

Experimental Studies of the Universal Chemical Key (UCK) Algorithm on the NCI Database of Chemical Compounds

Robert Grossman[†], Donald Hamelberg[†], Pavan Kasturi[†], Bing Liu[‡]
grossman@uic.edu, don@lac.uic.edu, pavan@lac.uic.edu, liub@cs.uic.edu

[†]Laboratory for Advanced Computing
[‡]Department of Computer Science
University of Illinois at Chicago
Chicago, IL 60607

Keywords: Chemical id, graph, chemical compounds, molecular structure, unique key, NCI.

Abstract

We have developed an algorithm called the Universal Chemical Key (UCK) algorithm that constructs a unique key for a molecular structure. The molecular structures are represented as undirected labeled graphs with the atoms representing the vertices of the graph and the bonds representing the edges. The algorithm was tested on 236,917 compounds obtained from the National Cancer Institute (NCI) database of chemical compounds. In this paper we present the algorithm, some examples and the experimental results on the NCI database. On the NCI database, the UCK algorithm provided distinct unique keys for chemicals with different molecular structures.

Introduction

Chemical compounds usually have several common names. Although unique identifiers attached to chemical compounds would be useful for a variety of purposes, there is no consensus about how to do this. Currently most nomenclatures for chemical compounds either do not provide unique keys or the unique keys provided are based upon convention, such as when the compound was entered into a database. For this reason, determining whether a compound was entered into a database twice or comparing compounds across databases is difficult.

An illustration of the structural formula of such a compound, Testosterone, is depicted as an example in Figure 1. This molecule as entered in the National Cancer Institute (NCI) database of chemical compounds has 54 names associated with it and a unique id of 9700, which is also different from its Chemical Abstract Services (CAS) id of 58-22-0.

Because of examples like this, it is very important to construct a unique key that is derived from the structural features of the compound. Using such a key, properties of a chemical contained in a database in one location could be combined with properties of the same chemical compound contained in a database in another location automatically. With the increasing use of distributed infrastructures for computing, such as data grids and web service-based platforms, having universal chemical keys that can be used to combine distributed data about chemical compounds is of growing importance. Indeed, we have used the UCK algorithm described here

to build distributed data web applications for docking chemical compounds in proteins from the Protein Data Bank (PDB).

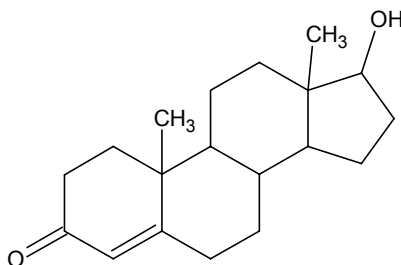


Figure 1 Structural formula of Testosterone, C₁₉H₂₈O₂. Testosterone has the NCI id¹ of 9700 and the CAS id 58-22-0. Some of the other names Testosterone goes by are 17-hydroxyandrost-4-en-3-one, Androlin, Cristerona T, and Homosteron.

Our algorithm for computing what we call a Universal Chemical Key or UCK is based upon abstracting the chemical compound as a labeled graph, with atoms represented by nodes and bonds represented by edges. The nodes are labeled with the symbols corresponding to the atoms they represent. Note that two labeled graphs representing molecular structures are the same or isomorphic if they are labeled using the same labels and can be mapped onto each other such that the labels of nodes or atoms and edges or bonds are conserved.

In this paper, we introduce an algorithm, which given a labeled graph representing a chemical compound, produces a long string, which is the UCK. This string has the properties: i) Chemical compounds associated with the same labeled graph are identical and produce the same UCK. ii) The UCKs of different labeled graphs are different in practice according to our experiments. Since the problem of distinguishing labeled graphs in general is NP-hard, it is not reasonable to expect a fast algorithm to do this 100% of the time. On the other hand, we show that in practice, on large collections of chemical compounds such as the NCI database, our UCK algorithm does have this property. Since the UCK strings can be quite long, we associate a shorter string using a standard hashing algorithm called MD5 [1]. Although the MD5 hash is not guaranteed to be unique, in practice it almost always is unique.

In this paper, we present some background of the problem for finding unique keys or nomenclatures for chemical compounds in the Related Work section. Our algorithm is presented in the Computational Methods and Algorithm section. We tested the algorithm on the National Cancer Institute (NCI) database of chemical compounds, and the results are detailed in the Results section with some relevant examples. In closing we give a brief summary of this study and discuss some future work.

Related Work

Numbering and ordering of atoms and groups of atoms of molecular structures have always been done by organic chemists, which subsequently led to several different systems of naming compounds. The International Union of Pure and Applied Chemistry (IUPAC)

¹ In the NCI database, it is also called the NSC id.

nomenclature rules [2] are the most widely used. However, these rules only work effectively for very small molecules and generally are inconsistent, hard to understand, and easily cause mistakes [3, 4, 5]. The idea of using graph theory in solving the above problem was put forward some twenty years ago [6, 7], but never really caught the attention of chemists.

Recently, the International Union of Pure and Applied Chemistry (IUPAC) established a project to develop unique keys for chemical compounds. This is an ongoing project, which was announced [8] a couple of years ago. The approach that the scientists are using is in part based on graph theory. However, the details are yet to be published. The documentation and algorithm will be published at the end of the project. Their aim is to generate a unique key from a graphical input of structural information of known and as yet unknown compounds. The keys, known as IUPAC Chemical Identifiers are alphanumeric text strings that can lead back to the structure of the compound and that can be digitized to be used in printed and electronic information.

In calculating a unique identifier for a chemical compound, one can represent the molecule as a graph as discussed in the Introduction. Therefore, two molecules are the same if the graph representations are isomorphic. Several algorithms [9, 10, 11] have been developed to check whether two graphs are isomorphic. However, direct comparison of molecular structures will not be adequate for this study because the aim is to develop an algorithm that is capable of generating unique ids for chemical compounds whether they are presently available or not. The chemical id problem can be solved if one can uniquely order the nodes in a labeled graph irrespective of their original order. These types of algorithms are in fact very useful in the graph isomorphism problem. Tinhofer and coworkers [12] developed an algorithm to generate canonical numbers for a labeled graph. They do this by considering all adjacent matrices of the graph that belong to the isomorphism class. Each matrix is read row by row as a binary number and the matrix with the smallest number is selected. The graph with this numbering is considered to be canonically numbered. Also, Brendan D. Marky [13, 14] has developed an algorithm to generate a canonical labeling map of a labeled graph. This algorithm has been implemented in the program called “nauty.”

Computational Methods and Algorithms

Overview

We begin with an overview of the algorithm. The first step in generating the UCK is to represent different types of covalent bonds, i.e., single, double or triple by just a single bond between atoms of the molecule, so that we can focus only on the connectivity and also because there is no unique way to determine double and triple bonds. We also disregard the overall charge on the molecule. If desired, variants of the algorithm below could incorporate this information as additional labels. The molecule is now considered as an undirected labeled graph $G = (V, E)$ consisting of a finite set of vertices V and a finite set of edges E . The set of atoms in the molecule is the vertex set, V and the set of covalent bonds between the atoms is the edge set, E . Each edge has a unity weight. At this stage, the labels in the graph G represent atoms and do not reflect any of the local structure around the atom.

The next step in the UCK algorithm is to replace the labels with new labels that capture some of the local connectivity and chemical environment around each atom. Atoms with similar connectivity will end up having similar labels. In the next step of the UCK algorithm, the

lengths of the shortest paths between each pair of vertices of the vertex set V of the graph G are generated. The path labels are produced by concatenating the source label, the path length, and the destination label. At every stage of labeling we follow a rule based lexicographical ordering so that the whole procedure is invariant to the changes in ordering of the vertex set V . A lexicographical ordering of the labels of all pair of shortest path sets is done. The labels are concatenated to form a string and prefixed by the molecular formula of the molecule. The string thus obtained uniquely represents the molecule.

Here is part of an example. A more detailed description of the algorithm follows. A schematic representation of a molecule that may or may not exist is shown in Figure 2 with single, double and triple bonds and shows an overview of the algorithm. We represent the double and triple bonds as single bonds, so that it can be represented as a graph $G = (V, E)$ where the vertex set $V = \{1, 2, 3, 4, 5, 6\}$ and the edge set $E = \{(1, 2), (2,3), (3,4), (4,5), (5,6)\}$. The list $\langle O, C, C, C, C, C \rangle$ is the label list for elements corresponding to V . Observe that though the connectivity of the second vertex is different from that of the third to the sixth they all have the same labels.

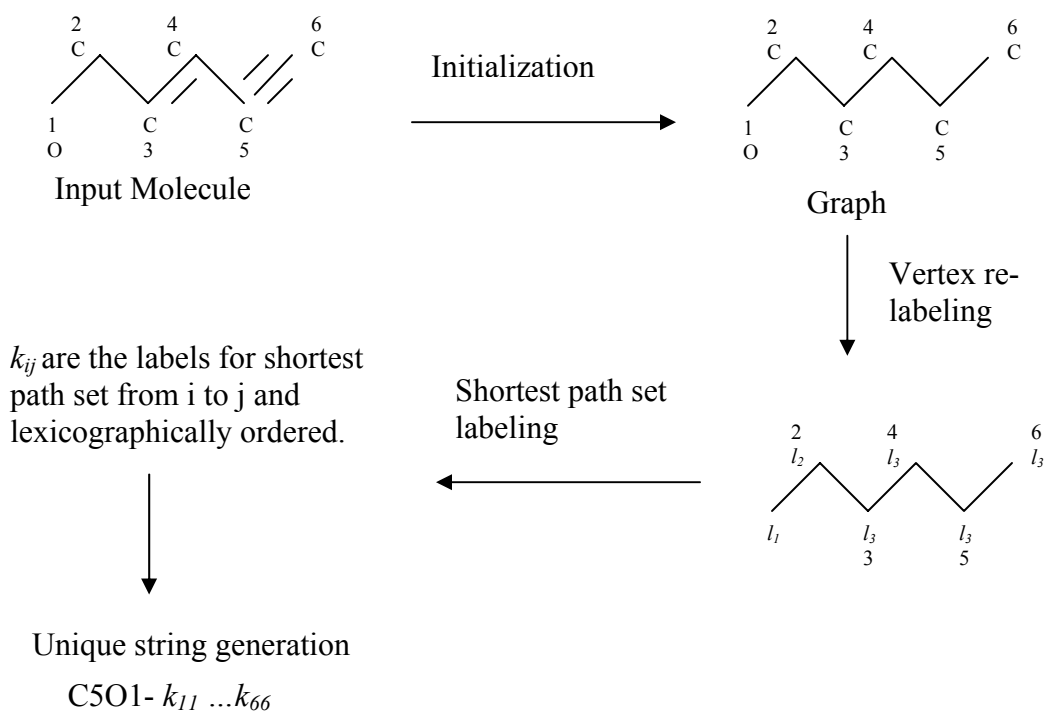


Figure 2 An example showing the procedure to generate a unique string for a given molecule

Algorithm

Initial setup:

The first step is to represent different types of covalent bonds as single bonds. Next we consider the molecule as an undirected graph $G = (V, E)$ consisting of a finite set of vertices V and a set of edges E . The set of atoms in the molecule is the vertex set, V and the set of covalent

bonds between the atoms is the edge set, E . The weight function is defined as $w: E \rightarrow \{1\}$.

Labeling of the vertices:

For a depth d , we define $\lambda^{(d)}$ inductively from $\lambda^{(d-1)}$. $\lambda^{(d)}$ re-labels each node in G .

Step 0:

Define $\lambda^{(0)}(b) = \text{label}(b)$ where b is a node in Graph G and $\lambda^{(d)}$ is the label map.

Step 1:

Fix node 'a' and let b_1, \dots, b_k be its children.

Step 2:

Compute $\lambda^{(d-1)}(b_1), \dots, \lambda^{(d-1)}(b_k)$

Step 3:

Remove all occurrences of a from $\lambda^{(d-1)}(b_1), \dots, \lambda^{(d-1)}(b_k)$, lexicographically order them to produce the string $\lambda^{(d-1)}(b_{i1}) \dots \lambda^{(d-1)}(b_{ik})$.

Step 4:

Define $\lambda^{(d)}(a) = \text{label}(a) \lambda^{(d-1)}(b_{i1}) \dots \lambda^{(d-1)}(b_{ik})$

Labeling of shortest paths:

Re-label G with labels $\lambda^{(2)}$ (we use $d = 2$ in this work). Call this G' . Fix $a \in V$ of G' for each $b \in V$ of G' .

Define $\mu_a(b) = \text{The concatenated string } \langle \text{label}(a) \ n \ \text{label}(b) \rangle$, where $n = \text{length of shortest path from } a \text{ to } b$.

Generate unique key: $\mu(G)$ for graph G

$\mu(G) = \text{Chemical formula:Lexicographically Ordered string } \langle \mu_a(b) \mid a \in V, b \in V \text{ of } G' \rangle$

$\mu(G)$ is the unique key.

Example:

Consider benzoic acid ($C_7H_6O_2$).

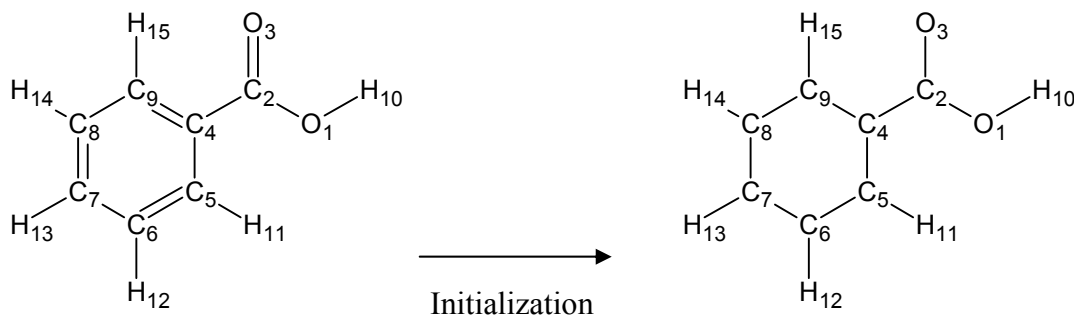


Figure 3 Structure of benzoic acid (NCI number = 149).

New labels for the vertices:

<i>Vertex Number</i>	<i>New Label $\lambda^{(2)}$</i>
1	OCCOH
2	CCCCOOH
3	OCCO
4	CCCHCCHCOO
5	CCCCCCHH
6	CCCHCCHH
7	CCCHCCHH
8	CCCHCCHH
9	CCCCCCHH
10	HOC
11	HCCC
12	HCCC
13	HCCC
14	HCCC
15	HCCC

Table 1 Showing new labels generated.

The unique string $\mu(G)$ is:

C7H6O2:CCCCCHH0CCCCCHHCCCCCHH0CCCCCHHCCCCCHH1CCCHCCHCO0CCCCCHH1CCCHCCHCO
OCCCCCHH1CCCHCCHHCCCCCHH1CCCHCCHHCCCCCHH1HCCCCCCCCCHH1HCCCCCCCCCHH2CCCCCH
HCCCCCHH2CCCCCHHCCCCCHH2CCCCOOHCCCCCHH2CCCCOOHCCCCCHH2CCCHCCHHCCCCCHH2C
CCHCCHHCCCCCHH2HCCCCCCCCCHH2HCCCCCCCCCHH3CCCHCCHHCCCCCHH3CCCHCCHHCCCCCHH3H
CCCCCCCCCHH3HCCCCCCCCCHH3HCCCCCCCCCHH3HCCCCCCCCCHH3OCCOCCCCCHH3OCCOCCCCCHH3
OCCOHCCCCCHH3OCCOHCCCCCHH4HCCCCCCCCCHH4HCCCCCCCCCHH4HOCCCCCCCCCHH4HOCCCCOOH0
CCCCOOHCCCCOOH1CCCHCCHCO0CCCCOOH1OCCOCCCCOOH1OCCOHCCCCOOH2CCCCCHHCCCCOOH2CC
CCCCNHCCCCOOH2HOCCCCOOH3CCCHCCHHCCCCOOH3CCCHCCHHCCCCOOH3HCCCCCCCCOOH3HCCCC
COOH4CCCHCCHHCCCCOOH4HCCCCCCCCOOH4HCCCCCCCCOOH5HCCCCCCHCCHCOO0CCCHCCHCO0CCCHC
HCOO1CCCCCHHCCCCCHCO01CCCCCHHCCCCCHCO01CCCCOOHCCCCCHCO02CCCHCCHHCCCCCH
COO2CCCHCCHHCCCCCHCO02HCCCCCCHCCHCO02HCCCCCCHCCHCO02OCCOCCCCCHCO02OCCOHCCC
HCCCHCO03CCCHCCHHCCCCCHCO03HCCCCCCHCCHCO03HCCCCCCHCCHCO03HOCCCCCHCCHCO04HCCC
CCHCCHH0CCCHCCHHCCCCCHH0CCCHCCHHCCCCCHH0CCCHCCHHCCCCCHH1CCCCCHHCCCCCHH1C
CCCCCHHCCCCCHH1CCCHCCHHCCCCCHH1CCCHCCHHCCCCCHH1CCCHCCHHCCCCCHH1CCCHCCHH
CCHCCHH1HCCCCCHCCHH1HCCCCCHCCHH1HCCCCCHCCHH2CCCCCHHCCCCCHH2CCCCCHHCCCCCH
CHH2CCCHCCHCO0CCCHCCHH2CCCHCCHCO0CCCHCCHH2CCCHCCHHCCCCCHH2CCCHCCHHCCCCCHH2
HCCCCCHCCHH2HCCCCCHCCHH2HCCCCCHCCHH2HCCCCCHCCHH2HCCCCCHCCHH2HCCCCCHCCHH
3CCCCCHHCCCCCHH3CCCCCHHCCCCCHH3CCCCOOHCCCCCHH3CCCCOOHCCCCCHH3CCCHCCHCO
CCCHCCHH3HCCCCCHCCHH3HCCCCCHCCHH3HCCCCCHCCHH3HCCCCCHCCHH4CCCCOOHCCCCCHH4
HCCCCCHCCHH4HCCCCCHCCHH4OCCOCCCCCHH4OCCOCCCCCHH4OCCOHCCCCCHH4OCCOHCCCC
CHH5HOCCCCCHCCHH5HOCCCCCHCCHH5OCCOCCCCCHH5OCCOCCCCCHH6HOCHCCC0HCCCCCCC0HCCCC
CCC0HCCCCCCC0HCCCCCCC0HCCCCCCC1CCCCCHHCCCC1CCCCCHHCCCC1CCCHCCHHCCCC1CCCHCCHH
HCCC1CCCHCCHHCCCC2CCCCCHHCCCC2CCCCCHHCCCC2CCCHCCHCOOHCCC2CCCHCCHCOOHCCC2CC
HCCCHHCCC2CCCHCCHHCCCC2CCCHCCHHCCCC2CCCHCCHHCCCC2CCCHCCHHCCCC2CCCHCCHHCCCC3CC
CCCCCHHCCCC3CCCCCHHCCCC3CCCCCHHCCCC3CCCCOOHCCC3CCCCOOHCCC3CCCCOOHCCC3CCC
HCCCHCOOHCCC3CCCHCCHCOOHCCC3CCCHCCHHCCCC3CCCHCCHHCCCC3CCCHCCHHCCCC3CCCHCCHHCCC
C3HCCCHCCCC3HCCCHCCC3HCCCHCCC3HCCCHCCC3HCCCHCCC3HCCCHCCC3HCCCHCCC3HCCCHCCC4CCCC
CCHHCCCC4CCCCCHHCCCC4CCCCOOHCCC4CCCCOOHCCC4CCCHCCHCOOHCCC4CCCHCCHHCCCC4CCC
HCCCHHCCCC4HCCCHCCC4HCCCHCCC4HCCCHCCC4HCCCHCCC4HCCCHCCC4HCCCHCCC4HCCCHCCC4HCCCH

CCC4OCCOHCCC4OCCOHCCC4OCCOHHCCC4OCCOHHCCC5CCCCOHHCCC5HCCCHCCC5HCCCHCCC5HCCCH
 CCC5HCCCHCCC5HOCHCCC5HOCHCCC5OCCOHCCC5OCCOHCCC5OCCOHCCC5OCCOHCCC6HOCHCCC6HO
 CHCCC6OCCOHCCC6OCCOHHCCC7HOCHOC0HOCHOC1OCCOHHOC2CCCCOHHOC3CCCHCCHCOOHOC3OCCO
 HOC4CCCCCHHHOC4CCCCCHHHOC5CCCHCCHHHOC5CCCHCCHHHOC5HCCCHOC5HCCCHOC6CCCHCCHHH
 OC6HCCCHOC6HCCCHOC7HCCCOCCO0OCCO0CCO1CCCCOHOCCO2CCCHCCHCO0OCCO2OCCOHOCCO3CCC
 CCCHHOCCO3CCCCCHHOCCO3HOCOCO4CCCHCCHHOCCO4CCCHCCHHOCCO4HCCCOCCO4HCCCOCCO5CC
 CHCCHHOCCO5HCCCOCCO5HCCCOCCO6HCCCOCCOHO0CCOHOCCO1CCCCOHOCCO1HOCOCOHO2CCCH
 CCHCO0OCCO2OCCO0CCO3CCCCCHHOCCO3CCCCCHHOCCO4CCCHCCHHOCCO4HCCCOCCO4HCCCOCCO
 H4HCCCOCCO4HCCCOCCO5CCCHCCHHOCCO5HCCCOCCO5HCCCOCCO6HCCC

As we can see, the length of the generated string is very large. So we used the *md5* algorithm to generate a *hex digest* for the above string. The *hex digest* is 39BF9B334B172E4E71E76B93C830B47E. From this point on, we will only use *hex digest* of *md5* to represent the unique string.

Results

We implemented the algorithm in C and tested it on 236,917 chemical compounds in the NCI database of chemical compounds. The input was a trimmed version of a PDB file that contains the atomic symbols and their spatial coordinates. We calculated the bond information and initialized it. We represented the graph in the form of an adjacency matrix and ran our UCK algorithm. The output was then piped to the *md5* algorithm to generate the UCK *hex digest*.

We generated UCK strings and the UCK *md5 hex digest* for 236,917 data sets consisting of a variety of drug molecules, including, in many cases, variants of the same molecule. The results are summarized in Table 2. By variants of the same molecule we mean data sets (which represent chemical compounds) that differ in the ordering of the atoms and/or their spatial orientation. The NCI database has many molecules whose structures are very similar but are different molecules. There are also molecules that have the same molecular formula but different structural formulas.

Description	Number	Remarks
Total number of chemical compounds	236,917	Includes some compounds with duplicate entries
Number of chemical compounds with single entry	203,384	All gave unique UCK
Number of chemical compounds with two or more entries	33,533	The UCK algorithm gave unique keys to all distinct compounds. The UCK algorithm gave the same key to the same compound occurring in multiple entries.

Table 2 Summary of number of compounds studied

Our algorithm produces excellent results, generating the same key for variants of the same molecule and different keys for different compounds. The algorithm is invariant to changes

<i>NCI Number</i>	<i>Formula</i>	<i>CAS</i>	<i>#Names</i>	<i>md5 hex digest generation of UCK</i>
91771	C ₃₈ H ₄₂ N ₂ O ₆	23495-89-8	6	5C0F9A8F0ECC0BAF32CBCA62DA571F42
97338	C ₃₈ H ₄₂ N ₂ O ₆	477-57-6	6	5C0F9A8F0ECC0BAF32CBCA62DA571F42

Table 4 Snapshot of NCI database and the hex digest (last column) generated by our algorithm for the two molecules with a free bond rotation.

The algorithm is sensitive to changes in connectivity even at the remotest portions of the molecule, which makes it very effective in detecting different chemical compounds that are very similar. Figure 6 shows a pair of compounds, which are quite similar. Table 5 gives the results of our algorithm. We can observe that their UCK keys are different. Figure 7 shows another pair of similar compounds, and Table 6 gives the UCK keys produced by our algorithm. Again, we can see that their keys are quite different.

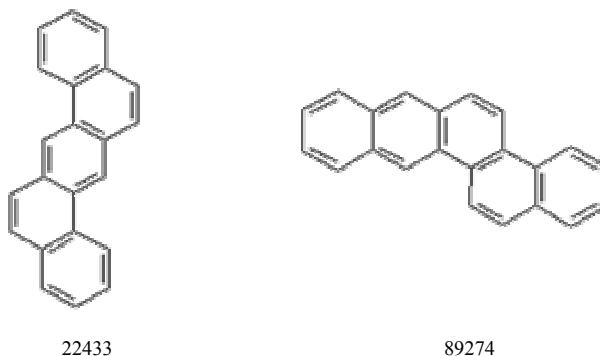


Figure 6 Structures of dibenzo[a,h]anthracene (22433) and benzo[b]chrysene (89274).

<i>NCI Number</i>	<i>Formula</i>	<i>CAS</i>	<i>#Names</i>	<i>md5 hex digest generation of UCK</i>
22433	C ₂₂ H ₁₄	53-70-3	11	A15F6359F7AC44C1A90C0F90598664B4
89274	C ₂₂ H ₁₄	214-17-5	10	7A4A4AF922300C1270423810B748FAF5

Table 5 Snapshot of NCI database and the hex digest (last column) generated by our algorithm for molecules with changes at remote parts of the molecule.

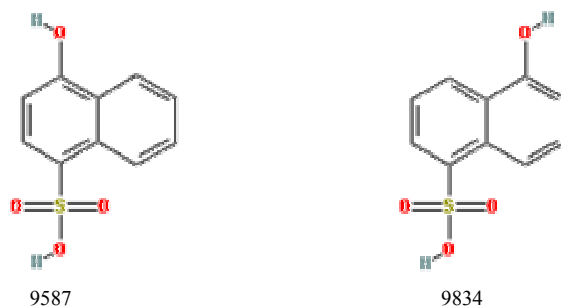


Figure 7 Structures of 4-hydroxy-1-naphthalenesulfonic acid (NCI 9587) and 5-hydroxy-1-naphthalenesulfonic acid (NCI 9834).

<i>NCI Number</i>	<i>Formula</i>	<i>CAS</i>	<i>#Names</i>	<i>md5 hex digest generation of UCK</i>
9587	C ₁₀ H ₈ O ₄ S	84-87-7	12	6BEAF3C856A2C1318F306DDB0E7F888A
9834	C ₁₀ H ₈ O ₄ S	5419-77-2	1	64373D3DB8329F56B6AFAE59F07AAFC3

Table 6 Snapshot of NCI database and the last column the hex digest generated by our algorithm for molecules with changes at remote parts of the molecule.

Execution time: The proposed algorithm is very efficient. Even for the largest compound in the NCI database, it takes less than a second to produce the UCK key on a Pentium 4 Linux PC with 512MB of memory.

Summary and Conclusions

We have developed an algorithm called the UCK algorithm that generates unique keys for a wide variety of chemical compounds. The UCK algorithm views molecular structures as undirected labeled graphs. The atoms are represented as the vertices and the edges as the bonds. The algorithm was experimentally tested on 236,917 compounds from the NCI database, and generated unique keys for all the uniquely identified structures. We call these keys Universal Chemical Keys or UCKs.

We have used the UCKs to build distributed web-service based applications involving protein docking with data pulled from multiple distributed databases. Without a unique key such as the UCK, the application would have no way of knowing whether two distributed proteins or chemicals were the same or not.

The UCK algorithm depends only upon the structure of the labeled graph. Distinct labeled graphs give rise to distinct UCKs. On the other hand, two different labeled graphs could, in theory, give rise to the same UCK. This is a calculated trade-off. Distinguishing arbitrary labeled graphs is NP-hard and hence a fast deterministic algorithm cannot be expected. On the other hand, chemical compounds give rise to a restricted class of labeled graphs – it is likely that our algorithm can be proven to be unique on various restricted classes of chemical compounds. We plan to investigate these types of results and to verify experimentally the properties of our UCK algorithm on additional databases of chemical compounds.

In particular, in this study the UCK algorithm uses a depth $d = 2$, for labeling the vertices. Using different depths and dynamically assigning the depth depending upon the compound gives rise to UCKs that are more expensive to compute but stronger. We plan on investigating these trade-offs in future work.

To summarize, the UCK algorithm is a fast and effective algorithm that provides intrinsic and unique keys for a wide class of commonly occurring chemical compounds.

References

- 1 R. Rivest. The MD5 message digest algorithm. RFC 1321, 1992.
- 2 IUPAC, Nomenclature of Organic Chemistry. Pergamon Press: Oxford, 1979.

- 3 M. H. Klin, O. V. Lebedev, T. S. Pivina, N. S. Zefirov: Nonisomorphic cycles of maximum length in a series of chemical graphs and the problem of application of IUPAC nomenclature rules. *MATCH* 27, 1992, 133-151.
- 4 R. C. Read: The Coding of various Kinds of unlabeled Trees. In: Read R. C. (ed.): *Graph Theory and Computing*. Academic Press, New York, 1972.
- 5 R. C. Read: A new system for the design of chemical compounds. 2. Coding of cyclic compounds. *Journal of Chem. Inf. and Comp. Sci.* 25, 1985, 116-128.
- 6 M. Randić: On recognition of identical graphs representing molecular topology. *J. Chem. Phys.* 60, 1974, 3920-3928.
- 7 A. Prokurowski: Search for unique incidence matrix of a graph. *BIT* 14, 1974, 209-226.
- 8 *Chemistry International* 23, 3, 2001.
- 9 V. L. Arlazarov, I. I. Zuev, A. V. Uskov and I. A. Faradzev: An algorithm for the reduction of finite non-oriented graphs to canonical form. *Zh. Vycheil. Mat. Mat. Fiz.* 14, 3 (1974) 737-743.
- 10 T. Beyer and A. Proskurowski: Symmetries in graph coding problem. *Proc. NW76 ACM/CIPC Pac. Symp.* (1976) 198-203.
- 11 C. Böhm and A. Santolini: A quasi-decision algorithm for the p-equivalence of two matrices. *ICC BULLETIN* 3, 1 (1964) 57-69.
- 12 M. Klin, C. Rucker, G. Rucker, G. Tinhofer: Algebraic combinatorics in mathematical chemistry. Methods and algorithms. I. Permutation Graphs and coherent (Cellular) algebras. Technical Report, Technische Universität München, TUM-M9510 (1995).
- 13 B. D. McKay: Computing automorphism and canonical labeling of graphs. *International Conference on Combinatorial Mathematics, Canberra (1977)*, *Lecture Notes in Mathematics* 686, Springer-Verlag 223-232.
- 14 B. D. McKay: Practical Graph Isomorphism. *Congressus Numerantium*, 30 (1981), 45-87.