists	0	
angiotensin II AT1 antagonists	1183	167666, 167676, 168383, 169111, 169337, 169338, 169339, 169340, 169341, 169342, 169343, 169344, 169345, 169346, 169347, 169348, 169349, 169352, 169353, 169354, 169355, 169356, 169357, 169358, 169359, 169360, 169361, 169362, 169363, 169364, 169365, 169369, 169696, 169781, 169792, 170466, 170467, 170468, 170469, 170470, 170471, 170472, 170473, 170474, 170475, 170476, 170477, 170478, 170479, 170480, 170481, 170482, 170483, 170484, 170485, 170486, 170487, 170488, 170489, 170939, 171508, 172247, 172248, 172249, 172250, 172251, 172252, 172253, 172254, 172255, 172256, 172257, 172258, 172259, 172260, 172261, 172262, 172263, 172264, 172265, 172266, 172267, 172268, 172269, 172270, 172301, 172302, 172377, 172582, 172601, 173354, 174350, 174351, 174352, 174353, 174354, 174355, 174356, 174357, 174358, 174359, 174360, 174361, 174362, 174363, 174364, 174562, 174803, 174805, 174806, 175231, 175270, 175284, 175285, 175286, 175287, 175288, 175289, 175290, 175291, 175292, 175293, 175294, 175295, 175296, 175474, 175475, 175476, 175477, 175566, 175567, 175568, 175569, 175570, 175571, 175572, 175573, 175592, 175593, 175594, 175595, 175596, 175597, 175598, 175599, 175600, 175601, 175602, 176245, 176293, 176541, 176544, 177140, 177141, 177142, 177223, 177374, 177375, 177376, 177526, 177535, 177568, 178069, 178093, 178094, 178095, 178096, 178097, 178098, 178099, 178669, 178670, 178671, 178672, 178673, 178674, 178675, 178676, 179012, 179013, 179014, 179015, 179016, 179017, 179018, 179019, 179020, 179021, 179022, 179023, 179024, 179025, 179026, 179027, 179028, 179029, 179030, 179031, 179032, 179033, 179034, 179035, 179036, 179037, 179038, 179039, 179040, 179041, 179364, 179709, 179710, 179711, 179712, 179713, 179763, 179764, 179765, 179766, 179767, 179907, 179908, 179909, 180112, 180297, 180298, 180299, 180300, 180310, 180311, 180312, 180313, 180314, 180315, 180316, 180317, 180318, 180451, 180452, 180453, 180454, 180455, 180456, 180481, 180482, 180483, 180484, 180485, 180486, 180487, 180488, 180489, 180490, 180867, 180959, 180960, 180961, 180962, 180963, 182059, 182341, 182342, 182343, 182344, 182345, 182346, 182347, 182360, 182361, 182362, 182363, 182364, 182535, 182622, 182712, 182713, 182714, 182715, 182716, 182717, 182718, 182719, 182720, 183123, 183759, 183793, 183952, 184326, 184327, 184328, 184329, 184330, 184331, 184332, 184333, 184336, 184337, 184338, 184339, 184551, 184596, 185251, 185252, 185253, 185254, 185255, 185256, 185257, 185258, 185259, 185260, 185261, 185262, 185263, 185264, 185762, 185763, 185764, 185765, 185766, 185767, 185768, 185769, 185987, 186289, 186712, 186884, 186885, 186886, 186887, 186888, 186889, 186891, 187000, 187001, 187002, 187003, 187004, 187005, 187006, 187007, 187008, 187009, 187010, 187011, 187012, 187075, 187213, 187282, 187283, 187284, 187285, 187286, 187287, 187402, 187889, 187890, 187891, 187892, 187893, 187894, 187895, 187896, 187897, 187898, 187899, 187900, 187901, 188394, 188410, 188436, 188437, 188438, 188439, 188851, 188871, 188900, 189578, 189579, 189580, 189581, 189582, 189583, 189713, 189737, 190265, 190399, 190486, 190567, 190568, 190569, 190570, 190571, 190572, 190573, 190574, 190603, 190604, 190605, 190606, 190607, 190608, 190609, 190610, 190611, 190612, 190613, 190732, 190733, 190844, 190872, 190879, 190882, 190910, 191271, 191354, 191379, 191685, 191686, 191687, 191688, 191799, 191806, 191815, 191853, 191934, 191935, 191936, 191937, 191938, 191939, 191960, 191961, 191962, 191963, 191964, 192016, 192590, 192659, 192660, 192661, 192662, 192663, 192664, 192725, 192726, 192727, 192728, 192729, 192730, 192731, 192732, 192929, 193038, 193039, 193040, 193041, 193042, 193044, 193045, 193193, 193194, 193195, 193196, 193197, 193250, 193251, 193252, 193253, 193254, 193255, 193256, 193257, 193258, 193274, 193285, 193450, 193451, 193452, 193453, 193454, 193455, 193465, 193466, 193467, 193468, 193469, 193477, 193478, 193479, 193480, 193481, 193482, 193483, 193484, 193485, 193486, 193553, 193554, 193555, 193556, 193557, 193558, 193559, 193560, 193561, 193562, 193563, 193564, 193565, 193566, 193897, 193936, 194103, 194104, 194105, 194106, 194107, 194108, 194109, 194139, 194344, 194345, 194346, 194347, 194672, 194686, 194687, 194688, 194689, 194690, 194691, 194692, 194693, 194726, 194727, 194728, 194729, 194730, 194731, 194732, 194869, 194906, 194941, 195059, 195060, 195061, 195105, 195106, 195107, 195108, 195109, 195110, 195111, 195112, 195157, 195158, 195476, 195477, 195478, 195479, 195480, 195481, 195482, 195483, 195484, 195485, 195486, 195487, 195577, 195649, 195678, 195679, 196045, 196046, 196047, 196048, 196049, 196145, 196146, 196249, 196250, 196251, 196252, 196253, 196314, 196849, 196850, 196884, 197032, 197033, 197076, 197077, 197078, 197079, 197080, 197081, 197082, 197083, 197084, 197085, 197086, 197089, 197616, 197653, 197654, 197663, 197664, 197665, 197757, 197806, 197807, 197808, 197809, 197810, 197929, 197930, 197931, 197932, 197933, 197934, 197935, 197936, 197937, 197938, 197939, 198183, 198528, 198558, 198879, 198882, 198883, 198884, 198885, 198886, 199155, 199156, 199157, 199158, 199159, 199160, 199536, 199537, 199538, 199539, 199540, 199541, 199542, 199543, 199544, 199545, 199546, 199572, 199609, 199813, 200023, 200025, 200026, 200027, 200028, 200029, 200030, 200031, 200032, 200033, 200034, 200035, 200078, 200107, 200108, 200120, 200121, 200199, 200917, 201029, 201444, 201445, 201446, 201447, 201448, 201449, 201541, 201542, 201561, 201580, 201722, 201723, 201724, 201725, 201726, 201727, 201728, 201979, 201998, 202024, 202075, 202076, 202095, 202096, 202108, 202226, 202227, 202252, 202358, 202510, 202726, 202727, 202728, 202729, 202730, 202731, 202732, 202733, 203051, 203052, 203053, 203244, 203245, 203246, 203247, 203248, 203249, 203250, 203251, 203252, 203294, 203858, 203859, 203894, 203895, 203933, 203934, 203935, 203936, 203957, 203958, 203959, 203960, 203962, 203963, 203964, 203965, 203966, 203967, 203968, 203969, 203970, 203971, 204107, 204109, 204111, 204112, 204113, 204158, 204179, 204180, 204181, 204182, 204183, 204184, 204185, 204186, 204187, 204188, 204189, 204248, 204249, 204250, 204251, 204252, 204253, 204287, 204304, 204306, 204585, 204586, 204587, 204588, 204589, 204713, 204714, 204715, 204716, 204744, 204911, 205227, 205266, 205267, 205268, 205269, 205270, 205271, 205274, 205275, 205277, 205278, 205367, 205614, 205615, 205616, 205617, 205618, 205619, 205620, 205621, 205622, 205623, 205624, 205625, 205626, 205627, 205833, 205834, 205835, 205836, 205837, 205838, 205839, 205840, 205841, 205842, 205843, 206006, 206007, 206008, 206009, 206010, 206011, 206012, 206013, 206232, 206308, 206561, 206562, 206563, 206564, 206565, 206578, 206794, 206823, 207275, 207276, 207277, 207278, 207335, 207388, 207468, 207631, 207632, 207633, 207634, 207635, 207636, 207637, 207638, 207639, 207640, 207641, 207719, 208052, 208053, 208054, 208055, 208056, 208104, 208105, 208106, 208107, 208108, 208109, 208110, 208111, 208112, 208113, 208114, 208115, 208116, 208117, 208279, 208280, 208366, 208367, 208368, 208496, 208504, 208505, 208506, 208507, 208508, 208509, 208510, 208599, 208600, 208601, 208602, 208603, 208604, 208605, 208606, 208607, 208608, 208609, 208610, 208626, 208627, 208999, 209394, 209414, 209415, 209416, 209417, 210116, 210138, 210278, 210458, 210724, 210825, 211633, 211637, 211680, 211682, 211685, 211696, 211699, 211705, 211711, 211905, 211906, 211907, 211908, 211909, 211910, 211911, 211912, 212537, 212608, 212609, 212610, 212611, 212612, 212613, 212644, 212669, 212706, 212743, 212944, 212945, 212946, 213331, 213665, 213682, 213683, 213783, 213860, 213861, 213862, 214430, 214431, 214432, 214433, 214434, 214435, 214462, 214604, 214605, 214626, 214922, 214923, 215148, 215149, 215470, 215471, 215472, 215475, 215476, 215477, 215478, 215479, 215546, 215547, 215548, 215549, 215550, 215551, 215744, 215807, 215825, 216194, 216379, 216380, 216381, 216382, 216383, 216384, 216385, 216386, 216387, 216388, 216389, 216455, 217089, 217090, 217091, 217092, 217093, 217094, 217095, 217096, 217108, 217109, 217116, 217117, 217118, 217119, 217120, 217121, 217122, 217123, 217124, 217125, 217126, 217127, 217128, 217129, 217130, 217307, 217335, 217364, 217603, 217813, 218061, 218062, 218063, 218064, 218065, 218281, 218282, 218305, 218534, 218723, 218724,

Table 3 (Continued)

activity class	no. new actives	MDDR IDs
		218727, 218728, 218729, 218730, 218732, 218772, 218773, 218774, 218775, 219193, 219195, 219712, 219713, 220136, 220137, 221563, 221648, 221649, 221651, 221768, 221769, 221770, 221771, 221834, 221997, 221998, 221999, 222000, 223039, 223040, 223041, 223042, 223043, 223044, 223045, 223046, 223047, 223291, 223986, 223987, 224651, 224693, 225477, 225478, 225479, 225545, 225550, 225552, 225890, 225891, 225892, 225893, 226523, 227336, 227337, 227338, 227432, 227433, 227821, 227857, 229236, 229237, 229238, 229239, 229240, 229241, 229242, 229946, 229947, 229948, 229949, 229950, 229951, 229952, 232644, 234800, 234939, 235170, 235173, 235363, 235669, 235670, 235862, 235863, 235864, 235865, 235866, 235867, 235868, 237081, 237151, 237153, 237154, 237155, 237225, 237496, 237497, 237498, 237499, 237500, 237501, 237502, 237503, 237504, 237505, 238192, 238193, 238194, 238195, 238330, 239665, 240314, 240315, 240316, 240317, 240318, 240319, 240320, 240321, 240322, 240906, 241300, 241634, 241635, 241636, 241637, 241638, 241639, 243538, 243539, 244665, 245194, 245195, 245196, 245197, 245198, 248644, 249072, 249073, 255037, 255872, 255873, 255874
cyclooxygenase inhibitors	2	160981, 222486
D2 antagonists	8	091385, 115395, 162578, 163701, 185541, 268677, 275032, 276141
HIV-1 protease inhibitors	28	155135, 157851, 157852, 157853, 157854, 157855, 157856, 172409, 172410, 172411, 197356, 197435, 199182, 200419, 200420, 200421, 200422, 200423, 200424, 200425, 200631, 200632, 200633, 200634, 257268, 258127, 258128, 258129
protein kinase C inhibitors	5	189740, 198239, 200729, 200731, 274904
renin inhibitors	9	150274, 150275, 150276, 150277, 150278, 150279, 150280, 150281, 150282
substance P antagonists	17	155099, 158755, 175662, 183306, 183307, 192747, 235932, 239472, 258380, 264027, 270190, 272352, 272353, 272354, 273720, 273724, 277119
thrombin inhibitors	6	221049, 256922, 277454, 282385, 282386, 282389

It appears that the Maybridge (Figure 8a) as well as the NCI data set (Figure 8b) show a nearly linear behavior for the investigated properties.

To obtain the performance listed in Table 1, the feature trees have to be held in memory, which results in large memory requirements. It should be especially noted that the index creation, as described above, only needs to be done once for a data set.

Applying SwiFT to 1−*n* Comparisons. Exemplarily, we investigated 1−*n* comparisons for 47 out of the 50 top-selling (nonsteroidal) drugs of 2003:[37,38] Excluding azithromyzin, hydrocodone and levofloxacin due to their complex ring systems which cannot be reasonably represented with feature trees, we screened the Maybridge, NCI, Zinc leadlike, and Hert data sets against the remaining 47 compounds. We used two subsets of around 375 000 compounds for the Zinc leadlike subset due to memory restrictions during index creation. The average run times for the standard match-search and SwiFT are presented in Figure 9. Figure 10 shows typical results of a feature tree similarity calculation using SwiFT.

Considering both index-based methods, there is a run time reduction of ∼53% for the Maybridge data set, 57% for the NCI data set, and 46% for the Hert data set (Figure 9). In the case of the Hert data set, the method for finding duplicates leads only to a small difference. For the Zinc leadlike subset, the match-search extension decreases the run time by ∼75%, while the method of finding duplicates leads to a reduction of ∼50%. Using both methods together results in a run time reduction of ∼80%.

Applying the SwiFT Index to *n*−*n* Comparisons. For performing an all-against-all comparison for a data set, a parallel calculation is necessary. Consequently, the data set is split into distinct segments. In the case of SwiFT, where the input of the match-search algorithm is extended by the global matrix, the dimension of the global matrix and consequently the index size is restricted by the amount of available RAM and the process size. To investigate the influence of the segment size on the run time, we performed several *n*−*n* comparisons of the Bionet data set using different segment sizes. The run times of the standard match-search and SwiFT are shown in Figure 11.

Table 4. Active Sets Used for Clustering

active set	no. compounds
5HT1A agonists	827
5HT reuptake inhibitors	359
cox inhibitors	636
substance P antagonists	1,246
5HT3 antagonists	752
D2 antagonists	395
HIV protease inhibitors	750
renin inhibitors	1,130
thrombin inhibitors	803
angiotensin II AT1 antagonists	943
additional angiotensin II AT1 antagonists	1,183

When SwiFT is used, the reduction in run time increases with the size of the chosen segments, although the run time of the standard match-search does not differ (apart from small variability influenced by cluster usage). Since the memory requirement increases as well (see Figure 12), we chose a segment size of 6000 molecules for the following experiments with the subsets of the Maybridge, NCI, and Zinc data sets. Performance can be further increased if machines with more memory are available. An additional removal of the duplicates leads to a run time reduction of ∼50% (using a segment size of 6000).

To investigate the impact of different data on the run time reduction, we used three subsets with 30 000 compounds each of the Maybridge, NCI, and Zinc leadlike set, respectively. The run times are displayed in Figure 13. SwiFT reduces the run time by 49% (Maybridge), 33% (NCI), and 33% (Zinc).

The size of the used data sets is an important factor in this test case. The larger the data set, the more important is the removal of redundant feature trees for an *n*−*n* comparison. Furthermore, using only every *i*th feature tree leads to the data being more diverse because normally related molecules are located close to each other in data sets.

Hert Data Set−An Application Scenario. On the basis of the MDDR database release in 2005, we created an activity data set using Hert's[35] compound lists.[40] Creating a nonredundant feature tree file for this data set, we investigated

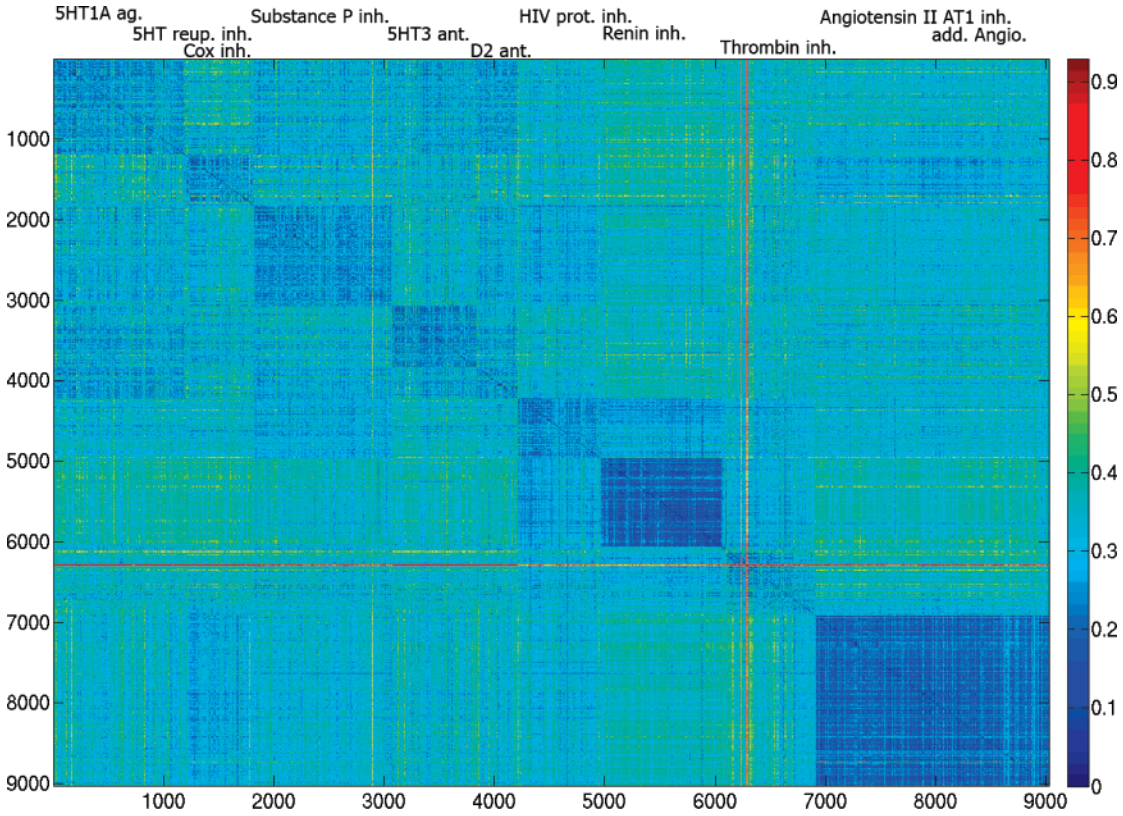

Figure 14. Distance matrix of 9024 active molecules. A pair distance of 0 (highest similarity) is coded in dark blue, and a pair distance of 1 (smallest similarity) is coded in dark red (figure generated with Matlab 15[47]).

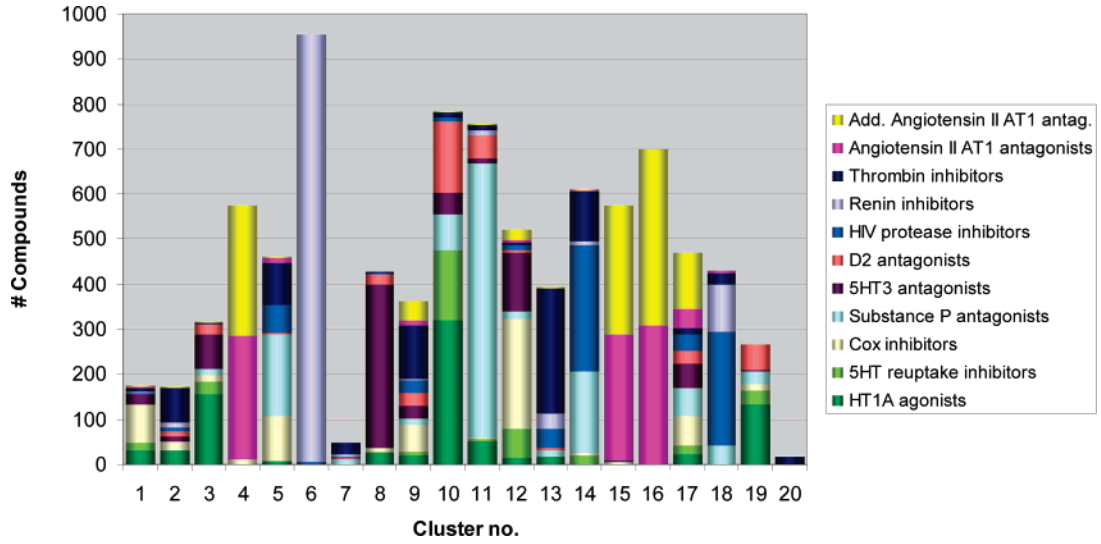

Figure 15. Cluster composition. The compounds of the active classes are color-coded for the 10 clusters resulting from a *k*-means clustering with $k = 10$.

that several compounds were assigned to two or more activity classes (Table 2). Additionally, while screening valsartan, an angiotensin II AT1 antagonist (see Figure 10), against this data set, 10 of the 20 most similar compounds (including valsartan itself as the number 1 hit with a similarity of 1) were part of the angiotensin II AT1 antagonists active set. Examining the MDDR SDfile[41] of the data set (2005), we found a few compounds not yet marked as active for 10 of the 11 active classes. This can probably be explained by the fact that more activities have been found in recent years. However, for the angiotensin II AT1 antagonist activity class, we found 1183 additional compounds in the random set which were annotated in the SDfile as angiotensin II AT1 antagonists (Table 3). These additional angiotensin II AT1 antagonists were included in the data set to investigate whether they are similar to the ones already contained. Furthermore, we excluded protein kinase C inhibitors mostly because they split into pharmacophorically different subclasses, for example, binding to the Cys2 activator-binding domain[42] or the catalytic domain.[43,44]

Using these active sets (Table 4), we performed a so-called "*k*-means clustering" with $k = 10$, resulting in 10 clusters.

We used the *k*-means method by Hartigan et al.,[45] which is implemented in R[46]. The distance matrix was calculated with FTrees. Feature trees, which are assigned to more than one activity class (Table 2), occur for each class. The active classes were used in the order given by Table 4.

When the distance matrix (distance = 1 − similarity) of the 9024 feature trees used in this experiment is analyzed (Figure 14), a higher similarity can be observed along the diagonal, which differentiates the activity classes very well. Especially the angiotensin II AT1 antagonists (including the additional ones) and the renin inhibitors show a high similarity to the other members of their activity class. Furthermore, the relationship between several activity classes is recognizable. Exemplarily, there is a high similarity between the 5HT1A agonists, 5HT reuptake inhibitors (both displayed by the first dark blue rectangle along the diagonal), and the D2 antagonists (the fifth rectangle). This is supported by Table 2, which shows that there are several molecules belonging to two of these classes. The clustering (Figure 15) also supports the observation that some active classes have common molecules by the resulting mixed clusters. On the other hand, the renin inhibitors (cluster no. 3) and the angiotensin II AT1 antagonists (cluster no. 6) are separated very well, which is reflected also by the distance matrix (last rectangle).

For calculation of the distance matrix, we need 43% less time with SwiFT (22 h 4 min compared to 38 h 14 min with the standard match-search). By removing redundant feature trees, the size of the distance matrix and the number of clustering steps can be decreased, too.

CONCLUSIONS

We have implemented a new indexing method (SwiFT) for a faster comparison of reduced graph descriptors like feature trees. The method reuses already computed partial results by indexing the subtrees of a compound data set. The match-search algorithm enhanced by SwiFT produces the identical similarity values like the standard one. Screening different data sets, we have shown that SwiFT leads to a substantial reduction in run time for $1-n$ as well as for $n-n$ and $n-m$ comparisons. Obviously, an indexing method becomes more valuable if a higher number of compounds are used. Regarding the wish to further speed up the comparison process and to reduce the memory requirement for data storage, a parallel computation environment can be applied, splitting the data sets in distinct segments. When segments of, for example, 6000 compounds are chosen, the buildup of the indices takes only a few seconds and can be neglected concerning the overall run time. All large calculations shown in this paper are already based on segmentation. Therefore, SwiFT can be used in a parallel computing environment.

Furthermore, the search tree can be used to find identical feature trees in data sets without performing an $n-n$ comparison. This makes this preprocessing step extremely fast. In turn, it leads to a minimization of the comparison run time, especially for data sets containing a large number of isomers like the Zinc data set. Due to the implemented I/O routines, both preprocessing steps have to be done only once, which makes it particularly applicable for Web services (e.g., FTreesWeb[48] provides free Web-enabled access to the SwiFT technology) and in-house database searches.

ACKNOWLEDGMENT

The authors want to thank Jörg Degen and Patrick Maass of the ZBH, and Marcus Gastreich and Sally Hindle from BioSolveIT for valuable discussions and support throughout this project.

REFERENCES AND NOTES

(1) Lengauer, T.; Lemmen, C.; Rarey, M.; Zimmermann, M. Novel Technologies for Virtual Screening. *Drug Discovery Today* **2004**, *9*, 27−34.
(2) Johnson, E. G.; Maggiora, G. M. *Concepts and Applications of Molecular Similarity*; John Wiley & Sons: New York, 1990.
(3) Fischer, E. Einfluss der Konfiguration auf die Wirkung der Enzyme. *Ber. Dtsch. Chem. Ges.* **1894**, *27*, 2985−2993.
(4) Willett, P. Chemical Similarity Searching. *J. Chem. Inf. Comput. Sci.* **1998**, *38*, 983−996.
(5) Evers, A.; Klebe, G. Successful Virtual Screening for a Submicromolar Antagonist of the Neurokinin-1 Receptor Based on a Ligand-Supported Homology Model. *J. Med. Chem.* **2004**, *47*, 5381−5392.
(6) Bender, A.; Mussa, H.; Glen, R. Molecular Similarity Searching Using Atom Environments, Information-Based Feature Selection, and a Naive Bayesian Classifier. *J. Chem. Inf. Comput. Sci.* **2004**, *44*, 170−178.
(7) Hessler, G.; Zimmermann, M.; Matter, H.; Evers, A.; Naumann, T.; Lengauer, T.; Rarey, M. Multiple-Ligand-Based Virtual Screening: Methods and Applications of the Mtree Approach. *J. Med. Chem.* **2005**, *48*, 6575−6584.
(8) Todescini, R.; Consonni, V. Handbook of Molecular Descriptors. In *Methods and Principles in Medicinal Chemistry*; Mannhold, R., Kubinyi, H., Timmermann, H., Eds.; Wiley-VCH: Weinheim, Germany, 2000; Vol. 11.
(9) Sheridan, R. P.; Kearsley, S. K. Why Do We Need So Many Chemical Similarity Search Methods? *Drug Discovery Today* **2002**, *7*, 903−911.
(10) Rarey, M.; Dixon, J. S. Feature Trees: A New Molecular Similarity Measurement Based on Tree Matching. *J. Comput.-Aided Mol. Des.* **1998**, *12*, 471−490.
(11) Rarey, M.; Stahl, M. Similarity Searching in Large Combinatorial Spaces. *J. Comput. Aided Mol. Des.* **2001**, *15*, 497−520.
(12) Rarey, M.; Hindle, S.; Mass, P.; Metz, G.; Rummey, C.; Zimmermann, M. Feature Trees: Theory and Applications from Large-Scale Virtual Screening to Data Analysis. In *Pharmacophores and Pharmacophore Search*; Langer, T.; Hoffmann, R. D., Eds.; Wiley-VCH: Weinheim, Germany, 2005; Vol. 32, pp 81−116.
(13) *FTrees*, version 1.5.1; BioSolve IT GmbH: St. Augustin, Germany, 2006.
(14) Gillet, V. J.; Downs, G. M.; Holliday, J. D.; Lynch, M. F.; Dethlefsen, W. Computer Storage and Retrieval of Generic Chemical Structures in Patents. 13. Reduced Graph Generation. *J. Chem. Inf. Comput. Sci.* **1991**, *31*, 260−270.
(15) Gillet, V. J.; Willett, P.; Bradshaw, J. Similarity Searching Using Reduced Graphs. *J. Chem. Inf. Comput. Sci.* **2003**, *43*, 338−345.
(16) Barker, E. J.; Gardiner, E. J.; Gillet, V. J.; Kitts, P.; Morris, J. Further Development of Reduced Graphs for Identifying Bioactive Compounds. *J. Chem. Inf. Model.* **2003**, *43*, 346−356.
(17) 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*, 503−511.
(18) Harper, G.; Bravi, G. S.; Pickett, S. D.; Hussain, J.; Green, D. V. S. The Reduced Graph Descriptor in Virtual Screening and Data-Driven Clustering of High-Throughput Screening Data. *J. Chem. Inf. Model.* **2004**, *44*, 2145−2156.
(19) Birchall, K.; Gillet, V. J.; Harper, G.; Pickett, S. D. Training Similarity Measures for Specific Activities: Application to Reduced Graphs. *J. Chem. Inf. Model.* **2006**, *46*, 577−586.
(20) Baurin, N.; Baker, R.; Richardson, C.; Chen, I.; Foloppe, N.; Potter, A.; Jordan, A.; Roughley, S.; Parratt, M.; Greaney, P.; Morley, D.; Hubbard, R. E. Drug-Like Annotation and Duplicate Analysis of a 23-Supplier Chemical Database Totalling 2.7 Million Compounds. *J. Chem. Inf. Model.* **2004**, *44*, 643−651.
(21) Mayer, D.; Naylor, C. B.; Motoc, I.; Marshall, G. A Unique Geometry of the Active Site of Angiotensin-Converting Enzyme Consistent with Structure Activity. *J. Comput.-Aided Mol. Des.* **1987**, *1*, 3−16.
(22) *FlexV*, version 1.8.1; BioSolve IT GmbH: St. Augustin, Germany, 2006.

(23) Bellman, R. On the Theory of Dynamic Programming. *Proc. Natl. Acad. Sci. U.S.A.* **1952**, *38*, 716−719.
(24) Rarey, M.; Kramer, B.; Lengauer, T.; Klebe, G. A Fast Flexible Docking Method Using an Incremental Construction Algorithm. *J. Mol. Biol.* **1996**, *261*, 470−489.
(25) Sheridan, R. P.; Ramaswamy, N.; Rusinko, A.; Bauman, N.; Haraki, K. S.; Venkataraghavan, R. 3dsearch: A System for Three-Dimensional Substructure Searching. *J. Chem. Inf. Comput. Sci.* **1989**, *29*, 255−260.
(26) Xing, L.; Glen, R. Novel Methods for the Prediction of Log P, Pk_a and Log D. *J. Chem. Inf. Comput. Sci.* **2002**, *42*, 796−805.
(27) Adelson-Velskii, G. M.; Landis, E. M. An Algorithm for the Organization of Information. *Sov. Math. Dokl.* **1962**, *3*, 1259−1263.
(28) Giegerich, S.; Kurtz, S. From Ukkonen to Mccreight and Weiner: A Unifying View of Linear-Time Suffix Tree Construction. *Algorithmica* **1997**, *19*, 331−353.
(29) Cormen, T. H.; Leiserson, C. E.; Rivest, R. L. Hash Tables. In *Introduction to Algorithms*; MIT Press: Cambridge, England, 2001; pp 221−252.
(30) Key Organics Limited UK Bionet Screening Compounds Database. http://www.keyorganics.ltd.uk/screenin.htm (accessed Oct 2006).
(31) Maybridge Screening Collection. http://www.maybridge.com (accessed Sept 2006).
(32) Nci Data Set. http://dtp.nci.nih.gov/docs/3d_database/structural_information/structural_data.html (accessed Nov 2006).
(33) Irwin, J. J.; Shoichet, B. K. Zinc−A Free Database of Commercially Available Compounds for Virtual Screening. *J. Chem. Inf. Model.* **2005**, *45*, 177−182.
(34) Teague, S. J.; Davis, A. M.; Leeson, P. D.; Oprea, T. The Design of Leadlike Combinatorial Libraries. *Angew. Chem., Int. Ed.* **1999**, *38*, 3743−3748.
(35) Hert, J.; Willett, P.; Wilton, D. J.; Acklin, P.; Azzaoui, K.; Jacoby, E.; Schuffenhauer, A. Comparison of Fingerprint-Based Methods for Virtual Screening Using Multiple Bioactive Reference Structures. *J. Chem. Inf. Comput. Sci.* **2004**, *44*, 1177−1185.
(36) MDL Information Systems Inc. Mdl Drug Data Report. http://www.mdli.com (accessed Sept 2006).
(37) RXList: The Top 300 Prescriptions for 2003 by Number of U.S. Prescriptions Dispensed. http://www.rxlist.com/top200.htm (accessed Sept 2006).
(38) Boda, K.; Johnson, A. P. Molecular Complexity Analysis of de Novo Designed Ligands. *J. Med. Chem.* **2006**, *49*, 5869−5879.
(39) Fricker, P. C.; Gastreich, M.; Rarey, M. Automated Drawing of Structural Molecular Formulas under Constraints. *J. Chem. Inf. Model.* **2004**, *44*, 1065−1078.
(40) Cheminformatics.org. http://www.cheminformatics.org (accessed Sept 2006).
(41) Dalby, A.; Nourse, J. G.; Hounshell, W. D.; Gushurst, A. K. I.; Grier, D. L.; Leland, B. A.; 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.
(42) Zhang, G.; Kazanietz, M. G.; Blumberg, P. M.; Hurley, J. H. Crystal Structure of the Cys2 Activator-Binding Domain of Protein Kinase C Delta in Complex with Phorbol Ester. *Cell* **1995**, *81*, 917−924.
(43) Xu, Z.-B.; Chaudhary, D.; Olland, S.; Wolfrom, S.; Czerwinski, R.; Malakian, K.; Lin, L.; Stahl, M. L.; Joseph-McCarthy, D.; Benander, C.; Fitz, L.; Greco, R.; Somers, W. S.; Mosyak, L. Catalytic Domain Crystal Structure of Protein Kinase C-Theta (Pkctheta). *J. Biol. Chem.* **2004**, *279*, 50401−50409.
(44) Grodsky, N.; Li, Y.; Bouzida, D.; Love, R.; Jensen, J.; Nodes, B.; Nonomiya, J.; Grant, S. Structure of the Catalytic Domain of Human Protein Kinase C Beta II Complexed with a Bisindolylmaleimide Inhibitor. *Biochemistry* **2006**, *45*, 13970−13981.
(45) Hartigan, J. A.; Wong, M. A. A K-Means Clustering Algorithm. *Appl. Stat.* **1979**, *28*, 100−108.
(46) R Development Core Team. *R: A Language and Environment for Statistical Computing*, version 2.4.1; R Foundation for Statistical Computing: Vienna, Austria, 2006.
(47) *Matlab*, version 15; Mathworks Natick, Massachusetts, 2006.
(48) Fischer, J. R.; Fricker, P.; Rarey, M.; Gastreich, M.; Hindle, S.; Sonnenburg, F.; Lemmen, C. Ftreesweb: A Web Interface to Feature Trees. http://www.zbh.uni-hamburg.de/FTreesWeb (accessed Nov 2006).

CI700007B