An Approach to Automated Vocabulary Control in Indexes of Organic Compounds

CHARLES H. DAVIS
2631 E. 2nd Street, Bloomington, Indiana 47401

Received April 20, 1967

Algorithms are described for the automatic control of scattering which results from the use of synonymous organic compound names. The implications for computer-produced indexes are discussed, and the advantages of using truly systematic nomenclature in subject indexes are enumerated. COMIT II has been used as the programming language.

Vocabulary control is a necessary part of subject indexing if the indexes are to be cumulated and used for retrospective searching. If no such control is exercised, scattering of information will result—*i.e.*, information will appear under synonymous headings which are scattered throughout the alphabet. In indexes of organic compound names, vocabulary control is effected by a nomenclature system which allows an indexer to choose systematically among chemical synonyms.

In a typical conventional indexing situation, the structures of compounds appearing in original articles are drawn, and appropriate index names are assigned on the basis of the nomenclature system in use at that particular indexing service. A final edit may then be performed on these entries to determine their accuracy. A program which performed all or part of these functions would be beneficial, because it would relieve professional chemists from routine aspects of this work, and it would improve speed and accuracy of index preparation. In addition, a program which would automate or even partially automate vocabulary control procedures could be used in conjunction with computer-produced indexes such as *Index Chemicus*, thus making them more useful for retrospective searches as well as current awareness services. KWIC indexes might eventually be able to replace traditional subject indexes if they had such a capability.

In a nomenclature system such as that used by *Chemical Abstracts* in its subject indexes, names are dependent on an order of precedence of functions in the cases where a compound contains more than one functional group (*1*). Thus, an aldehyde which contains a carboxyl group is not named as an aldehyde but rather as an acid; an amine which contains a hydroxyl group is named as an alcohol or phenol rather than as an amine, etc. By following a series of rules based on such a hierarchy, one arrives at the name of a "parent" compound—a portion of the actual compound. All other groups present are then named as substituents on this parent compound. In an index, the parent compound is designated by that part of the chemical name which appears before the first comma, called the comma of inversion. The names of the substituents follow the comma of inversion in alphabetical order.

Three algorithms will be described, and they will show how a nomenclature system of this general type can be used in conjunction with a computer program which will automate or partially automate vocabulary control procedures. The programs were run on the IBM 7040 computer at the Indiana University School of Medicine, and COMIT II was used as the programming language. COMIT II, a second-generation list-processing language, was designed primarily for use in computational linguistics and is especially well suited for handling datum of this type (*2*).

A compromise approach to vocabulary control would be a program which considered only the names of the parent compounds. This would not be completely satisfactory, because the name of the parent compound is often itself a function of the substituents which follow it. For example, the name "Benzene, Amino-" is perfectly acceptable as far as the name of the parent compound is concerned, but it is not completely acceptable because the substituent "Amino-" should change the entire name to "Aniline" by the rules of the nomenclature system. However, such an approach would be somewhat useful, because it is the name of the parent which determines the gross alphabetic order of the index entries, and there are many "unacceptable" names of parents which would be eliminated by such a program. As an example of this there are the synonyms, "Acetaldehyde" and "Ethanal," both of which are frequently used. Some reduction of scattering would obviously occur if one were chosen over the other, regardless of what substituents followed. Figure 1 shows a generalized flow diagram of the program which was devised to handle this compromise situation. The actual programming rules can be seen in Figure 2.

The basic idea of this program is to shelve the comma of inversion and the names of the substituents while checking for the name of the parent in a "dictionary." Rule A finds everything up to the comma of inversion, the comma, a space, and everything following the space. Everything to the comma will be the parent, and so after all else has been shelved, the name of the parent is compressed and looked for in rule -DICT. Rule -DICT contains a sample of names of parents to illustrate how non-*CA* names can be converted to *CA* names. A dump of the contents of rule -DICT would essentially supply a list of cross references. If the name of the parent is found in the list, it is changed to the preferred name, and control goes to rule B; if it is not found, control

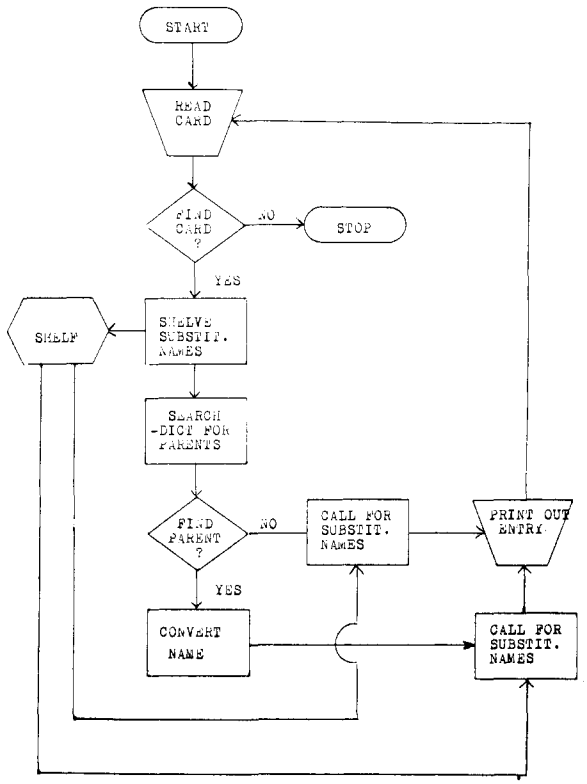

Figure 1. Flow diagram of the basic algorithm.

```
COMIT 7040/44 SYSTEM UNDER IBSYS OR IBJOB VERSION 1JULY66
   880002

          COM              DAVIS 001     CHEM 2
READ $=//*RCK1 **
* STOP
A $+,+-++=1+2+3+4//*Q23 2 3 4, *L1  DICT
-DICT ETHANAL=ACETALDEHYDE  B
                BENZENECARBOXALDEHYDE=BENZALDEHYDE  B
             JABBERWOCKY-17=DU-PONT-GREEN-OIL-3  B
           TRANS*-BUTENEDIOIC-ACID=FUMARIC-ACID  B
      CIS*-BUTENEDIOIC-ACID=MALEIC-ACID  B
      BETA*-NAPHTHOL=*2*-NAPHTHOL  B
B $=-+1+$0+*.//*A23 3, *WAM1 2 3 4  READ
STOP  *
END

    SUCCESSFUL COMPILATION. WORKSPACE CONTAINS 18651 REGISTERS.

ACETALDEHYDE, TRICHLORO-
BENZALDEHYDE, 2,4-DIBUTYL-
DU PONT GREEN OIL 3, ATTEMPTED PURIFICATION OF
FUMARIC ACID, METHYL-
MALEIC ACID, DIMETHYL-
2-NAPHTHOL, 1-CHLORO-3,4,5-TRIMETHYL-
     18463 REGISTERS OF THE WORKSPACE WERE UNUSED.

COMDUMP OF CHANGED DATA AFTER 27 RULES.
  THE WORKSPACE IS EMPTY.
  SHELF 23 IS EMPTY.
```

Figure 2. Computer printout of program CHEM 2.

also goes to rule B. This rule calls the comma and the substituents into workspace, spaces the printer, and prints out the entries a line at a time. In this case the following non-*CA* names have been converted to names having *CA*-approved parents (Jabberwocky 17 and du Pont Green Oil 3 are spurious examples to show that such a method would work for trivial names as well as systematic ones):

ETHANAL, TRICHLORO-
BENZENECARBOXALDEHYDE, 2,4-DIBUTYL-
JABBERWOCKY 17, ATTEMPTED PURIFICATION OF
Trans-BUTENEDIOIC ACID, METHYL-
Cis-BUTENEDIOIC ACID, DIMETHYL-
β-NAPHTHOL, 1-CHLORO-3,4,5-TRIMETHYL-

For this project, the data cards were arranged so that the names into which they were to be converted would appear on the printout sheet in alphabetical order. Naturally, if these programs were used in practice, they would have to be run in conjunction with a fast sorting program, such as IBSORT.

An interesting but difficult method for handling the problem of scattering would be to print out the entry at all of the headings where it might logically occur—that is, if a program could be written to convert one name into all or even most of the logical alternatives, then a chemist looking in the index would have a much better chance of finding what he wanted and would not have to follow cross references.

Program CHEM 3, which is shown in Figure 3, is meant to illustrate a general approach to such a program, and it deals only with the specific example of an entry made at "Benzene, Amino Carboxy." The traditional hyphens which follow substituent names in *CA*'s indexes (except for class names—*e.g.*, Ether, Ketone) have been omitted to simplify the search procedure.

CHEM 3 follows the same basic pattern as CHEM 2, the preceding program. The parent is compressed and looked for in the dictionary after the comma of inversion and the substituents have been shelved. If the name of the parent were unacceptable, it could be changed at this point. Whether the name of the parent is changed or not, it is shelved, and control goes to rule B, which calls the comma of inversion and the substituents (if any) into workspace. The next rule tests for an empty workspace. If anything is found, control goes to rule H, which duplicates the contents of workspace and shelves one copy. The following rule labeled "C," identifies substituent names as strings of characters between spaces,

```
COMIT 7040/44 SYSTEM UNDER IBSYS OR IBJOB VERSION 1JULY66
   880002

          COM              DAVIS 001     CHEM 3
READ $=//*RCK1 **
* STOP
A $+,+-++=1+2+3+4//*Q23 2 3 4, *L1  DICT
-DICT      BENZENE=     //*Q33 1  B
B $Q//*A23 1  *
* $1=          H
*  C
H $=1+1//*Q23 2  *
C -++=-1+2+3//*L2  LIST
*  QUIT
-LIST      AMINO=        //*Q43 1  C
           CARBOXY=      //*Q44 1  C
QUIT $Q//*A44 1  *
* $1=          2
C $Q//*A43 1 *
* $1=          F
G $=-+$0+$0+*.//*A33 2, *A23 3, *WAM1 2 3 4  READ
D $+$0=//*A23 1  *
* $=1+1//*Q23 2  *
* $+C+A+B+3+1+Y+Y+$//*Q53 1 9 *
* $=-BENZOIC-ACID+$2+*.//*A53 2, *WAM1 2 3  F
F $+$0=//*A23 1  *
* $=1+1//*Q23 2  *
* $+A+M+1+N+0+$//*Q53 1 7  *
* $=-ANILINE+$0+*.//*A53 2, *WAM1 2 3  G
STOP  *
END

    SUCCESSFUL COMPILATION. WORKSPACE CONTAINS 18515 REGISTERS.

BENZOIC ACID, AMINO
ANILINE,  CARBOXY
BENZENE, AMINO CARBOXY
     18140 REGISTERS OF THE WORKSPACE WERE UNUSED.

COMDUMP OF CHANGED DATA AFTER 26 RULES.
  THE WORKSPACE IS EMPTY.
  SHELF 23 IS EMPTY.
  SHELF 33 IS EMPTY.
  SHELF 43 IS EMPTY.
  SHELF 44 IS EMPTY.
  SHELF 53 IS EMPTY.
```

Figure 3. Computer printout of program CHEM 3.

compresses them, and looks for them in rule -LIST. "Amino" and "Carboxy" are put on shelves 43 and 44, respectively, and rule C will fail since there are no more substituents.

Provision should have been made at this point for substituent names not in -LIST; such a provision is made in program CHEM 4, but since CHEM 3 is only treating the case "Benzene, Amino Carboxy," it will run anyway. When rule C fails, control goes to the next rule, which simply routes control to rule QUIT.

At this point tests are inserted to determine whether anything has been shelved. This would be necessary in a complete program where the results were not a foregone conclusion. If both shelves 43 and 44 had been empty, control would have gone to rule G, and the original name would have been printed out as the only logical entry. Control would also have gone to rule G if rule B had failed to find any substituents. Since shelves 43 and 44 are not empty, control goes to rules D and E, which initiate similar operations. The contents of the workspace are deleted, and a null constituent is inserted. The substituents are called into workspace, are duplicated, and one copy is reshelved. The copy which is retained in workspace has the appropriate substituent name deleted as shown in the listing. Everything except the undesired name is put on shelf 53, and control falls to the next rule, which converts the contents of workspace into the desired parent compound name and calls the contents of shelf 53 into a null constituent which follows it. The new name is printed out, and control goes either to rule F or rule G. At least the name found by rule G will be printed, after which control goes back to rule READ, and the next card is read into workspace.

The problem of the order of precedence of functions is circumvented by CHEM 3, because all (given) possibilities are printed out, and no choice must be made among them. The problem of locants—*i.e.*, the relative positions of the substituents with respect to each other—has been ignored at this point.

Program CHEM 4, which is shown in Figure 4, is an attempt to mimic the processes involved in preparing subject index entries for conventional indexes such as those of *Chemical Abstracts*. The parent is found in a dictionary just as in CHEM 2 and CHEM 3, and the substituents are looked for in rule -LIST as in CHEM 3. Following rule -LIST, there is a rule which will shelve substituent names not found in -LIST in order to get them out of workspace so that the search can continue. When no more substituents are found, rule C fails and control goes to rule QUIT, which makes sure the workspace is empty. A series of tests determines which "forbidden" substituent names may have been present. Since the COMIT compiler sorts the left halves of list subrules alphabetically, the tests must be made in shelf number sequence in order to follow the *CA* order of precedence. If no "forbidden" substituent names are found, all the tests will fail, and the original name at the benzene heading will be printed out. The flow of control is such that the program will proceed to completion as soon as possible—*i.e.*, as soon as the functional group highest in the order of precedence is found. Rules D, E, F, and G all initiate similar routines. Everything but the undesired substituent name is shelved, the undesired sub-

```
COMIT 7040/44 SYSTEM UNDER IBSYS OR IBJOB VERSION 1JULY66
   880002


          COM        DAVIS 001   CHEM 4
      READ $=///*RCK1 **
      *   STOP
      A $+,+-+$=1+2+3+4///*Q23 2 3 4, *L1  DICT
      -DICT   BENZENE=    //*033 1   B
      B $0//*A23 1  *
      * $=1+1///*023 2  *
      C -+$+-=1+2+3///*L2  LIST
      *   QUIT
      -LIST   CARBOXY=///*31 1  C
           HYDROXY=///*Q2 1  C
           AMINO=///*03 1  C
           METHOXY=///*04 1  C
      * -+$+-=1+2+3///*099 2 3  C
      QUIT $=0 *
      * $0//*A1 1  *
      * $1=   D
      * $0//*A2 1  *
      * $1=   E
      * $0//*A3 1  *
      * $1=   F
      * $0//*A4 1  *
      * $1=   G
      * $=-+$0+$0+*,///*A33 2, *A23 3, *WAM1 2 3 4  READ
      D $=$0//*A23 1  *
      * $=0+A+B+P+0+X+Y+$///*053 1 9  *
      * $=-BENZOIC-ACID+$0+*,///*A53 2, *WAM1 2 3  *
      * $=$0+$0+$0+$0+$0///*A33 1, *A2 2, *A3 3, *A4 4, *A99 5  *
      * $=0  READ
      E $=$0///*A23 1  *
      * $+U+Y+D+$+0+X+Y+$///*053 1 9  *
      * $=-PHENOL+$0+*,///*A53 2, *WAM1 2 3  *
      * $=$0+$0+$0+$0///*A33 1, *A99 2, *A3 3, *A4 4  *
      * $=0  READ
      F $=$0///*A23 1  *
      * $+A+Y+1+N+0+$///*053 1 7  *
      * $=-ANILINE+$0+*,///*A53 2, *WAM1 2 3  *
      * $=$0+$0+$0///*A33 1, *A4 2, *A99 3  *
      * $=0  READ
      G $=$0///*A23 1  *
      * $+Y+E+T+H+I+X+Y+$///*053 1 9  *
      * $=-ANISOLE+$0+*,///*A53 2, *WAM1 2 3  *
      * $=$0+$0///*A33 1, *A99 2  *
      * $=0  READ
      STOP  *
      END

   SUCCESSFUL COMPILATION, WORKSPACE CONTAINS 18347 REGISTERS.
```

Figure 4. Listing of programming rules for CHEM 4.

```
   SUCCESSFUL COMPILATION, WORKSPACE CONTAINS 18347 REGISTERS.

ANILINE,
ANILINE,  METHOXY
ANILINE,  TETRAKIS(1,2-DIMETHYLCYCLOPROPYL)
ANISOLE,
ANISOLE, 1,3,5-TRIISOBUTYL
BENZOIC ACID,
BENZOIC ACID, AMINO
BENZOIC ACID, AMINO  HYDROXY METHOXY
BENZOIC ACID,  CHLORO FLUORO METHOXY
BENZOIC ACID,  HYDROXY
BENZOIC ACID,  METHOXY
PHENOL,
PHENOL, AMINO
PHENOL,  METHOXY
PHENOL,  DIPENTYL
     18002 REGISTERS OF THE WORKSPACE WERE UNUSED.

COMDUMP OF CHANGED DATA AFTER 2806 RULES.
  THE WORKSPACE IS EMPTY.
  SHELF 1 IS EMPTY.
  SHELF 2 IS EMPTY.
  SHELF 3 IS EMPTY.
  SHELF 4 IS EMPTY.
  SHELF 23 IS EMPTY.
  SHELF 33 IS EMPTY.
  SHELF 53 IS EMPTY.
  SHELF 99 IS EMPTY.
```

Figure 5. Computer printout of results of CHEM 4.

stituent name is replaced by the proper parent compound name, and the other substituents are called back into workspace to follow the new parent. The entry is then printed out, and control goes to the next rule, which calls for the contents of whatever shelves may have been used in the operation. The next rule clears the workspace and loops control back to rule READ.

CHEM 4, like CHEM 3, is only a prototype, as it cannot handle every eventuality. However, the printout sheet for CHEM 4, which is shown in Figure 5, illustrates how this program has been used to convert the following entries which might occur at the "Benzene" heading:

BENZENE, AMINO
BENZENE, AMINO METHOXY
BENZENE, AMINO TETRAKIS(1,2-DIMETHYL-CYCLOPROPYL
BENZENE, METHOXY
BENZENE, 1,3,5-TRIISOBUTYL METHOXY
BENZENE, CARBOXY
BENZENE, AMINO CARBOXY
BENZENE, AMINO CARBOXY HYDROXY METHOXY
BENZENE, CARBOXY CHLORO FLUORO METHOXY
BENZENE, CARBOXY HYDROXY
BENZENE, CARBOXY METHOXY
BENZENE, HYDROXY
BENZENE, AMINO HYDROXY
BENZENE, HYDROXY METHOXY
BENZENE, HYDROXY DIPENTYL

The program has been written so that the sequence of the data cards does not matter, but in this particular case the cards were again arranged so that the entries on the printout sheet would appear in alphabetical order.

CHEM 2, CHEM 3, and CHEM 4 represent a series of pilot projects. Future projects should investigate means of handling locants. Substituent names could probably be tagged with numerical subscripts and handled in some way, but FORTRAN or some other programming language would probably be better suited for the numerical computations which would surely be involved.

CHEM 2 is a relatively simple program which, in expanded form, could be used effectively with computer-produced indexes just as it is. By using magnetic tape to increase processing speed, it could be tied in with the procedures currently used for indexes such as *Index Chemicus*. As stated before, this would be a compromise, but some vocabulary control should be better than none.

The total times (including processing, execution, and loading) for the three programs which have been discussed are:

CHEM 2	16.3 seconds
CHEM 3	17.9 seconds
CHEM 4	46.9 seconds

It is fairly clear that existing nomenclature systems do not lend themselves to computer programming. The development of a nomenclature system similar to, but more consistent than, the one now used by *Chemical Abstracts* would be an enormous help. Having fewer exceptions to the rules would greatly facilitate programming. In effect, the names generated by such a system could be used as a code not unlike the various linear codes which have been developed; the use of conventional systematic nomenclature would have two major advantages: (1) the names can be pronounced, whereas the codes cannot; (2) the system could be used immediately by all chemists without requiring them to learn something altogether new. A truly systematic nomenclature would allow for substructure searching, and the names could be made unique and unambiguous so that they could be used by themselves as a registry code for storage and retrieval.

ACKNOWLEDGMENT

The author wishes to thank Dr. Robert Wall of the Linguistics Department, Indiana University, and Dr. Ann F. Painter of the Graduate Library School, Indiana University, for their encouragement and advice. Computations for this project were performed at the Indiana University School of Medicine Research Computation Center, which is supported in part by Public Health Research Grant Fr 00162-03.

LITERATURE CITED

(1) Chemical Abstracts Service, "The Naming and Indexing of Chemical Compounds," p. 12N, § 83, American Chemical Society, Easton, Pa., 1962.
(2) M.I.T., The Research Laboratory of Electronics and the Computation Center, *An Introduction to COMIT Programming* and *COMIT Programmers Reference Manual*. The M.I.T. Press, November 1963.