Algorithm for Naming Molecular Equivalence Classes Represented by Labeled Pseudographs

Yong-jin Xu[†] and Mark Johnson*

Pharmacia Corporation, 301 Henrietta Street, Kalamazoo, Michigan 49007

Received June 2, 2000

The emergence of large chemical databases imposes a need for organizing the compounds in these databases. Mapping the chemical graph in particular, and a molecular equivalence class represented by a labeled pseudograph[1] in general, to a unique number or string facilitates high-throughput browsing, grouping, and searching of the chemical database. Computing this number using a naming adaptation of the Morgan algorithm,[2] we observed a large classification noise in which nonisomorphic graphs were mapped to the same number. Our extensions to that algorithm greatly reduced the classification noise.

INTRODUCTION

The need for organizing large compound collections is growing in drug discovery research. We frequently need to study the behavior of the compounds within various molecular equivalence classes or compare the overall behavior among compounds from different molecular equivalence classes. A molecular equivalence class is an exhaustive subgroup of compounds that share a recognizable structural feature in a compound collection. For example, all compounds with two simple rings connected by one or more bridge bonds fall into a particular molecular equivalence class among the family of classes associated with the 2D topological arrangement of rings. Similarly, all compounds with two functional groups of −OH and −SH fall into another equivalence class among the family of classes associated with functional groups. These two classes can be represented pictorially by their unique structural features, as shown in Figure 1. Mathematically, the left-hand representation in Figure 1 can be thought of as a labeled pseudograph in which two loops are connected by an edge. Similarly, the right-hand representation is a labeled pseudograph with two components. In effect, a molecular equivalence function maps molecules to their molecular equivalence classes by mapping the molecules to the space of labeled pseudographs. Different molecular equivalence functions[3,4] partition the chemical space into different sets of molecular classes.

Compounds are assigned to a molecular equivalence class by assigning them the name for that class. We call this name the molecular equivalence number (meqnum). Since a molecular equivalence class can be uniquely represented by a labeled pseudograph as shown in Figure 1, naming molecular equivalence classes is equivalent to naming labeled pseudographs.

A number of methods have already been developed to name the chemical graph for different purposes. These methods include encounter indexing,[4] linear connection table codes,[5] fingerprints,[6] Randić's molecular identification,[7] and naming adaptations of the Morgan algorithm.[8−11] From here on, we shall refer to these adaptations as Morgan naming algorithms, a term which includes the algorithm that we will present. Since none of these methods was aimed at labeled pseudographs, we must find out which of these to modify to match our needs. Because most of the interesting equivalence classes include all of the stereochemical isomers constructable from a chemical graph, an algorithm that distinguishes stereoisomers is usually not necessary, although it would be nice.

Figure 1. Two equivalence classes.

We will take a look at what is required. First of all, the algorithm must generalize to labeled pseudographs. Second, we require short memorizable and voicable codes for human recognition, memory, communication, and substring querying. Short names are also required from considerations of computer memory and speed. In addition, a short name is critical because we need to concatenate these names to form more sophisticated classes. Third, a priori codes are required for comparing analysis results computed on different collections involving the same set of equivalence classes. Fourth, the algorithm must distinguish the vast majority of molecular equivalence classes. Although perfect resolution is related to a NP complete problem[12] and therefore gets very slow when graph size increases, frequent classification noise can be very disruptive and misleading. Therefore, negligible classification noise, which must be handled case by case, as are outliers in data analysis, is required.

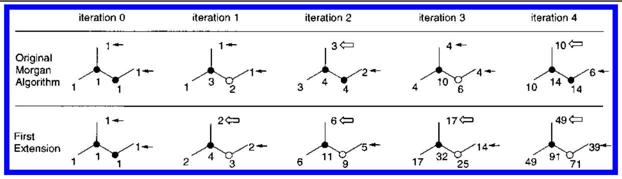
Among the available algorithms, linear connection table codes, as NP complete solutions, were felt to be too computationally intensive for our purpose. In addition, like fingerprints, they tend to be very long and humanly uncommunicable. Encounter indexing is not a priori. This leaves the Randić's molecular ID algorithm and the Morgan naming algorithms as bases on which to build an algorithm suited to our needs even though each of these algorithms has been shown to have significant classification noise.[7,8,10]

* To whom correspondence should be addressed. Telephone: (616) 833-7830. E-mail: marknmartha@mindspring.com.

[†] Telephone: (616) 833−4129. E-mail: yongjinxu@hotmail.com.

10.1021/ci0003911 CCC: $20.00 © 2001 American Chemical Society
Published on Web 01/22/2001

Table 1. Example of Oscillating Behavior of Morgan Algorithm[a]



[a] One set of highlighted vertices (marked as circles) gives the same value on even iterations and different values on odd iterations, while the other set of highlighted vertices (marked as arrows) gives the same values on odd iterations and different values on even iterations.

The Morgan naming algorithms and the molecular ID algorithm associate invariants[11] with the vertexes and edges, respectively, and then combine these invariants into a graph name. A more recent study of vertex-partitioning algorithms[13] addresses a number of approaches for improving the recognition of topological symmetry. Although a fuller partitioning of the vertexes of a graph plays an important role in the resulting naming algorithms, the tradeoff between better partitioning and computational complexity is far from clear. For example, even when this partitioning fails to fully resolve the topological symmetry, the constructed name might still be unique to that graph. And we shall give a common case in which a simple Morgan naming algorithm fails to distinguish two graphs by their names even though the underlying Morgan algorithm correctly perceives the topological symmetry in both graphs. Consequently, we chose to construct a naming algorithm suitable for labeled pseudographs starting with the original Morgan algorithm because of its excellent tradeoff between computational simplicity and resolution.

We begin with a straightforward extension of the Morgan algorithm to all labeled pseudographs. After illustrating two critical cases in which the algorithm based on this extension resulted in too much classification noise, we will present a modification that resolved these two critical cases. We close with a, hopefully convincing, demonstration that the classification noise of our final Morgan naming algorithm is negligible.

EXTENDING THE MORGAN ALGORITHM TO LABELED PSEUDOGRAPHS

As chemical graphs are labeled graphs, a naming algorithm based on the original "vertex-defined" Morgan algorithm works as follows:

1. Each vertex *i* is assigned an initial value υ_i corresponding to the atom type of the vertex.

2. For each vertex *i*, compute $\upsilon_i' = \Sigma_j \upsilon_j$, where *j* is a vertex connected to *i*.

3. Assign υ_i' to υ_i.

4. Repeat steps 2 and 3 until there is not a change in the grouping of vertexes by their associated values.

5. Take the summation or product of all vertexes' values as the ID number for the graph.

Step 5 is added as a functional extension for naming. Similar naming approaches have been reported by a number of authors.[8] It is widely recognized[9] that the algorithm ignores distinctions in the number of edges that might join two vertexes and the bond types associated with these edges. As shown in Table 1, it is also known to exhibit oscillating behavior[10] on υ_i in some graphs, which results in failures to resolve distinguishable vertexes. These will reduce the resolution of the codes constructed in step 5.

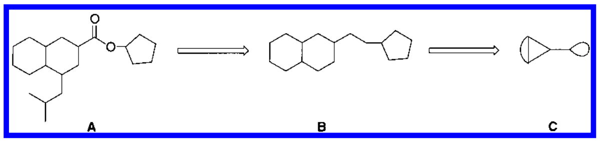


Figure 2. Examples of cyclic skeleton and reduced cyclic skeleton. Graph A is the original molecule. Graph B is the cyclic skeleton of A, and C is the reduced-cyclic skeleton with second-degree vertexes reduced.

These inadequacies are overcome in the following "edge-defined" extension of the Morgan naming algorithm:

1. Each vertex *i* is assigned an initial value υ_i according to the atom type and each edge *i* assigned e_i according to bond type.

2. For every vertex *i*, the following is computed: $\upsilon_i' = \upsilon_i + \Sigma_{j(i)}(\upsilon_{j(i)} + e_{j(i)})$, where the sum is over those edges incident with vertex *i*. $\upsilon_{j(i)}$ is υ_i if *j(i)* is a loop edge and is the value of the vertex other than vertex i otherwise.

3. Assign υ_i' to υ_i.

4. Repeat steps 2 and 3 *n* times.

5. $m = \Pi_i \upsilon_i$, where *m* is the overall product used as an ID number.

Note that if there are no loops or multiple edges and if e_i is identically 0, this reduces to the "vertex-defined" Morgan algorithm.

CLASSIFICATION NOISE IN THE EDGE-DEFINED MORGAN NAMING ALGORITHM

When generating high-throughput browsing indexes, we transformed the chemical graph into various types of subgraphs and reduced graphs. The cyclic skeleton is obtained by ignoring atom and bond types and then removing all side chains.

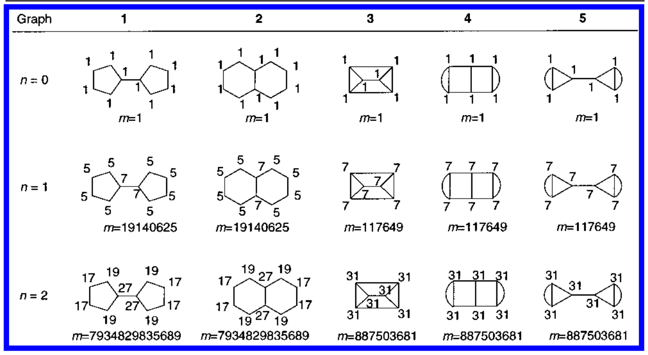
We say the second-degree vertex X2, connecting vertexes X1 and X3, is "reduced" if the edge pattern X1−X2−X3 is replaced with the edge X1−X3. Recursively reducing all second-degree vertexes[14] of the cyclic skeleton generates the reduced-cyclic skeleton. Figure 2 shows an example of the cyclic skeleton and reduced-cyclic skeleton. In this paper, we define the ring edge as the edge that is within a ring and the side chain edge as the edge that can be removed by recursively deleting the single degree terminal vertexes. We also define a bridge edge as an edge on a path connecting two ring systems. For example, graph A in Figure 2 contains 3 bridge edges, 16 ring edges, and 5 side chain edges in which the carbonyl edge is considered a side chain edge.

The preceding edge-defined Morgan naming algorithm works operationally and covers the labeled pseudographs that can arise via this reducing transformation. However, it does not discriminate some graphs that are common outcomes of the cyclic skeleton and reduced-cyclic skeleton equivalence functions. Two common examples encountered in our study are shown in Table 2 in which each group of graphs/pseudographs gives the same equivalence number.

ADDITIONAL EXTENSIONS TO THE MORGAN NAMING ALGORITHM

From the first three iterations shown in Table 2, it is easy to see that graphs 1 and 2 will give the same value for *m* no

Table 2. Examples That Our First Extension of the Morgan Naming Algorithm Can Not Distinguish[a]



Graph	1	2	3	4	5
$n = 0$	m=1	m=1	m=1	m=1	m=1
$n = 1$	m=19140625	m=19140625	m=117649	m=117649	m=117649
$n = 2$	m=7934829835689	m=7934829835689	m=887503681	m=887503681	m=887503681

[a] n is the number of iterations. Each vertex and edge is assigned an initial value 1.

matter how large n might be. The same thing happens to graphs 3−5. Interestingly, the underlying extension of the Morgan algorithm correctly perceives the topological symmetries in graphs 1 and 2, but the constructed names are still not unique!

Observing the difference between 1 and 2, we find that graph 1 has two ring systems connected by one bridge, while graph 2 is a fused ring. Therefore, they differ in the number of bridge edges. By counting the number of bridge edges, we resolve graphs 1 and 2. The number of bridge edges similarly resolves graphs 4 and 5.

The obvious difference between graph 3 and graph 4 or 5 is the number of multiple edges. Graph 3 has none, and graphs 4 and 5 have 4 each. Therefore, by halving, for example, the value originally assigned to each multiedge and loop, we distinguish graph 3 from graph 4 or 5.

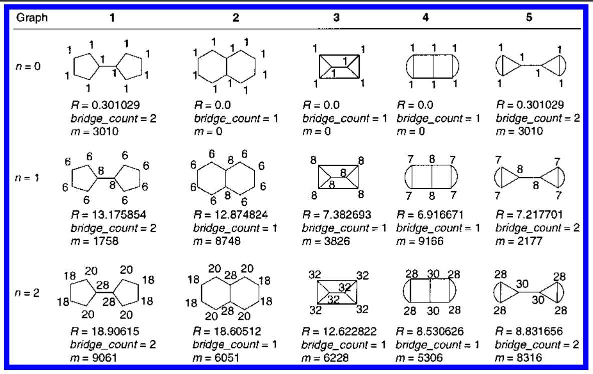
The number generated from the last step of the edge-defined Morgan naming algorithm is the product of all vertex values. We can imagine that there exist situations such that two different sets of vertex values give the same product. For example: $18 \times 7 \times 19 = 14 \times 171$. This duplication is resolved by hashing the value of each vertex to a prime number before the multiplication. By doing so, two different sets of integers are mapped to two different sets of prime numbers and then two different products according to the unique prime-factorization theorem.

The other problem we can see from Table 2 is that the value of m gets large very quickly with n. Even in a small graph such as 1 and $n = 2$, m is already larger than a 32 bit integer can accommodate. Commonly we set $n = 5$ in order to get a reasonable resolution. Consequently, we take the logarithm of the value for each vertex. The sum of these logarithms is the logarithm of the product. The value of our index function is defined as the first *size_of_output* digits of the mantissa of the logarithm sum, where *size_of_output* is the user-desired maximum number of digits for the index.

COMPUTATIONAL IMPLEMENTATION

Our extended Morgan naming algorithm is given by the following procedures, where *max_v_type*, n, and *size_of_output* are user defined parameters specifying the maximum number of different vertex types, the number of iterations, and the maximum number of output digits, respectively.

Table 3. Extended Morgan Naming Algorithm Enhances the Discrimination Power of Morgan Naming Algorithm[a]



Graph	1	2	3	4	5
$n = 0$	R = 0.301029 *bridge_count* = 2 m = 3010	R = 0.0 *bridge_count* = 1 m = 0	R = 0.0 *bridge_count* = 1 m = 0	R = 0.0 *bridge_count* = 1 m = 0	R = 0.301029 *bridge_count* = 2 m = 3010
$n = 1$	R = 13.175854 *bridge_count* = 2 m = 1758	R = 12.874824 *bridge_count* = 1 m = 8748	R = 7.382693 *bridge_count* = 1 m = 3826	R = 6.916671 *bridge_count* = 1 m = 9166	R = 7.217701 *bridge_count* = 2 m = 2177
$n = 2$	R = 18.90615 *bridge_count* = 2 m = 9061	R = 18.60512 *bridge_count* = 1 m = 6051	R = 12.622822 *bridge_count* = 1 m = 6228	R = 8.530626 *bridge_count* = 1 m = 5306	R = 8.831656 *bridge_count* = 2 m = 8316

[a] *Max_v_type* = 5000; *size_of_output* = 4.

1. First *max_v_type* prime numbers are computed, starting from 1, and they are stored in an array P. The array P is computed only once for a data set.

2. Each vertex i is assigned an initial value v_i based on the atom type, and each edge i assigned e_i based on the bond type.

3. For each loop and multiple edge, $e_i = e_i/2$.

4. The number of bridge edges is counted as *bridge_count*, *bridge_count* = 1 + (*bridge_count* mod *max_v_type*).

5.For every vertex i, the following is computed: $v_i' = v_i + \Sigma_{j(i)}(v_{j(i)} + e_{j(i)})$, where the sum is over those edges incident with vertex i. $v_{j(i)}$ is v_i if $j(i)$ is a loop edge and is the value of the vertex other than vertex i otherwise.

6. $v_i = (v_i'$ mod *max_v_type*$) + 1$.

7. Repeat steps 5 and 6 n times.

8. The index value is the first *size_of_output* digits of the mantissa of $R = \log(P[bridge\_count]) + \Sigma_i \log(P[v_i])$.

We see from Table 3 that this extended Morgan naming algorithm distinguishes graphs 1−5.

Our program for this extended[15] Morgan naming algorithm is called MEQNUM. It was implemented with C++ language on PC/Window NT and SGI/UNIX. The program contains about 450 lines. The input of the program is a SD file, which has been preprocessed with bond types (side chain, ring, and bridge) recognized. MEQNUM takes about 1 s for every 1000 compounds on a P-II 400 PC/Window NT.

The null graph is a graph with neither edges nor vertexes. With the Nilakantan equivalence function,[4] when it maps an acyclic compound to its component ring system, of which there are none, it gives the null graph. MEQNUM assigns a 0 to the null graph. However, it is likely that some other graphs may also give a meqnum 0. Because there are more graphs than numbers allowed by the algorithm, according to pigeonhole principle, some graphs will have to be mapped to the same number. This causes classification noise. We now show that the noise can be made negligible.

COMBINATORY DEMONSTRATION BASED ON THE CHOICE OF PARAMETER VALUES

The three user-defined parameters, n, *max_v_type* and *size_of_output*, play very important roles in minimizing classification noise. The number of iteration parameters n means that the vertexes n bonds away from vertex i influence the eventual value of v_i at step 8 in the algorithm. Therefore, if *n* is too small, vertexes will lose their uniqueness and thereby increase overall classification error. However, a big *n* is not practical and quickly redundant once most of the nonsymmetric vertexes are distinguished. The optimal *n* is the point where the number of distinguished vertexes stop increasing, which is usually no more than 5. In our study, we chose $n = 5$.

The parameter *max_v_type* must be large enough to accommodate a large number of possible vertex types. Usually, the number of different vertex types is greater than the total number of molecules. In the worst case, it is the total number of atoms in all molecules being studied. For example, in a hypothetical database of 250 000 compounds with 20 non-hydrogen atoms on average in which there are no atomic symmetries, there would be roughly 5 million distinct vertex types. Generating that many primes for the first step in the algorithm wastes time and memory. In fact, we are only interested in getting a unique number for each molecule, which is the product of the vertex values. Therefore, rather than requiring a unique value for each different vertex, we require a unique combination of vertex values for each graph. According to this reasoning, we need to choose a suitable *max_v_type* value so that there are enough combinations of vertex values.

To get an estimate of how many primes will suffice, let *x* be the average number of atoms in a molecule. Then the total number of vertex value combinations *C* is

$$\mathrm{C} = \frac{(max\_v\_type + x - 1)!}{x!(max\_v\_type - 1)!}$$

This is the number of ways that *max_v_type* balls (prime factors) can be placed in *x* cells (atoms).[16] Generally, we set *max_v_type* to 5000. Taking $x = 20$, the total number of combinations is about 4.1×10^{55}. When molecule size increases, for example *x* increases to 30; the total number of combinations also increases, in this case to 3.8×10^{78}. Thus, although many distinct vertex types are assigned the same prime, rarely are distinct graphs assigned the same index as a consequence.

The value of the parameter *size_of_output* defines the size of the output. In fact, it also defines the maximum number of different graphs that can be represented. For example, if *size_of_output* is 2, we will only be able to represent 100 different graphs because there are only 100 different numbers with 2 digits. We found that 7 digits were enough to distinguish the 240 000 compounds in our database. To enhance the sensitivity without increasing the number of digits, we added the option of expressing this number in base 35 using the characters {0, 1, 2, 3, ..., 9, A, B, C, ..., N, P, ..., Z} creating a “license-plate” representation of the meqnum.[17]

EMPIRICAL DEMONSTRATION OF UNIQUENESS

MEQNUM was used to compute the following structural browsing indexes in order to validate the program and verify the negligible classification noise of the new algorithm:

1. Stereoisomer index: The equivalence function here is the identity function that maps the chemical graph to itself. Only stereoisomers give identical meqnums.

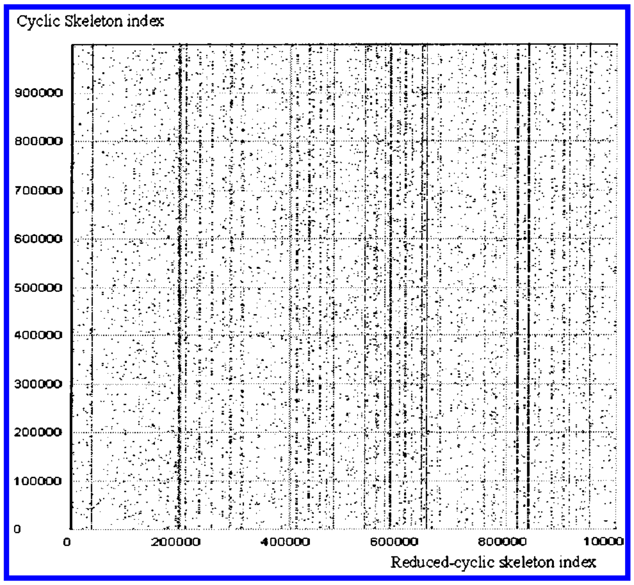


Figure 3. Plot of cyclic skeleton index vs reduced-cyclic skeleton index. Some reduced-cyclic skeletons, corresponding to the dense vertical lines, are better expressed in this database than others.

2. Cyclic skeleton index: The equivalence function here is the one in Figure 2 mapping chemical graph A to B.

3. Reduced-cyclic skeleton index: The equivalence function here is the one in Figure 2 mapping chemical graph A to C.

These indexes were computed for all compounds in our database. A classification error for the stereoisomer index occurs when two compounds which are not stereoisomers are assigned the same meqnum. Of the compounds in our database, only 28 339 compounds shared a meqnum with at least one other compound. To show that compounds having the same value for the stereoisomer index were stereoisomers or racemates, etc., we computed their MACCS fingerprints.[6] In each case, they agreed in their MACSS fingerprints as well. Of course, agreement in the MACSS fingerprint does not imply that two compounds are stereoisomers, but it does imply they are very likely to have identical or very similar chemical graphs. This implication was examined for the 136 randomly selected classes. The compounds within each of these classes (973 compounds checked in total) proved to be the same compound or stereoisomers.

Although interesting, the preceding test is not stringent enough for our needs. Even the edge-extended Morgan naming algorithm does fine against chemical graphs because of the structural diversity associated with atom and bond typing. Classification noise is encountered principally when such distinctions are ignored as is the case for the cyclic skeleton and reduced cyclic skeleton indexes. We further verified the algorithm by plotting[18] the reduced-cyclic skeleton index versus the cyclic skeleton index (Figure 3). We picked the 118 reduced-cyclic skeletons at random, each of which contained multiple cyclic skeletons, i.e., had multiple *Y* values. From each set of multiple cyclic skeletons sharing a common reduced-cyclic skeleton meqnum, we randomly picked 20 cyclic skeletons (all if less than 20) and checked if they belonged to the same reduced-cyclic skeleton. All of these cyclic skeletons appropriately belonged to their reduced-cyclic skeleton class.

These systematic checks are useful, but limited in diversity. The examples in which the edge-extended Morgan naming algorithm failed, five of which are given in Table 3, were found during our use of structural browsing indexes in a diverse range of contexts, not in such systematic checks. Our confidence in MEQNUM rests primarily on our having yet to find, during the diverse needs of practical use, a case where MEQNUM mapped two nonisomorphic labeled pseudographs to the same name.

From the vertical-pattern spots in Figure 3, it is easy to see that some reduced-cyclic skeletons are well-developed with a large variety of cyclic skeletons, while some others are minimally developed with few cyclic skeletons. The implication of using such plots based on various structural-browsing indexes is under investigation.

SUMMARY

Our first extension of the algorithm to labeled pseudographs solved the oscillation problem of the edge-based Morgan naming algorithm by fixing the number of iterations and by including the current value of each vertex into its iteration sum. It reduced the classification noise as well by taking edge types into account. But the classification noise was still too large and only a limited number of iterations could be handled because of the overflow problem associated with a large number for the iterative sums and their final product. These difficulties led us to modify the iterative process by taking into account the number of bridges and the number of multiple edges, hashing vertex numbers to prime numbers, and taking the logarithm of the final vertex numbers and the final product to overcome the overflow problem. The resulting classification noise should be negligible after an appropriate choice of the values for parameters *n*, *max_v_type*, and *size_of_output*. We have yet to encounter distinct molecular equivalence classes assigned the same name by MEQNUM, our implementation of this extended Morgan naming algorithm.

ACKNOWLEDGMENT

Thanks are expressed to the reviewers of this paper, especially Dr. John M. Barnard, for the helpful comments and corrections.

REFERENCES AND NOTES

(1) Behzad, M.; Chartrand, G.; Lesniak-Foster, L. Graphs and Digraphs. *Mathematics* **1979**, 406.

(2) Morgan, H. L. The Generation of a Unique Machine Description for Chemical Structure—A Technique Developed at Chemical Abstracts Service. *J. Chem. Doc.* 1965, *5*, 107.

(3) Bemis, G. W.; Murcko, M. A. The Properties of Known Drugs. 1. Molecular Frameworks. *J. Med. Chem.* 1996, *39*, 2887.

(4) (a) Johnson, M. *Advances in Molecular Similarity*; JAI Press Inc.: London, 1998; Vol. 2 (Browsable Structure—Activity Datasets), p 153. (b) Nilakantan, R.; Bauman, N.; Haraki, K.; Venkataraghavan, R. A Ring-Based Chemical Structural Query System: Use of a Novel Ring-Complexity Heuristic. *J. Chem. Inf. Comput. Sci.* 1990, *30*, 65.

(5) (a) Wipke, W. T.; Dyott, T. M. Stereochemically Unique Naming Algorithm. *J. Am. Chem. Soc.* 1974, *96*, 4834. (b) Weininger, D. SMILES: A Chemical Language and Information System. 1. Introduction to Methodology and Encoding Rules. *J. Chem. Inf. Comput. Sci.* **1988**, *28*, 31.

(6) Such as MACCS, TAD, TAT etc. which can be computed with the MOE software package developed by Chemical Computing Group Inc., Montreal, Quebec, Canada, http://www.chemcomp.com.

(7) Randić, M. Molecular ID Numbers: By Design. *J. Chem. Inf. Comput. Sci.* 1986, *26*, 134.

(8) (a) Randić, M. On Unique Numbering of Vertexes and Unique Codes for Molecular Graphs. *J. Chem. Inf. Comput. Sci.* 1975, *15*, 105. (b) Uchino, M. Algorithm for Unique and Unambiguous Coding and Symmetry Perception of Molecular Graph Diagram. I. Vector Functions for Automorphism Partitioning. *J. Chem. Inf. Comput. Sci.* **1980**, *20*, 116.

(9) (a) Evans, L. A.; Lynch, M. F.; Willett, P. Structural Search Codes for On-line Compound Registration. *J. Chem. Inf. Comput. Sci.* 1978, *18*, 146. (b) Bawden, D.; Catlow, J. T.; Devon, T. K. Evaluation and Implementation of Topological Codes for Online Compound Search and Registration. *J. Chem. Inf. Comput. Sci.* **1981**, *21*, 83.

(10) Zheng, O.; Yuan, S.; Brandt, J.; Zheng, C. An Effective Topological Symmetry Perception and Unique Numbering Algorithm. *J. Chem. Inf. Comput. Sci.* 1999, *39*, 299.

(11) Shelley, C. A.; Munk, M. E. An Approach to the Assignment of Canonical Connection Tables and Topological Symmetry Perception. *J. Chem. Inf. Comput. Sci.* 1979, *19*, 247.

(12) Read, R. C.; Corneil, D. G. The Graphs Isomorphism Disease, *J. Graph Theory* 1977, *1*, 339.

(13) Liu, X.; Balasubramanian, K.; Munk, M. E. Computational Techniques for Vertex Partitioning of Graphs. *J. Chem. Inf. Comput. Sci.* 1990, *30*, 263.

(14) Balaban, A. T.; Filip, P.; Balaban, T. S. Computer Program for Finding All Possible Cycles in Graphs. *J. Comput. Chem.* 1985, *6*, 316.

(15) Patent pending.

(16) Feller, W. In *An Introduction to Probability Theory and Its Applications*; John Wiley & Sons: New York, 1971; Chapter 2, section 5.

(17) The 5-digit base 35 number can accommodate 52 million compounds and the 7-digit number can accommodate 64 billion compounds. Because the meqnum is defined as the first *size_of_output* digits of the mantissa of *R*, a lower digit meqnum for the same graph is always a substring of a higher digit meqnum.

(18) We used Spotfire, a commercial data visualization package marketed by Spotfire AB, Göteberg, Sweden, http://www.spotfire.com.

CI0003911