Automated Discovery of Active Motifs in Three Dimensional Molecules

Xiong Wang
Comp. & Infor. Sci.
New Jersey Inst. Tech.
Newark, NJ 07102
xwang@homer.njit.edu

Jason T. L. Wang
Comp. & Infor. Sci.
New Jersey Inst. Tech.
Newark, NJ 07102
jason@village.njit.edu

Dennis Shasha
Courant Institute
New York University
New York, NY 10012
shasha@cs.nyu.edu

Bruce Shapiro
Lab. of Math. Biology
National Cancer Institute
Frederick, MD 21702
bshapiro@ncifcrf.gov

Sitaram Dikshitulu
Data/Knowledge Eng. Lab.
New Jersey Inst. Tech.
Newark, NJ 07102

Isidore Rigoutsos
IBM Watson Research Ctr.
Yorktown Heights, NY 10598
rigoutso@watson.ibm.com

Kaizhong Zhang
Univ. of Western Ontario
Ontario, Canada N6A 5B7
kzhang@csd.uwo.ca

Abstract

In this paper we present a method for discovering approximately common motifs (also known as active motifs) in three dimensional (3D) molecules. Each node in a molecule is represented by a 3D point in the Euclidean Space and each edge is represented by an undirected line segment connecting two nodes in the molecule. Motifs are rigid substructures which may occur in a molecule after allowing for an arbitrary number of rotations and translations as well as a small number (specified by the user) of node insert/delete operations in the motifs or the molecule. (We call this "approximate occurrence.") The proposed method combines the geometric hashing technique and block detection algorithms for undirected graphs. To demonstrate the utility of our algorithms, we discuss their applications to classifying three families of molecules pertaining to antibacterial sulfa drugs, anti-anxiety agents (benzodiazepines) and antiadrenergic agents (β receptors). Experimental results indicate the good performance of our algorithms and the high quality of the discovered motifs.[1]

Introduction

Recently, a significant body of research has been performed for data mining in non-traditional data types such as sequences (Bailey & Elkan 1995; Hofacker *et al.* 1996; Mannila, Toivonen, & Verkamo 1995; Wang *et al.* 1994), trees (Wang *et al.* 1996) and graphs (Conklin, Fortier, & Glasgow 1993; Djoko, Cook, & Holder 1995) that commonly arise in scientific disciplines (Fayyad, Haussler, & Stolorz 1996; Shek *et al.* 1996). This paper focuses on a specific scientific domain, namely biochemistry, and presents techniques for discovering approximately common motifs (also known as *active motifs*) in molecules. Topologically, molecules are three dimensional (3D) graphs (Rigoutsos, Platt, & Califano 1996). In biology, the tertiary structures of proteins are also 3D graphs (Fischer *et al.* 1992).

3D Graph Representation of Molecules

Each node in a molecule is represented by a 3D point in the Euclidean Space. Each edge is represented by an undirected line segment connecting two nodes in the molecule. A molecule can be divided into one or more rigid substructures (Rigoutsos, Platt, & Califano 1996). In chemical compounds, for example, a ring is a rigid substructure. Thus, a molecule can be represented by a 3D graph and a rigid substructure is a connected subgraph in it.

Example 1. Consider the graph G in Fig. 1 containing two rigid substructures. Nodes in the substructures are numbered 0, 1, 2, 3, 4, 5 and 6, 7, 8, 9, 10, respectively. The two substructures in the graph can rotate with respect to each other around the edge $\{5, 6\}$.

The graph can be divided into substructure 0 (Str_0) and substructure 1 (Str_1) (see Fig. 2). Thus, Str_0

consists of nodes numbered 0, 1, 2, 3, 4, 5 as well as edges connecting these nodes (Fig. 2(a)). Str_1 consists of nodes numbered 6, 7, 8, 9, 10 as well as edges connecting them (Fig. 2(b)). $\square$

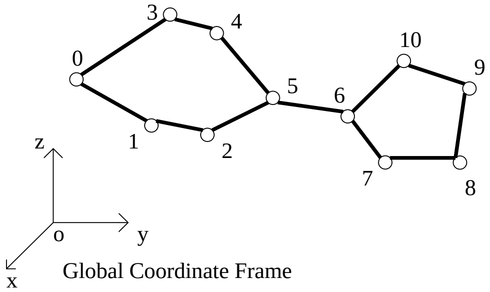

Fig. 1. A 3D graph.

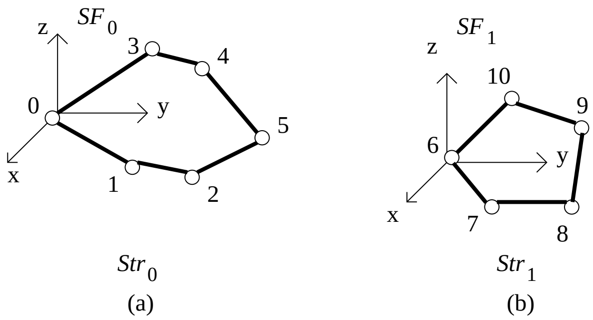

Fig. 2. The rigid substructures of the graph in Fig. 1.

We attach a local coordinate frame to each substructure. For instance, let us focus on the substructure Str_0 in Fig. 2. We attach a local coordinate frame to the node numbered 0 as shown in Fig. 2(a). This local coordinate frame is represented by three basis points P_{b_1}, P_{b_2} and P_{b_3}, with coordinates $P_{b_1}(x_0, y_0, z_0)$, $P_{b_2}(x_0+1, y_0, z_0)$ and $P_{b_3}(x_0, y_0+1, z_0)$, respectively. The origin is P_{b_1} and the three basis vectors are $\vec{V}_{b_1,b_2}$, $\vec{V}_{b_1,b_3}$, and $\vec{V}_{b_1,b_2} \times \vec{V}_{b_1,b_3}$. (Here, $\vec{V}_{b_1,b_2}$ represents the vector starting at point P_{b_1} and ending at point P_{b_2}. $\vec{V}_{b_1,b_2} \times \vec{V}_{b_1,b_3}$ stands for the cross product of the two corresponding vectors.) We refer to this coordinate frame as Substructure Frame 0, denoted SF_0. Note that, for any i in the graph with global coordinate $P_i(x_i, y_i, z_i)$, we can find a local coordinate of i with respect to SF_0, denoted P'_i, where

$$P'_i = \vec{V}_{b_1,i} = (x_i - x_0, y_i - y_0, z_i - z_0).$$

Active Motifs in Graphs

We consider a motif in a graph G to be a rigid substructure of G. Deleting a node v from a motif (graph) means to remove the corresponding point from the Euclidean Space and make the edges touching v connect with one of its neighbors v'. [This amounts to contraction of the edge between v and v' (Gabow, Galil, & Spencer 1984).] Inserting a node v into a graph is to add the corresponding point to the Euclidean Space and make a node v' and a subset of its neighbors become the neighbors of v. Graph G matches graph G' with n mutations if by applying an arbitrary number of rotations and translations as well as n node insert/delete operations, one can transform G to G'. A motif M *approximately occurs* in a graph G (or G approximately contains M) within n mutations if M matches some subgraph of G with n mutations or fewer where n is chosen by the user. We are interested in finding motifs that approximately occur in a set of graphs.

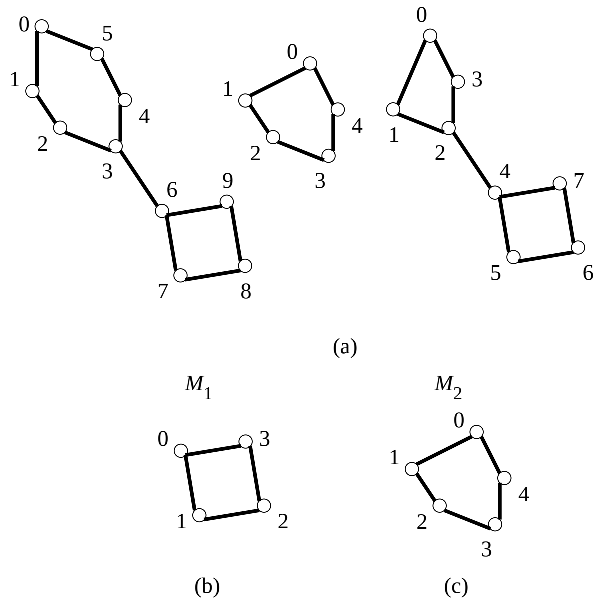

Fig. 3. (a) The set $\mathcal{S}$ of three graphs; (b) the motif exactly occurring in two graphs in $\mathcal{S}$; (c) the motif approximately occurring, within one mutation, in all the three graphs.

Example 2. Consider the set $\mathcal{S}$ of three graphs in Fig. 3(a). Suppose only exactly coinciding substructures (i.e., no mutations) occurring in at least two graphs and having size greater than 2 are considered as 'active motifs.' Then $\mathcal{S}$ contains one active motif shown in Fig. 3(b). If rigid substructures having size greater than 4 and approximately occurring in all the three graphs within one mutation (i.e. one node delete/insert is allowed in matching a motif with a graph), then $\mathcal{S}$ contains one active motif shown in Fig. 3(c). $\square$

To discover active motifs, we first find candidate substructures from the graphs and then evaluate their activity using the geometric hashing technique (Lamdan & Wolfson 1988). To our knowledge, there are two research groups having done work that is closely related to ours: Djoko *et al.* (Djoko, Cook, & Holder 1995) studied techniques for substructure discovery in two dimensional graphs; Fischer *et al.* (Fischer *et al.* 1992) searched for known motifs in 3D proteins. The novelty of our approach is to discover approximately common 3D motifs without prior knowledge of their topologies, positions, or occurrence frequency in the graphs. We use chemical molecules as an example in the paper, though our techniques should generalize to any scientific domain where data is naturally represented as 3D graphs.

Discovery Algorithm

Terminology

Let $\mathcal{S}$ be a set of 3D graphs. The occurrence number of a motif M is the number of graphs G in $\mathcal{S}$ that approximately contain M within the allowed number of mutations. Formally, the occurrence number of a motif M with respect to mutation d and set $\mathcal{S}$, denoted $occurrence\_no^d_{\mathcal{S}}(M)$, is k if there are k graphs in $\mathcal{S}$ that contain M within d mutations. For example, consider Fig. 3 again. Let $\mathcal{S}$ contain the three graphs in Fig. 3(a). Then $occurrence\_no^0_{\mathcal{S}}(M_1) = 2$; $occurrence\_no^1_{\mathcal{S}}(M_2) = 3$.

Given a set $\mathcal{S}$ of 3D graphs, our algorithm tries to find all the motifs M where M occurs in at least $Occur$ graphs in $\mathcal{S}$ within the allowed number of mutations Mut and $|M| \geq Size$, where $|M|$ represents the size, i.e., the number of nodes, of the motif M. (Mut, $Occur$ and $Size$ are user-specified parameters.) We can use the discovered results in several ways. For example, chemists may attempt to evaluate whether the approximately common motifs are in fact the active sites; biologists and computer scientists may use the motifs to classify new molecules into one family or another as we will show in the "Experimental Results" section.

Generating Candidate Motifs

Our algorithm consists of two phases: (1) find candidate motifs from the graphs in $\mathcal{S}$; and (2) evaluate the activity of the candidate motifs to determine which of them satisfy the specified requirements.

In phase (1), for each graph G in $\mathcal{S}$, we start with the node having the largest degree and use a modified depth-first search algorithm for finding blocks (McHugh 1990) to decompose the graph into substructures. Instead of locating all edges belonging to a block as described in the original algorithm, we locate all nodes of that block. The algorithm then marks out the detected blocks and partitions the remaining portion of the graph into unordered subtrees where each subtree is rooted at some node v in a block or at a node connected to v.[2] We then throw away the substructures (blocks and unordered subtrees) M where $|M| < Size$. The remaining substructures constitute the candidate motifs generated from G. This algorithm runs in time linearly proportional to the number of edges in G.

Hashing Candidate Motifs

Phase (2) of the discovery algorithm consists of two subphases. In subphase A of phase (2), we hash the candidate motifs generated from all graphs of $\mathcal{S}$ into a three dimensional hash table. For purposes of exposition, consider the substructure Str_0 in Fig. 2. We choose any three nodes in Str_0 and calculate their three dimensional hash function value as follows. Suppose the chosen nodes are numbered i, j, k and have global coordinates $P_i(x_i, y_i, z_i)$, $P_j(x_j, y_j, z_j)$ and $P_k(x_k, y_k, z_k)$, respectively. Calculate l_1, l_2, l_3 where

$$\begin{aligned}
l_1 &= ((x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2) \\
&\quad \times 100 \bmod Prime \bmod Nrow, \\
l_2 &= ((x_i - x_k)^2 + (y_i - y_k)^2 + (z_i - z_k)^2) \\
&\quad \times 100 \bmod Prime \bmod Nrow, \\
l_3 &= ((x_k - x_j)^2 + (y_k - y_j)^2 + (z_k - z_j)^2) \\
&\quad \times 100 \bmod Prime \bmod Nrow.
\end{aligned}$$

Here, we use a multiplier 100 and truncate the digits after the 2^{nd} position on the right of the decimal point. The reason for using the multiplier is that we want the digits after the decimal point to contribute to the distribution of the hash values. We truncate the values because only the first 2 digits are accurate. (The multiplier is a parameter value observed in the experiments and is adjustable for different data.) $Prime$ is a large prime number and $Nrow$ is the cardinality of the hash table in each dimension. The node-triplet (i, j, k) will be hashed to the three dimensional bucket $h[l_1][l_2][l_3]$. Intuitively we use the squares of the lengths of the three edges connecting the three chosen nodes to determine the bucket address. Stored in that address are the graph identification number and the substructure identification number. In addition, we store the coordinates of the basis points of Substructure Frame 0 (SF_0) with respect to the three chosen nodes.

Specifically, suppose the chosen nodes i, j, k are not collinear. We can construct another local coordinate

[2] Thus, for example, running the algorithm on the graph in Fig. 1 yields the two substructures in Fig. 2.

frame using $\vec{V}_{i,j}$, $\vec{V}_{i,k}$ and $\vec{V}_{i,j} \times \vec{V}_{i,k}$ as basis vectors.[3] The coordinates of P_{b_1}, P_{b_2}, P_{b_3} with respect to this local coordinate frame, denoted $SF_0[i,j,k]$, form a 3×3 matrix which is calculated as follows:

$$SF_0[i,j,k] = \begin{pmatrix} \vec{V}_{i,b_1} \\ \vec{V}_{i,b_2} \\ \vec{V}_{i,b_3} \end{pmatrix} \times A^{-1}$$

where

$$A = \begin{pmatrix} \vec{V}_{i,j} \\ \vec{V}_{i,k} \\ \vec{V}_{i,j} \times \vec{V}_{i,k} \end{pmatrix}$$

Thus suppose the graph in Fig. 1 has identification number 12. The hash table entry for the three chosen nodes i, j, k is $(12, 0, SF_0[i,j,k])$. Since there are 6 nodes in the substructure Str_0, we have $\begin{pmatrix} 6 \\ 3 \end{pmatrix} =$ 20 node-triplets generated from the substructure and therefore 20 entries in the hash table for the substructure.

Example 3. Suppose the global coordinates for the nodes numbered 0, 1, 2, 3, 4 in the substructure Str_0 of Fig. 2(a) are

$$P_0(1.0178, 1.0048, 2.5101),$$
$$P_1(1.2021, 2.0410, 2.0020),$$
$$P_2(1.3960, 2.9864, 2.0006),$$
$$P_3(0.7126, 2.0490, 3.1921),$$
$$P_4(0.7610, 2.7125, 3.0124).$$

The basis points of SF_0 have global coordinates

$$P_{b_1}(1.0178, 1.0048, 2.5101),$$
$$P_{b_2}(2.0178, 1.0048, 2.5101),$$
$$P_{b_3}(1.0178, 2.0048, 2.5101).$$

Fig. 4 shows the local coordinates, with respect to SF_0, of the nodes 0, 1, 2, 3 and 4.

[3] Note that the proper order of choosing the nodes in a triplet is significant. We determine the order of the three nodes by considering the triangle formed by them. The first node chosen always opposes the longest edge of the triangle and the third node opposes the shortest edge. Thus, the order is unique if the triangle is not isosceles or equilateral, which usually holds when the coordinates are floating point numbers. In other cases we store all configurations obeying the longest-shortest rule described above. When constructing the local coordinate frame, we use the first node as the origin of coordinates and the vector corresponding to the shortest edge as the X-axis.

$$P_0'(0.0000, 0.0000, 0.0000),$$
$$P_1'(0.1843, 1.0362, -0.5081),$$
$$P_2'(0.3782, 1.9816, -0.5095),$$
$$P_3'(-0.3052, 1.0442, 0.6820),$$
$$P_4'(-0.2568, 1.7077, 0.5023).$$

Fig. 4. The local coordinates, with respect to SF_0, of nodes 0, 1, 2, 3, 4 in the substructure Str_0 of Fig. 2(a).

Now, let $Prime$ be 1009 and $Nrow$ be 62. Thus, for example, for the nodes numbered 1, 2 and 3, the hash table address is $h[31][41][28]$ and

$$SF_0[1,2,3] = \begin{pmatrix} -1.0567 & 0.3578 & 0.1739 \\ -0.8758 & 0.0719 & 0.9072 \\ -0.0359 & 0.4175 & 0.0240 \end{pmatrix}$$

As another example, for the nodes numbered 1, 4 and 2, the hash table address is $h[31][26][42]$ and

$$SF_0[1,4,2] = \begin{pmatrix} 0.3694 & -1.3082 & -0.2429 \\ -0.0435 & -0.8571 & -1.0067 \\ 0.4552 & -0.3437 & -0.0869 \end{pmatrix}$$

Similarly, for the substructure Str_1, we attach a local coordinate frame SF_1 to node 6 as shown in Fig. 2(b). There are 10 hash table entries for Str_1, each having the form $(12, 1, SF_1[l,m,n])$ where l, m, n are any three nodes in Str_1. □

Evaluating the Activity of a Motif

Let $\mathcal{H}$ be the resultant hash table after hashing the candidate motifs (substructures) generated from all graphs in $\mathcal{S}$. In subphase B of phase (2) of our discovery algorithm, we evaluate the activity of each candidate motif M by rehashing M into $\mathcal{H}$. Each substructure (motif) in $\mathcal{H}$ is associated with a counter, which is updated as illustrated by the following example.

Suppose the two substructures (motifs) of graph G with identification number 12 in Fig. 1 have already been stored in the hash table $\mathcal{H}$. Suppose i, j, k are three nodes in the substructure Str_0 of G. Thus for this triplet, its entry in the hash table is $(12, 0, SF_0[i,j,k])$. Now consider another motif M. We hash M using the same hash function. Let u, v, w be three nodes in M that have the same hash address as i, j, k (i.e. the triplet $[u, v, w]$ hits the substructure Str_0). Calculate

$$SF_M = SF_0[i,j,k] \times \begin{pmatrix} \vec{V}_{u,v} \\ \vec{V}_{u,w} \\ \vec{V}_{u,v} \times \vec{V}_{u,w} \end{pmatrix} + \begin{pmatrix} P_u \\ P_u \\ P_u \end{pmatrix}$$

Intuitively, SF_M contains the coordinates of the three basis points of the Substructure Frame 0 (SF_0)

with respect to the global coordinate frame in which the motif M is given. The counter of the substructure Str_0 equals the total number of node-triplets of M that hit Str_0 and that yield the same SF_M.

Example 4. Consider the motif M in Fig. 5.

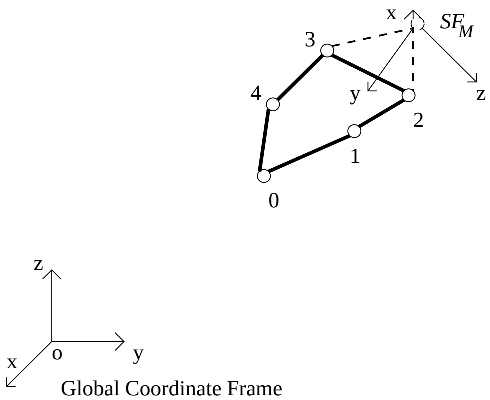

Fig. 5. A substructure (motif) M.

In M, the nodes numbered 0, 1, 2, 3, 4 match, after rotation, the nodes numbered 5, 4, 3, 1, 2 in the substructure Str_0 in Fig. 2(a). The node numbered 0 in Str_0 does not appear in M (i.e. it is to be deleted). Thus, M matches Str_0 with 1 mutation. If the user-specified mutation number Mut is 1 or greater, then M satisfies that requirement.

Now suppose in M, the global coordinates of nodes numbered 1, 2, 3 and 4 are

$$P_1(-0.269000, 4.153153, 2.911494),$$
$$P_2(-0.317400, 4.749386, 3.253592),$$
$$P_3(0.172100, 3.913515, 4.100777),$$
$$P_4(0.366000, 3.244026, 3.433268).$$

For the nodes numbered 3, 4 and 2, the bucket address in the hash table is $h[31][41][28]$ and

$$SF_M = \begin{pmatrix} -0.012200 & 5.005500 & 4.474200 \\ 0.987800 & 5.005500 & 4.474200 \\ -0.012200 & 4.298393 & 3.767093 \end{pmatrix}$$

For the nodes numbered 3, 1 and 4, the bucket address is $h[31][26][42]$ and

$$SF_M = \begin{pmatrix} -0.012200 & 5.005500 & 4.474200 \\ 0.987800 & 5.005500 & 4.474200 \\ -0.012200 & 4.298393 & 3.767093 \end{pmatrix}$$

Referring to Example 3, these two matches (hits) have the same SF_M, and therefore the counter for the substructure Str_0 of graph 12 is incremented by 2. In total, the counter of Str_0 after hashing M has value 10 in this example. □

Note that, for any node i in the motif M with global coordinate $P_i(x_i, y_i, z_i)$, it has a local coordinate with respect to SF_M, denoted P_i', where

$$P_i' = \vec{V}_{c_1,i} \times \begin{pmatrix} \vec{V}_{c_1,c_2} \\ \vec{V}_{c_1,c_3} \\ \vec{V}_{c_1,c_2} \times \vec{V}_{c_1,c_3} \end{pmatrix}$$

Here P_{c_1}, P_{c_2} and P_{c_3} are the three basis points of SF_M and $\vec{V}_{c_1,i}$ is the vector starting at P_{c_1} and ending at P_i. Thus, for example, the local coordinates, with respect to SF_M, of nodes 3, 4 and 2 in M are

$$P_3'(0.184300, 1.036200, -0.508100),$$
$$P_4'(0.378200, 1.981600, -0.509500),$$
$$P_2'(-0.305200, 1.044200, 0.682000).$$

They match the local coordinates, with respect to SF_0, of nodes 1, 2 and 3 of the substructure Str_0 (cf. Fig. 4). Likewise, the local coordinate, with respect to SF_M, of node 1 in M is

$$P_1'(-0.256800, 1.707700, 0.502300),$$

which matches the local coordinate, with respect to SF_0, of node 4 of the substructure Str_0 (cf. Fig. 4).

Proposition. *Let n be the counter value of a substructure (motif) S in the hash table $\mathcal{H}$ after rehashing a candidate motif M. Let G be the graph from which S is generated. Let $|M| \geq Mut + 4$. If $n > \Theta_M$ where*

$$\Theta_M = \frac{(N-1) \times (N-2) \times (N-3)}{6}$$

and $N = |M| - Mut$, then M matches S (i.e. M approximately occurs in G) within Mut mutations.

Proof Sketch. When $n \geq 1$, there are at least three node matches between S and M. Since three nodes are enough to set SF_M at a fixed position and direction, all the other nodes in M will have definite coordinates under this SF_M. When another node-triplet match yielding the same SF_M occurs, it means that there are at least one more node match between S and M. If there are $N-1$ node matches, then $n \leq \Theta_M$. Therefore when $n > \Theta_M$, there are at least N node matches between S and M. □

Intuitively, our scheme is to hash node-triplets and match the triplets. This proposition says that we need a sufficient number of matches between triplets in order to infer there is a match between the corresponding motif and graphs.[4] The larger Mut is, the fewer triplet matches are needed. In Example 4, suppose the user-specified mutation number Mut is 1. The counter value of Str_0 is 10 after hashing M, which is greater than Θ_M = (5-2)(5-3)(5-4)/6 = 1. By the proposition, M should match the structure Str_0 within 1 mutation, which is correct.

Thus, after rehashing each candidate motif M into the hash table $\mathcal{H}$, we check the counter values of the substructures in $\mathcal{H}$. If there are less than $Occur$ graphs containing substructures whose counter value $> \Theta_M$, discard M. The remaining candidates are qualified motifs.

Experimental Results

We have implemented the proposed algorithms using the C programming language on a SunSPARC 20 workstation running Solaris version 2.4. The data used came from three families of 3D molecules pertaining to antibacterial sulfa drugs (Family 1), anti-anxiety agents (benzodiazepines) (Family 2) and antiadrenergic agents (β receptors) (Family 3). The 3D structures for each of the compounds were generated from the 2D molecular structure data listed in (Hansch, Sammes, & Taylor 1990). We used the molecular modeling software package SYBYL (version 6.0) to generate energy minimized structures.[5] Table 1 shows the statistics concerning the data and the motifs discovered from them. The parameter values used were $Size = 6$, $Mut = 2$, and $Occur = 3$.

Family	#of molecules	Max size	Min size	#of motifs found
1	43	62	19	46
2	26	59	30	20
3	48	59	20	17

Table 1. The number of chemical molecules used in the experiments, their sizes, and the number of motifs discovered from the molecules.

To evaluate the quality of the discovered motifs, we applied them to classifying the molecules using 10-way cross-validation. That is, we partitioned each family of molecules randomly into 10 subsets of as nearly equal size as possible. For each such subset $\mathcal{S}$, we took all molecules excluding $\mathcal{S}$ as the training sample and used the molecules in $\mathcal{S}$ as the test data.[6] We then found the active motifs from the training sample of each family using the discovery algorithm described in the previous section (the parameter values used were $Size = 6$, $Mut = 2$, $Occur = 1$). Each motif M of Family i is associated with a weight d where $d = (n_i - \max_j\{n_j\})/(|M|)$. Here $1 \leq i, j \leq 3$, $i \neq j$, and n_i is M's occurrence number in the training sample of Family i. Intuitively, the more frequently a motif occurs in its own family and the less frequently it occurs in the other two families, the higher its weight is. The motifs with the weight greater than zero were then used as the characteristic motifs for their corresponding family. The averaging motif weight for a family is the sum of the weights of the family's characteristic motifs divided by the total number of the family's characteristic motifs.

When classifying a test molecule Q, we first decomposed Q into rigid substructures Q^k, $1 \leq k \leq m$. We then calculated the number of characteristic motifs in Family i, denoted n_i^k, $1 \leq i \leq 3$, that matched Q^k within 2 mutations. Each Family i obtained a score N_i where $N_i = \sum_{k=1}^{m} n_i^k$. The molecule Q was classified into Family i with maximal N_i. If two families tied on their scores, Q was classified into the family with the smaller averaging motif weight. If the scores were 0 for all families (i.e. the test molecule did not match any characteristic motif), then the "no-opinion" verdict was given.

The metrics used to evaluate the effectiveness of our classification algorithm are precision rate (PR) and no-opinion rate (NR), where $PR = (NumCorrect)/(NumTest) \times 100\%$ and $NR = (NumNoOpinion)/(NumTest) \times 100\%$. $NumCorrect$ is the number of test molecules classified correctly, $NumNoOpinion$ is the number of test molecules obtaining the "no opinion" verdict, and $NumTest$ is the total number of test molecules. Our experimental results indicated a 91% precision rate and a 5% no-opinion rate on average.

The majority of misclassified molecules came from Family 3 (antiadrenergic agents). A close look at the data revealed why this happened. This family's averaging motif weight was less than 1. Also, very few active motifs existed in the family.

[4] Note that the proposition provides only the "sufficient" (but not the "necessary") condition for finding the motifs. In general, our algorithms may miss some motifs, though this case does not happen in the experimental results presented in the next section.

[5] SYBYL is a trademark of TRIPOS Associates, Inc. at St. Louis, Missouri.

[6] Thus, the training sample consisted of 90% and the test data consisted of 10% of the molecules. Subsequent experiments revealed that training with 50% of the molecules gave nearly the same results, suggesting that relatively little training data is enough to achieve good results.

Conclusion

Geometric hashing techniques have been used in many different applications, though few of them consider substructure matching. The only exception is (Rigoutsos, Platt, & Califano 1996), in which the authors proposed to use magic vectors for substructure matching. The choice of magic vectors is domain dependent and is based on the type of each individual graph. We extend the ideas there and provide a framework for substructure discovery in 3D graphs. Currently we are combining the techniques presented in the paper with the algorithms for acyclic graph matching (Zhang, Wang, & Shasha 1996) and integrating them into our previously developed pattern discovery toolkit for scientific and biochemical databases (Wang *et al.* 1994).

Acknowledgments

This work was supported by NSF grants IRI-9224601, IRI-9224602, IRI-9531548, IRI-9531554, and by the Natural Sciences and Engineering Research Council of Canada under Grant No. OGP0046373. We thank the KDD referees whose comments helped to improve the paper.

References

Bailey, T. L., and Elkan, C. 1995. The value of prior knowledge in discovering motifs with MEME. In *Proceedings of the 3rd International Conference on Intelligent Systems for Molecular Biology*, 21–29.

Conklin, D.; Fortier, S.; and Glasgow, J. 1993. Knowledge discovery in molecular databases. *IEEE Transactions on Knowledge and Data Engineering*, 5(6):985–987.

Djoko, S.; Cook, D. J.; and Holder, L. B. 1995. Analyzing the benefits of domain knowledge in substructure discovery. In *Proceedings of the 1st International Conference on Knowledge Discovery and Data Mining*, 75–80.

Fayyad, U.; Haussler, D.; and Stolorz, P. 1996. KDD for science data analysis: Issues and examples. In *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*, 50–56.

Fischer, D.; Bachar, O.; Nussinov, R.; and Wolfson, H. J. 1992. An efficient automated computer vision based technique for detection of three dimensional structural motifs in proteins. *J. Biomolec. Struct. Dynam.*, 9(4):769–789.

Gabow, H. N.; Galil, Z.; and Spencer, T. H. 1984. Efficient implementation of graph algorithms using contraction. In *Proceedings of the 25th Annual IEEE Symposium on Foundations of Computer Science*, 347–357.

Hansch, C.; Sammes, P. G.; and Taylor, J. B. 1990. *Comprehensive Medicinal Chemistry – The Rational Design, Mechanistic Study and Therapeutic Application of Chemical Compounds*, volume 6. Pergamon Press, Oxford, UK.

Hofacker, I. L.; Huynen, M. A.; Stadler, P. F.; and Stolorz, P. 1996. Knowledge discovery in RNA sequence families of HIV using scalable computers. In *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*, 20–25.

Lamdan, Y., and Wolfson, H. 1988. Geometric hashing: A general and efficient model-based recognition scheme. In *Proceedings of International Conference on Computer Vision*, 237–249.

Mannila, H.; Toivonen, H.; and Verkamo, A. I. 1995. Discovering frequent episodes in sequences. In *Proceedings of the 1st International Conference on Knowledge Discovery and Data Mining*, 210–215.

McHugh, J. A. 1990. *Algorithmic Graph Theory*. Prentice Hall, Englewood Cliffs, New Jersey.

Rigoutsos, I.; Platt, D.; and Califano, A. 1996. Flexible substructure matching in very large databases of 3D-molecular information. Research Report, IBM T. J. Watson Research Center.

Shek, E. C.; Muntz, R. R.; Mesrobian, E.; and Ng, K. 1996. Scalable exploratory data mining of distributed geoscientific data. In *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*, 32–37.

Wang, J. T. L.; Chirn, G.-W.; Marr, T. G.; Shapiro, B. A.; Shasha, D.; and Zhang, K. 1994. Combinatorial pattern discovery for scientific data: Some preliminary results. In *Proceedings of the 1994 ACM SIGMOD International Conference on Management of Data*, 115–125.

Wang, J. T. L.; Shapiro, B. A.; Shasha, D.; Zhang, K.; and Chang, C.-Y. 1996. Automated discovery of active motifs in multiple RNA secondary structures. In *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*, 70–75.

Zhang, K.; Wang, J. T. L.; and Shasha, D. 1996. On the editing distance between undirected acyclic graphs. *International Journal of Foundations of Computer Science*, 7(1):43–57.