match no. 23 pp. 65-87 1988

MOLECULAR TOPOLOGY. 3.[1] A NEW CENTRIC CONNECTIVITY INDEX (CCI)

Mircea V. Diudea[a]* and Bazil Pârv[b]

[a] University of Cluj - Napoca, Faculty of Chemical Technology
Arany Janos Str., 11, 3400 Cluj - Napoca, Romania

[b] Cluj - Napoca University Computing Center, Romania

(Received: October 1987)

Abstract

A "centric criterion" CC vector was derived from the distance frequency matrix F by means of a new power function. CC enables one to drastically simplify the procedure for determining the centre/polycentre of a graph G. Based on CC, a new "centric connectivity index" CCI was developped. This topological index can account for the presence of heteroatom as well as of unsaturation or aromaticity. For organic radicals, a CCIR version is proposed. Two computer programs, based on vertex adjacency data, were performed for CCI and CCIR computation.

Introduction

In graph theory, a vertex v_i with the property that the maximal distance to any other vertex in the graph G is minimal, is called the centre of the graph.[2] The distance between v_i and the most distant vertex in G is called the radius of the graph, $r_{max,ij}$:

---

* Author to whom all correspondence should be addressed.

$$r_{max,ij} = \min \; ; \quad j = 1, 2, \ldots N \qquad (1)$$

where N is the number of vertices in G.

All other vertices in G will have their radii larger than $r_{max,ij}$. According to the Jordan theorem, any tree has a unique centre (vertex) or bicentre (two adjacent vertices).

Based on the graph centre notion, Balaban[3] introduced the <u>centric index</u> B , eq. 2 :

$$B = \sum_{i=1}^{n} \delta_i^2 \qquad (2)$$

where δ_i represents a pruning sequence (see[3, 4]).

A normalized <u>centric index</u> C was also introduced, in order to have a topological index reflecting the shape of alkanes, eq. 3 :

$$C = 1/2 \; (B_{tree} - B_{chain}) = 1/2 \; (B_{tree} - 2 N - U) \qquad (3)$$

where $U = [1 - (-1)^N]/2$. It is obvious that for n-alkanes (chain graphs) C = 0 . The centric index C belongs to the "quadratic" class of topological indices (see[4]).

Balaban's procedure for determining the centre is applicable only to acyclic graphs. In cyclic systems, sometimes more than one vertex could appear as central. To resolve this problem on cyclic **graphs**, a generalized concept for the graph centre and several centric indices were proposed.[5, 6]

The new definition for the graph centre is based on the distance matrix D and consists of four hierarchically ordered criteria: 1) the minimal maximum distance in the row or column of the vertex ; 2) the smallest sum of all distances d_{ij} (in D) in the row or column corresponding to vertex i ; 3) the smallest number of largest entries in the vertex distance code, VDC ; 4) criteria 1) - 3) are repeated to the pseudocentre of the graph until on two successive cycles no more vertices are rejected. The **polycentre** thus obtained may contain one vertex or several adjacent or non-adjacent vertices.

Centric Criterion (CC) Vector

In order to simplify the procedure for determining the centre/polycentre of a graph, we proposed a new function, eq. 4 :

$$dd(i) = \sum_{j=1}^{d_{max}} (f_{ij})^{j/d_{max}} \qquad (4)$$

where : f_{ij} - represent the elements of the distance frequency matrix, $F_{N,d_{max}}$.

d_{max} - is the largest value for a path (or distance) in G :

$$d_{max} = \max d_{ij} \; ; \quad i,j = 1, 2, \ldots N \qquad (5)$$

and d_{ij} - the elements of distance matrix $D_{N,N}$ (or simply D).

dd(i) - denote the elements of the resulting "distance distribution" ($DD_{N,1}$) vector.

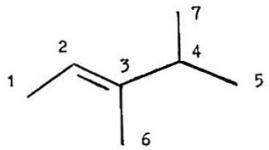
The vector DD shows a distance sequence often different of that resulting by simple summation of all entries in the i-th row or column of D (see [7]). The minimal value among dd(i) - values will discriminate for the centre of the graph :

$$r_{dd}(G) = \min dd(i) \; ; \quad i = 1, 2, \ldots N \qquad (6)$$

By normalizing the vector DD by min dd(i), we derived the "centric criterion" ($CC_{N,1}$) vector :

$$cc(i) = dd(i) \,/\, \min dd(i) \; ; \quad i = 1, 2, \ldots N \qquad (7)$$

Thus, CC shows a $r_{cc}(G) = 1$. The subscript dd and cc in the graph radius expression, r (G), suggest the radius values derived from DD and CC, respectively. Exemplifying, the following matrices can be writen for the graph G1 :



G1

$$
\begin{array}{c}
\\ 1\\2\\3\\4\\5\\6\\7
\end{array}
\begin{array}{c}
1\ 2\ 3\ 4\ 5\ 6\ 7\\
\begin{Vmatrix}
0&1&0&0&0&0&0\\
1&0&1&0&0&0&0\\
0&1&0&1&0&1&0\\
0&0&1&0&1&0&1\\
0&0&0&1&0&0&0\\
0&0&1&0&0&0&0\\
0&0&0&1&0&0&0
\end{Vmatrix}
\end{array}
\longrightarrow
\begin{Vmatrix}
0&1&2&3&4&3&4\\
1&0&1&2&3&2&3\\
2&1&0&1&2&1&2\\
3&2&1&0&1&2&1\\
4&3&2&1&0&3&2\\
3&2&1&2&3&0&3\\
4&3&2&1&2&3&0
\end{Vmatrix}
\longrightarrow
\begin{Vmatrix}
1&1&2&2\\
2&2&2&0\\
3&3&0&0\\
3&2&1&0\\
1&2&2&1\\
1&2&3&0\\
1&2&2&1
\end{Vmatrix}
\longrightarrow
\begin{Vmatrix}
5.6818\\
4.2852\\
3.0481\\
3.7303\\
5.0960\\
4.6937\\
5.0960
\end{Vmatrix}
\longrightarrow
\begin{Vmatrix}
1.8640\\
1.4059\\
1.0000\\
1.2238\\
1.6718\\
1.5399\\
1.6718
\end{Vmatrix}
$$

A D F DD CC

Fig. 1 - Matrices corresponding to the graph G1

To observe the vertex v_3 which has : 1) the smallest maximal distance (2) ; 2) the minimal distance sum (9) and 3) the smallest number of largest entries in VDC (zero) . The last criterion requires additional explanations.

Randić introduced[8] the so-called atomic and molecular path codes, in intention to have an unique characterization of molecular structures. The atomic (vertex) path code of the atom i, VPC(i), can be writen as a power series :

$$VPC(i) = p_1^{f_{i1}}\ p_2^{f_{i2}} \ldots p_{max}^{f_{i,max}} \tag{8}$$

where p_j is the path of length j , and f_{ij} represents its frequency number (e. g. the elements of F). For polycyclic structures, Bonchev (see[4], loc. cit.[59]) replaced the path with the distance notion, thus defining the vertex distance code, VDC(i) :

$$VDC(i) = d_1^{f_{i1}}\ d_2^{f_{i2}} \ldots d_{max}^{f_{i,max}} \tag{9}$$

The molecular (graph) codes are obtained similarly, by summation over all atoms (vertices) in the molecule (graph G) :

$$GPC(G) = p_1^{f_1}\ p_2^{f_2} \ldots p_{max}^{f_{max}} \tag{10}$$

$$GDC(G) = d_1^{f_1}\ d_2^{f_2} \ldots d_{max}^{f_{max}} \tag{11}$$

where :

$$f_j = 4/2 \sum_{i=1}^{N} f_{ij} \qquad (12)$$

represents the half sum in the column j of F. More frequently, these codes are used in linear form :

$$VDC = (f_{i1}, f_{i2}, \ldots f_{i,max}) \qquad (13)$$

It is obvious that the i-th row in F represents just the VDC of vertex v_i.

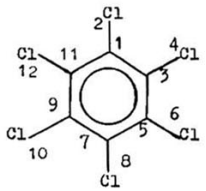
The above codes were converted into topological indices (see[4]) of quadratic type :

$$VDI = \sum_j (f_{ij})^2 \qquad (14)$$

$$GDI = \sum_j (f_j)^2 \qquad (15)$$

Returning to G1, the VDC(3) = 3,3,0,0,which agrees with the criterion 3) of Bonchev et al.[5, 6] for the graph centre.

For G2, only CC is given; one can see that G2 has a polycentre. All other vertices out of the ring have their cc(i)-values larger than unity.



G2

CC(G2)
1.0000
1.3556
1.0000
1.3556
1.0000
1.3556
1.0000
1.3556
1.0000
1.3556
1.0000
1.3556

Fig. 2 - The CC vector for G2

Centric Connectivity Index, CCI

As shown before, CC provides a good topological information and therefore we converted it into a "centric connectivity index", CCI, accord-

ing to eq. 16 :

$$CCI(G) = m \sum_{\text{all edges}} b_{ij} \left[cc(i)S(i) \times cc(j)S(j) \right]^{-1/2} \qquad (16)$$

where : m - is a multiplier, introduced by the same reasons as for the Balaban's J index (see[7]).

$$m = 1/(\mu + 1) \qquad (17)$$

μ - is the cyclomatic number, or the number of rings, in the classical definition of polycyclic compounds (see Baeyer, IUPAC and Chemical Abstracts nomenclature systems)

$$\mu = q - N + 1 \qquad (18)$$

with q = the number of edges in G.

b_{ij} - represents the bond order (most often the conventional order: 1, 2 and 3 for simple, double and triple bonds, respectively, and 1.5 for aromatic bonds).

cc(i), cc(j) - are the values in CC corresponding for two adjacent vertices .

S(i),S(j) - stand for the degrees of electronegativity of vertices i and j, respectively, as defined by us in ref.[1]

There is a limited analogy between CCI and the Balaban's J^{X}_{het} index amended for heteroatoms.[9] The use of group electronegativities S(i) along with the cc(i) values makes from CCI a highly discriminating topological index. Following the connectivity in G, CCI also accounts for the type of bonds (by means of b_{ij}). For G1 and G2, the computed S(i) and CCI values are listed below :

S(G1)	S(G2)
1.6216	1.2780 (atoms: 1,3,5,7,9,11)
1.3869	1.9937 (atoms: 2,4,6,8,10,12)
1.2873	
1.2780	
1.6216	
1.6216	
1.6216	

CCI(G1) = 3.9081 CCI(G2) = 5.1352

Fig. 3 - The computed S(i) and CCI for G1 and G2

Radical Approach - CCIR

In a series of drug molecules, with similar biological properties, only the radical attached to the same skeleton varies. Moreover, some properties could be a function of local topology (meaning just the radical) or local electronic distribution. Thus, one imposes to compute a topological index only for radicals.

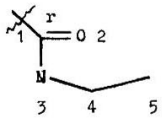
Balaban and Filip proposed[7] an approach for the J index which considers the radical as a rooted graph. They assigned to the root vertex a distance sum which is ten times smaller than the minimal distance sum among all graph vertices.

By analogy, we addopted for the root vertex a dd(i) value 0.1 less than the minimal dd(i) value :

$$dd(root) = \min dd(i) - 0.1 \qquad (19)$$

Since CCI uses of double characterization of a vertex, this approach has to account for the modified adjacency to root vertex when computing S(root) = the group electronegativity of root vertex.[1]

Subsequently, DDR is normalized by dd(root) value, thus obtaining the CCR vector. CCIR is then computed by the same eq. 16. Exemplifying, the computed values for a radical (rooted graph) RG3 (from a butyrylcholinesterase inhibitor molecule - see[10]) are listed below :



RG3

(r-stands for root vertex)

CC(RG3)	S(RG3)	CCIR = 2.8210
1.0000	1.2873	
1.5978	1.9115	
1.0399	1.4223	
1.2739	1.3825	
1.5978	1.6216	

Fig. 4 - The computed CC, S(i) and CCIR for RG3

Computer Programs for CCI and CCIR

Two computer programs were performed for CCI and CCIR computation. First program, SCRGRAF - reads from input terminal informations about the

chemical structure of a given compound and stores this information on a disk file. The second program, GRAFØ1 - computes CCI and CCIR by using two disk files: first file was created by SCRGRAF program and the second file, named ELNEGS.DAT contains the Sanderson's electronegativities of atoms. This second file was created by using a text editor. Its structure is the following : - Sanderson's electronegativities

- group of atom in the short form of periodic chart
- symbol of atom

The programs were written in FORTRAN 80, on a Romanian CUB - Z microcomputer (based on Z - 80 microprocessor).

Conclusions

The new function introduced enabled us to derive, from the distance frequency matrix F, the "centric criterion" CC vector. This vector is an expeditive tool for determining the graph centre/polycentre and brings a valuable topological information.

The "centric connectivity index", CCI , derived on the basis of CC appears to be a powerful topological index, as will be shown in the next paper of this series. For biological activity correlations, the radical approach CCIR can be used instead of (or along with) CCI. These topological indices account for the presence of heteroatom as well as of unsaturation or aromaticity.

Figures 5 to 14 present the computer programs for CCI and CCIR computation.

Acknowledgements

We are deeply indebted to professor A. T. Balaban, from the Polytechnic Institute Bucharest, for encouragements and helpful discussions.

```
c
c       SCRGRAF - read from input terminal :
c               - name of chemical compound
c               - number of vertices (atoms)
c               - number of edges
c               - element symbol for each atom
c               - for each edge :
c                 - start vertex
c                 - end vertex
c                 - bond  type
c
c       These informations are stored on a disk file in above
c       mentioned order
c
        byte n,m,i1,i2
        integer el
        integer den(40)
        write(1,1)
1       format(' SCRGRAF - formulae reading , as a graph '//)
        write(1,2)
2       format(' Enter name of file in which formula  is stored : '/)
        call assdsk(7)
        write(1,19)
19      format(' Name of chemical compound : '/)
        read(1,17) den
17      format(40a2)
        write(7,18) den
18      format(1x,40a2)
        write(1,3)
3       format(' Number of atoms : ')
        read(1,4) n
4       format(i3)
        write(1,5)
5       format(' Number of edges : ')
        read(1,4) m
        write(7,16) n,m
16      format(1x,2i4)
        do 50 i=1,n
        write(1,13) i
13      format(' Element symbol for atom ',i3,' : ')
        read(1,14) el
        write(7,15) i,el
14      format(a2)
15      format(1x,i4,a2)
50      continue
        do 100 i=1,m
        write(1,6) i
6       format(' Edge ',i3)
        write(1,7)
7       format(' Start atom ')
```

Fig. 5. Program SCRGRAF (in FORTRAN)

```
51              read(1,4) i1
52              write(1,8)
53      8       format(' End atom ')
54              read(1,4) i2
55              write(1,9)
56      9       format(' Bond  type ')
57              read(1,11) b
58      11      format(f10.0)
59              write(7,12) i1,i2,b
60      12      format(1x,2i4,f10.2)
61      100     continue
62              endfile 7
63              stop
64              end
```

Fig. 5. (Continued)

```
c
c       GRAF01 - program for determining :
c              - adjacency matrix
c              - distance matrix
c              - frequency matrix
c              - distance distribution
c              - centric criterion
c
c       The following subroutines are called  :
c
c       CITGRAF (which call ASSDSK)
c       ADIAC
c       DISTA
c       FRECV
c       SCRMS1
c       SCRMG1
c       COMPCC
c
c
        program GRAF01
        integer den(40)
        byte n,m,k,f(1000),d(1000),a(1000),ga(100)
        byte leg(100,2),v(100),h(100)
        dimension tip(100)
        dimension sa(100),si(100),sg(100)
        dimension dd(100),cc(100)
        write(1,1)
1       format(///' Program GRAF01 '/)
        call citgrf(n,m,den,leg,tip,sa,ga)
        write(1,19) den
19      format(//,' Computing are performed for compound '/,
     -          5x,40a2,//)
        call adiac(n,m,leg,a)
        write(1,2)
2       format(/' Adjacency matrix : '/)
        call scrms1(a,n,1)
        call dista(n,m,a,d)
        write(1,3)
3       format(/' Distance matrix : '/)
        call scrms1(d,n,1)
        call frecv(n,d,k,f)
        write(1,4)
4       format(/' Frequency matrix : '/)
        call scrmg1(f,n,k,1)
        ki=k
        do 10 i=1,n
        dd(i)=0.0
        do 10 j=1,k
        aux=float(j)/float(ki)
        ii=(j-1)*n+i
```

Fig. 6. Program GRAF01 (in FORTRAN)

```
51              kk=f(ii)
52      10      dd(i)=dd(i)+kk*aux
53              dmin=9.0e20
54              do 20 i=1,n
55              if(dmin.gt.dd(i)) dmin=dd(i)
56      20      continue
57              write(1,25)
58              read(1,22) i
59              if(i.eq.2) go to 50
60              do 30 i=1,n
61      30      cc(i)=dd(i)/dmin
62      25      format(' Do you want computations for whole molecule ',/,
63                     ' (1 = yes , 2 = no) ')
64              na=0
65              nl=0
66              call compcc(n,m,a,sa,ga,tip,leg,cc,v,sg,h,si,cci,na,nl)
67              write(1,11)
68      11      format(/' Atom        DD            CC            SI'/)
69              do 40 i=1,n
70      40      write(1,12) i,dd(i),cc(i),si(i)
71      12      format(1x,i5,3f12.4)
72              write(1,8) cci
73      8       format(' CCI  = ',f12.4/)
74              go to 100
75      50      write(1,21)
76      21      format(/' No. of atom which is rooted ')
77              read(1,22) na
78      22      format(i3)
79              if(na.le.0) go to 50
80              if(na.gt.n) go to 50
81              write(1,26)
82      26      format(' No. of broken edges to this atom : ')
83              read(1,22) nl
84              dmin=dmin-0.1
85              dd(na)=dmin
86              do 70 i=1,n
87      70      cc(i)=dd(i)/dmin
88              call compcc(n,m,a,sa,ga,tip,leg,cc,v,sg,h,si,cci,na,nl)
89              write(1,11)
90              do 80 i=1,n
91      80      write(1,12) i,dd(i),cc(i),si(i)
92              write(1,23) cci
93      23      format(' CCIR = ',f12.4//)
94      100     stop
95              end
```

Fig. 6. (Continued)

```
c
c       CITGRAF - subroutine which reads data about graph from disk
c               file to core memory
c       Disk file was created with program SCRGRAF and its  structure
c       is :
c         - first record contains compound name
c         - second record contains number of vertices (atoms) and
c           number of edges
c         - next records (one for each atom) contain  name of the
c           corresponding element
c         - next records (one for each edge) contain  :
c             - first atom of bond
c             - second atom of bond
c             - bond  type
c
c       Memory variables :
c
c       DEN(40)  - name of compound
c       LEG(M,2) - atom index for each edge
c       TIP(M) - bond type
c       SA(N) - atom electronegativity
c       GA(N) - group from periodic chart
c
        subroutine CITGRF(N,M,DEN,LEG,TIP,SA,GA)
        integer den(40)
        integer EL,EL1
        byte N,M,ga1,LEG(100,2),GA(100)
        dimension tip(100),SA(100)
        write(1,1)
1       format(' Please enter name of file which contains formula : ')
        call assdsk(7)
        read(7,19) den
19      format(40a2)
        read(7,2) n,m
2       format(2i4)
        call open(8,'ELNEGS  DAT ',1)
        do 500 i=1,n
        read(7,4) j,el
4       format(i4,a2)
        rewind 8
100     read(8,5,end=200) sa1,ga1,el1
5       format(f8.3,i1,a2)
        if(el.eq.el1) go to 300
        go to 100
200     write(1,6) el
6       format(1x,a2,' element not found in ELNEGS!/)
        go to 500
300     sa(i)=sa1
        ga(i)=ga1
500     continue
```

Fig. 7. Subroutine CITGRAF (in FORTRAN)

```
      do 10 i=1,m
      read(7,3) leg(i,1),leg(i,2),tip(i)
3     format(2i4,f10.2)
10    continue
      return
      end
```

Fig. 7. (Continued)

```
c
c       ASSDSK - assigns a disk file for a  logical unit number LUN
c
        subroutine ASSDSK(LUN)
        byte numef(8),extf(3),nume(12)
        integer nrunit,lun
2       format(' Name of disk file (with capital letters) : ')
3       format(' File extension (with capital letters) : ')
4       format(' Unit (1=A , 2=B) : ')
5       format(8A1)
6       format(i2)
        do 30 i=1,8
30      numef(i)=' '
        write(1,2)
        read(1,5) (numef(i),i=1,8)
        do 40 i=1,3
40      extf(i)=' '
        write(1,3)
        read(1,5) (extf(i),i=1,3)
        do 50 i=1,8
50      nume(i)=numef(i)
        do 60 i=1,3
        ii=i+8
60      nume(ii)=extf(i)
        nume(12)=' '
70      write(1,4)
        read(1,6) nrunit
        if(nrunit.lt.1.or.nrunit.gt.2) go to 70
        call open(lun,nume,nrunit)
        return
        end
```

Fig. 8. Subroutine ASSDSK (in FORTRAN)

```
c
c       ADIAC - adjacency matrix determination for a graph with
c             N - number of vertices
c             M - number of edges
c             LEG(M,2) - edge vector (start vertex, end vertex)
c
c       The adjacency matrix is symmetrical and is stored in vector
c       form , columnwise (only the upper triangle).
c
        subroutine ADIAC(N,M,LEG,A)
        byte n,m,leg(100,2),i1,i2,a(1000)
        integer z
        na=n
        na=na*(n-1)/2
        do 10 i=1,na
10      a(i)=0
        do 20 k=1,m
        i1=leg(k,1)
        i2=leg(k,2)
        if(i1.lt.i2) go to 15
        z=i1
        i1=i2
        i2=z
15      z=(i2-1)*(i2-2)/2+i1
20      a(z)=1
        return
        end
```

Fig. 9. Subroutine ADIAC (in FORTRAN)

```
c
c       DISTA -distance matrix determination
c       Input parameters :
c             N - number of vertices
c             M - number of edges
c             A - adjacency matrix (upper triangle)
c       Results :
c             D - distance matrix (upper triangle)
c
c
        subroutine DISTA(N,M,A,D)
        byte n,m,d1
        byte a(1000),d(1000)
        na=n
        na=na*(n-1)/2
        do 10 k=1,na
10      d(k)=0
        do 20 k=1,na
        if(a(k).eq.0) go to 20
        d(k)=1
20      continue
        do 40 i=1,n
        do 40 j=1,n
        if(i.eq.j) go to 40
        if(j.lt.i) go to 11
        k1=(j-1)*(j-2)/2+i
        go to 12
11      k1=(i-1)*(i-2)/2+j
12      continue
        do 30 l=1,n
        if(l.eq.i.or.l.eq.j) go to 30
        if(l.lt.j) go to 21
        k2=(l-1)*(l-2)/2+j
        go to 22
21      k2=(j-1)*(j-2)/2+l
22      if(d(k1)*d(k2).eq.0) go to 30
        d1=d(k1)+d(k2)
        if(l.lt.i) go to 23
        k3=(l-1)*(l-2)/2+i
        go to 24
23      k3=(i-1)*(i-2)/2+l
24      if(d(k3).ne.0and.d1.ge.d(k3)) go to 30
        d(k3)=d1
30      continue
40      continue
        return
        end
```

Fig. 10. Subroutine DISTA (in FORTRAN)

```
c
c       FRECV - frequency matrix F determination
c
c       Input data :
c             N - number of vertices
c             D - distance matrix (upper triangle)
c       Results :
c             K - maximal element of matrix D
c             F - frequency matrix , with N rows and K columns
c                 (stored columnwise)
c
c
        subroutine FRECV(N,D,K,F)
        byte f(1000),d(1000)
        byte n,k
        na=n
        na=na*(n-1)/2
        k=1
        do 10 i=1,na
        if(d(i).gt.k) k=d(i)
10      continue
        nf=n*k
        do 20 i=1,nf
20      f(i)=0
        do 50 i=1,n
        do 50 j=1,n
        if(i-j) 30,50,40
30      l=(j-1)*(j-2)/2+i
        go to 45
40      l=(i-1)*(i-2)/2+j
45      m=d(l)
        ii=(m-1)*n+i
        f(ii)=f(ii)+1
50      continue
        return
        end
```

Fig. 11. Subroutine FRECV (in FORTRAN)

```
c
c       SCRMS1 -writes a symmetrical matrix A of order n on a file with
c              logical unit number LUN
c
c       Matrix A is stored columnwise , upper triangle , without
c       diagonal elements , which are null.
c       Matrix type is BYTE.
c
        subroutine scrms1(a,n,lun)
        byte n,a(1000),v(100)
        do 20 i=1,n
        do 10 j=1,n
        if(i.ne.j) go to 5
        v(j)=0.0
        go to 10
5       i1=i
        j1=j
        if(i1.lt.j1) go to 7
        k=i1
        i1=j1
        j1=k
7       k=(j1-1)*(j1-2)/2+i1
        v(j)=a(k)
10      continue
        write(lun,1) (v(k),k=1,n)
1       format(' ',20i4)
20      continue
        return
        end
```

Fig. 12. Subroutine SCRMS1 (in FORTRAN)

```
c
c       SCRMG1 - writes a general matrix A with M rows and N columns
c              on a file with logical unit number LUN.
c
c       Matrix is stored columnwise.
c       Matrix type is BYTE.
c
        subroutine scrmg1(a,m,n,lun)
        byte n,m,a(1),v(100)
        do 20 i=1,m
        do 10 j=1,n
        k=(j-1)*m+i
10      v(j)=a(k)
        write(lun,1) (v(k),k=1,n)
1       format(' ',20i4)
20      continue
        return
        end
```

Fig. 13. Subroutine SCRMG1 (in FORTRAN)

```
c
c       COMPCC - determination of following parameters :
c
c        V   - adjacency sum of vertex
c        SG  - group electronegativity of vertex
c        H   - number of hydrogen atoms on vertex
c        SI  - vertex degree of electronegativity
c        CCI - centric connectivity index
c
c       Input data :
c        N   - number of vertices
c        M   - numver of edges
c        A   - adjacency matrix
c        SA  - vertex electronegativity vector
c        GA  - vector of group in periodic chart of atom
c        TIP - bond  type vector
c        LEG - bond  vector
c        CC  - centric criterion vector
c
        subroutine COMPCC(N,M,A,SA,GA,TIP,LEG,CC,V,SG,H,SI,CCI,NA,NL)
        byte n,m,a(1000),leg(100,2),ga(100),v(100),h(100)
        dimension sa(100),tip(100),cc(100),sg(100),si(100)
        do 100 i=1,n
        v(i)=0
        do 100 j=1,n
        if(i-j) 10,100,20
10      l=(j-1)*(j-2)/2+i
        go to 30
20      l=(i-1)*(i-2)/2+j
30      v(i)=v(i)+a(l)
100     continue
        do 200 i=1,n
        is=0
        do 150 j=1,m
        if(leg(j,1).ne.i.and.leg(j,2).ne.i) go to 150
        is=is+int(tip(j))
150     continue
        h(i)=8-ga(i)-is
        if(i.eq.na) h(i)=h(i)-nl
        if(h(i).lt.0) h(i)=0
200     continue
        do 300 i=1,n
        j=h(i)
        sg(i)=(sa(i)*2.592**j)**(1./(1.+j))
300     continue
        do 400 i=1,n
        j=v(i)
        if(i.eq.na) j=j+nl
        si(i)=sg(i)**(1./(1.+j))
400     continue
```

Fig. 14. Subroutine COMPCC (in FORTRAN)

```
      miu=m-n+1
      am=1./(float(miu)+1.)
      cci=0.
      do 500 i=1,m
      i1=leg(i,1)
      i2=leg(i,2)
      cci=cci+tip(i)*(si(i1)*cc(i1)*si(i2) :c(i2))**(-1./2.)
500   continue
      cci=am*cci
      return
      end
```

Fig. 14. (Continued)

References

1. Parts 1 and 2 : M. V. Diudea and I. Silaghi-Dumitrescu, to be published in Rev. Roum. Biochim.

2. F. Harary, "Graph Theory", Addison Wesley, Reading (Mass), 1971 .

3. A. T. Balaban, Theoret. Chim. Acta, 53, 355 (1979).

4. A. T. Balaban, I. Motoc, D. Bonchev and O. Mekenyan, in "Steric Effects in Drug Design", Topics Curr. Chem., 114, 21 (1983),(eds. M. Charton and I. Motoc).

5. D. Bonchev, A. T. Balaban and O. Mekenyan, J. Chem. Inf. Comput. Sci., 20, 106 (1980).

6. D. Bonchev, A. T. Balaban and M. Randić, Internat. J. Quantum Chem., 19, 61 (1981).

7. A. T. Balaban and P. Filip, Math. Chem., 16, 163 (1984).

8. M. Randić, Math. Chem., 7, 5 (1979).

9. A. T. Balaban, Math. Chem., 21, 115 (1986)

10. J. M. Clayton and W. P. Purcell, J. Med. Chem., 12, 1087 (1969).