RASCAL: Calculation of Graph Similarity using Maximum Common Edge Subgraphs

JOHN W. RAYMOND[1], ELEANOR J. GARDINER[2] AND PETER WILLETT[2]

[1]*Pfizer Global Research and Development, Ann Arbor Laboratories, 2800 Plymouth Road, Ann Arbor, Michigan 48105, USA*
[2]*Krebs Institute for Biomolecular Research and Department of Information Studies, University of Sheffield, Western Bank, Sheffield S10 2TN, UK*
Email: john.raymond@pfizer.com

A new graph similarity calculation procedure is introduced for comparing labeled graphs. Given a minimum similarity threshold, the procedure consists of an initial screening process to determine whether it is possible for the measure of similarity between the two graphs to exceed the minimum threshold, followed by a rigorous maximum common edge subgraph (MCES) detection algorithm to compute the exact degree and composition of similarity. The proposed MCES algorithm is based on a maximum clique formulation of the problem and is a significant improvement over other published algorithms. It presents new approaches to both lower and upper bounding as well as vertex selection.

Received 16 August 2001; revised 29 April 2002

1. INTRODUCTION

It is often desired in many applications to compare objects represented as graphs and to determine the degree of similarity between the objects. This is sometimes referred to as inexact graph matching. Inexact graph matching is of importance in many applications such as protein ligand docking [1], video indexing [2] and computer vision [3, 4]. One algorithmic approach to this problem is to use the maximum common subgraph (MCS) between the graphs being compared. In this paper, a novel and efficient algorithm based on the edge induced formulation of the MCS problem is presented. The algorithm has been developed for use in chemical similarity searching applications where molecular structures are represented as graphs such that the atoms correspond to labeled vertices and the chemical bonds correspond to labeled edges, but it is directly applicable to a comparison of any collection of labeled graphs. For a more detailed description of the graphical representation of a chemical structure, the reader is referred to a text on chemical graph theory [5].

The similarities between the graphs representing molecules play an important role in many aspects of chemistry and, increasingly, biology. Examples include database searching [6], the prediction of biological activity [7], the design of combinatorial syntheses [8] and the interpretation of molecular spectra [9]. A wide range of types of similarity measures has been described [10]. However, many of these are too time-consuming when very large numbers of molecules need to be considered, as is the case for applications involving the searching and clustering of chemical databases which typically contain tens or even hundreds of thousands of molecules. In such cases, molecules are represented by a simple bit-string, in which bits are set to denote the presence in a molecule of pre-defined molecular substructures, such as a particular ring system or functional group. The similarity between two molecules is then calculated using an association coefficient based on the numbers of bits in common between the bit-strings describing those two molecules [10]. Although widely used for similarity purposes, such simple molecular representations have several limitations [11]; most importantly, a bit-string does not preserve the connectivity of the parent molecule. MCS-based similarity measures do take full account of molecular connectivity but have been little used to date in chemical information systems because of their computational requirements. The work reported here has been undertaken to address this situation.

The proposed inexact graph matching procedure RASCAL (RApid Similarity CALculation) consists of two components, screening and rigorous graph matching. The screening procedure is intended to determine rapidly whether the graphs being compared exceed some specified minimum similarity threshold (without resulting in any false dismissals) in order to avoid unnecessary calls to the more computationally demanding, graph matching procedure. The screening procedure is described in Section 2 along with an example calculation. The latter graph matching process consists of an efficient determination of the MCS using the graph similarity concept. Section 3 formulates the

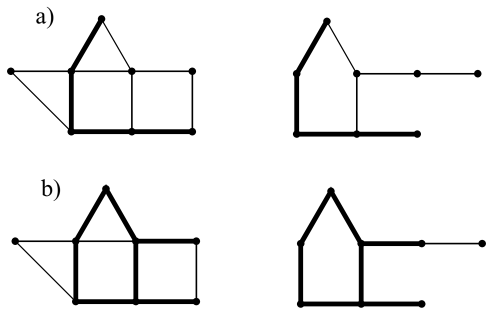


FIGURE 1. (a) Maximum common induced subgraph; (b) maximum common edge subgraph.

MCS graph matching procedure in terms of the maximum clique problem and presents improvements to existing clique detection algorithms for this purpose, including several new contributions. In Section 4, the RASCAL algorithm is compared with other published algorithms in several experimental trials where it is shown to be considerably more efficient.

1.1. Definitions and terminology

All graphs referred to in the following text are assumed to be simple, undirected graphs. For an introduction to graph-related concepts and notation, the reader is referred to an introductory text on graph theory [12]. $N_G(v_i)$ will denote the set of vertices adjacent to a vertex v_i. A *maximum clique*, $\omega(G)$, of a graph G is the largest set of mutually adjacent vertices in G. A *line graph* $L(G)$ is a graph whose vertex set consists of the edge set of G; therefore, if (v_i, v_j) is an edge in G it is also a vertex in $L(G)$. A pair of vertices in $L(G)$ are adjacent if the two corresponding edges in G are incident on each other.

A pair of graphs are said to be *isomorphic* if there is a one-to-one correspondence between their vertices and an edge only exists between two vertices in one graph if an edge exists between the two corresponding vertices in the other graph. An *induced subgraph* is a set S of vertices of a graph G and those edges of G with both endpoints in S. A graph G_{12} is a *common induced subgraph* of graphs G_1 and G_2 if G_{12} is isomorphic to induced subgraphs of G_1 and G_2. A *maximum common induced subgraph* (MCIS) consists of a graph G_{12} with the largest number of vertices meeting the aforementioned property. Related to the MCIS is the *maximum common edge subgraph* (MCES). An MCES is a subgraph consisting of the largest number of edges common to both G_1 and G_2. Note that the MCIS or MCES between two graphs is not necessarily connected or unique by definition. Figure 1a illustrates an MCIS between two graphs (highlighted in bold), and Figure 1b demonstrates an MCES between the same two graphs.

2. SCREENING PROCEDURE

Using the classification of Sanfeliu and Fu [13], graph distance/similarity measures can be categorized into two classes:

(1) *Feature-based distances.* A set of features or invariants is established from a structural description of a graph, and these features are then used in a vector representation to which various distance or similarity measures can be applied.
(2) *Cost-based distance.* The distance or similarity between two graphs reflects the number of prescribed edit operations that are required in order to transform one graph into the other.

The proposed two-tiered screening system is a cost-based method that has been developed to be used in conjunction with an MCES algorithm. It will be referred to as the cost-vector approach in order to differentiate it from feature-based methods. While the new screening system is almost as simple as the feature-based approach from a computational perspective, it has the desirable features that it cannot result in any false dismissals of graphs that are in fact similar, and it provides an upper bound on the size of the MCES between two molecular graphs. The procedure forms a screening hierarchy whereby the calculated upper bound for the second tier is always less than or equal in magnitude to the value calculated in the first tier. Following the formal treatment in Sections 2.1 and 2.2, we present an example illustrating the proposed screening procedure.

2.1. First tier cost-vector

Degree sequences of a graph have been used by other authors to establish upper bounds on graph invariants [14, 15] and recently for indexing graph databases [16]. In our proposed first tier screening procedure, we have used degree sequences to develop a novel method to calculate an upper bound on the size of an MCES between a pair of graphs. First, the set of vertices in each graph is partitioned into l partitions by label type, and then sorted in non-increasing order by degree. Let L_i^1 and L_i^2 denote the sorted degree sequences in partition i in graphs G_1 and G_2, respectively. An upper-bound on the similarity between a pair of graphs can be given as follows:

$$V(G_1, G_2) = \sum_{i=1}^{l} \min\{|L_i^1|, |L_i^2|\}$$

$$E(G_1, G_2) = \left\lfloor \sum_{i=1}^{l} \sum_{j=1}^{\max\{|L_i^1|,|L_i^2|\}} \frac{\min\{d(v_j^1), d(v_j^2)\}}{2} \right\rfloor$$

$$\mathrm{sim}^{\mathrm{cv}}(G_1, G_2) = \frac{(V(G_1, G_2) + E(G_1, G_2))^2}{(|V(G_1)| + |E(G_1)|) \cdot (|V(G_2)| + |E(G_2)|)}.$$

Note that the term to the right of $E(G_1, G_2)$ is divided by two since each edge in this formulation is counted twice and, if $|L_i^1| \neq |L_i^2|$, then entries of value zero are appended to the shorter sequence to make the sequences of equal length. It is clear that the similarity measure $\mathrm{sim}^{\mathrm{cv}}(G_1, G_2)$ ranges from

0 to 1 and that it obeys the following inequality,

$$\mathrm{sim}^{\mathrm{cv}}(G_1, G_2) \geq \frac{(|V(G_{12})| + |E(G_{12})|)^2}{(|V(G_1)| + |E(G_1)|) \cdot (|V(G_2)| + |E(G_2)|)},$$

where G_{12} is the MCES between graphs G_1 and G_2. This measure has been investigated in depth by Johnson [17]. The value $\mathrm{sim}^{\mathrm{cv}}(G_1, G_2)$ can serve as an upper bound on the size of the MCES between any two graphs. For screening purposes, all that is necessary is to specify a minimum acceptable value for the MCES-based graph similarity measure. If the value determined by $\mathrm{sim}^{\mathrm{cv}}(G_1, G_2)$ is less than the minimum acceptable similarity, then a rigorous MCES comparison can be avoided. Since chemical graphs are of bounded degree, this procedure can be performed using a bucket sort in $O(n)$ time for chemical graphs or $O(n \log n)$ otherwise, where $n = \max\{|L_i^1|, |L_i^2|\}$.

2.2. Second tier cost-vector

The first tier procedure provides a rapid screening mechanism which takes advantage of local connectivity and vertex labels to help eliminate unnecessary and costly MCES comparisons, but it does not take account of edge labeling. For this purpose, a second tier screening scheme has been developed. Since it is more costly from a computational perspective, it is intended to be used on pairs of graphs that have passed the initial cost-vector screen.

An unambiguous integer code is assigned to each edge incident on a given vertex which incorporates the edge label along with the labels of both vertex endpoints. Instead of simply assigning the respective degree to each vertex in L_i^1 or L_i^2, each vertex is assigned its set of incident edge codes. If $f(L_i^1, L_i^2)$ is defined to be a linear assignment of compatible edge codes associated with each pair of vertices in the sequences L_i^1 and L_i^2 such that each vertex in L_i^1 is compared to each vertex in L_i^2, then the second tier cost-vector upper bound can be given as

$$E(G_1, G_2) = \left\lfloor \sum_{i=1}^{l} \frac{f(L_i^1, L_i^2)}{2} \right\rfloor$$

with $V(G_1,G_2)$ and $\mathrm{sim}^{\mathrm{cv}}(G_1, G_2)$ again being calculated as previously discussed. It has been found that the linear assignment algorithm described by Carpaneto *et al.* [18] performs well for this purpose. This algorithm has a worst case time complexity of $O(n^3)$ where $n = \max\{|L_i^1|, |L_i^2|\}$.

2.3. Example of cost-vector screening

The following example illustrates the operation of the cost-vector screening approach. Figure 2 depicts two molecular graphs, methadone and meperidine, with their respective MCESs highlighted in bold. Since there are 16 bonds in the MCES, $|E(G_{12})| = 16$, and since the number of atoms in common between the two structures is 17 (including any

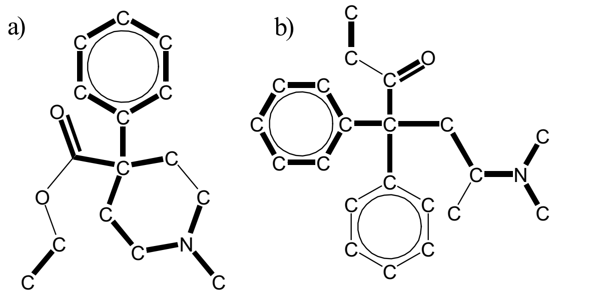


FIGURE 2. An example MCES comparison: (a) meperidine; (b) methadone.

isolated atoms), $|V(G_{12})| = 17$. In order to calculate an MCES-based similarity measure, the following parameters are used:

methadone:	$N_{\mathrm{a}}^{\mathrm{meth}} = 23$	(number of atoms in methadone)
	$N_{\mathrm{e}}^{\mathrm{meth}} = 24$	(number of bonds in methadone)
meperidine:	$N_{\mathrm{a}}^{\mathrm{mep}} = 18$	(number of atoms in meperidine)
	$N_{\mathrm{e}}^{\mathrm{mep}} = 19$	(number of bonds in meperidine).

The MCES-based similarity is then calculated using the Johnson metric to be 0.63.

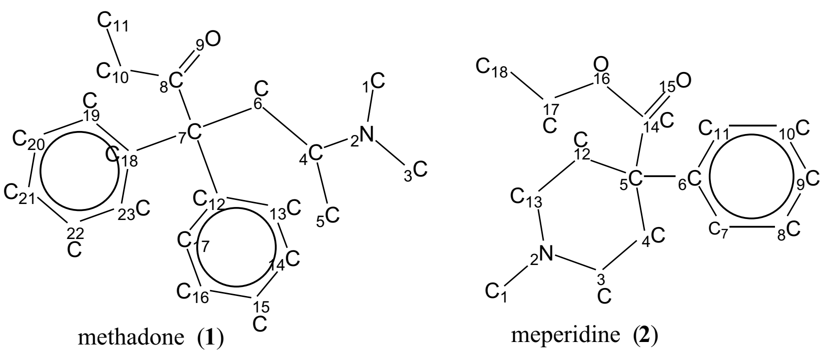
Applications of similarity in chemical information systems focus on pairs of molecules having a high degree of structural resemblance. For example, in drug discovery programs, given a molecule that has been shown to exhibit some useful biological activity (e.g. lowering blood pressure or restricting tumor growth), molecules in a chemical database having a high degree of similarity are also likely to exhibit some activity and can hence be selected for biological testing [19, 20]. It is hence common to consider just the nearest neighbors of a query molecule (i.e. those having a similarity greater than some minimum threshold value).

In the present case, if we were to set this threshold to 0.70 then it is clear that this pair of molecular graphs does not fulfill the requirement, but since a rigorous MCES comparison is computationally expensive, it is desirable to see if it is possible to skip the unnecessary MCES comparison.

2.3.1. *First tier screen*

In order to initiate the first tier cost-vector comparison, the vertices are first separated into partitions according to label type. Any vertex whose corresponding label is not present in both graphs is removed from consideration. Since, in this case, both molecules have only carbon, oxygen and nitrogen atoms, this pruning is not applicable. This is shown in Figure 3. The entry $d(v_1) = 4$ (7) in column L_1^1 represents vertex number 7 in graph 1 (i.e. methadone) with degree 4 and of type carbon. Note that since column L_2^1 has one fewer element than L_2^2, it is padded with a zero to make the columns the same size.

$|V(G_1, G_2)|$ is calculated by summing the size of the set L_i with the fewer non-null elements in graphs 1 and 2. Since L_1^2 has 15 non-null elements and L_1^1 has 21, $|V(G_1, G_2)|$ is incremented by 15. Considering the other



methadone (**1**) meperidine (**2**)

L^1_1=C	L^1_2=O	L^1_3=N	L^2_1=C	L^2_2=O	L^2_3=N	min $\{d(v^1_j), d(v^2_j)\}$		
						L_1=C	L_2=O	L_3=N
$d(v_1)$= 4 (7)	$d(v_1)$= 1 (9)	$d(v_1)$= 3 (2)	$d(v_1)$= 4 (5)	$d(v_1)$= 2 (16)	$d(v_1)$= 3 (2)	$d(v_1)$ = 4	$d(v_1)$= 1	$d(v_1)$= 3
$d(v_2)$= 3 (8)	0		$d(v_2)$= 3 (6)	$d(v_2)$= 1 (15)		$d(v_2)$ = 3	$d(v_2)$= 0	
$d(v_3)$= 3 (4)			$d(v_3)$= 3 (14)			$d(v_3)$ = 3		
$d(v_4)$= 3 (18)			$d(v_4)$= 2 (3)			$d(v_4)$ = 2		
$d(v_5)$= 3 (12)			$d(v_5)$= 2 (4)			$d(v_5)$ = 2		
$d(v_6)$= 2 (6)			$d(v_6)$= 2 (7)			$d(v_6)$ = 2		
$d(v_7)$= 2 (10)			$d(v_7)$= 2 (8)			$d(v_7)$ = 2		
$d(v_8)$= 2 (13)			$d(v_8)$= 2 (9)			$d(v_8)$ = 2		
$d(v_9)$= 2 (14)			$d(v_9)$= 2 (10)			$d(v_9)$ = 2		
$d(v_{10})$= 2 (15)			$d(v_{10})$= 2 (11)			$d(v_{10})$= 2		
$d(v_{11})$= 2 (16)			$d(v_{11})$= 2 (12)			$d(v_{11})$= 2		
$d(v_{12})$= 2 (17)			$d(v_{12})$= 2 (13)			$d(v_{12})$= 2		
$d(v_{13})$= 2 (19)			$d(v_{13})$= 2 (17)			$d(v_{13})$= 2		
$d(v_{14})$= 2 (20)			$d(v_{14})$= 1 (1)			$d(v_{14})$= 1		
$d(v_{15})$= 2 (21)			$d(v_{15})$= 1 (18)			$d(v_{15})$= 1		
$d(v_{16})$= 2 (22)			0			$d(v_{16})$= 0		
$d(v_{17})$= 2 (23)			0			$d(v_{17})$= 0		
$d(v_{18})$= 1 (1)			0			$d(v_{18})$= 0		
$d(v_{19})$= 1 (3)			0			$d(v_{19})$= 0		
$d(v_{20})$= 1 (5)			0			$d(v_{20})$= 0		
$d(v_{21})$= 1 (11)			0			$d(v_{21})$= 0		
					SUM =	32	1	3 = 36

FIGURE 3. An example of first tier cost-vector screening.

two partitions L_2 and L_3 results in $|V(G_1, G_2)| = 17$. $|E(G_1, G_2)|$ is determined using $\min\{d(v^1_j), d(v^2_j)\}$. This procedure is also illustrated in Figure 3. $|E(G_1, G_2)|$ is calculated by summing each column in the $\min\{d(v^1_j), d(v^2_j)\}$ set of columns, adding the resulting values together and then integer dividing the result by two. Since the resulting sum equals 36, $|E(G_1, G_2)| = 18$. Now the cost-vector similarity value $\text{sim}^{\text{cv}}(G_1, G_2)$ is calculated to be 0.70 using the Johnson measure. Since this value is equal to the minimum acceptable similarity of 0.70, the second tier screening process must be implemented.

2.3.2. *Second tier screen*

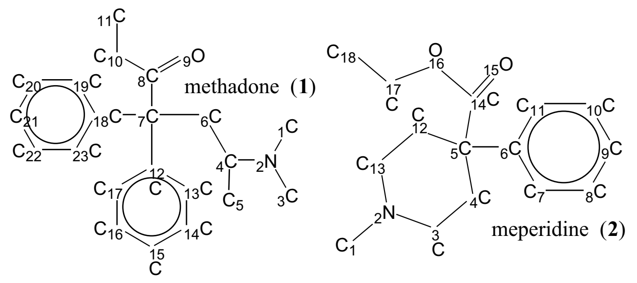
Since there are three distinct vertex labels present in both molecular graphs, three separate linear assignments will need to be performed in order to calculate the updated value of $|E(G_1, G_2)|$. The three cost matrices associated with these assignments are presented in Figure 4.

Considering the assignment of L_3 first, it is clear from Figure 3 that the assignment cost matrix will consist of only one element. In Figure 3, $d(v_1)$ in L^1_3 corresponds to vertex 2 in methadone and $d(v_1)$ in L^2_3 represents vertex 2 in meperidine. Since both vertices have three incident edges and all incident edges are the same type (N–C), all incident edges in one graph have a compatible edge in the other graph. The corresponding element in the cost matrix will therefore be 3. Since there is only one element in the cost matrix, the linear assignment is trivial and $f(L^1_3, L^2_3) = 3$.

This process has been repeated in Figure 4 for L_1 and L_2 consisting of vertices of type carbon and nitrogen. The L_1 cost matrix is considerably more complex than the other two. In this case an efficient linear assignment algorithm is necessary. The maximum assignment in each cost matrix is highlighted. For the L_1 cost matrix, the resulting maximum score is $f(L^1_1, L^2_1) = 29$ (i.e. the sum of the boldface elements). Note that it is irrelevant whether the computed assignment is unique as the upper bound is a scalar value. Therefore,

$$E(G_1, G_2) = \left\lfloor \sum_{i=1}^{l} \frac{f(L^1_i, L^2_i)}{2} \right\rfloor = \left\lfloor \frac{29}{2} + \frac{1}{2} + \frac{3}{2} \right\rfloor = 16.$$

The updated second tier cost-vector similarity is then calculated to be 0.63.



L_2 (oxygen) :

(i)	(v_i)	1	2
		16	*15*
1	*9*	0	**1**

L_3 (nitrogen):

(i)	(v_i)	1
		2
1	*2*	**3**

L_1 (carbon) :

(i)	(v_i)	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
		5	*6*	*14*	*3*	*4*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*17*	*1*	*18*
1	*7*	**4**	1	1	1	2	0	0	0	0	0	2	1	1	0	1
2	*8*	2	1	**2**	1	2	0	0	0	0	0	2	1	1	0	1
3	*4*	2	1	1	**2**	2	0	0	0	0	0	2	2	1	1	1
4	*18*	1	3	1	1	1	2	2	2	2	2	1	**1**	1	0	1
5	*12*	1	**3**	1	1	1	2	2	2	2	2	1	1	1	0	1
6	*6*	2	1	1	1	2	0	0	0	0	0	**2**	1	1	0	1
7	*10*	2	1	1	1	**2**	0	0	0	0	0	2	1	1	0	1
8	*13*	0	2	0	0	0	2	2	2	2	2	0	0	0	0	0
9	*14*	0	2	0	0	0	2	2	2	2	2	0	0	0	0	0
10	*15*	0	2	0	0	0	2	2	2	2	2	0	0	0	0	0
11	*16*	0	2	0	0	0	2	2	2	2	2	0	0	0	0	0
12	*17*	0	2	0	0	0	2	2	2	2	2	0	0	0	0	0
13	*19*	0	2	0	0	0	2	2	2	2	**2**	0	0	0	0	0
14	*20*	0	2	0	0	0	2	2	2	**2**	2	0	0	0	0	0
15	*21*	0	2	0	0	0	2	2	**2**	2	2	0	0	0	0	0
16	*22*	0	2	0	0	0	2	**2**	2	2	2	0	0	0	0	0
17	*23*	0	2	0	0	0	**2**	2	2	2	2	0	0	0	0	0
18	*1*	0	0	0	1	0	0	0	0	0	0	0	1	0	1	0
19	*2*	0	0	0	1	0	0	0	0	0	0	0	1	0	**1**	0
20	*5*	1	1	1	1	1	0	0	0	0	0	1	1	1	0	**1**
21	*11*	1	1	1	1	1	0	0	0	0	0	1	1	**1**	0	1

FIGURE 4. Second tier linear assignment cost matrices.

This upper bound value is less than 0.7, the minimum acceptable similarity; in this case, indeed, it is equal to the actual MCES-based similarity value that we would have obtained if the MCES calculation was carried out. The proposed screening method would, therefore, prevent the unnecessary MCES comparison between these two molecules. In concluding this section, it is important to emphasize that the currently proposed screening system is highly dependent upon the selection of an acceptable minimum similarity index (*MSI*). This implies that the user knows *a priori* what *MSI* constitutes a sensible threshold for similarity.

3. MCES ALGORITHM

3.1. Modular product

The MCS problem can be reduced to determining the maximum clique in the modular product graph, denoted as $G_1 \Diamond G_2$. This concept has been discovered on several occasions [21, 22, 23]. The modular product of two labeled factor graphs G_1 and G_2 is defined on the vertex set $V(G_1 \Diamond G_2) = V(G_1) \times V(G_2)$ where the respective vertex labels are compatible and two vertices (u_i, v_i) and (u_j, v_j) being adjacent whenever

$$(u_i, u_j) \in E(G_1) \text{ and } (v_i, v_j) \in E(G_2)$$
$$\text{and } w(u_i, u_j) = w(v_i, v_j)$$

or

$$(u_i, u_j) \notin E(G_1) \text{ and } (v_i, v_j) \notin E(G_2),$$

where $w(u_i, u_j) = w(v_i, v_j)$ indicates that the vertex and edge labels for each respective pair of vertices are compatible.

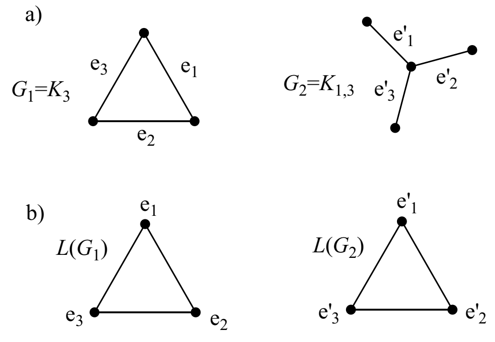


FIGURE 5. ΔY exchange.

3.2. Edge-induced isomorphism

The transformation of the MCIS formulation to the MCES problem is based upon the work of Whitney [24], as illustrated in Figure 5. Figure 5a shows two graphs $G_1 = K_3$ and $G_2 = K_{1,3}$, respectively. It is evident by visual inspection that the two graphs in Figure 5a are not isomorphic. Figure 5b depicts the line graphs of G_1 and G_2, respectively, and it is clear by inspection that the line graphs are isomorphic. This is referred to as a ΔY exchange. Whitney proved that, provided that a ΔY exchange does not occur, an isomorphism between two line graphs $L(G_1)$ and $L(G_2)$ induces an edge isomorphism between the root graphs (G_1 and G_2) of the two line graphs. By preventing the occurrence of a ΔY exchange during clique detection, it is then possible to use the modular product approach to determine the MCES between two graphs rather than the MCIS [25, 26].

This procedure is illustrated with the example in Figure 6. The chemical graphs G_1 and G_2 are shown in Figure 6a, and their respective line graphs are depicted in Figure 6b. Each vertex in the line graph $L(G_1)$ is labeled with its respective edge and vertex endpoint labels in G_1. Take for instance edge number 1 in graph G_1 of Figure 6a. This corresponds to vertex number 1 in the line graph $L(G_1)$ in Figure 6b. This vertex is labeled with the adjacent vertex pair C–O.

The edges in a labeled line graph are also labeled. Each edge between two vertices, v_i and v_j, in a labeled line

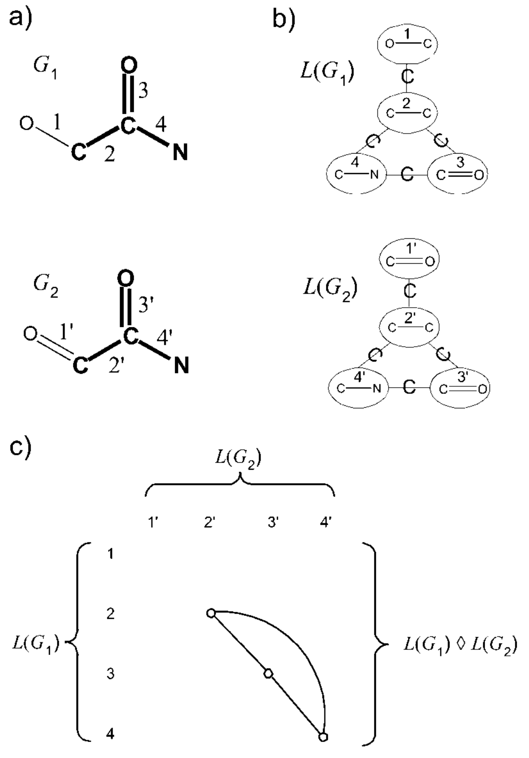


FIGURE 6. The modular product of line graphs.

graph $L(G_1)$ is labeled with the vertex label of the incident vertex in G_1 between the two edges in G_1 corresponding to v_i and v_j. For instance, edges 3 and 4 in graph G_1 of Figure 6a correspond to vertices 3 and 4 in the line graph $L(G_1)$ in Figure 6b. Since edges 3 and 4 in G_1 are incident on an atom of type carbon (C), then the corresponding edge in $L(G_1)$ is also labeled as being of type carbon.

Since a labeled line graph can then simply be assumed to be an arbitrary labeled graph, the modular product can be constructed as previously described. The modular product graph for the line graphs depicted in Figure 6b is presented in Figure 6c. Since no ΔY exchange has occurred, the clique depicted in the modular product graph corresponds to the MCES between the two labeled graphs in Figure 6a. The MCES is highlighted in bold face in the respective factor graphs.

In practice, it is straightforward to verify whether a ΔY exchange has occurred during clique detection. Since a clique in the modular product graph corresponds to a subgraph common to both G_1 and G_2, the respective degree sequences for a common subgraph projected on to G_1 and G_2 should be identical when the vertices are sorted according to degree. However, when a ΔY exchange has occurred the degree sequences of the common subgraph in G_1 and G_2 will not be the same, as evidenced by Figure 5.

3.3. Proposed clique-detection algorithm

Our clique-detection procedure is a branch and bound algorithm that incorporates improvements in both lower and upper bounding as well as node selection and pruning. A branch and bound algorithm attempts to explore the search tree by iteratively obtaining subproblems of previous problem instances visualized as branches in the search tree and is a common approach to the maximum clique problem [27]. This process can be described succinctly as the recursive implementation of the relation $\omega(G) = \max\{1 + \omega(N_G(v_i)), \omega(G\backslash v_i)\}$.

Bounding techniques for traversing the search tree generated using this procedure can be categorized into two specific types: (1) lower bounds and (2) upper bounds. A straightforward approach to reducing the number of branch traversals necessary to determine the maximum clique involves using the size of the largest clique so far detected [27] as a lower bound. Upper bounds set a largest possible size on the maximum clique that can exist at a specific search tree node. If this size is smaller than the size required to increase the size of the maximum clique thus far discovered, then it is not necessary to extend the search tree from the current search tree node and a backtrack can be instanced.

3.3.1. *Lower bounds*

Since it is widely known that the order of the largest clique thus far detected during a clique detection procedure can be useful in potential pruning heuristics, it seems reasonable to assume that an efficient heuristic capable of determining a relatively large clique in a graph G prior to initiating the clique detection could potentially prevent the investigation of numerous unnecessary search tree branches. In the proposed algorithm, an MCES-based minimum graph similarity heuristic capable of establishing a relatively large lower bound on the size of the maximum clique in a modular product graph is developed.

As noted previously, the MCES between two molecular graphs only has potential value if the two molecules are meaningfully similar; it follows that it is reasonable to determine the MCES between two molecular structures only if the similarity between the two structures is greater than some minimum established threshold. If a value for the minimum similarity index (*MSI*) is established as the minimum threshold, a lower bound on the size of the maximum clique, $|E(G_{12})|$, can be constructed by observing $\text{sim}(G_1, G_2) \geq \mathit{MSI}$ which for the previously discussed Johnson metric [17] can be shown to yield

$$|E(G_{12})| \geq (MSI(|V(G_1)| + |E(G_1)|)(|V(G_2)| + |E(G_2)|))^{1/2} - |V(G_1)| + \Delta V(G_1),$$

where $|V(G_1)| \leq |V(G_2)|$ and $\Delta V(G_1)$ is defined as the number of vertices in graph G_1 with no equivalent label in G_2.

3.3.2. *Upper bounds*

As previously stated, an upper bound sets a maximum value for the largest clique that can exist at a specific search tree node. The two schemes that have been investigated include approximate coloring and edge projection onto the factor graphs.

3.3.3. *Coloring approaches*

A vertex *k-coloring* of a graph G is a partition of the vertices $V(G)$ into independent sets $\{P_1, P_2, \ldots, P_k\}$ such that no vertex in any one partition set P_i is adjacent to any other vertex in the same partition set. The minimum k for which a graph G can be colored (i.e. partitioned) is called the *chromatic number* of G and is denoted by $\chi(G)$. Determining $\chi(G)$ has been shown to be in the same class of problems as the maximum clique and MCES problems, which are known as NP-complete, for which no known polynomial time complexity solutions are known [28]. It is clear from the definition of the chromatic number that $\omega(G) \leq \chi(G)$; therefore, a requirement for forward traversal in the search tree can be given by $h + \chi(G) > M$ where h is the current depth in the search tree and M is the size of the current maximum clique (i.e. lower bound).

Grimmett and McDiarmid [29] and Manvel [30] have shown that as the number of nodes of a graph G increases, the chromatic number starts to become an arbitrarily poor estimate for the upper bound of the maximum clique. Moreover, approximate coloring methods such as the greedy algorithm can give arbitrarily poor estimates of the chromatic number, further frustrating efforts to bound the size of the maximum clique in a graph. We have found that the estimated chromatic number obtained using the greedy algorithm [31] provides an effective upper bound within the range $125 \leq |V(G)| \leq 175$ when used in conjunction with edge projection, with poorer performances resulting if the value is set outside of this range. The median value of $|V(G)| = 150$ has hence been selected as the threshold for initiating greedy coloring in the proposed algorithm.

3.3.4. *Projection bounds*

Bessonov [32] has implemented a bounding technique based on the projection of vertex labels onto the graphs under consideration for the MCIS problem. Although a simple bound, the projective bound is almost always superior to the chromatic bound in practice. While the Bessonov implementation involved a projection bound for the MCIS problem, it is a simple matter to extend it to the MCES problem. The projective bounds are easily conceptualized by considering the modular product in a matrix representation (see Figure 6). Since each row in the modular product represents an edge in the graph G_1 and each column represents an edge in the graph G_2, each node in the modular product graph is assigned two labels, e_1^i for the ith edge in graph G_1 and e_2^j for the jth edge in graph G_2. To determine the two projective upper bounds K_α and K_β, all the vertices under consideration in the modular product graph are projected back onto the factor graphs G_1 and G_2; the number of distinct edges conserved in the projection of the modular product onto each respective factor graph G_1 and G_2 then represents the projective bounds K_α and K_β, respectively.

When the simple modular product in Figure 6 is projected onto the factor graphs $L(G_1)$ and $L(G_2)$, edges $\{2, 3, 4\}$ and

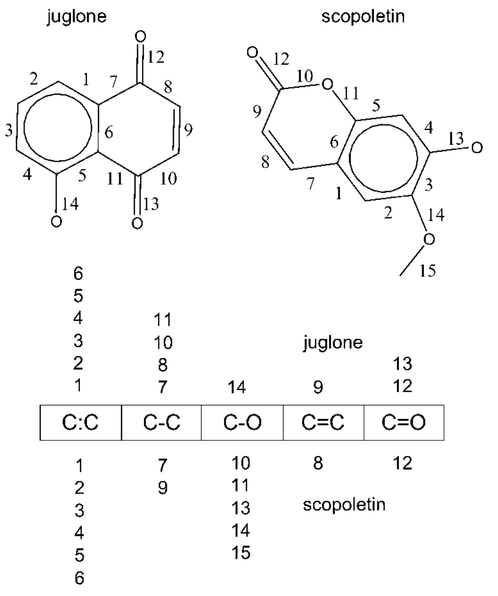


FIGURE 7. An example of a labeled projection upper bound.

$\{2', 3', 4'\}$ are conserved in $L(G_1)$ and $L(G_2)$, respectively. This results in $K_\alpha = 3$ and $K_\beta = 3$. The Bessonov projection upper bound is then given as $\min\{K_\alpha, K_\beta\}$ and, in this case, the projection bound is trivial since the maximum clique constitutes the modular product.

Although the simple projection of the modular product onto the factor graphs G_1 and G_2 provides a fast and effective upper bound for the size of the maximum clique in the modular product, we have found that this simple procedure does not take optimal advantage of vertex and edge labeling in the factor graphs. Let i represent an unambiguous identifier for a pair of adjacent vertices in a graph incorporating the edge label as well as the labels for the two endpoint vertices. Let S_i^1 indicate the set of all adjacent vertex pairs with the unique identifier i in graph G_1 and $\mathbf{S}^1$ denote the set of all such S_i^1. $\mathbf{S}^1 = \{S_1^1, S_2^1, \ldots, S_N^1\}$ where N is the number of unique S_i^1. We propose a new and sharper upper bound (K_{wp}) that is given as

$$K_{\text{wp}} = \sum_{i=1}^{N} \min\{|S_i^1|, |S_i^2|\}.$$

This process is best illustrated using an example. Figure 7 demonstrates the procedure for two molecules, juglone and scopoletin. It is clear from the figure that all bonded pairs of atoms in both molecular graphs have a corresponding bonded pair of atoms with the same label identifier in the other graph; therefore, the simple projection procedure will yield the upper bounds $K_\alpha = 14$ and $K_\beta = 15$. The value of K_{wp} is calculated as shown in Figure 7 by $6 + 2 + 1 + 1 + 1 = 11$.

The initial partition used for the proposed vertex selection procedure is based on the initial labeled projection (i.e. $h = 0$), and the upper bound resulting from this partitioning process will be referred to as K_{a}. Since the value of K_{a} at $h > 0$ will be dependent upon the algorithm's vertex

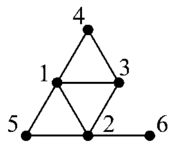


FIGURE 8. An undirected graph G.

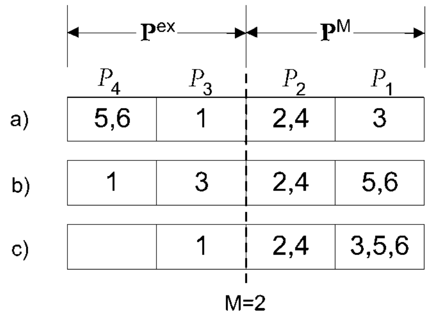


FIGURE 9. The vertex partitioning procedure.

selection procedure, the upper bound K_{wp} will be computed at each forward search tree node instance to account for the possibility that $K_{\text{wp}} < K_{\text{a}}$.

In the event of a backtrack instance, a modular product vertex is removed from consideration at a particular search tree node. It may then be possible to decrement K_{wp}. If the vertex v_i under consideration during a backtrack instance is the only vertex in a particular partition used to calculate K_{wp}, such as the vertex corresponding to edge 9 in juglone and edge 8 in scopoletin, then K_{wp} can be decremented.

3.3.5. *Vertex selection*

Selkow [33] has shown that an upper bound for the clique number in a graph can be given as $\omega(G) \leq \max\{\min\{1 + \varphi(N_G(v_i)), k\}\}$, where $v_i \in V(G)$ and k represents the index for the independent set partition P_k of vertices of $V(G)$ containing v_i and $\varphi(N_G(v_i))$ denotes an upper bound for the maximum clique of the graph induced by the vertices adjacent to vertex v_i. This observation serves as the impetus for the novel method for node selection in the proposed branch and bound algorithm. For instance, an arbitrary partitioning of the graph in Figure 8 into independent partitions is depicted in Figure 9a. If the current estimate for the maximum clique is $M = 2$ (i.e. the lower bound), then it is apparent that $\varphi(N_G(v_i))$ need only be calculated for vertices 1, 5 and 6, but if the partitions are sorted in non-increasing order according to the number of vertices in each partition, it is only necessary to calculate $\varphi(N_G(v_i))$ for vertices 1 and 3 (Figure 9b).

Figure 9b serves to illustrate the concept that, when trying to determine the maximum clique in a graph, the branch and bound procedure need only consider the vertices contained in the excess partitions $\mathbf{P}^{\text{ex}} = \{P_k, P_{k-1}, \ldots, P_{M+1}\}$. Furthermore, the number of vertices in the partitions $\mathbf{P}^{\text{ex}}$ and possibly the number of partitions k can be reduced by reassigning vertices in $\mathbf{P}^{\text{ex}}$ to partitions in $\mathbf{P}^M = \{P_M, P_{M-1}, \ldots, P_1\}$. If there exists a vertex v_i in a partition $\{P_j \mid P_j \in \mathbf{P}^{\text{ex}}\}$ and a partition $\{P_m \mid P_m \in \mathbf{P}^M\}$ such that $N_G(v_i) \cap P_m = \varnothing$, then v_i can be reassigned to partition P_m. This is illustrated in Figures 9b and 9c where node $v_i = 3$ in partition P_3 can be placed into partition P_1, thus reducing the number of partitions in $\mathbf{P}^{\text{ex}}$ and the number of nodes in $\mathbf{P}^{\text{ex}}$ that must be investigated.

The branching and node selection procedure then consists of the following operations.

(1) Perform an initial partitioning of the vertices for the root node of the search tree (i.e. the graph G) using the labeled edge projection procedure. Sort the resulting partitions $\mathbf{P}_{h=0} = \{P_{k0}, P_{k0-1}, \ldots, P_1\}$ such that $|P_{k0}| \leq |P_{k0-1}| \leq \ldots \leq |P_1|$ and reassign vertices in excess partitions $\mathbf{P}^{\text{ex}} = \{P_{k0}, P_{k0-1}, \ldots, P_{M+1}\}$ to partitions $\mathbf{P}^M = \{P_M, P_{M-1}, \ldots, P_1\}$ if possible.
(2) Choose a vertex v_i from partition P_{kh} in $\mathbf{P}^{\text{ex}}$ and copy the nodes to partitions in the set $\mathbf{P}_{h+1}$ using $P_{h+1} := \mathbf{P}_h \cap N_G(v_i)$. Increment the depth h by one. Sort the partitions $\mathbf{P}_h = \{P_{kh}, P_{kh-1}, \ldots, P_1\}$ as before and reassign vertices in excess partitions $\mathbf{P}^{\text{ex}} = \{P_{kh}, P_{kh-1}, \ldots, P_{M+1}\}$ to M-partitions $\mathbf{P}^M = \{P_M, P_{M-1}, \ldots, P_1\}$ if possible. Repeat (2).

The node selection procedure just outlined has the desirable property that the node v_i will typically have large degree in $N_G(v_i)$ since, on average, it is more difficult to partition (color) vertices of larger degree than those with a lower degree, resulting in sparsely populated partitions in $\mathbf{P}_h$. This has the tendency to cause the maximum clique to be detected early in the traversal of the search tree. It has been found that when $|V(G)| > 400$ in the modular product of chemical graphs, the efficiency of the algorithm can be improved by re-sorting the partition blocks according to the non-increasing numbers of vertices in each partition block following the $\mathbf{P}^{\text{ex}}$ to $\mathbf{P}^M$ vertex re-assignment procedure. This re-sorting should only be performed at depth $h = 0$ (i.e. the root node). Performing the re-sorting at search tree nodes $h > 0$ does not improve the run-time for the algorithm.

3.3.6. *Horizontal pruning*

Horizontal pruning can be defined as a procedure for reducing the number of nodes or edges in $N_G(v_i)$ given an easily calculable property of $N_G(v_i)$. The horizontal pruning techniques proposed here attempt to take maximum advantage of the separation of $\mathbf{P}_h$ into $\mathbf{P}^{\text{ex}}$ and $\mathbf{P}^M$. Since any clique larger than M, the current maximum clique, must have at least one node in $\mathbf{P}^{\text{ex}}$, only the nodes contained in $\mathbf{P}^{\text{ex}}$ need be considered in order to find any clique larger than M. Therefore, any horizontal pruning to eliminate any vertices from $\mathbf{P}_h$ need only be applied to the vertices in $\mathbf{P}^{\text{ex}}$.

3.3.7. *Kikusts pruning*

Kikusts [34] has proposed a recursive relation for the determination of the maximum independent set of a graph. This heuristic can be transformed in terms of the maximum

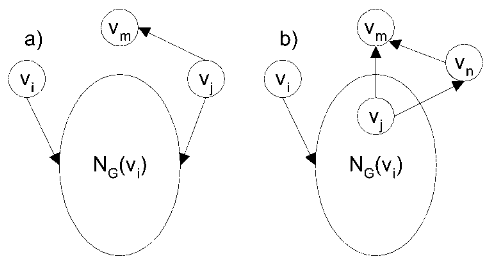


FIGURE 10. Kikusts heuristic.

clique by the following recursive relations:

$$\omega(G) = \max\{1 + \omega(N_G(v_i)), \varpi(v_i)\}$$

and

$$\varpi(v_i) = \max\{|C|, C \subseteq (G\backslash v_i) \text{ is complete and } |C \cap N_G(v_i)| \geq 2\}.$$

Assume that a node v_i is selected at a particular search tree node at depth h, and the recursive search for whether v_i is contained in a clique larger than M has backtracked to the search tree node from which v_i was selected. At this point, the current value of M represents the term $\omega(G) = 1+\omega(N_G(v_i))$ in the previous equation. It is then known that the largest clique that $N_G(v_i)$ can be involved in is $M - 1$. Therefore, if there is a vertex v_j in G that is not adjacent to v_i, then the largest clique that $v_j \cup N_G(v_i)$ can be involved in is also size M. So, in order for v_j to be involved in a clique larger than M, it must be adjacent to another vertex v_m such that v_m is not adjacent to v_i. This is illustrated in Figure 10a.

Similarly, if v_j is adjacent to v_i, then v_j is in $N_G(v_i)$. This means that the largest clique in $v_j \cup N_G(v_i)$ is $M - 1$. Therefore, in order for v_j to be involved in a clique larger than M, it must be adjacent to at least two vertices, v_m and v_n, not adjacent to v_i. This can be strengthened slightly by requiring that v_m and v_n be adjacent. This situation is represented in Figure 10b.

If any node v_j at depth h fails to satisfy either requirement illustrated in Figure 10, then it can be deleted from G because it cannot possibly be involved in a clique larger than M. This means that it may be possible to remove other vertices in addition to v_i upon a backtrack instance. Rather than implementing this test for all vertices in $G\backslash v_i$, the separation of G (i.e. $\mathbf{P}_h$) into $\mathbf{P}^{\text{ex}}$ and $\mathbf{P}^M$ allows the simplification of this procedure as this test only need be performed on the vertices contained in $\mathbf{P}^{\text{ex}}$. This procedure will be referred to as Kikusts pruning, denoted KP($v_i, \mathbf{P}^{\text{ex}}$) and is performed during each backtrack instance in an attempt to reduce the number of vertices in $\mathbf{P}^{\text{ex}}$ beyond $\mathbf{P}^{\text{ex}}\backslash v_i$.

3.3.8. *Equivalence class pruning*

One inherent difficulty associated with the computation of an MCES is degeneracy due to symmetry in a set of query graphs, as many molecules exhibit a fair degree of structural symmetry (e.g. the two molecules shown in Figure 11). This often results in a modular product graph containing large numbers of MCES or near MCES maximal cliques making clique detection more difficult as the upper-bounding operations become less effective. We have found that, by addressing symmetry in the graphs being compared prior to clique detection, the efficiency of the matching process can be significantly improved in many cases.

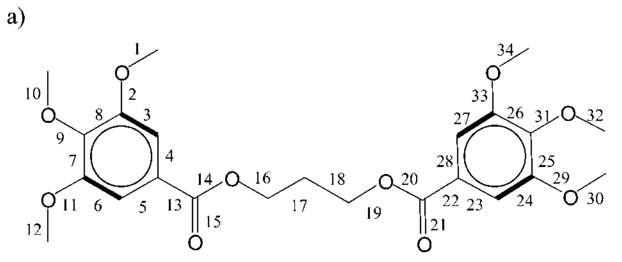
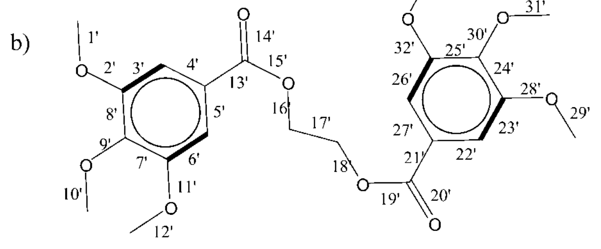



FIGURE 11. Symmetrical molecular graphs.

TABLE 1. The equivalence class listing of Figure 10.

Graph (a)		Graph (b)	
Equivalence class	Edge ID	Equivalence class	Edge ID
1	1, 12, 30, 34	1	1′, 12′, 29′, 33′
2	2, 11, 29, 33	2	2′, 11′, 28′, 32′
3	9, 31	3	9′, 30′
4	10, 32	4	10′, 31′
5	3, 6, 24, 27	5	7′, 8′, 24′, 25′
6	7, 8, 25, 26	6	3′, 6′, 23′, 26′
7	4, 5, 23, 28	7	4′, 5′, 22′, 27′
8	13, 22	8	13′, 21′
9	15, 21	9	14′, 20′
10	14, 20	10	15′, 19′
11	16, 19	11	16′, 18′
12	17, 18	12	17′

Table 1 lists the edges for the molecular graphs in Figure 11 in their respective equivalence classes. Take edge 3 in Figure 11a and edge 6′ in Figure 11b. It is clear that the edge pair (3, 6′) is equivalent to edge pairs (3, 3′), (3, 23′), (3, 26′), (6, 3′), (6, 6′), (6, 23′), (6, 26′), (24, 3′), (24, 6′), (24, 23′), (24, 26′), (27, 3′), (27, 6′), (27, 23′), (27, 26′). Any MCES rooted at any of these edge pairs cannot be larger than an MCES rooted at edge pair (3, 6′). Therefore, once the MCES rooted at edge pair (3, 6′) is discovered, it is no longer necessary to consider any of the

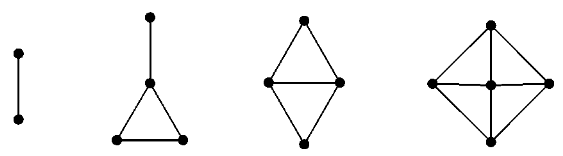


FIGURE 12. Forbidden subgraphs for line graph automorphism.

clique search tree branches rooted at these other equivalent edge pairs. Since the equivalent edge pairs correspond to equivalence classes 5 (Graph (a)) and 6 (Graph (b)) in Table 1, it is simple to determine which edge pairs in the modular product graph are equivalent once the factor graphs have been processed using an equivalence testing algorithm. Faulon [35] has shown that equivalence class testing of planar molecular graphs which comprise almost all molecular graphs can be performed in $O(n^2)$ time. We have used the algorithm of Fan *et al.* [36, 37] in our experiments, which is also of $O(n^2)$ time.

Since the MCES modular product is constructed using the line graphs of the factor graphs, it must first be determined for what cases the automorphism groups of line graphs, $L(G)$, correspond to the automorphism groups of the root graphs, G. It has been shown that this correspondence will hold provided that the molecular graphs being compared are not isomorphic to the graphs illustrated in Figure 12 [38]. Therefore, in the absence of such a forbidden isomorphism, a simple horizontal pruning heuristic based on equivalence classes can be employed to simplify clique detection. If one or both of the graphs being matched happen to be isomorphic to one of the forbidden graphs in Figure 12, it is still possible to use the equivalence class pruning heuristic provided that the equivalence class detection algorithm is modified to account for the forbidden graphs.

The function $\mathrm{P}_{\mathrm{eq}}(v_i, \mathbf{P}^{\mathrm{ex}})$ is defined as the pruning of symmetrically equivalent vertices from the modular product graph and is only performed at depth $h = 0$. Let $EQ_1^{v_i}$ and $EQ_2^{v_i}$ be the equivalence class labels of the edges in graphs G_1 and G_2, respectively, corresponding to the currently selected vertex v_i in the modular product graph. Then $\mathrm{P}_{\mathrm{eq}}(v_i, \mathbf{P}^{\mathrm{ex}})$ prunes any vertices v_j in $\mathbf{P}^{\mathrm{ex}}$ of the modular product graph where $EQ_1^{v_i} = EQ_1^{v_j}$ and $EQ_2^{v_i} = EQ_2^{v_j}$.

3.3.9. *Overall algorithm*

The proposed algorithm can be summarized in the following pseudo-code.

Algorithm RASCAL

Step 0: (Construct modular product) If the current comparison passes both cost-vector screening stages then construct the modular product graph.

Step 1: (Initialization) Set $h := 0$ and $M :=$ the clique lower bound determined using a specified *MSI*. Partition $V(G)$ into independent sets $\mathbf{P}_{h=0} := \{P_{Ka0}, P_{Ka0-1}, \ldots, P_1\}$ using the labeled projection technique. Sort $\mathbf{P}_0$ in order of non-increasing number of vertices in each partition block (i.e. $|P_{Kah}| \le |P_{Kah-1}| \le \ldots \le |P_1|$) and re-assign vertices in $\mathbf{P}^{\mathrm{ex}}$ to $\mathbf{P}^M$ if possible. Update K_{ah} if re-assignment resulted in the reduction in the number of partitions. If $\bigcup_{i=1}^{Kah} |P_i| \ge 400$, then re-sort the partitions P_i in order of non-increasing number of vertices in each partition block.

Step 2: (Upper bound) Determine $K_{\mathrm{wp}h}$. If $K_{\mathrm{wp}h} < K_{ah}$ then $\{K_h := K_{\mathrm{wp}h}\}$ else $\{K_h := K_{ah}\}$. If $\bigcup_{i=1}^{Kah} |P_i| \le 150$ and $h + K_h > M$, then use the greedy coloring algorithm to estimate the chromatic number $\chi(G)$. If $\chi(G) < K_h$ then $K_h := \chi(G)$. If $h + K_h \le M$ then {go to Step 4} else {go to Step 3}.

Step 3: (Branching) Choose a vertex $v_h \in P_{Kah} \mid P_{Kah} \in \mathbf{P}^{\mathrm{ex}}$. Set $\mathbf{P}_{h+1} := \mathbf{P}_h \cap N_G(v_h)$, $K_{ah+1} := |\mathbf{P}_{h+1}|$, and $P_{Kah} := P_{Kah} \backslash v_h$. If $P_{Kah} = \varnothing$ then decrement K_{ah}. Set $h := h + 1$. If $K_{ah} = 0$ then $\{M := \left|\bigcup_{i=0}^{h-1} v_i\right|$ and go to Step 4} else {sort $\mathbf{P}_h$ in order of non-increasing number of vertices in each partition block (i.e. $|P_{Kah}| \le |P_{Kah-1}| \le \ldots \le |P_1|$) and reassign vertices in $\mathbf{P}^{\mathrm{ex}}$ to $\mathbf{P}^M$ if possible}. Update K_{ah} if re-assignment resulted in the reduction in the number of partitions. Go to Step 2.

Step 4: (Backtracking) If $h = 0$ then end ($M =$ max clique). While $h + K_h \le M$, decrement h. If $h = 0$ then $\mathrm{P}_{\mathrm{eq}}(v_h, \mathbf{P}^{\mathrm{ex}})$. Perform $\mathrm{KP}(v_h, \mathbf{P}^{\mathrm{ex}})$. Update $K_{\mathrm{wp}h}$. Go to Step 2.

4. EXPERIMENTAL EVALUATION

Unlike other graph problems such as the maximum clique or the coloring problem, the MCIS/MCES problem does not have a standardized set of benchmark graphs with which to test the efficiency of published algorithms. Since RASCAL has been developed for chemical information handling and no standardized test graphs are available, the experimental evaluation of the algorithm has been performed using a data set of chemical graphs. While chemical graphs are sparse with respect to many of the types of graphs of interest to other research fields, it is assumed that they can be used to satisfactorily test the efficiency of the proposed algorithm. Using the definition of the modular product, it is clear that the resulting modular product will be very dense as most edge pairs in both factor graphs will not be adjacent.

The data set of compounds selected for testing purposes consisted of a collection of 200 compounds comprising steroids, melatonins, statins, anti-leukemia actives and other various compounds from miscellaneous activity classes. The average number of vertices is 25.6 with a standard deviation of 6.7. In order to investigate the efficiency and robustness of the proposed algorithm, all possible pairwise MCES comparisons between two molecules in a data set were examined. For this data set, 19,900 pairwise comparisons were performed.

For comparative purposes, three separate algorithms were coded and tested against the RASCAL algorithm. Two of these algorithms are advanced maximum clique detection algorithms for arbitrary graphs. These are the

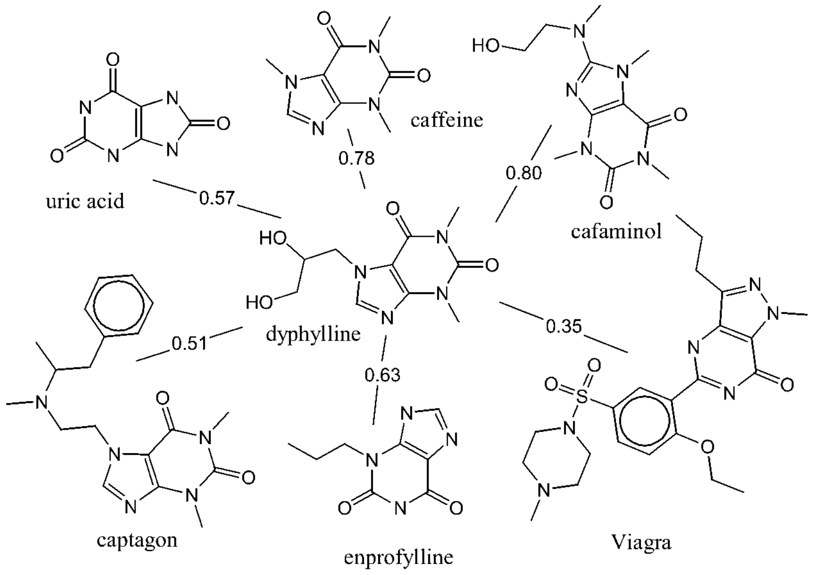


FIGURE 13. An example of chemical graph similarity.

TABLE 2. 'Screen-out' performance (%).

	MSI							
	0.5	0.55	0.6	0.65	0.7	0.75	0.8	0.85
First tier	24.7	34.3	46.2	59.8	72.9	85.0	93.4	97.6
Second tier	59.0	54.9	47.3	36.8	25.4	14.3	6.2	2.1
Total	83.7	89.2	93.5	96.6	98.3	99.3	99.6	99.7

MC1 algorithm of Wood [31] employing a greedy coloring upper bound and the Ostergard [39] algorithm. The other algorithm is due to Bessonov [32] which was developed specifically for maximum clique detection in a modular product graph and uses the simple edge projection technique as an upper bounding procedure. All three of these algorithms employ different vertex selection schemes for traversing the search tree. Of the three, the vertex selection procedure used by Wood is most like the one employed in the RASCAL algorithm. For the sake of uniformity, all algorithms were coded in Visual C++ 6.0 using the same data structures where appropriate and they were executed on a 400 MHz Intel Celeron processor with 128 MB RAM using Windows 98. To further ensure uniformity, all four algorithms were run in conjunction with the cost-vector screening procedure previously presented as well as the minimum similarity index (*MSI*) lower bound for the size of the maximum clique, with the exception of the Ostergard algorithm which was run using only the cost-vector screening. The unique branching procedure of the Ostergard algorithm precludes the use of a lower-bounding procedure for the size of the maximum clique.

The specification of an acceptable *MSI* value will depend in a large part on the application as well as the connectivity of the graphs being considered. It will also be dependent upon the number of vertices and density of the graphs. Figure 13 demonstrates a simple example depicting various similarity measures calculated for a set of small molecules. In this figure, dyphylline is compared to a set of other small xanthine compounds. For the efficiency evaluation, trial simulations were run at *MSI* values ranging from 0.5 to 0.85, and modular product simplification heuristics [40] were used to simplify the modular product based on chemical knowledge prior to clique detection whenever possible.

To evaluate the performance of the cost-vector screening procedure, the 'screen-out' was calculated at the same *MSI* values. The 'screen-out' is defined as the percentage of comparisons correctly excluded from further consideration relative to the total number of comparisons whose MCES-based similarity measure is calculated as being below the specified *MSI* threshold. Table 2 lists the results of the 'screen-out' simulation. It can be seen that the proposed screening method is effective in screening unnecessary comparisons and that the first tier cost-vector approach becomes increasingly effective as the value of *MSI* increases; however, the second tier cost-vector screen becomes more effective relative to the first tier screen as the *MSI* value decreases.

Table 3 presents the total time for each simulation trial; a time limit of 24 hours was used for each trial. It is clear

TABLE 3. Algorithm time comparison results (seconds).

	MSI							
Algorithm	0.5	0.55	0.6	0.65	0.7	0.75	0.8	0.85
Bessonov	>24 h	>24 h	>24 h	>24 h	>24 h	>24 h	>24 h	93
Ostergard	>24 h	>24 h	>24 h	>24 h	>24 h	>24 h	>24 h	>24 h
Wood	>24 h	>24 h	>24 h	>24 h	>24 h	>24 h	12,898	13
RASCAL	304	188	120	80	43	22	11	7

TABLE 4. RASCAL time trial results (total (per comparison) in seconds (milliseconds)).

					MSI			
Data set	No. of molecules	No. of comparisons	Avg. $\|V(G)\|$	Std. Dev. $\|V(G)\|$	0.7	0.75	0.8	0.85
ASX	5386	14,501,805	32.5	5.0	71,806 (4.95)	37,229 (2.57)	17,570 (1.21)	7058 (0.487)
CNS	16,000	127,992,000	23.3	4.7	55,340 (0.432)	22,216 (0.174)	8462 (0.0661)	3269 (0.0255)
NAN	2715	3,684,255	26.3	5.5	1883 (0.511)	826 (0.224)	349 (0.0947)	150 (0.0407)

TABLE 5. Proportion of MCES similarity comparisons exceeding the MSI threshold.

	MSI			
Data set	0.7	0.75	0.8	0.85
ASX	1,698,872/11.7%	892,315/6.15%	422,383/2.91%	150,385/1.04%
CNS	1,425,165/1.11%	485,830/0.38%	167,584/0.13%	50,920/0.040%
NAN	48,279/1.31%	22,224/0.60%	9749/0.26%	3522/0.096%

from Table 3 that the RASCAL algorithm outperformed the other published algorithms. This is especially so at the lower *MSI* thresholds, which permit the identification of more distant structural relationships between molecules and which are of most potential interest in the discovery of novel bioactive molecules. While all of the heuristics presented in the RASCAL algorithm provide efficiency improvements, the majority of the decrease in run-time is due to the combination of the labeled edge projection upper bound in association with the partition re-ordering procedure. The equivalence testing heuristic was also found to have a significant impact on run-time for molecular graphs with a high degree of symmetry.

To further test the efficiency and durability of the RASCAL algorithm, it was tested on three separate data sets of molecular structures. These consisted of a set of 5386 chemical structures supplied by Asinex (http://www.asinex.com), another 2715 chemical structures acquired from Nanosyn (http://www.nanosyn.com) and a set of 16,000 central nervous system targeted molecules obtained from ChemBridge (http://www.chembridge.com). These data sets are denoted ASX, NAN and CNS, respectively. All possible pairwise comparisons for each data set were evaluated using *MSI* values of 0.7, 0.75, 0.8 and 0.85. The minimum *MSI* of 0.7 was selected because *MSI* values below 0.7 failed to discriminate between the molecular structures in a chemically sensible manner to the degree desired. Table 4 presents the results of these trial simulations. These results indicate that rates in the range of thousands of comparisons per second using the specified processor can be achieved, even at the lower *MSI* threshold values, making it feasible to carry out MCES-based searching of large chemical databases in reasonable amounts of time, something that has not previously been possible in systems for chemical information management [41].

Table 4 shows that the per comparison time results for the CNS and NAN are comparable. However, the time results for the ASX data set differ significantly. Although the average size of a molecular graph for the ASX data set is larger than for both the CNS and NAN data sets, this difference does not fully explain the discrepancy in the performance of the algorithm. The difference in performance is chiefly attributable to the relative similarity of the graphs in each data set. A simple measure of diversity was calculated by determining the percentage of pairwise comparisons whose MCES-based similarity exceeds the specified *MSI* relative to the total number of pairwise comparisons possible in a data set. Each cell in Table 5 lists the number of comparisons as well as the percentage of comparisons exceeding the stated *MSI* value. It is evident from Table 5 that the proportion of comparisons whose similarity exceeds the specified *MSI* value is larger for the ASX dataset than for the CNS and NAN data sets, indicating that the ASX data set requires proportionately more calls to the computationally expensive graph matching routine. This difference corresponds closely with the noted difference in per comparison time results between the data sets.

5. CONCLUSION

An efficient graph similarity procedure RASCAL has been proposed based on the concept of the maximum common edge subgraph. The proposed algorithm consists of several heuristics discovered in the literature as well as several new heuristics such as the first and second tier screening procedure, the suggested vertex selection procedure, a sharper upper bound based on graph projections and the equivalence class pruning method to account for symmetry in the graphs being compared. It is relatively straightforward to implement and has proven to be an efficient tool for similarity calculations in chemical graph databases, thus providing an alternative to existing measures of chemical similarity based on bit-strings. While RASCAL has been designed for use in chemical information management, the algorithm is conceptually general and is directly applicable to any graph-based similarity application.

It may be possible to significantly improve the performance of the algorithm by incorporating other more efficient upper-bounding techniques in addition to the approaches suggested here. This possibility remains for future work on the algorithm, as does its extension to the calculation of similarity for 3-D chemical graphs and its application to chemical problems such as searching and the prediction of biological activity.

ACKNOWLEDGEMENTS

The authors would like to extend their gratitude to Pfizer (Ann Arbor) for funding of this project and thank Alain Calvet, George Cowan, Eric Gifford, Christine Humblet and David Wild of Pfizer and Mark Johnson of Pananugget Consulting for their support. The authors would also like to thank the reviewers for their helpful comments and suggestions for improving the manuscript. The Krebs Institute for Biomolecular Research is a designated centre of the Biotechnology and Biological Science Research Grant.

REFERENCES

[1] Kuhl, F., Crippen, G. and Friesen, D. (1984) A combinatorial algorithm for calculating ligand binding. *J. Comp. Chem.*, **5**, 24–34.

[2] Shearer, K., Bunke, H. and Venkatesh, S. (2001) Video indexing and similarity retrieval by largest common subgraph detection using decision trees. *Pattern Recog.*, **34**, 1075–1091.

[3] Horaud, R. and Skordas, T. (1989) Stereo correspondence through feature grouping and Maximal Cliques. *IEEE Trans. Pattern Anal. Mach. Intell.*, **11**, 1168–1180.

[4] Pelillo, M., Siddiqi, K. and Zucker, S. W. (1999) Matching hierarchical structures using association graphs. *IEEE Trans. Pattern Anal. Mach. Intell.*, **21**, 1105–1120.

[5] Trinajstic, N. (1992) *Chemical Graph Theory*. CRC Press, Boca Raton, FL.

[6] Willett, P. (1999) Matching of chemical and biological structures using subgraph and maximal common subgraph isomorphism algorithms. *IMA Vol. Math. Appl.*, **108**, 11–38.

[7] Gifford, E., Johnson, M., Smith, D. and Tsai, C. (1996) Structure-reactivity maps as a tool for visualizing xenobiotic structure-reactivity. *Network Science*, **2**, 1–33.

[8] Lewis, R. A., Mason, J. S. and McLay, I. M. (1997) Similarity measures for rational set selection and analysis of combinatorial libraries: the diverse property-derived (DPD) approach. *J. Chem. Inf. Comput. Sci.*, **37**, 599–614.

[9] Chen, L. and Robien, W. (1994) Application of the maximal common substructure algorithm to automatic interpretation of 13C NMR spectra. *J. Chem. Inf. Comput. Sci.*, **34**, 934–941.

[10] Gillet, V. J., Wild, D. J., Willett, P. and Bradshaw, J. (1998) Similarity and dissimilarity methods for processing chemical structure databases. *Comp. J.*, **41**, 547–558.

[11] Flower, D. R. (1998) On the properties of bit string-based measures of chemical similarity. *J. Chem. Inf. Comput. Sci.*, **38**, 379–386.

[12] Diestel, R. (2000) *Graph Theory*. Springer, New York.

[13] Sanfeliu, A. and Fu, K. S. (1983) A distance measure between attributed relational graphs for pattern recognition. *IEEE Trans. Syst. Man, Cybern.*, **13**, 353–362.

[14] Hansen, P. (1979) Upper bounds for the stability number of a graph. *Rev. Roumaine Math. Pures Appl.*, **24**, 1195–1199.

[15] Liu, R. Y. (1989) An upper bound on the chromatic number of a graph. *J. Xinjiang Univ. Natur. Sci.*, **6**, 24–27.

[16] Papadopoulos, A. N. and Manolopoulos, Y. (1999) Structure-based similarity search with graph histograms. In *Proc. 10th Int. Workshop on Database and Expert Systems Applications*, Florence, Italy, September 1–3, 1999, pp. 174–178. Springer, Berlin.

[17] Johnson, M. (1985) Relating metrics, lines and variables defined on graphs to problems in medicinal chemistry. In Alavi, Y., Chartrand, G., Lesniak, L., Lick, D. and Wall, C. (eds), *Graph Theory and Its Applications to Algorithms and Computer Science*, pp. 457–470. Wiley, New York.

[18] Carpaneto, G., Martello, S. and Toth, P. (1988) Algorithms and codes for the assignment problem. *Ann. Oper. Res.*, **13**, 193–223.

[19] Brown, R. D. and Martin, Y. C. (1996) Use of structure-activity data to compare structure-based clustering methods and descriptors for use in compound selection. *J. Chem. Inf. Comput. Sci.*, **36**, 572–584.

[20] Patterson, D. E., Cramer, R. D., Ferguson, A. M., Clark, R. D. and Weinberger, L. E. (1996) Neighborhood behavior: a useful concept for validation of 'molecular diversity' descriptors. *J. Med. Chem.*, **39**, 3049–3059.

[21] Levi, G. (1972) A note on the derivation of maximal common subgraphs of two directed or undirected graphs. *Calcolo*, **9**, 341–352.

[22] Barrow, H. and Burstall, R. (1976) Subgraph isomorphism, matching relational structures and maximal cliques. *Inf. Proc. Lett.*, **4**, 83–84.

[23] Vizing, V. G. (1974) Reduction of the problem of isomorphism and isomorphic entrance to the task of finding the nondensity of a graph. In *Proc. Third All-Union Conference on Problems of Theoretical Cybernetics*, Novosibirsk, p. 124 (in Russian).

[24] Whitney, H. (1932) Congruent graphs and the connectivity of graphs. *Amer. J. Math.*, **54**, 150–168.

[25] Nicholson, V., Tsai, C., Johnson, M. and Naim, M. (1987) A subgraph isomorphism theorem for molecular graphs. In King, R. B. and Rouvray, D. H. (eds), *Graph Theory and Topology in Chemistry*. Elsevier, Athens, GA.

[26] Kvasnicka, V. and Pospichal, J. (1990) Maximal common subgraphs of molecular graphs. *Reports Molecular Theory*, **1**, 99–106.

[27] Pardalos, P. and Xue, J. (1994) The maximum clique problem. *J. Global Optimiz.*, **4**, 301–328.

[28] Garey, M. R. and Johnson, D. S. (1979) *Computers and Intractability*. W. H. Freeman, San Francisco, CA.

[29] Grimmett, G. R. and McDiarmid, C. J. H. (1975) On colouring random graphs. *Math. Proc. Camb. Phil. Soc.*, **77**, 313–324.

[30] Manvel, B. (1981) Coloring large graphs. *Congr. Numer.*, **33**, 197–204.

[31] Wood, D. (1997) An algorithm for finding a maximum clique in a graph. *Oper. Res. Lett.*, **21**, 211–217.

[32] Bessonov, Y. E. (1985) On the solution of a problem on the search for the best intersection of graphs on the basis of an analysis of the projections of the subgraphs of the modular product. *Vychisl. Sistemy*, **121**, 3–22 (in Russian).

[33] Selkow, S. (1978) New bounds for the clique number of a graph. *Inf. Proc. Lett.*, **7**, 173–174.

[34] Kikusts, P. (1986) Another algorithm determining the independence number of a graph. *Elektron. Inf. Verarb. Kybern.*, **22**, 157–166.

[35] Faulon, J. L. (1998) Isomorphism, automorphism partitioning, and canonical labeling can be solved in polynomial-time for molecular graphs. *J. Chem. Inf. Comput. Sci.*, **38**, 432–444.

[36] Fan, B. T., Barbu, A., Panaye, A. and Doucet, J. P. (1996) Detection of constitutionally equivalent sites from a connection table. *J. Chem. Inf. Comput. Sci.*, **36**, 654–659.

[37] Fan, B. T., Panaye, A. and Doucet, J. P. (1999) Comment on 'Isomorphism, automorphism partitioning, and canonical labeling can be solved in polynomial-time for molecular graphs'. *J. Chem. Inf. Comput. Sci.*, **39**, 630–631.

[38] Teh, H. H. and Chen, C. C. (1970) On the automorphism groups of interchanged graphs. *Nanta Math.*, **4**, 72–78.

[39] Ostergard, P. (2002) A fast algorithm for the maximum clique problem. *Discrete Appl. Math.*, **120**, 195–205.

[40] Raymond, J., Gardiner, E. and Willett, P. (2002) Heuristics for rapid similarity searching of chemical graphs using a maximum common edge subgraph algorithm. *J. Chem. Inf. Comput. Sci.*, **42**, 305–316.

[41] Willett, P., Barnard, J. and Downs, G. (1998) Chemical similarity searching. *J. Chem. Inf. Comput. Sci.*, **38**, 983–996.