A New System for the Designation of Chemical Compounds. 1. Theoretical Preliminaries and the Coding of Acyclic Compounds[†]

RONALD C. READ

Department of Combinatorics and Optimization, University of Waterloo, Waterloo, Ontario, Canada N2L 3G1

Received April 20, 1982

This paper and its sequel describe in detail a method whereby a unique designation (or "code") of any chemical compound can be derived from its structural formula. This code is a linear string of standard typewriter, or computer, characters and is well-suited for use in many applications concerning retrieval of information about chemical compounds. The advantages of this system over other such systems at present in use are threefold. First, the system is "closed" in the sense that a fixed set of rules suffice to generate the code for any compound whatsoever, provided only that its structural formula is known. Thus there can be no occasion to have to add further rules, list further parent structures, and so on as a consequence, for example, of the discovery of some new kind of molecular structure. Second, the decoding process, retrieving the structural formula from the code, is simple and can be performed by hand if necessary. Third, the derivation of the code does not rely on any appeal to chemical intuition but is based on graph-theoretical principles that make it comparatively easy to program the coding and decoding processes on a computer. This paper gives a survey of the basic problem of chemical nomenclature, its computational complexity, and its relation to the problem of graph isomorphism. There follows a description of the procedure for deriving the code of an acyclic compound. Also discussed is a method whereby information concerning stereoisomerism can be included in the code. The sequel to this paper will describe how to derive the code of a cyclic compound.

(1) INTRODUCTION

There was a time in the history of organic chemistry when the question of nomenclature presented no problems; the number of known compounds was sufficiently small that each compound could be given an arbitrarily chosen name (a "trivial" name) without overtaxing either the ingenuity of those who chose the names or the memory of those that had occasion to use them. Later this system was extended by combining two or more trivial names in various more or less self-explanatory ways, making such combinations as "methyl acetate", "butyl alcohol", and so on.

As the number of different compounds increased, this simple method became less and less practical for the purpose of allocating distinctive names. There was clearly a need for a more systematic way of giving each compound a name, or at least some kind of designation. A number of chemical nomenclature and notation systems were devised to fill this need. The IUPAC system, based on the work of Dyson,[3] and the Wiswesser Line Notation (see ref 14) are notable examples of notation systems that have been put to considerable practical use. Many other methods for designating compounds have also been proposed.

Regarded from a mathematical point of view, each such system is essentially an algorithm—a set of instructions—which takes as its input the structural formula of a molecule (such as that shown in Figure 1) or something equivalent to it, such as a connection table, and gives, as output, a designation for the molecule. This designation is not generally a name; it is more likely to be a string of symbols with little or no pretense at being pronounceable.

Today the number of known organic chemical compounds is several millions. Nevertheless, the chemical nomenclature and notation systems that are at present in use are able to keep track of these compounds and to give a designation for each. It might seem pointless therefore to propose yet another system (as will be done in this paper and its sequels). There are reasons, however, for doing this, reasons that will appear later; but first we need to look at the graph-theoretical concepts that underlie the problem of chemical designation.

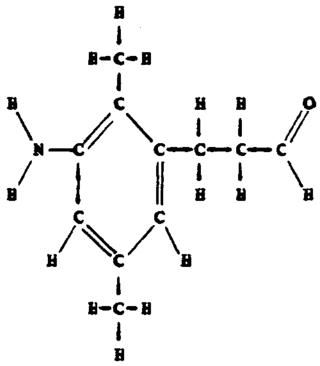


Figure 1.

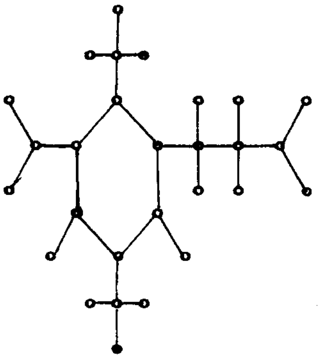


Figure 2.

(2) SOME GRAPH-THEORETICAL CONCEPTS

The problem of distinguishing between different chemical compounds and of giving them distinctive designations is basically a variation on a famous (or perhaps one should say notorious) problem of graph theory, the isomorphism problem. A graph, in the sense that graph theorists use the word, is a set of vertices (which we can denote by points in space) some pairs of which are "adjacent" or "joined". The adjacency of

[†] The research for this paper was supported by Grant A8142 from the National Science and Engineering Research Council of Canada.

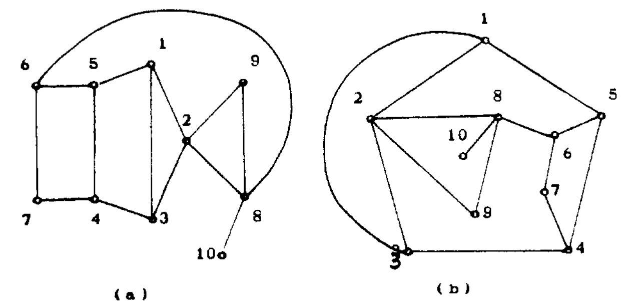


Figure 3.

two vertices can be indicated diagramatically by drawing a line (either straight or curved) between the points that represent them. Such a line is called an edge. Thus the diagrams in Figures 2 and 3 depict typical graphs.

Now there are some obvious similarities between Figures 1 and 2, as well as some significant differences. In both figures, for example, we have an example of a configuration known to chemists as a "ring" (the benzene ring in Figure 1) and to graph theorists as a "circuit" or "cycle" (the vertices and edges in the corresponding part of Figure 2). Among the differences is the fact that in the structural formula shown in Figure 1, which can be regarded as a graph of a rather special type, there are several different *kinds* of vertices, according to the atoms that they represent, whereas the vertices in Figure 2 are not, of themselves, different from each other. A more important difference is that in Figure 1 some of the bonds are double, whereas there are no "double edges" in a graph according to the usual definition. Graph theorists occasionally allow the possibility of double edges—two vertices joined by two edges—but when they do they tend to regard these two edges as forming a cycle (of length 2). This is clearly not in accord with chemical practice; we would not want to class ethylene, for example, as a cyclic compound because of its double bond. Thus, in interpreting a structural formula as a graph, it is better to regard single bonds, double bonds, triple bonds, etc. as being different *kinds* of edges. In this way the word "acyclic" will have a uniform meaning, and various other possible sources of confusion can be avoided.

It is clear how Figures 1 and 2 are related. Figure 2 is obtained from Figure 1 by ignoring the differences between the atoms and between different types of bonds. A graph obtained from a structural formula in this way can be called a "chemical structure graph" or just a "structure graph" (see Goodson[4]). This concept is useful in discussing the general structure of a molecule. Also useful is the intermediate concept in which the distinction between different kinds of bonds is ignored while differences between atoms are respected. This latter concept will be important in the coding of cyclic compounds.

A structural formula therefore is a special type of graph in which the vertices are of several kinds and in which there are several different kinds of edges. These are quite minor modifications to the original concept of a graph, and all the remarks made below about graphs apply (with the appropriate slight changes) to structural formulas.

Any graph, whether a structural formula, a structure graph, or some other kind, can be represented geometrically in many ways, since there is nothing that dictates where the vertices are to be placed. Thus the two diagrams in Figure 3, though seemingly very different, represent the same graph. This can be seen by checking the labels placed next to the vertices. It will be seen that for each pair of labels i and j, the corresponding vertices are either joined in both graphs or joined in neither. This property, loosely described as that of "being the same graph", is called "isomorphism". Two graphs are isomorphic if and only if their vertices can be labeled with the integers 1 up to n (the number of vertices) in such a way that adjacency is preserved, i.e., so that, as in the above example, vertices i and j are either adjacent in both graphs or nonadjacent in both.

An important problem in graph theory (the isomorphism problem) is that of devising a "good" algorithm for determining whether two graphs are isomorphic or not. The sense of the all-important word "good" here is that the algorithm should be "polynomial"; that is, that the number of operations required to carry out the determination (or, equivalently, the running time of a computer program that implements the algorithm) should be bounded by an expression of the form An^k, where A and k are constants and n, as before, is the number of vertices.

Now there is an obvious method of testing for isomorphism, namely, to label one graph arbitrarily and then run through all possible labelings of the other graph, testing to see whether adjacency is preserved by the current labeling. Unfortunately this method does not satisfy the requirement of being "good"; the number of labelings to be tried is $n!$, a number which increases very rapidly with n and which as n becomes large will exceed any expression An^k, whatever the values of A and k may be.

An "exponential algorithm" is one whose running time is of the order of $A\alpha^n$, where A and α are constants. Exponential algorithms tend not to be practical since their running time increases so rapidly with the value of n; hence the insistence on a "good" algorithm in the statement of the isomorphism problem. The running time of the algorithms just described, depending as it does on $n!$, is even worse than exponential and therefore does not qualify as a solution to the isomorphism problem. Despite the vast amount of research that has been carried out on this problem in many places, no algorithm for isomorphism of graphs in general has yet been devised which is any better than exponential. For a survey of the graph isomorphism problem, including some of its chemical aspects, see ref 12.

Hand in hand with the isomorphism problem goes the "coding problem". This is the problem of devising a good algorithm that will associate with each graph a "code", which can be thought of as some linear string of symbols, in such a way that two graphs have the same code if and only if they are isomorphic. Think of this as the problem of giving a distinctive "name" (in a somewhat abstract sense) to every graph and the connection with the problem of chemical documentation becomes apparent. It is clear that a solution to this coding problem would at once provide a solution to the isomorphism problem. To test whether two graphs were isomorphic, we would merely have to compute (in polynomial time) their two codes and compare the two linear strings obtained. Thus these two problems are closely linked. Naturally, in view of what has just been said, the coding problem is also unsolved—there is no known polynomial algorithm for coding graphs in general.

Thus it is seen that the basic problem of chemical nomenclature is the graph coding problem applied to the specific class of graphs, namely, the structural formulas, with their different kinds of vertices and of edges. Unfortunately, this restriction to structural formulas does not bring with it any radical simplification of the problem. However, for certain more special kinds of graph, the isomorphism problem can be solved. We take a brief look at these possibilities in the next section.

(3) SPECIAL GRAPHS

A "tree" is a graph that is connected and without circuits. It corresponds therefore to the structural formula of a single acyclic molecule. Figure 4 shows a tree, in the graph-theoretical sense, and a typical acyclic compound whose structure

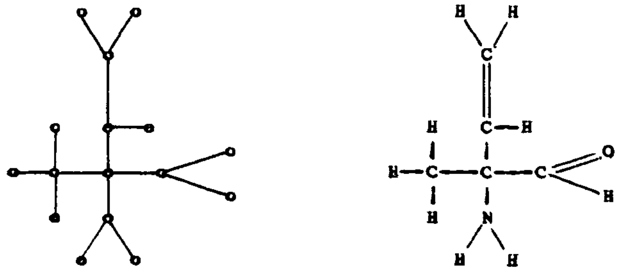


Figure 4.

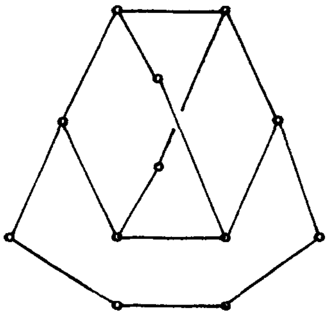


Figure 5.

graph is that same tree. Trees have many special properties which tend to make them easier to handle than graphs in general.

In particular, the isomorphism problem and the coding problem for trees are not difficult. It has been known for some time that trees can be coded very efficiently. With skillful programming and careful choice of data structures, computer programs can be devised that will code trees in a time directly proportional to the number of vertices. In other words, we have $k = 1$ in the expression An^k. Such a program or algorithm is said to be "linear". For a discussion of various methods of coding trees see ref 11; for more on the linear algorithm to code trees see ref 1. Hence, as far as acyclic compounds are concerned, there are no theoretical problems associated with their coding, and the coding system described in the latter part of this paper is basically one of these tree coding algorithms modified to take into account some of the special properties of structure graphs and (an important point) to organize the output into a chemically meaningful format (more on this later).

A "planar graph" is one that can be drawn in the plane without any of its edges intersecting another. All the graphs in Figures 1–4 are planar, but Figure 5 shows one that is not. This graph has been drawn with just one intersection of edges; it could have been drawn differently with more intersections, but it can be proved that it cannot be drawn so as to have no intersections at all.

Planar graphs, too, have many special properties. In particular, polynomial algorithms exist for testing planar graphs for isomorphism, and polynomial coding algorithms are also known. Hence, if all structural formulas were planar, we could use these algorithms to devise a good coding system for cyclic chemical compounds. It should be pointed out, however, that these algorithms, unlike those for trees, are not particularly straightforward; they would be less easy to implement in a chemical context, and it would be difficult to organize the output into a convenient and meaningful form.

But are structural formulas necessarily planar? Undoubtedly most if not all presently known compounds have structural formulas which can be drawn in the plane without having to make bonds intersect, but there is no reason to suppose that this will always be so. The growing skill of practical chemists at synthesizing intricate molecular structures makes it only a matter of time before many such compounds are known, and they will need to be documented along with the others. Moreover, the philosophy behind the coding system described in these papers has been that it should be one that can be applied to any conceivable structural formula (i.e., any collection of atoms bonded together in any way) irrespective of whether it corresponds to a feasible chemical compound. For only in this way can we be quite sure that the system will be able to cope with any compound, however bizarre, that may come our way.

For these reasons the existing algorithms for coding planar graphs are not suitable for our purposes, and although they are of interest and worth a brief mention, we must pass them by.

The "degree" of a vertex in a graph is the number of edges that "come together" (in an obvious sense) at that vertex. It corresponds roughly to the valency of an atom in a structural formula, except in so far as we are regarding multiple bonds as different kinds of single edges. We shall, in fact, use the term degree in precisely this way in what follows to denote the number of bonds (of whatever multiplicity) that have the atom in question at one end. In short, it is the number of other atoms to which the given atom is connected. A third class of special graph to be mentioned consists of those graphs whose vertices have a bounded degree, that is to say, a class for which no vertex has a degree exceeding some fixed integer m.

The chemical analogue of such a class, for the case $m = 12$, will include all structural formulas corresponding to chemically feasible compounds, since in real life no atom has a confirmed connectivity greater than 12. Hence a good algorithm for coding graphs of bounded degree would be directly applicable to the chemical coding problem.

Such an algorithm may exist. Some very recent work of Luks[7] has shown that isomorphism of graphs with bounded degree can be tested in polynomial time. This would suggest that it might be possible to devise a coding algorithm for these graphs which would also be polynomial. Of course, the degree of the polynomial (the value of k in the expression An^k) would increase with m and might be quite high for $m = 12$, so that the algorithm, though polynomial, might not be all that practical. Still, it is an interesting possibility.

It is too soon to say for sure whether this recent result offers any scope for application to the chemical coding problem. The indications are that it does not. The result is highly theoretical, involving group-theoretical and other concepts that are far removed from those usually employed by chemists to describe and document compounds. Hence there will be no attempt to use this new result in the present system of documentation, though its potential usefulness in this field is noted.

It follows then that a system that will code *any* structural formula will most likely be in the form of an exponential algorithm. In consequence, the coding algorithm that will be presented in part 2, for coding cyclic compounds, is exponential. It should be remembered, however, that the designation "exponential" represents a worst-case judgment. An algorithm is exponential because *some* graphs (possible only a few) cause it to take exponential time in handling them; the majority of graphs may be handled very quickly. The algorithm of part 2 makes use of some special properties of structural formulas to ensure that almost all compounds are handled comparatively quickly. Under these circumstances one can tolerate the occasional compound which, because of some exceptional characteristic, takes up a lot of computer time.

(4) DESIRABLE PROPERTIES OF A CHEMICAL CODE

We now consider the most important question concerning chemical nomenclature, namely, "What form should the code of a chemical compound take?" This is a question to which no single answer can be given; for a system of nomenclature that is ideal in one context may be quite unsuitable in another. Thus, for example, a system that is designed to enable chemists to talk about compounds among themselves or to refer to them in presenting papers will need to produce codes that are tolerably brief and reasonably pronounceable. At the other extreme, a system might be intended only for computer use, in which case the code could be any kind of symbol string—not necessarily brief, almost certainly not pronounceable, and probably not intelligible, except with difficulty, to humans. Let us imagine, for example, a computer program to implement the basic problem of chemical information retrieval. It might work as follows. The user inputs the details of a structural formula to the program and receives a reply such as "That is a known molecule; it is referenced in ...", or perhaps "I have no references to any such compound." Such a program would first compute the code of the given compound and then search for this code among its stored information concerning chemical compounds. This exemplifies a type of coding with which humans would never even come into direct contact. In between these two extremes are other possibilities—codes that might normally be computed by a computer but which can be understood by chemists and used by them in day to day work.

In a research report produced some years ago[8] R. S. Milner and I drew up a list of attributes that might, in various contexts, be desired in a chemical coding system. Many of these attributes are clearly incompatible with others, but it will be useful to consider them in turn and say something about them. Here is a slightly extended version of that list. (1) The codes should be linear strings of symbols. (2) The information on the structural formula should give rise to a unique code by a clearly defined process (the coding algorithm). (3) The information on the structural formula should be recoverable from the code by a clearly defined process (the decoding algorithm). (4) The coding process should be simple; for preference it should be possible for a chemist to code a compound by hand, without the use of a computer. (5) The decoding process should be simple, again, preferably one that can be carried out by hand. (6) The coding process should not depend at all on chemical intuition or on knowledge of the properties of chemical compounds. (7) The coding process should not need to have recourse to any lists of names or other nonsystematic items. (8) The code should be brief. (9) The code should be pronounceable. (10) The symbols used in the code should be familiar. No symbol should be used that is not present on a standard typewriter or computer terminal keyboard. (11) The code should be easily comprehensible to chemists. (12) The coding and decoding algorithms should be efficient (i.e., polynomial) rather than exponential or worse.

Here are some comments on the points raised above.

(1) Codes should be linear symbol strings. This attribute is desirable for any chemical documentation system. The basic operation in a data-retrieval program of the type described above is that of locating one particular item among a large number of possibilities. This implies some form of look-up process, one of the simplest and commonest of which is that of lexicographical search, exactly as in looking up a word in a dictionary. In order for this to be possible the items must be in the form of linear symbol strings. Most systems so far developed give their output in this form. Here are some typical examples: (a) 1-bromo-2,3-dichloro-3-methylbutane ("classical" chemical nomenclature; see below), (b) C_5·C,3·X,1,5,6 (Dyson's system, from which the IUPAC system was derived[6]), (c) L56TJ CVQ GNW (Wiswesser Line Notation[14]), and so on. Even when the code is basically in a nonlinear form (as is the case, for example, in the system due to Morgan[9] where the code is a standardized connection table), it would probably be convenient to convert to some linear form before using the code for information retrieval.

(2 and 3) The coding and decoding processes should yield unique results. These are absolute requirements for any reliable and unambiguous system. If they are not met, then it becomes possible to arrive at more than one code for a compound. If a chemist, searching a list for a particular compound, has arrived at a different code from that used by the compiler of the list, then the item in question will not be found. Similarly, if requirement 3 is not met, then it would not be possible to interpret a code unambiguously and to determine what compound the code stood for.

Failure to achieve this necessary uniqueness could arise from the coding algorithm being sensitive to the particular form in which the structural formula is presented to it or from a lack of clarity in the coding rules, whereby a rule could be interpreted in more than one way.

With some systems it is very difficult to be sure whether they possess attribute 2 or not. This is true, for example, of the "classical" chemical nomenclature (see, for example, ref 2 and 5). For simple straight-chain organic compounds with few substituents attached to the main hydrocarbon chain, the rules of classical chemical nomenclature work quite well. Even here, however, the number of nonsystematic items in the vocabulary of the system is quite large. For example, each basic chain has its own trivial name, and if one considers more complicated branched chain structures, the situation becomes worse. The rules require that such a compound should be described in terms of substitution in a certain basic straight-chain structure. This basic chain is defined as the longest straight chain present in the molecule; but locating this chain may be tedious. What is more, there may well be several candidates for the role of longest chain, so further rules are required in order to pick out the particular chain structure that will form the basis of the code of the compound. Even for acyclic structures this first step can be quite time consuming and not altogether straightforward to program on a computer. For cyclic compounds the situation is much more difficult. To handle them a large list of nonsystematic items is introduced, namely, the ring structures and their associated numberings.

With such a system it is not at all easy to answer the vital question "Will this set of rules always succeed in defining an unambiguous code, no matter to what compound it is applied?"

(4 and 5) Coding and decoding should be simple, and preferably performable by hand. It is clearly a desirable attribute that coding and decoding should be as simple as possible. It would be unreasonable, however, to expect a system to be so simple that the coding of *any* compound could be carried out by hand. The size and complexity of many of the chemical compounds that are known today precludes this possibility. Nevertheless, it is reasonable to ask that the only factor that would prevent coding by hand should be the size of the compound in question, rather than any intrinsic difficulties in the coding process itself. In other words, a good coding system should be capable of being used, without a computer, on any compound with not too many atoms.

One would expect that decoding would usually be easier than coding, if for no other reason than that it requires less in the way of safeguards to ensure a unique result. Thus we may reasonably ask that decoding should be feasible by hand for a range of compounds at least as large as those that can be coded by hand.

(6) The coding process should not depend on chemical intuition or on knowledge of the properties of chemical compounds. The number of compounds that have to be dealt with in an information retrieval context is so large that the coding and decoding processes must be capable of being carried out

on computers. It would therefore be an embarrassment to a system of nomenclature if, shall we say, the decoding rules gave two possible outcomes and if this ambiguity had to be resolved by an intuitive judgment (perhaps to the effect that one of the results did not "make sense"). Although the study of artificial intelligence progresses apace, it would be altogether too impractical to simulate chemical intuition on a computer. Hence the present requirement.

What this means, essentially, is that once the structure of the molecule to be coded has been accurately specified, the coding algorithm should be able to proceed to completion by using *only* this information. What is meant by "accurately specified" may depend on the context for which the system is designed. In this paper it will be assumed that a compound is specified when its structural formula, or equivalently a connection table, is given. An important example of further information that might be required is that pertaining to the three-dimensional structure of the molecule. This raises the question of stereoisomerism, which will be briefly discussed later.

Computer-oriented nomenclature systems can be divided into two classes: those that are extensions of classical nomenclature with additional ordering rules to avoid the intuitive questions and those that are essentially graph-theoretical exercises.

Among nomenclatures in the first class is the IUPAC system,[6] the history of which dates back to before the large-scale application of computers. Although its vocabulary is strange, it is a conventional approach and can be learned quickly by the practicing chemist. The main drawback of this and related approaches is the essential complexity of the coding process. In general there is no proof that the coding algorithm is effective. The codes produced by the IUPAC system are reasonably concise, usually being shorter than those produced by the classical nomenclature.

Systems belonging to the second class usually make no attempt to produce codes that are concise, pronounceable, or immediately intelligible (attributes 8, 9, and 11). Instead, they concentrate on the efficiency and simplicity of the coding and decoding processes. One satisfactory system of this kind is that due to Morgan,[9] in which the central concept is the connection table (or adjacency matrix). The code consists of this table, together with information on the nature of the atoms and peculiarities of bonding. Essentially Morgan's coding algorithm chooses one particular connection table from the many that are possible for a given compound.

Morgan's system is suitable for fully automated applications, of the kind that was mentioned earlier in this section, but it has the disadvantage that the code is not immediately or even easily comprehensible.

(7) The coding process should not need to have recourse to any lists of names or other nonsystematic items. Consider a computer program to produce the classical name of a compound that is input to it, such as the compound whose classical name is the one given in example a above. Such a program would determine, from the input data alone, that the longest chain has four carbon atoms. It would then need to know that such a chain is known as "butane". This is an example of a nonsystematic item or arbitrary designation. The program cannot derive this information from the input data alone; it would have to consult some list of designations in order to find it. This might be time consuming and hence is to be avoided where possible. Accordingly, a desirable feature of a good coding system is that it should have few, if any, nonsystematic items. Most systems at present in use have a large number of such items.

The grave disadvantage of the presence of nonsystematic items is that it places limitations on the range of compounds that can be coded. Thus the classical nomenclature requires that the ring structure of a cyclic compound be given a name, but this name can be found only by looking up the structure in a list. Thus if a new compound is found with a ring structure unlike any previously known, the system cannot code that compound. The usual solution to this problem is to make some ad hoc modification to the system, say by adding the new ring structure to the list; but this is clearly unsatisfactory. Most systems of nomenclature that are at present in use are of this "open-ended" type. A system should ideally be "closed" in the sense that its rules suffice to produce a code for *any* compound presented to it, no matter how complicated or unusual it might be.

(8 and 9) Codes should be brief and pronounceable. These attributes are clearly Utopian and are incompatible with most of the other attributes and with the size of the corpus of known compounds. There are far too many compounds for them to all have brief names, even arbitrary ones. However, one can ask that codes should not be unnecessarily long.

The attribute of being pronounceable is best regarded as unattainable. Even the codes produced by the classical system can be described as "pronounceable" only by considerably stretching the meaning of that word! Many systems do not attempt to produce a code that can be pronounced (other than by pronouncing each symbol separately). Special pronounceable names are routinely given to compounds of particular importance, of course; but we are concerned here with designation systems that will cover *all* possible compounds.

(10) The symbols used in the code should be familiar. This requirement is mainly for convenience. It makes it easier to input coded information to a computer. Moreover, when codes are used in a noncomputerized setting, such as when they are quoted in scientific papers, it is clearly an advantage if they can be easily typed. The main symbols used in current systems that are not of this kind are superscript and subscript numerals. An ideal system would avoid these symbols (though they could be made optional provided this did not introduce ambiguity).

(11) The code should be easily comprehensible to chemists. There are probably no nomenclature systems for which the code always gives the chemist an immediate and clear picture of what the compound in question looks like, and it would be unreasonable to hope to find such a system. The output from the computer-oriented systems is usually quite opaque, requiring extensive paperwork (or a decoding program) to give the structural formula in a recognizable form. In each of the examples a–c above, some information about the molecule can be seen immediately by those familiar with the coding system, though even example a requires a certain amount of thought before the precise nature of the whole molecule becomes apparent.

Thus, although immediate full comprehension is too much to expect, it is not unreasonable to ask that, unless a system is designed to be used only in a fully automated context, the code that it produces should at least give *some* information about the compound immediately, without the need for any decoding.

(12) The coding and decoding algorithms should be efficient. This requirement was discussed in sections 2 and 3. As pointed out there, prospects for a polynomial algorithm for cyclic compounds do not appear to be bright.

Let us now review the extent to which these attributes might be combined in a coding system. For greater generality we shall assume that the system is not intended only for a computerized context but will also be used in documents (to be read or consulted) and that a certain amount of coding will be done by hand. The above considerations then show that such a system should have the following characteristics.

The code should be a linear string of familiar typewriter symbols. It should be uniquely defined by the coding algorithm given a suitable description of the input compound, and in the derivation of the code no chemical intuition or lists of nonsystematic items should be needed. The coding algorithm should be able to provide this unique code no matter what compound is input to it.

The code itself should convey some information about the compound even without any decoding being done. The decoding algorithm should always recover from the code all the input information. It should be possible to perform the coding and decoding processes by hand, except in so far as the sheer size of the molecule might make this infeasible. Finally, the coding algorithm should be polynomial, at least when applied to acyclic compounds. (Decoding is easier than coding, so one would expect the decoding algorithm to be always polynomial; indeed, it should be linear.)

The system that will be described in what follows, and in part 2 of this paper, fulfills these requirements. The codes can all be typed,[16] and, for the most part, the characters are the familiar atom designations and other symbols habitually used by chemists. As will be seen, each code is an example of a "line formula"; but whereas this term is normally used for any method of stringing out the formula of a molecule in some linear way (with no guarantee of uniqueness), the coding rules here force one particular line formula to be chosen as the code. In this way the uniqueness that is essential for data retrieval is grafted onto an already familiar method of presenting chemical formulas.

Coding and decoding can be proved to be unique, though the proofs will not be given here. (An outline of the proof for acyclic compounds is given in ref 8). Furthermore, the coding algorithm is closed; it works for any kind of input compound.

The coding process for acyclic compounds is simple, calling for no sophisticated graph-theoretical manipulations such as finding longest chains. The only information used is that concerning adjacency, and this is derived directly from the input data. The coding algorithm for acyclic compounds is polynomial; in fact, it can be made linear in the size of the input. The coding of cyclic compounds is, naturally, more complicated. Even so, it works directly with the information on adjacency. When applied to cyclic compounds, the coding algorithm will not, in general, be polynomial, but its overall performance is good, since the exponential part of the algorithm applies only to the coding of the ring structure, and for many compounds the ring structure is quite small even though the whole molecule may be large. Hence coding by hand is possible to a considerable extent. Decoding is much easier and is possible by hand virtually without restriction.

No chemical intuition is required to perform the coding, and no lists of names or other nonsystematic items are needed. Thus persons knowing no chemistry at all can code a compound, provided that they have learned the (comparatively few) rules.

The main advantage of this system as compared with others at present in use, such as IUPAC, for example, resides in its comparative simplicity, the absence of nonsystematic elements, and the fact that it has a "closed" set of rules. In consequence, it is quite easy to write a computer program that will encode *any* chemical compound and that will not need to be modified or updated even if markedly more complicated compounds make their appearance.

Such a program has already been written for the IBM 370 computer at the University of Waterloo. This program, written in FORTRAN, is reasonably short, about 850 lines of code, including comments, and it handles all compounds, both acyclic and cyclic. Tests on a sample of compounds having up to 50 atoms indicate that the time taken to code an acyclic compound works out to about 250 μs/atom.

The only limitations on the program at present are a restriction to just seven kinds of atoms (C, H, I, N, O, P, and S) and limits of 99 on the number of atoms in the molecule and of 150 on the number of bonds. These restrictions were imposed purely for convenience. The inclusion of the whole gamut of chemical elements would not appreciably affect the speed of the program, since this information is required only on input. Extension to larger molecules would require only changes to the input routines (which currently expect a two-digit label for the atoms) and the redimensioning of the appropriate arrays.

A program of this kind does not need a large or powerful computer. The program in question has also been compiled and run on a minicomputer (a PDP 11/20) and could almost certainly be implemented without much difficulty on a microcomputer, though I have not tried the experiment. Thus it would be within the reach of small establishments, or even of individual chemists, to obtain the unique code of any compound for which the structural formula was known. Granted the existence of lists of codes of compounds, either as appendices to books or as publications in their own right, the chemist can then be in a position to search for any compound whose structure is known, with the assurance that if the compound is in the list then he will find it and (perhaps more important) that if he does not find it then it is not there.

Silk[13] has made a critical appraisal of present nomenclature systems and has questioned whether the use of systematic names is really as important as it seems to be. His conclusion that "the time for devising a new, comprehensive system of organic nomenclature is past" seems unduly pessimistic, and one with which not all chemists will agree. The present system is offered in the belief that there *is* a need for a simple and comprehensive system of nomenclature and that the coding procedure here described meets both these requirements.

The codes produced by this system tend to be longer than those given by some other systems, partly because brevity has been regarded as less important than simplicity. It would be quite easy to add further rules in order to abbreviate the code, especially if requirements 6 and 7 above were relaxed, but such changes would be incidental to and not basic to the coding process. Brevity is not all that important for a computerized application, since storage space is not at such a premium as it once used to be. For hand computation the added length of the code is the price paid to keep the process simple. In any case, the codes are not unreasonably long.

The following sections contain a description of the new coding process for acyclic compounds. This is important in its own right (since there are many acyclic compounds) and also because it forms the first step in the coding of cyclic compounds. In order to avoid too much material in one paper, I shall give the description of the coding process for cyclic compounds separately in part 2.

(5) SOME ILLUSTRATIVE EXAMPLES

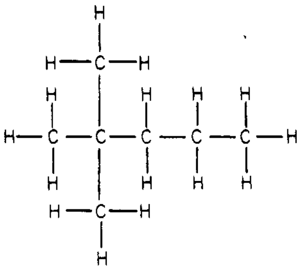
In order to describe the procedure for deriving the code of an acyclic compound, certain concepts, rules, and definitions will have to be introduced. However, the reader may not find the formal definitions very illuminating without some prior idea of the nature of the coding process. Accordingly, we first look at how the algorithm handles the coding of some specific compounds. As these examples are being coded, the relevant concepts and rules will be informally described, and the reasons for them will be explained. More formal definitions of the concepts and statements of the rules will be set out in sections 6–8, after the reader has already seen them in action. Finally, the coding and decoding algorithms will be formally set out in sections 9 and 10.

These illustrative examples will be considered from the point of view of someone carrying out the coding by hand, using pencil and paper. The coding process requires no involved concepts or computations, so it is quite easy to go from the hand-coding procedure to a computerized version of the same process.

There are two concepts that need to be introduced informally at this stage. The term "cluster" will denote an aggregation of atoms in a molecule which, for the purposes of the coding procedure, is regarded as a single unit. It is, so to speak, a "superatom", and the coding process handles it as if it were a single atom. Indeed, a cluster will often consist of just one atom. But however many atoms there are in a cluster, only one of them (called the "root") will be connected, by chemical bonds, to atoms outside the cluster.

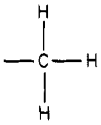
We recall that the "degree" of an atom, or of a cluster, is the number of other atoms or clusters to which it is connected. For an atom, therefore, the degree is what the valency becomes if we regard all bonds as single bonds.

Example 1. We are now ready to consider an example. The coding process starts with a drawing of the structural formula of the molecule in its most extended form, that is, with no abbreviations and with all atoms and bonds indicated. Such a drawing is that of **5.1**, the first example.



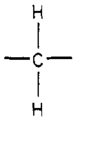
5.1

The coding algorithm consists of one basic operation, which is repeated over and over again until the desired code has been found. The first iteration of this operation is slightly different from those which follow and will be called the "preliminary iteration". To carry out this preliminary iteration, we note all atoms of degree 1, in this example the hydrogens, and the atoms to which they are attached. These latter atoms will be the roots of the clusters of atoms that we shall construct. Thereafter, these clusters will be treated in all essential respects as if they were single atoms. Thus we shall replace



by the cluster $-CH_3$

and replace

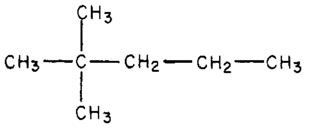


by the cluster $-CH_2-$

and so on.

In a string of symbols like "CH_3" and "CH_2", representing a cluster, the symbols must be written in a standard order. In the formal description of the coding procedure certain "ordering rules" will be given which specify the order in which symbols must be written to get the correct unique code, but we shall not discuss them in this informal presentation. We note, however, that the first symbol in a cluster must be the root, the one and only atom of the cluster which is connected to an atom in the rest of the molecule. Thus the valency bonds in $-CH_2-$ are understood to emanate from the carbon atom.

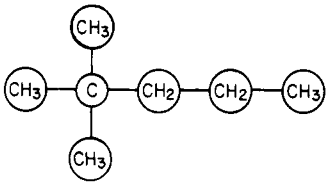
If we now replace each cluster by its appropriate symbol, we obtain **5.2**.



5.2

So far, this is all in accordance with standard chemical notation. In fact, one would not normally consider the extended form (**5.1**) of the structural formula but would start with something more like **5.2**. Care must be taken, however, to ensure that any contractions are in strict accordance with the coding rules. Here, for example, we must not write "H_3C" for the left-hand methyl group (as is occasionally done), since the root of a cluster must come first.

It is important to keep track of which atoms belong to which cluster, and to facilitate this we shall circle all clusters in the drawing of the molecule. Moreover, any atoms that are not connected to any atoms of degree 1 will be regarded as clusters and will also be circled. Thus we write structure **5.3**.



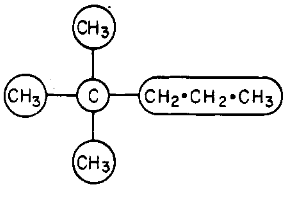
5.3

We now have a molecule consisting of seven clusters, and we proceed to the first "general" iteration. We look at each of the clusters of degree 1, the four methyl groups, and pick out the "chain" which it determines. To do this, we look at the cluster to which it is connected and see if this has degree 2. If it has, then we look at the cluster to which *it* is connected and so on until we come to a cluster which is not of degree 2. Thus, starting with the right-hand methyl group in **5.3**, we obtain a chain which we can write as

$$CH_2{\cdot}CH_2{\cdot}CH_3 \text{ or } (CH_2)_2{\cdot}CH_3$$

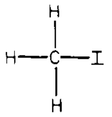
where it is understood that the free bond, connecting with the rest of the molecule, is with the cluster on the extreme left and that this bond is single unless otherwise specified. In the diagram **5.4** this chain is circled. The other chains consist of just one cluster each, written as "CH_3".

We now have a cluster consisting of a single carbon atom joined to four chains, namely



5.4

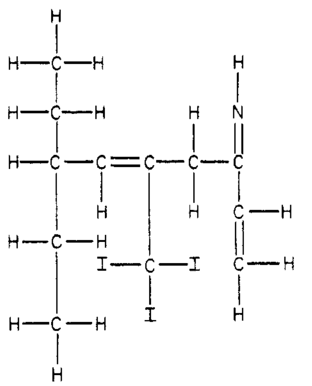
and just as we would simplify



to CH_3I

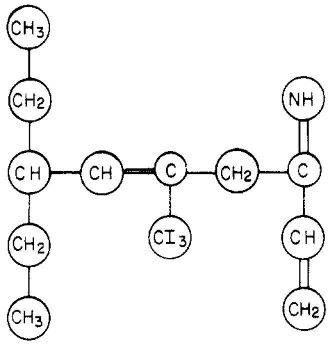
so we simplify **5.4** to $C(CH_3)_3((CH_2)_2{\cdot}CH_3)$ (**5.5**), which is the required code.

Example 2. In the first example there were no multiple bonds, so that "degree" was the same as "valency". In the present example we shall see how double bonds are handled. We consider the compound **5.6**



5.6

The preliminary iteration gives the clusters shown in **5.7**.



5.7

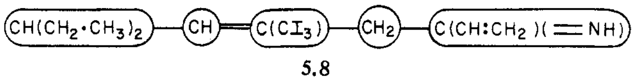
We look at the clusters of degree 1, which it will be convenient to call "end clusters"; they are the two CH_3 groups, the end CH_2 group, the CI_3 group, and the NH group. The chains determined by these clusters will be written as follows: $CH_2{\cdot}CH_3$, $CH_2{\cdot}CH_3$, $CH{:}CH_2$, CI_3, and =NH.

Here again we see that a chain may consist of a single cluster, as with the last two here. In the last chain we have indicated that the free bond is double by prefixing "=" to the notation for the chain. Note, however, that in the code for the third chain the double bond inside the chain is denoted by ":". This illustrates an important point, namely, that to avoid ambiguity it is necessary to have two ways of indicating multiple bonds. When such a bond occurs in a the middle of a chain, it is denoted by a "dot" bond, but when it is the bond connecting a chain or a cluster to the rest of the molecule, it is denoted by a "dash" bond, such as "=". However, as stated in the first example, a single bond in this position is assumed and not indicated. For this reason the single dash bond "–" never occurs in the code of an acyclic compound.

We now form the clusters rooted at the three carbon atoms of degree 3 to which the above chains are connected; they are $CH(CH_2{\cdot}CH_3)_2$, $C(CI_3)$, and $C(CH{:}CH_2)({=}NH)$.

The order of the chains in the last cluster is determined by an ordering rule (rule 1a) by which chains attached by a single bond precede those attached by a double bond.

The molecule now looks like **5.8**.



5.8

If we now repeat the process and look at the end clusters and the chains they define, we see that the whole molecule is one *closed* chain, i.e., terminated at *both* ends by a cluster of degree 1. No extra parentheses are required in writing down the expression for a chain, except possibly for repetitions of a cluster, since the dots for valency bonds separate the clusters making up the chain. So we could now write the whole molecule as **5.9**. But we could just as well start from the other end of the chain, and obtain **5.10**. Which coding is the correct one?

$$CH(CH_2{\cdot}CH_3)_2{\cdot}CH{:}C(CI_3){\cdot}CH_2{\cdot}C(CH{:}CH_2)({=}NH)$$

5.9

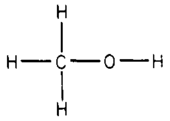
$$C(CH{:}CH_2)({=}NH){\cdot}CH_2{\cdot}C(CI_3){:}CH{\cdot}CH(CH_2{\cdot}CH_3)_2$$

5.10

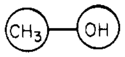
An ordering rule, given in more detail later, says that the larger of the two end clusters (more specifically, the one with the longer code) should appear on the left. This implies that the second of the above, viz **5.10**, is the correct code for this example.

We have already remarked that the preliminary iteration of the coding procedure is slightly different from those which follow. This difference is that in the preliminary iteration we do not allow the formation of chains of more than one cluster. In other words, if an atom of degree 1 is joined to an atom of degree 2, we do *not* go past this second atom to the atom beyond; the atom of degree 2 becomes the root of a cluster. Some examples will illustrate the difference that this makes.

In coding methanol

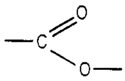


the preliminary iteration gives us

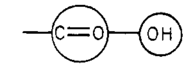


whence we obtain the code $CH_3{\cdot}OH$. Had we allowed the formation of chains during the preliminary iteration, thus allowing the formation of the chain O·H we would obtain $CH_3(O{\cdot}H)$ as the code.

This special treatment of the first iteration needs justification. The main reason for it is that it results in codes that are more elegant and more chemically appealing. For example, $CH_3{\cdot}OH$ is more acceptable than $CH_3(O{\cdot}H)$. Again, if the preliminary iteration were the same as the others, the carboxyl radical



would give rise to two chains, namely, =O and O·H. The same rule 1a mentioned above would then determine $-C(O{\cdot}H){=}O$ as the code for this radical. As it is, however, we first get the chain

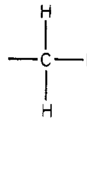


which then gives $-C{=}O{\cdot}OH$, a much closer approximation to the conventional –COOH. Note that the radical –N=O remains unchanged (we cannot write it as –NO since this would imply a monovalent oxygen atom and a divalent nitrogen).

Although this is an informal presentation, and no proper definitions have been given, it can be seen that the general form of notation for a cluster is something like **5.11**, where R is the root and X, Y, and Z are the codes (in some prescribed order) for the chains connected to R. But if this were always so, we would have to write

$$R(X)(Y)(Z)$$

5.11

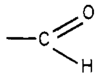


as $C(H)(H)(H)$ or as $C(H)_3$

and in either case we would have an unfamiliar symbolism with superfluous parentheses. For this reason, when a chain consists of a single atom, we dispense with the parentheses, and the general form for a cluster is thus

$$Rabc...(X)(Y)(Z)...$$

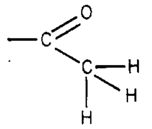
where a, b, c, ... are the symbols for the single atoms (if any) connected to R. It is important to note that the rule that a chain consisting of a single atom need not be enclosed in parentheses when incorporated into a cluster remains true even when this atom is connected by a double bond and that, just as with chains in general, this bonding is indicated by prefixing "=" to the symbol for the atom. Thus for the aldehyde radical



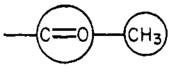
we have two chains H and =O; and by virtue of rule 1a again, the cluster is coded as $CH{=}O$. The symbol "=", when not preceded by a left parenthesis, must be taken with the symbol to its right as a single unit. A further example is provided by the carboxyl radical which, as we have just seen, is coded as $C{=}O{\cdot}OH$.

These simple examples indicate that care must be exercised to obey the rules to the letter, since the temptation to write the more conventional "CHO" and "COOH" for these radicals would be strong. Note that there is no ambiguity in the notation $CH{=}O$. It could not represent $-C{-}H{=}O$. For even if such a combination were possible, it would be coded differently, as $C(H{=}O)$.

It is instructive to see what happens if the hydrogen in the aldehyde radical is replaced by a methyl group. The result is

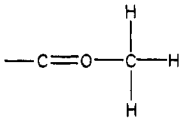


which becomes



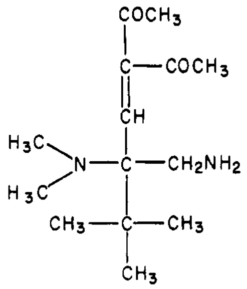
after the preliminary iteration and becomes the chain $C{=}O{\cdot}CH_3$ after the second. This illustrates an important point; namely, that if one compound is obtained from another by some kind of substitution, for example, by replacing a hydrogen atom by a monovalent radical, then it is the exception rather than the rule that the two codes will differ only by a similar substitution. Generally, any kind of substitution upsets the timing of the formation of chains and clusters, and a code of a completely different form may result. This apparent defect in the system is not really a defect; it can be shown that it is inevitable in any coding system.

Note that this last chain, $C{=}O{\cdot}CH_3$, is not the same as $C{:}O{\cdot}CH_3$. The latter code is not very plausible, chemically speaking, since it would stand for the chain

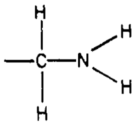


Nevertheless, it is a basic tenet of this coding system that no appeal needs to be made to chemical intuition or feasibility in order to resolve ambiguities. This example shows why two ways of indicating double bonds (: or = according to circumstances) are necessary to avoid ambiguity.

Example 3. Suppose we are given the structural formula **5.12.** We note first that some of the abbreviations are not in accordance with the rules: this must be remedied. If in doubt, we go back to the extended structural formula and we find, for example, that **5.13** will become $-CH_2{-}NH_2$ at the preliminary iteration of the coding procedure.

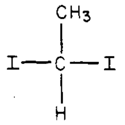


5.12



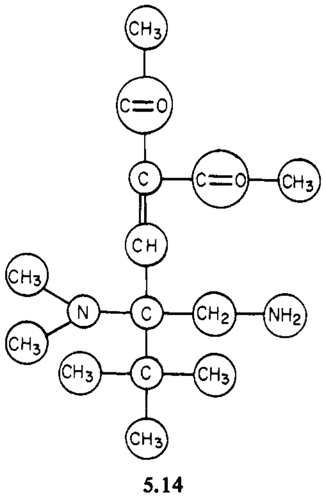
5.13

It should be remarked here that the coding rules are simple and easy to apply. The main danger is that chemical intuition may tempt us to break them. As a simple example, if we fail to look at the extended structural formula for a compound given as



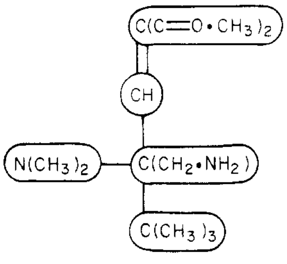
we might code it as $CHI_2(CH_3)$. This would be wrong, since the first iteration of the coding procedure should determine the clusters CH_3 and CHI_2. This gives a closed chain of two clusters, and the correct coding for this compound is therefore $CHI_2{\cdot}CH_3$. Although the two codes are not so very different, it is essential for the purposes of information retrieval that the coding algorithm should yield a unique code. This it will always do, provided the rules are scrupulously observed.

Bearing these points in mind we return to our example and see that after the preliminary iteration we have **5.14**.



5.14

In the next iteration of the coding process we identify eight chains: two, coded as $C{=}O{\cdot}CH_3$, attached to the same carbon atom, three methyl groups attached to another carbon atom, two more methyl groups attached to a nitrogen atom, and the chain $CH_2{\cdot}NH_2$ attached to a carbon atom. Forming the clusters rooted at the four above-mentioned atoms, we obtain **5.15**.



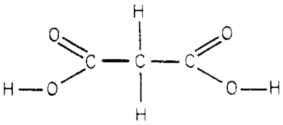
5.15

We now have three chains converging on the cluster $C(CH_2{\cdot}NH_2)$. The chains are $CH{:}C(C{=}O{\cdot}CH_3)_2$, $N(CH_3)_2$, and $C(CH_3)_3$. Hence the final cluster (which is the whole molecule) has the code

$$\underbrace{C(CH_2{\cdot}NH_2)}_{\text{root cluster}}\underbrace{(C(CH_3)_3)}_{\text{chain 1}}\underbrace{(N(CH_3)_2)}_{\text{chain 2}}\underbrace{(CH{:}C(C{=}O{\cdot}CH_3)_2)}_{\text{chain 3}}$$

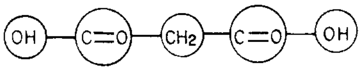
The correct order for writing the chains in a cluster is determined by rules that will be given later.

Example 4. One further feature of the coding system remains to be illustrated. Consider the formula **5.16**



5.16

After the preliminary iteration this becomes



which is a closed chain. When we have a closed chain we have to decide from which end to start writing the code. In the course of making this decision for the present chain we discover that it is symmetrical about the central cluster. When this happens, we can regard the chain as a cluster, rooted at the central cluster. That is to say, instead of

$$OH{\cdot}C{=}O{\cdot}CH_2{\cdot}C{=}O{\cdot}OH$$

we can write the shorter, and more elegant code

$$CH_2(C{=}O{\cdot}OH)_2$$

Decoding. The procedure for decoding the code of a compound, that is, recovering the structural formula, is simply the reverse of the coding procedure, but it is easier, since one does not need any rules to guarantee uniqueness. A formal algorithm for it will be given later, but the general principles are easily described.

The code of a cluster starts with the symbol for the root (possibly preceded by a multiple bond, e.g., "="). Then follow the symbols for the single atoms (if any) connected to the root (these may also be preceded by bond symbols) and a number of expressions in parentheses. These expressions are chains, the clusters of which are separated by dot bonds. Thus clusters can be decoded in terms of shorter chains, and chains can be decoded in terms of shorter clusters.

In this way, the whole code (which must be in the form either of a closed cluster or of a closed chain) is eventually expressed in terms of clusters which are single atoms, and all connections between the atoms are thereby determined.

(6) DEFINITIONS

Before starting the rules for the various parts of the coding procedure we must first define precisely the concepts, terms, and symbols to be used.

Alphabet. The symbols that may be used in the code of a chemical compound are of five categories, as follows. (a) The usual symbols for the elements, e.g., C, H, Br, Cl, etc. These symbols are treated, and counted, as single symbols, even when they contain two letters (Br, Cl, etc.). (b) The integers 1, 2, 3, ... In deference to common chemical practice these will be written here as subscripts; but they can just as well be placed on the line if necessary and would normally be so placed in computer output. An integer consisting of two or more digits is still treated as a *single* symbol. (c) Dot bonds, e.g., ·, :, ⋮, etc. (d) Dash bonds, e.g., =, ≡, etc. (e) Parentheses, e.g., (and).

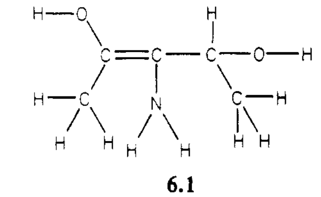
The reason for the two methods (dot and dash) of depicting valency bonds has already been explained. A single dash "–" is not used in the final code of a compound.

Note that almost all these symbols are available on typewriters and computer printers. The only exceptions are the triple (or more) dot and dash bonds. These have been retained because of their familiarity, but one can easily substitute something else for them such as $\underline{:}$ and $\underline{=}$.

Ordering the Symbols. The above symbols have a particular order associated with them. Two symbols of different categories are ordered according to the order (a, b, c, d, e) of the categories detailed above. Within these five categories the orderings are respectively as follows: (a) atom symbols, lexicographic or dictionary ordering, that is, by the first (or only) letters, if these are different and by the second letter if the first letters are the same; (b) integers, increasing order, i.e., 1, 2, 3, 4, ...; (c) dot bonds, by increasing valency; (d) dash bonds, also by increasing valency; (e) parentheses, the order is left parenthesis, right parenthesis.

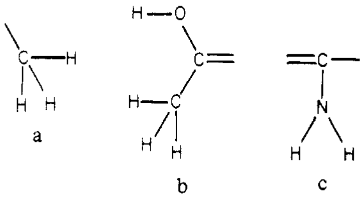
Degree. The degree of an atom is the number of other atoms to which it is connected. Alternatively, it is the number of bonds at the atom, irrespective of whether these are single or multiple. The degree of a cluster is defined below.

Cluster. A cluster may be roughly defined as a connected portion of a chemical compound having the property that at most one atom of the cluster is connected to an atom (or atoms) not in the cluster. In the compound

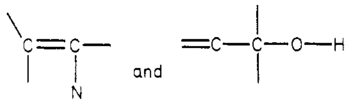


6.1

the portions



are clusters, whereas



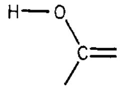
are not. In a cluster, the unique atom having connections with one or more other atoms elsewhere in the molecule will be known as the "root". The number of atoms, not in the cluster, to which the root is connected will be called the "degree" of the cluster. Thus the degrees of the clusters a–c above are 1, 1, and 2, respectively. A cluster of degree 1 will be called an "end cluster".

A more formal, recursive definition of the term cluster is as follows.

(1) Each atom of the molecule is a cluster. The root of such a cluster is the atom itself, and the degree of the cluster is the degree of the atom, that is, the number of other atoms to which it is connected.

(2) If X is a cluster having root R and degree d and if R is connected to an atom which is the root of an end cluster Y, that is, a cluster of degree 1, then the portion of the molecule consisting of X, Y, and the bond between their roots is also a cluster. Its root is R and its degree is $d - 1$.

Note that many of the clusters defined in this way will not appear during the course of the coding algorithm, since the algorithm will often incorporate several end clusters simultaneously into a larger cluster. Thus, for example, the portion



of the molecule of **6.1** conforms to the definition (with Y as the hydroxyl group) and is therefore a cluster. However, when the coding algorithm is applied, the methyl group connected to this carbon atom will become an end cluster at the same time as the hydroxyl group, and the algorithm will insist on incorporating *both* end clusters into the larger cluster rooted at the carbon atom.

Closed Clusters. A closed cluster is a cluster whose root is not connected to any atom not in the cluster; in other words, it is a cluster having degree zero. Such a cluster is therefore a complete molecule and will occur only when the algorithm terminates.

Chain. A chain is a sequence of clusters, say $C^{(1)}$, $C^{(2)}$, ..., $C^{(k)}$, the last of which, $C^{(k)}$ (the end of the chain), has degree 1 while the remainder have degree 2 and where the root of $C^{(i)}$ is connected to the root of $C^{(i+1)}$ ($i = 1, 2, ..., k - 1$). The root of $C^{(1)}$ will be connected to exactly one atom in the rest of the molecule (this follows since the degree of $C^{(1)}$ is 2).

Every end cluster of a molecule determines a chain of which it is the end. For we merely have to trace backward from the end cluster, through clusters of degree 2, until we come to a cluster of degree 3 or more. These clusters, excluding the latter, constitute the chain determined by the given end cluster.

Note that a chain may consist of only one cluster.

Closed Chain. A closed chain is a sequence of clusters $C^{(0)}$, $C^{(1)}$, $C^{(2)}$, ..., $C^{(k)}$ in which the root of $C^{(i)}$ is connected to the root of $C^{(i+1)}$ ($i = 0, 1, ..., k - 1$), $C^{(0)}$ and $C^{(k)}$ are of degree 1, i.e., are end clusters, and the remaining clusters are of degree 2. Clearly such a closed chain is a complete molecule in itself; it will occur only when the algorithm is about to terminate.

(7) CODING OF CLUSTERS AND CHAINS

We now give the definitions of the code of a cluster and of a chain. These will be defined recursively in terms of the codes of smaller chains or clusters. The definitions will call for clusters and chains to be ordered in a standard manner, but the standard ordering will not be defined here. It will be specified in the next section.

Chain Coding. The code for a chain has the general form of **7.1**, $eC^{(1)}dC^{(2)}dC^{(3)}d$ ... $dC^{(k)}$ where $C^{(1)}$, $C^{(2)}$, ..., $C^{(k)}$ are the codes for the clusters, and the d's represent the symbols for the bonds (dot bonds) between them. The nature of the "free" bond to the rest of the molecule is given by the symbol whose location is indicated by the "e". To obtain the coding for the chain of **7.1**, we replace each "d" by the appropriate dot bond. If "e" represents a single bond, then we do not indicate this, but if it is a multiple bond, then we replace "e" by the appropriate *dash* bond. Thus dots are used for valency bonds inside a chain, while dashes are used for bonds between the chain and the rest of the molecule.

It may be possible to contract the code for a chain if the chain contains repeated clusters. If this can be done, it must be done, and the rules for doing so will be given below.

A closed chain is coded in much the same way as an ordinary chain except that no bond "e" will appear or be implied in front of the left-most cluster. A closed chain can usually be coded in two ways according to which of the two end clusters is taken on the left. The rule for choosing one of these possibilities in preference to the other is given below.

Cluster Coding. Clusters are formed during the course of the algorithm by incorporating together a root atom or cluster and a number of chains connected to it. The code for a typical cluster is

$$\mathrm{Rabc} \; ... \; (\mathrm{P}^{(1)})(\mathrm{P}^{(2)})(\mathrm{P}^{(3)}) \; ...$$

Here R represents the symbol for the root atom, and a, b, c, ... denote the symbols for single atoms (if any) connected to R. $P^{(1)}$, $P^{(2)}$, $P^{(3)}$, ... are the codes for the chains which are attached to R. These atoms and chains must be written in a prescribed order specified by the ordering rules given below. Note that any single atoms to which R is connected are, in this context, chains. Hence if they are attached to R by a multiple bond they must be preceded by a dash bond. A dash bond used in this way is recognizable because it does not follow a left parenthesis, and since it and the atom symbol which follows it are regarded as forming one atom symbol, there is no need to place these two symbols in parentheses.

The code of a closed cluster is formed in exactly the same way.

(8) ORDERING AND CONTRACTION RULES

In the construction of a cluster during an iteration, the code of the root cluster must be followed by the codes of the chains written in the correct order. This order is specified by the following rule.

Rule 1. Order of Chains in a Cluster. (a) Chains attached to the root by an i-ple bond precede those attached by a j-ple bond, where $i < j$. Thus those with a single bond come first, then those with a double bond, and so on.

(b) For chains that are similarly bonded to the root, shorter chains precede longer ones. (The length of a chain is the number of symbols in its code. Remember that atom symbols like "Br" count as *one* symbol.)

(c) For chains that are similarly bonded to the root and are of the same length, the order is lexicographic, or dictionary, order. That is, the codes of two chains are compared, symbol by symbol, from left to right until the symbols are different. The chain with the earlier symbol takes precedence. (Example: $C(C(CH_3)_3)$ precedes $C(C(NH_2)_3)$ since in the fifth place, where they first differ, C precedes N.)

Note that rule 1 applies to the ordering of chains during a particular iteration of the algorithm. When a cluster is formed during a particular iteration, the code for the root cluster is written first, and this will include all the chains added to the root atom in previous iterations. The added chains will follow all these. Hence, automatically, a chain added in one iteration will follow any chain added in a previous iteration irrespective of their length, etc. In particular, we note that chains consisting of single atoms will immediately follow the root, since they must have been added during the preliminary iteration.

Thus if we look at all the chains that follow the symbol for the root in the code of a cluster, their order is determined by the following criteria, in order of priority: (i) iteration, (ii) bonding, (iii) length, (iv) lexicographic order.

As an example of criterion i, refer back to the coding of the carboxyl group. It was –C=O·OH. At first sight this may appear to contradict the rule that single bonds precede double bonds, but this is not so, for the cluster C=O was formed during the preliminary iteration; the chain OH (consisting of a single cluster, also formed during the preliminary iteration) was added to the root cluster during the next iteration. Thus the coding process itself automatically takes care of point i above; only ii–iv need to be invoked by the coder to ensure the correct ordering of the chains when a new cluster is being formed.

It is assumed in the above that the codes for the chains have been contracted according to rule 4 given below.

The formation of chains is governed by rule 2.

Rule 2. Order of Clusters in a Chain. The clusters in a chain are written from right to left in sequence, beginning with the cluster which is attached to the rest of the molecule and ending with the cluster of degree 1.

This merely repeats what was said under the heading Chain Coding above. With ordinary chains there are no ambiguities to be resolved.

In a closed chain, however, the clusters are ordered from one end to the other in sequence, and there are therefore two possible codings according to which end appears on the left. This ambiguity is resolved by the following rule.

Rule 3. To determine which of the two possible codes for a closed chain is the correct one, compare the clusters at the two ends. If they are of different lengths, then the correct code is the one which has the longer cluster on the left. If of the same length, then they are compared lexicographically, and the coding for which the left-hand cluster comes *after* the right-hand cluster is chosen. If the end clusters are identical, compare the bonds that connect them to the rest of the molecule. These will normally be identical in any chemically feasible molecule, but if they are different, the chain is oriented so that the bond of higher multiplicity is to the left. If the bonds are the same, examine the clusters adjacent to those previously compared, and compare them in the same way. If these are the same, compare the associated bonds, then the next clusters, and so on, working in from each end of the closed chain.

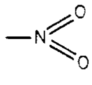
This rule will always distinguish the two possibilities, provided they are different. If the closed chain is its own mirror image (treating the clusters as units), this will be discovered in the course of the above examination. If so, then there may be a central cluster (if the number of clusters in the chain is odd), in which case we code the whole molecule as a cluster having the central cluster as root and two identical chains (see section 3, example 4). If the chain has an even number of clusters we leave the code as it is. Rule 3 is illustrated by the choice between **5.6** and **5.7** given earlier.

To conclude this section, we give two rules for contracting the code of a cluster or chain when there are multiple occurrences of some of its parts.

Rule 4. Contraction Rule for Clusters. This is used when, on applying rule 1c above (or simply on inspection), two or more chains are found to be identical.

If a chain occurs *r* times in a cluster, these *r* occurrences can be replaced by one occurrence, and a subscript *r* placed after the right-hand parentesis which follows the code of the chain or, if the chain is a single atom, after the symbol for the atom. Thus $C(NH_2)(NH_2)(NH_2)$ is replaced by $C(NH_2)_3$ and CHHH by CH_3.

Note that no parentheses are needed for a single atom and that this applies even when the atom has a multiple bond. Thus



becomes N═O═O which contracts to $N{=}O_2$.

Rule 5. Contraction Rule for Chains. The sequence $\cdot X \cdot X \cdot X \cdot \ldots \cdot X \cdot$ (**8.1**), in which cluster X occurs *r* times, can be replaced by $\cdot (X)_r \cdot$, provided that this repetition is maximal. That is, there is no other cluster X with another single bond to the right or to the left of **8.1**.

Comments. Note that according to this rule the enclosing of the repeated cluster in parentheses applies even when the cluster is a single atom. Thus $-O-O-CH_3$ would become $(O)_2 \cdot CH_3$.

The repeated cluster must have a dot bond on either side of it, though the one on the left can be the implied bond at the beginning of a cluster in which case it will not actually appear.

At present the system does not allow this kind of contraction unless all the bonds involved are of the same kind. Thus it does not contract ... –CH═CH—CH═CH—CH═CH– ... to ... $-(CH)_6-$ ..., the reason being that the code is not distinguishable from that of ... -CH-CH-CH-CH-CH-CH- ... and that this ambiguity cannot be resolved without appealing to chemical knowledge (the fact that the carbons are tetravalent). The system could undoubtedly be modified so as to permit this very useful kind of contraction, but in the interests of simplicity it will be left as it is for now.

Finally, one must not "cheat" by splitting an existing cluster in order to gain an additional replication. Thus $CH_2 \cdot CH_2 \cdot CH_2 \cdot CH_3$ must be contracted to $(CH_2)_3 \cdot CH_3$ and not to $(CH_2)_4 \cdot H$!

The above definitions and rules govern the way in which clusters and chains are constructed and coded. All that now remains is to present the algorithm itself.

(9) CODING ALGORITHM

Step 1. Start with the structural formula for the molecule. (Comment: This could be a drawing (for a hand computation) or something like a connection table for a computer program. The only information needed is that concerning which atoms are present, in what numbers, and which pairs of atoms are connected by which kind of bond.)

Step 2. The Preliminary Iteration. Each atom of degree 2 or more becomes the root of a cluster. Each such cluster consists of the root together with any atoms of degree 1 to which it is connected. The code for a cluster consists of the symbol for the root followed by the symbols of the other atoms, in the order prescribed by rule 1 and contracted by rule 4 where applicable.

If there is only one cluster, then this is the whole molecule, and its code is the code obtained for the cluster. (Comment: This can happen only for very simple molecules, for example methane, CH_4, formaldehyde, $CH_2{=}O$, etc.)

If there is more than one cluster, continue with step 3. (Comment: The molecule now consists of a number of clusters connected by chemical bonds. Two clusters are joined by an *r*-ple bond if and only if their root atoms are joined by an *r*-ple bond in the original molecule.)

Step 3. The General Iteration. (a) Determine all end-clusters. (b) For each end cluster identify the chain that it determines by tracing this chain through clusters of degree 2 until a cluster, X, of some other degree is encountered. (Comment: The chain may consist of the end cluster alone.)

If cluster X is of degree 1, then the whole molecule is a single closed chain. The code for the molecule is then found from rule 3.

If cluster X has degree 3 or more, this cluster is a "new root". Determine the code of the chain by rule 2.

(c) Construct a new cluster at each new root. The cluster consists of the new root, together with the chains attached to it. The code for the new cluster consists of the code for the root followed by the codes for the chains, taken in the order given by rule 1, each enclosed in parentheses and contracted by rule 4 when there are multiple occurrences of a chain.

(d) Clusters that are not new roots, i.e., at which no chain was attached, remain clusters as before.

(e) If there is now only one cluster, this cluster is the whole molecule and its code is the required code for the molecule.

(f) If there is more than one cluster, repeat the general iteration (step 3). (Comment: The molecule is now expressed in terms of the new clusters constructed in c and the clusters remaining, as in d, with the connections between them determined as before.)

End of the algorithm.

It is clear that this algorithm will terminate. If at some stage there are no clusters of degree 3 or more, then the molecule is a chain of clusters, and its code is determined. In the contrary case there will be some end clusters (at least three), and at least one new cluster will be formed. This will decrease the total number of clusters. Since the number of clusters steadily decreases, the algorithm must terminate.

Furthermore, the code produced by the algorithm is unique. The only information used is that of the connections between the atoms in the molecule. These determine the connections between the clusters at each stage, and the codes for the clusters are uniquely determined by the rules for coding chains and clusters. In short, at no stage is there any choice as to what to do next. Hence, whether carried out by hand or by a computer program, the final result is uniquely determined.

It still remains to show that decoding is unique, i.e. that we cannot have the situation where two different molecules give rise to the same code. To this end we give, in the next section, an algorithm for the decoding process, which recovers the structural formula, connection table, or equivalent information, from the code.

(10) DECODING ALGORITHM

We now state the algorithm for decoding the code of a chemical compound; that is, retrieving the structural formula from it. This will be done in two stages. First comes an algorithm for identifying the root and the chains in a given cluster (which may be the whole molecule). Then comes an algorithm for identifying the clusters that make up a chain. By the use of these two algorithms the code of the compound can be systematically expanded into a structural formula.

Decoding Algorithm for Clusters. (1) Note the first (leftmost) symbol of the code. This will be the symbol of the root atom.

(2) Note the symbols which follow until either a left parenthesis, a dot bond, or the end of the code is reached. These symbols will be atom symbols, integers, or dash bonds. Each atom thus designated is connected to the root atom by the preceding (dash) bond, if there is one, or by a single bond if there is not. A subscript *n* denotes that *n* replicas of the preceding atom are joined to the root in the indicated manner.

(3) If the end of the code has been reached, this algorithm terminates.

(4) If a dot bond has been reached, this algorithm terminates. The cluster in question belongs to a chain, the next cluster of which lies beyond the dot bond.

(5) If a left parenthesis is reached, find the corresponding right parenthesis. What is contained between these parentheses is a chain which is connected to the root atom in the indicated way.

(6) Look at the next symbol. If it is not an integer, carry on from step 3. If it is an integer, *n*, then *n* chains of the form just noted are connected to the root atom. Look at the next symbol and carry on from step 3.

This algorithm isolates the chains connected to the root atom. These chains must now be decoded in turn. Since the single-atom chains have already been dealt with, we need only consider those which are enclosed in parentheses and shall assume that these parentheses have been removed.

Decoding Algorithm for Chains. (1) Look at the first symbol. If it is a dash bond, then the chain is connected to the root by the indicated bonding. Otherwise the connection is understood to be a single bond. If a dash bond was present, look at the following symbol.

(2) The symbol now being looked at is either a left parenthesis or an atom symbol. If it is an atom symbol carry on with step 3. If it is a left parenthesis, find the corresponding right parenthesis: this will be followed by an integer. Note this integer, and then delete it and the two parentheses from the code. Look at the symbol following the (now deleted) left parenthesis and continue with step 3.

(3) This symbol is the root of the first (or next) cluster of the chain. The end of this cluster can be identified by the cluster-decoding algorithm.

(4) If the cluster ended at the end of the code, this algorithm terminates.

(5) If the cluster ended at a dot bond, we look at the integer (if any) noted in step 2. If no integer was noted, then the root of the cluster just isolated is connected by the indicated bond to the root of the next cluster of the chain. If an integer *n* was noted in step 2, we attach a chain of $n - 1$ replicas of the cluster (just noted) to the original, with single bonds between the roots. The last of these replicas is connected (by a single bond) to the root atom of the next cluster of the chain. In either case the root of the next cluster in the chain is given by the symbol following the dot bond which indicated the end of the cluster. Look at this symbol and carry on from step 2.

This algorithm isolates the clusters in a chain and determines the bonds between them. By alternating these two algorithms, the decoding of clusters and chains can eventually be reduced to that of clusters or chains of single atoms, the decoding of which is trivial.

A Note on Starting the Decoding Algorithm. The complete code of a compound has the form of either a closed chain or a closed cluster. A closed chain can be decoded exactly as for an ordinary chain, the sole difference being that the left-most cluster will not have a free bond. Similarly, a closed cluster can be decoded as if it were an ordinary cluster. However, it may not be immediately obvious whether a given code is that of a closed chain or a closed cluster. The solution is to start decoding as for a closed cluster. If the end of the first cluster (the one that would be the root of the closed cluster) is signaled by a dot bond, this indicates that this cluster is the first in a closed chain, and the decoding can proceed accordingly.

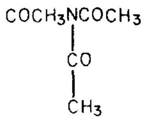
(11) NOTE ON THE QUESTION OF STEREOISOMERISM

In what has gone before, the starting point has been a description of the molecule which specifies merely which atoms are connected to which others. The coding system described in this paper is designed to do no more than to codify this information into a unique linear string of symbols that can, in turn, be decoded so as to give back this original information. Hence this system cannot distinguish stereoisomers from one another, since the information that it uses is not sufficient to allow this to be done.

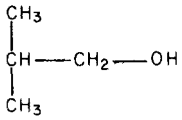
Nevertheless, it would be highly desirable to have a system that could distinguish between stereoisomers if required, and one naturally asks whether the present system could be expanded so that it can. It is not intended to discuss this matter here in any detail—the main purpose of this paper is to set out the basic system, without too much discussion of what might be added to it later—but a few observations may be of interest.

Consider an asymmetric carbon atom in a molecule. The symbols for the atoms that are bonded to this carbon will all

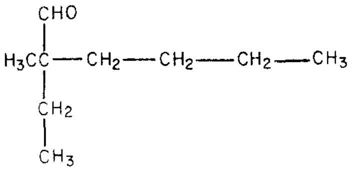
Chart I



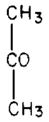
1



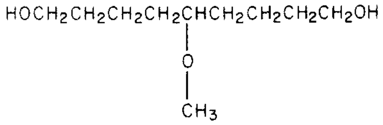
2



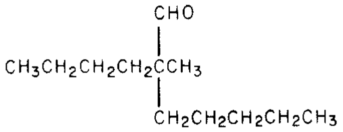
3



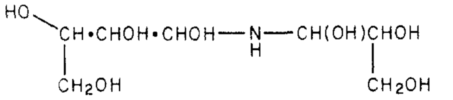
4



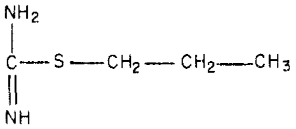
5



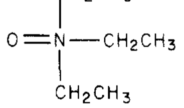
6



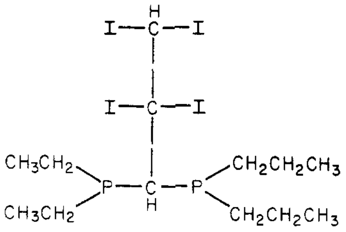
7



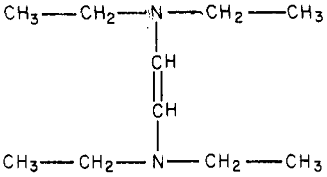
8



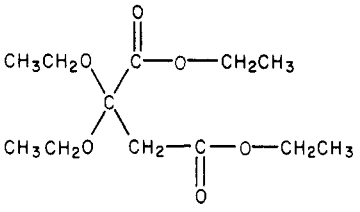
9



10



11



12

Solutions

1	$N(C{=}O{\cdot}CH_3)_3$
2	$CH(CH_3)_2(CH_2{\cdot}OH)$
3	$C(CH_3)(CH{=}O)(CH_2{\cdot}CH_3)((CH_2)_3{\cdot}CH_3)$
4	$C{=}O(CH_3)_2$
5	$CH(O{\cdot}CH_3)((CH_2)_4{\cdot}CH)_2$
6	$C(CH_3)(CH{=}O)((CH_2)_3{\cdot}CH_3)((CH_2)_4{\cdot}CH_3)$
7	$CH(OH)(CH_2{\cdot}OH){\cdot}(CH(OH))_2{\cdot}NH{\cdot}CH(OH){\cdot}CH(OH)(CH_2{\cdot}OH)$
8	$C(NH_2)({=}NH)(S{\cdot}(CH_2)_2{\cdot}CH_3)$
9	$N{=}O(CH_2{\cdot}CH_3)_3$
10	$P((CH_2)_2{\cdot}CH_3)_2{\cdot}CH(CI_2{\cdot}CHI_2){\cdot}P(CH_2{\cdot}CH_3)_2$
11	$N(CH_2{\cdot}CH_3)_2{\cdot}CH{:}CH{\cdot}N(CH_2{\cdot}CH_3)_2$
12	$C(O{\cdot}CH_2{\cdot}CH_3)_2(C{=}O{\cdot}O{\cdot}CH_2{\cdot}CH_3)(CH_2{\cdot}C{=}O{\cdot}O{\cdot}CH_2{\cdot}CH_3)$

occur somewhere in the code for the molecule. We can therefore associate a unique order with these atoms, the order in which their symbols occur in the code. Let the atoms be denoted by W, X, Y, and Z. We can then stipulate that, in a standard representation of the molecule, these atoms will be arranged round the carbon atom as shown in Figure 6; that is, if the bond C–W is directed downward, then X, Y, and Z occur in counterclockwise order as viewed from above.

This can be done with every asymmetric carbon atom in the molecule, and in this way the code of the molecule can be made to specify one particular stereoisomer (at least as far as asymmetric carbon atoms are concerned).

In much the same way, if we have a double bond with dissimilar pairs of chains at each end, we can take the order in which the roots of these chains occur in the code to specify one particular isomer of this configuration. Thus we can specify a particular isomer with respect to cis and trans isomerism also.

Now if the compound to be coded does not have this particular set of configurations, all that need be done is to specify which of these asymmetric atoms or double bonds have their appended chains going the other way. This could be done by attaching some mark (an asterisk, prime, etc.) to the appropriate symbol in the code. Thus, with the addition of only one extra symbol in the coding system, we can take care of these kinds of isomerism. This shows that the system is sufficiently flexible that it can be adapted to take account of stereoisomerism, if necessary. This approach to the problem has much in common with that discussed in ref 10. For an extension of Morgan's system of nomenclature to include stereoisomerism see ref 15. As already remarked, this problem will not be pursued further here; it will form the topic of a later paper.

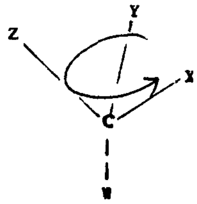


Figure 6.

ACKNOWLEDGMENT

My thanks are due to R. S. Milner, who assisted me with the initial report on the coding of acyclic compounds, from which the rest ultimately followed, and to M. Gordon, D. Bonchev, A. T. Balaban, N. Lozac'h, D. Rouvray, and others for their valuable help and suggestions. Above all, I thank A. L. Goodson of Chemical Abstracts Service for his constant encouragement and advice, without which this paper might never have been written.

APPENDIX A. SOME EXAMPLES

To understand a coding system well, one needs to have some practice at using it. Accordingly, a few examples of acyclic compounds are given in this section so that readers can assess their understanding of the coding and decoding processes. Given in Chart I are 12 structural formulas. They are not in any particular standard form but include the kind of abbreviations with which chemists are familiar. Thus the first step in coding will be to expand these abbreviations. The correct codes for these formulas are then given.

REFERENCES AND NOTES

(1) Aho, A. V.; Hopcroft, J. E.; Ullman, J. D. "The Design and Analysis of Computer Algorithms"; Addison-Wesley: Reading, MA, 1974.

(2) Cahn, R. S.; Dermer, C. S. "An Introduction to Chemical Nomenclature", 5th ed.; Butterworths: London, 1979.

(3) Dyson, G. M. "A New Notation and Enumeration System for Organic Compounds", 2nd ed.; Longmans: London, 1949.

(4) Goodson, A. L.; Lozac'h, N.; Powell, W. H. "Nodal Nomenclature-General Principles". *Angew. Chem., Int. Ed. Engl.* **1979**, *18*, 887-899.

(5) "Handbook for Chemical Society Authors"; The Chemical Society: London, 1960.

(6) "IUPAC, Nomenclature of Organic Chemistry"; Pergamon Press: Oxford, 1979; Sections A-F, H.

(7) Luks, E. M. "Isomorphism of Graphs of Bounded Valence Can Be Tested in Polynomial Time". Proceedings of the 21st Annual I.E.E.E. Symposium on the Foundations of Computer Science; IEEE: New York, 1980; pp 42-49.

(8) Milner, R. S.; Read, R. C. "A New System for the Designation of Chemical Compounds for the Purposes of Data Retrieval. I. Acyclic Compounds". Unpublished Research Report, University of the West Indies, 1969. Revised (1978) as Research Report CORR 78-42, Department of Combinatorics and Optimization, University of Waterloo.

(9) Morgan, H. L. "The Generation of a Unique Machine Description for Chemical Structures-A Technique Developed at Chemical Abstracts Service". *J. Chem. Doc.* **1965**, *5*, 107-113.

(10) Petrarca, A. E.; Lynch, M. F.; Rush, J. E. "A Method for Generating Unique Computer Structural Representations of Stereoisomers". *J. Chem. Doc.* **1967**, *7*, 154-165.

(11) Read, R. C. "The Coding of Various Kinds of Unlabelled Trees. Graph Theory and Computing"; Read, R. C., Ed.; Academic Press: New York, 1972; pp 153-182.

(12) Read, R. C.; Corneil, D. G. "The Graph Isomorphism Disease". *J. Graph Theory* **1977**, *1*, 339-363.

(13) Silk, J. A. "Realistic vs. Systematic Nomenclature". *J. Chem. Inf. Comput. Sci.* **1981**, *21*, 146-148.

(14) Smith, E. G. "The Wiswesser Line-Formula Chemical Notation". McGraw-Hill: New York, 1968.

(15) Wipke, W. T.; Dyott, T. N. "Stereochemically Unique Naming Algorithm". *J. Am. Chem. Soc.* **1974**, *96*, 4834-4842.

(16) The writing of numerals as subscripts is optional; they can be on the line.