Asian Journal of Chemistry Vol. 21, No. 5 (2009), 3683-3696

Computing the Szeged and PI Indices of $HAC_5C_7[p,q]$ and $HAC_5C_6C_7[p,q]$ Nanotubes by GAP Program

B. TAHERKHANI†, ALI IRANMANESH* and YASER ALIZADEH
Department of Mathematics, Tarbiat Modares University, P.O. Box: 14115-137, Tehran, Iran
Fax: (98)(21)88006544; Tel: (98)(21)88009730; E-mail: iranmana@modares.ac.ir

In this paper we give a GAP program for computing the Szeged and the PI indices of any graph. We also compute the Szeged and PI indices of $HAC_5C_7[p,q]$ and $HAC_5C_6C_7[p,q]$ nanotubes by this program.

Key Words: Szeged index, PI-index, Nanotubes, GAP programming.

INTRODUCTION

One of the main distinctive characteristics of modern chemistry is the use of theoretical tools for the molecular modeling of physicochemical processes, chemical reaction, medicinal and toxicological events, *etc*., in which chemicals are involved. The success of the molecular modeling is judged by the insights that it offers on the nature of the processes studied, which permit better comprehension and a rational modification of them. These properties, measured experimentally, are almost invariably expressed in quantitative terms, *e.g.*, boiling point, refraction index, transition state energy, percentage of inhibition of some enzymatic activity, lethal dose and so forth. The paradigm for the modeling of such properties is the relationship that exists between them and the molecular chemical structure. This fact presupposes for the first challenge for the molecular modeling. The properties are expressed as number while the molecular structure is not. The way to solve this problem is by using molecular descriptors that are numbers representing information about different molecular features to describe quantitatively the properties under study. These models are known as quantitative structure-property (QSPR) and quantitative structure-activity relationships (QSAR), depending on the physico-chemical or biological nature of the properties studied, respectively.

Topological indices of nanotubes are numerical descriptors that are derived from graph of chemical compounds. Such indices based on the distances in graph are widely used for establishing relationships between the structure of nanotubes and their physicochemical properties. Usage of topological indices in biology and chemistry began in 1947 when chemist Harold Wiener[1] introduced Wiener number and the name of Wiener index was given by Hosoya[2]. Wiener index (W) originally defined on trees and studied its use for correlation of physico-chemical properties

†Department of Mathematics, Payame Noor University, Takestan/Gazvin, 3481699445, Iran.

of alkenes, alcohols, amines and their analogous compounds. A number of successful QSAR studies have been made based in the Wiener index and its decomposition forms[3].

Starting from the middle of the 1970s, the Wiener index gained much popularity and, since then, new results related to it are constantly being reported. For a review, historical details and further bibliography on the chemical applications of the Wiener index[4-6]. Another topological index was introduced by Gutman[7] which was later on named as the Szeged index and abbreviated as Sz[8]. The Szeged index is a modification of Wiener index to cyclic molecules. The Szeged index was conceived by Gutman at the Attila jozsef university in Szeged. This index received considerable attention. It has attractive mathematical characteristics.

Let G be a connected graph. The vertex-set and edge-set of G denoted by V(G) and E(G), respectively. The distance between the vertices u and v, d(u,v), in a graph is the number of edges in a shortest path connecting them. Two graph vertices are adjacent if they are joined by a graph edge. Let e be an edge of a graph G connecting the vertices u and v. Define two sets $N_1(e|G)$ and $N_2(e|G)$ as follows:

$N_1(e|G) = \{x \in V(G)|d(x,u) < d(x,v)\}$ and $N_2(e|G) = \{x \in V(G)|d(x,v) < d(x,u)\}$.

The number of elements of $N_1(e|G)$ and $N_2(e|G)$ are denoted by $n_1(e|G)$ and $n_2(e|G)$, respectively.

The Szeged index of the graph G is defined as $Sz(G) = \sum_{e \in E(G)} n_1(e|G)\, n_2(e|G)$.

For the reason of the coincidence of Wiener and Szeged indices in case of trees, the authors[9,10] introduced another Szeged/Wiener-like topological index and named it Padmakar-Ivan index, abbreviated as PI. In fact the PI index of the graph G is denoted by PI(G) and defined as follows:

$$PI(G) = \sum_{e \in E(G)} n_1(e|G) + n_2(e|G)$$

Applications of the PI index to QSRP/QSAR were studied by Khadikar *et al.*[11]. The index was mostly compared with the Wiener and the Szeged index. It turned out that the PI index has similar discriminating power as the other two indices and in many cases (for instance to model ¢max, the so called difference in doublet of deformation mode, of unbranched cycloalkanes) it gives better result. As already mentioned, the Szeged index incorporates the distribution of vertices of a molecular graph, while the PI index does this job for the edges. Hence it seems that a combination of both could give good results in QSRP/QSAR studies. Indeed, the combination of the PI index and the Szeged index is the best for modeling polychlorinated biphenyls (PCBs) in environment among the 3 possible pairs of indices selected from the PI index, the Szeged index and the Wiener index[12]. For the Wiener and the Szeged index such studied were previously done[7,13]. The Szeged and PI indices of some nanotubes computed by previous workers[14-23].

Algorithm for the computation of the Szeged and PI indices for an arbitrary graph

We give an algorithm that enables us to compute the Szeged and PI indices of any graph. For this purpose, the following algorithm is presented:

1) We assign to any vertex one number.

2) We determine all of adjacent vertices set of the vertex i, $i \in V(G)$ and this set denoted by N(i). The set of vertices that their distance to vertex i is equal to $t(t \geq 0)$ is denoted by $D_{i,t}$ and consider $D_{i,0} = \{i\}$. Let e = ij be an edge connecting the vertices i and j, then we have the following result:

a) $V = U_{t\geq 0} D_{i,t}$, $i \in V(G)$.

b) $(D_{i,t} \setminus D_{j,t}) \subseteq (D_{j,t-1} \cup D_{j,t+1}), t \geq 1.$

c) $(D_{i,t} \cap D_{j,t-1}) \subseteq N_2(e \mid G)$ and $D_{i,t} \cap D_{j,t+1} \subseteq N_1(e \mid G) t \geq 1.$

d) $(D_{i,t} \cup \{i\}) \setminus (D_{i,1} \cup \{j\}) \subseteq N_1(e \mid G)$ and $(D_{j,1} \cup \{j\}) \setminus (D_{i,1} \cup \{i\}) \subseteq N_2(e \mid G).$

According to the above relations by determining $D_{i,t}$, $t \geq 1$, we can obtain $N_1(e|G)$ and $N_2(e|G)$ for each edge e and therefore the Szeged and PI indices of the graph G is computed. In continuation, we obtain the $D_{i,t}$, $t \geq 1$, for each vertex i.

3) The distance between vertex i and its adjacent vertices is equal to 1, therefore $D_{i,1} = N(i)$. For each $j \in D_{i,t}$, $t \geq 1$, the distance between each vertex of set $N(j) \setminus (D_{i,t} \cup D_{i,t-1})$ and the vertex i is equal to t+1, thus we have

$$D_{i,t+1} = \bigcup_{j \in D_{i,t}} (N(J) \setminus (D_{i,t} \bigcup D_{i,t-1}), t \geq 1.$$

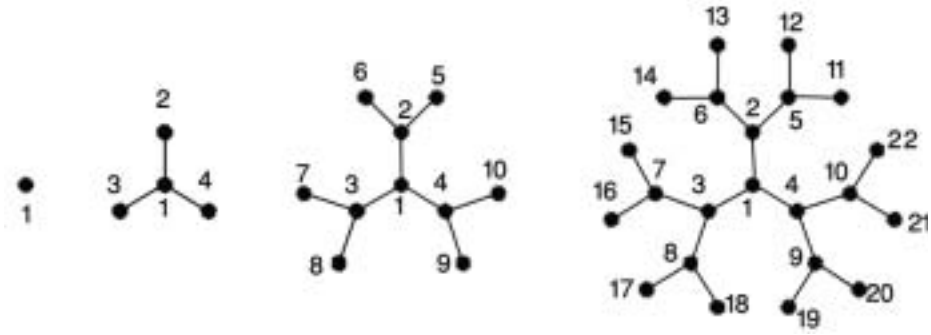
According to the above equation we can obtain $D_{i,t}$ $t \geq 2$ for each $i \in V(G)$.

4) In the start of program we set SZ and PI equal to zero and T equal to empty set. In the end of program the values of SZ and PI are equal to the Szeged and PI indices of the graph G, respectively. For each vertex $i, 1 \leq j \leq n$, and each vertex j in N(i), we determine $N_1(e|G)$ and $N_2(e|G)$ for edge e = ij, then add the values of $n_1(e|G) \cdot n_2(e|G)$ and $n_1(e|G) + n_2(e|G)$ to S Z and PI, respectively. Since the edge ji is equal to ij, we add the vertex i to T and continue this step for the vertex i + 1 and for each vertex in N(i+1)\T.

GAP stands for Groups, Algorithms and Programming[24]. The name was chosen to reflect the aim of the system, which is group theoretical software for solving computational problems in group theory. A rapid spread of interest in the understanding, design and even implementation of group theoretical algorithms. GAP software was constructed by GAP's team in Aachen. We encourage the reader to consult the work of Ashrafi[25] and Trinajstic[26] for background materials and computational techniques related to applications of GAP in solving some problems in chemistry and biology. According to the above algorithm, we prepare a GAP program to compute the Szeged and PI indices of dendrimers $T_{k,3}$.

Example 2.1: The Wiener index of tree dendrimers $T_{k,3}$, $k \geq 1$ is computed by Entringer *et al.*[27] and Gutman *et al.*[28]. Since the Wiener and Szeged index coincide

on trees[29,30], thus the Szeged index of $T_{k,d}$ is equal to its Wiener index. According to the above algorithm, we prepare a GAP program to compute the Szeged and PI indices of $T_{k,3}$, which the yield results can be compared with the results Entringer *et al.*[27] and Gutman *et al.*[28].



$T_{0,3}$, $T_{1,3}$, $T_{2,3}$, $T_{3,3}$

Fig. 1. Denderimers $T_{k,3}$

The following results are obtained[27,28];

$$n(T_{k,3}) = 1 + 3\,(2^k - 1)$$

and its Szeged index is equal to Wiener index, *i.e.*

$$W(T_{k,3}) = (9k - 15)2^{2k} + 18 \times 2^k - 3.$$

For computing of the Szeged and PI indices of $T_{k,3}$ by above program, at first we assign to any vertex one number (Fig. 1). According to this numbering, the set of adjacent vertices to each vertex $1 \le i \le n$ is obtained by the following program (part 1). In fact part 1 of the program is the presentation of the graph. We use the part 2 to compute the Szeged and PI indices of the graph.

```
k:=3;# (For example)
n:=1+3*(2^k -1);
N:=[];
K1:=[2,3,4];
N[1]:=K1;
for i in K1 do
 if k=1 then N[i]:=[1];
 else
  N[i]:=[2*i +1,2*(i+1)];
  Add(N[i],1);fi;
od;
K2:=[5..1+3 *(2^(k-1) -1)];
for i in K2 do
 N[i]:=[2*i +1,2*(i+1)];
 Add(N[i],Int((i-1)/2));
```

```
od;
K3:=[2+3 *(2^(k-1) -1)..n];
for i in K3 do
if k=1 then N[i]:=[1];
else N[i]:=[Int((i-1)/2)];fi;
od;
# (Part2)
D:=[];
for i in [1..n] do
  D[i]:=[];
  u:=[i];
  D[i][1]:=N[i];
  u:=Union(u,D[i][1]);
  s:=1;
  t:=1;
  while s<>0 do
    D[i][t+1]:=[];
    for j in D[i][t] do
      for m in Difference(N[j],u) do
        AddSet(D[i][t+1],m);
      od;
    od;
  u:=Union(u,D[i][t+1]);
     if D[i][t+1]=[] then
    s:=0;
  fi;
  t:=t+1;
  od;
od;
T:=[];
sz:=0;
pi:=0;
for i in [1..n-1] do
N1:=[];
 for j in Difference(N[i],T) do
N2:=[];
   N1[j]:=Union(Difference(N[i],Union([j],N[j])),[i]);
   N2[i]:=Union(Difference(N[j],Union([i],N[i])),[j]);
    for t in [2..Size(D[i])-1] do
      for x in Difference(D[i][t],Union(D[j][t],[j])) do
        if not x in D[j][t-1] then
          Add Set(N1[j],x);
```

```
            elif x  in D[j][t-1] then
             AddSet(N2[i],x);
           fi;
          od;
      od;
        sz:=sz+Size(N1[j])*Size(N2[i]);
        pi:=pi+Size(N1[j])+Size(N2[i]);
       od;
       Add(T,i);
      od;
      sz;# (The value of sz is equal to Szeged index of the graph)
      pi;# (The value of pi is equal to PI index of the graph)
```

Computing the Szeged and PI indices of $HAC_5C_7[p,q]$ nanotube by GAP program

A C_5C_7 net is a trivalent decoration made by alternating C_5 and C_7. It can cover either a cylinder or a torus. In this section we compute the Szeged and PI indices of $HAC_5C_7[p,q]$ nanotube by GAP program.

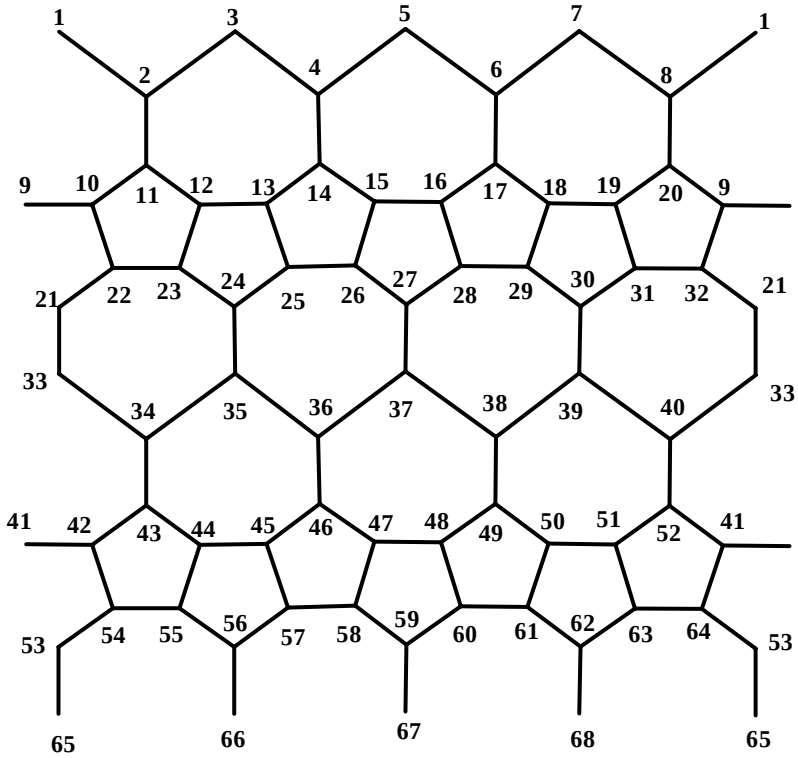


Fig. 2. HAC_5C_7 [4,2] nanotube

We denote the number of heptagons in one row by p. In this nanotube, the three first rows of vertices and edges are repeated alternatively, and we denote the number of this repetition by q. In each period there are 8p vertices and p vertices which are joined to the end of the graph and hence the number of vertices in this nanotube is equal to 8pq + p.

We partition the vertices of this graph to following sets:

K_1: The vertices of first row whose number is 2p.

K_2: The vertices of the first row in each period except the first one whose number is 2p(q-1).

K_3: The vertices of the second rows in each period whose number is 3pq.

K_4: The vertices of the third row in each period whose number is 3pq.

K_5: The last vertices of the graph whose number is p.

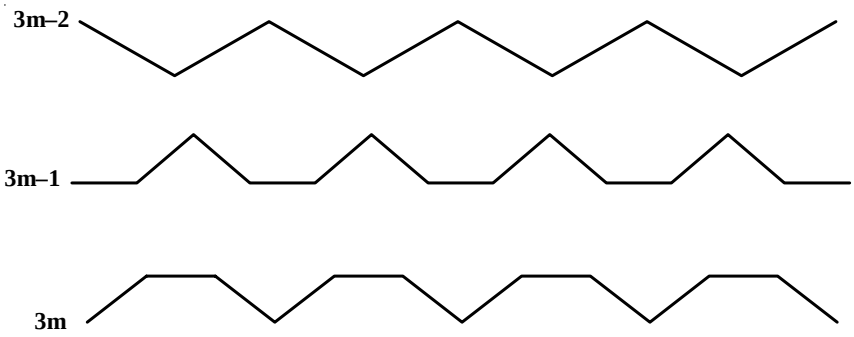


Fig. 3. m-th period of $HAC_5C_7[p,q]$

We write a program to obtain the adjacent vertices set to each vertex in the sets K_i, i = 1…5. We can obtain the adjacent vertices set to each vertex by the join of these programs. In this program, the value of x is the assign number of vertex i in that period.

The following program computes the Szeged and PI indices of $HAC_5C_7[p,q]$ nanotube for arbitrary p and q.

```
p:=3; q:=7;# (For example)
n:=8*p*q +p;
N:=[];
K1:=[1..2*p];
V1:=[2..2*p-1];
N[1]:=[2,2*p];
N[2*p]:=[2*p-1,5*p,1];
for i in V1 do
 if i mod 2=0 then N[i]:=[i-1,i+1,3/2 *i+2*p];
  else N[i]:=[i-1,i+1];fi;
od;
```

```
k:=[2*p+1..8*p*q];
k2:=Filtered(k,i->i mod (8*p)in [1..2*p]);;
for i in k2 do
 x:= i mod (8*p);
 if x mod 2 =1 then N[i]:=[i-1,i+1,(x-1)*(3/2) +1+i-x-3*p];
  else N[i]:=[i-1,i+1,x*(3/2) +2*p+i-x];fi;
 if x=1 then N[i]:=[i+1,i-1+2*p,i-3*p];fi;
 if x=2*p then N[i]:=[i-1,i+3*p,i-2*p+1];fi;
od;

k3:=Filtered(k,i->i mod (8*p) in[2*p+1..5*p]);;
for i in k3 do
 x:=i mod (8*p);
 if (x-(2*p)) mod 3 =1 then N[i]:=[i-1,i+1,i+3*p-1];
  elif (x-(2*p)) mod 3 =2 then N[i]:=[i-1,i+1,i+3*p];
   elif (x-(2*p)) mod 3 =0 then N[i]:=[i-1,i+1,(2/3) *(x-2*p)+i-x];fi;
 if x=2*p+1 then N[i]:=[i-1+3*p,i-1+6*p,i+1];fi;
 if x=5*p   then N[i]:=[i-3*p,i-3*p+1,i-1];fi;
od;

k4:=Filtered(k,i->i mod (8*p) in Union([5*p+1..8*p-1],[0]) );;
for i in k4 do
 x:=i mod (8*p);
 if (x-(5*p)) mod 3 =1 then N[i]:=[i-1,i+1,(x-(5*p)-1)*(2/3) +1+(i-x)+8*p];
  elif (x-(5*p)) mod 3 =2 then N[i]:=[i-1,i+1,i-3*p];
   elif (x-(5*p)) mod 3 =0 then N[i]:=[i-1,i+1,i-3*p+1];fi;
 if x=5*p+1 then N[i]:=[i+3*p-1,i+1,i+3*p];fi;
 if x=0 then N[i]:=[i-1,i-3*p+1,i-6*p+1];fi;
od;
K5:=[8*p*q+1 ..8*p*q+p];

for i in K5 do
 x:=i-8*p*q;
 y:=8*p*(q-1)+5*p+3*x-2;
 N[i]:=[y];
 N[y][3]:=i;
od;
D:=[];
for i in [1..n] do
   D[i]:=[];
   u:=[i];
   D[i][1]:=N[i];
```

```
   u:=Union(u,D[i][1]);
   s:=1;
   t:=1;
   while s<>0 do
    D[i][t+1]:=[];
    for j in D[i][t] do
     for m in Difference(N[j],u) do
      AddSet(D[i][t+1],m);
     od;
    od;
   u:=Union(u,D[i][t+1]);
     if D[i][t+1]=[] then
    s:=0;
   fi;
   t:=t+1;
   od;
od;
A:=[];
T:=[];
sz:=0;
pi:=0;
e:=[];
for i in [1..n-1] do
N1:=[];
 for j in Difference(N[i],T) do
N2:=[];
   N1[j]:=Union(Difference(N[i],Union([j],N[j])),[i]);
   N2[i]:=Union(Difference(N[j],Union([i],N[i])),[j]);
    for t in [2..Size(D[i])-1] do
     for x in Difference(D[i][t],Union(D[j][t],[j])) do
      if not x in D[j][t-1] then
       AddSet(N1[j],x);
      elif x  in D[j][t-1] then
       AddSet(N2[i],x);
      fi;
     od;
    od;
  sz:=sz+Size(N1[j])*Size(N2[i]);
  pi:=pi+Size(N1[j])+Size(N2[i]);
 od;
 Add(T,i);
od;
```

```
sz;# The value of sz is equal to szeged index of the graph
pi;# The value of pi is equal to pi index of the graph
```

Computing the Szeged and PI indices of $HAC_5C_6C_7$[p,q] nanotube by GAP program

A $C_5C_6C_7$ net is a trivalent decoration made by alternating C_5, C_6 and C_7. It can cover either a cylinder or a torus. In this section we compute the Szeged and PI indices of $HAC_5C_6C_7$ nanotube similar to the previous section.

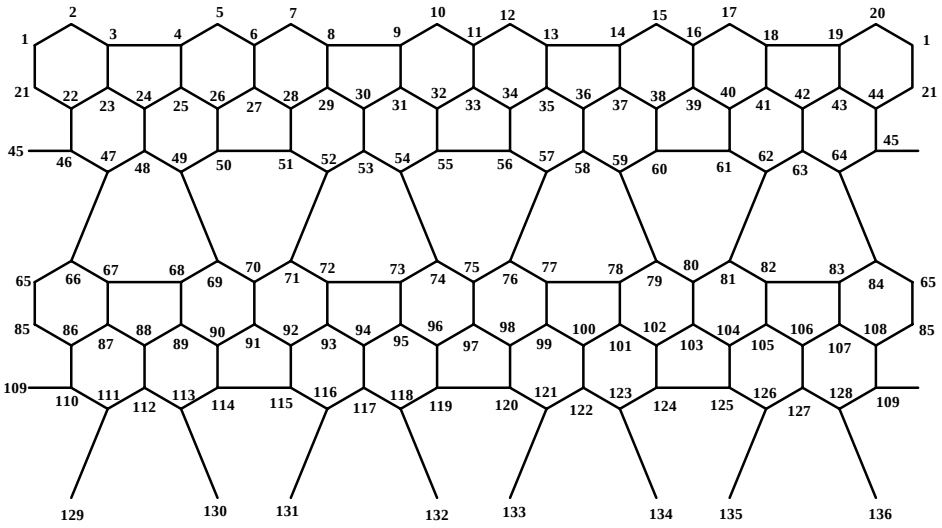


Fig. 4. $HAC_5C_6C_7$[4,2] nanotube

We denote the number of pentagons in the first row by p. In this nanotube the three first rows of vertices and edges are repeated alternatively; we denote the number of this repetition by q. In each period, there are 16p vertices and 2p vertices are joined to the end of the graph and hence the number of vertices in this nanotube is equal to 16pq + 2p.

The following program is the same as the last program.

```
p:=8; q:=3; # (For example)
n:=16*p*q+2*p;
N:=[];
K1:=[1..5*p];
V1:=[2..5*p-1];
for i in V1 do
 if i mod 5 =1 then N[i]:=[i-1,i+1,(i-1)*(6/5)+1+5*p];
  elif (i mod 5) in [0,2] then N[i]:=[i-1,i+1];
```

```
   elif i mod 5=3 then N[i]:=[i-1,i+1,(i-3)*(6/5) +3+5*p];
    elif i mod 5=4 then N[i]:=[i-1,i+1,(i-4)*(6/5)+5+5*p]; fi;
od;
N[1]:=[2,5*p,5*p+1]; N[5*p]:=[1,5*p-1];
k:=[5*p+1..16*p*q];

k2:=Filtered(k,i->i mod (16*p) in [1..5*p]);
for i in k2 do
 x:=i mod (16*p);
 if x mod 5=1 then N[i]:=[i-1,i+1,(x-1)*(6/5) +1+i-x+5*p];
  elif x mod 5=2 then N[i]:=[i-1,i+1,i-5*p+1];
   elif x mod 5=3 then N[i]:=[i-1,i+1,(x-3)*(6/5)+3+i-x+5*p];
    elif x mod 5=4 then N[i]:=[i-1,i+1,(x-4)*(6/5)+5+i-x+5*p];
     elif x mod 5=0 then N[i]:=[i-1,i+1,i-5*p];fi;
 if x=1 then N[i]:=[i+1,i-1+5*p,i+5*p];fi;
 if x=5*p then N[i]:=[i-1,i-5*p,i-5*p+1];fi;
od;

k3:=Filtered(k,i->i mod(16*p) in [5*p+1..11*p]);
for i in k3 do
 x:= (i-5*p) mod (16*p);
 if x mod 6=1 then N[i]:=[i-1,i+1,(x-1)*(5/6)+ i-x-5*p+1];
  elif x mod 6=2 then N[i]:=[i-1,i+1,(x-2)*(5/6)+2+i-x+6*p];
   elif x mod 6=3 then N[i]:=[i-1,i+1,(x-3)*(5/6)+3+i-x-5*p];
    elif x mod 6=4 then N[i]:=[i-1,i+1,(x-4)*(5/6)+4+i-x+6*p];
     elif x mod 6=5 then N[i]:=[i-1,i+1,(x-5)*(5/6)+4+i-x-5*p];
      elif x mod 6=0 then N[i]:=[i-1,i+1,x*(5/6)+1+i-x+6*p];fi;
 if x=1 then N[i]:=[i+1,i+6*p-1,i-5*p];fi;
 if x=6*p then N[i]:=[i-1,i+1,i-6*p+1];fi;
od;

k4:=Filtered(k,i-> i mod (16*p) in Union([11*p+1..16*p-1],[0]));
for i in k4 do
 x:=(i-11*p) mod (16*p);
  if x mod 5 =1 then N[i]:=[i-1,i+1,(x-1)*(6/5)+i-x-6*p];
   elif x mod 5 =2 then N[i]:=[i-1,i+1,(x-2)*(6/5)+2+i-x-6*p];
    elif x mod 5 =3 then N[i]:=[i-1,i+1,i-1+5*p];
     elif x mod 5 =4 then N[i]:=[i-1,i+1,(x-4)*(6/5)+4+i-x-6*p];
      elif x mod 5 =0 then N[i]:=[i-1,i+1,i+5*p];fi;
 if x=1 then N[i]:=[i-1,i+1,i-1+5*p];fi;
 if x=5*p then N[i]:=[i-1,i-5*p+1,i+5*p];fi;
od;
```

```
K5:=[16*p*q+1..n];
for i in K5 do
 x:=i-16*p*q;
 if x mod 2=0 then
    y:=(5/2)*x +16*p*q -5*p;
  else  y:=(5/2 )*(x-1) +3+16*p*q -5*p; fi;
 N[i]:=[y];
 N[y][3]:=i;
od;
D:=[];
for i in [1..n] do
   D[i]:=[];
   u:=[i];
   D[i][1]:=N[i];
   u:=Union(u,D[i][1]);
   s:=1;
   t:=1;
   while s<>0 do
     D[i][t+1]:=[];
     for j in D[i][t] do
       for m in Difference(N[j],u) do
         AddSet(D[i][t+1],m);
       od;
     od;
   u:=Union(u,D[i][t+1]);
      if D[i][t+1]=[] then
     s:=0;
   fi;
   t:=t+1;
   od;
od;
A:=[];
T:=[];
sz:=0;
pi:=0;
e:=[];
for i in [1..n-1] do
N1:=[];
  for j in Difference(N[i],T) do
N2:=[];
    N1[j]:=Union(Difference(N[i],Union([j],N[j])),[i]);
    N2[i]:=Union(Difference(N[j],Union([i],N[i])),[j]);
```

```
      for t in [2..Size(D[i])-1] do
       for x in Difference(D[i][t],Union(D[j][t],[j])) do
        if not x in D[j][t-1] then
          AddSet(N1[j],x);
        elif x  in D[j][t-1] then
         AddSet(N2[i],x);
        fi;
       od;
      od;
     sz:=sz+Size(N1[j])*Size(N2[i]);
     pi:=pi+Size(N1[j])+Size(N2[i]);
    od;
    Add(T,i);
  od;
  sz; # The value of sz is equal to Szeged index of the graph
  pi; # The value of pi is equal to PI index of the graph
```

ACKNOWLEDGEMENT

This research is supported by Payame Noor University, Iran.

REFERENCES

1. H. Wiener, *J. Am. Chem. Soc.*, **69**, 17 (1947).
2. H. Hosoya, *Bull. Chem. Soc. (Japan)*, **44**, 2322 (1971).
3. M.V. Diudea, M.S. Florescu and P.V. Khadikar, Molecular Topology and Its Applications, EFICON Press, Bucharest (2006).
4. S. Nikolic, N. Trinajstic and Z. Mihalic, *Croat. Chem. Acta*, **68**, 105 (1995).
5. I. Gutman and J.H. Potgieter, *J. Serb. Chem. Soc.*, **62**, 185 (1997).
6. I. Gutman, Y.N. Yeh, S.L. Lee and J.C. Chen, *MATCH Commun. Math Comput. Chem.*, **30**, 103 (1994).
7. P.P. Khadikar, N.V. Deshpande, P.P. Kale, A.A. Dobrynin, I. Gutman and G. Domotor, *J. Chem. Inf. Comput. Sci.*, **35**, 545 (1995).
8. P.V. Khadikar, S. Karmarkar, V.K. Agrawal, J. Singh, A. Shrivastava, I. Lukovits and M.V. Diudea, *Lett. Drug. Design. Disco.*, **2**, 606 (2005).
9. P.V. Khadikar, *Nat. Acad. Sci. Lett.*, **23**, 113 (2000).
10. P.V. Khadikar, S. Karmarkar and V.K. Agrawal, *Nat. Acad. Sci. Lett.*, **23**, 165 (2000).
11. P.V. Khadikar, S. Karmarkar and V.K. Agrawal, *J. Chem. Inf. Comput. Sci.*, **41**, 934 (2001).
12. P.V. Khadikar and S. Karmarkar, *Acta Chim. Slov.*, **49**, 755 (2002).
13. K.C. Mathur, S. Singh, S. Mathur and P.V. Khadikar, *Pollut. Res.*, **18**, 405 (1999).
14. A.R. Ashrafi, M. Ghorbani and M. Jalali, *J. Theoret. Comput. Chem.*, **7**, 221 (2008).
15. A.R. Ashrafi and A. Loghman, *J. Comput. Theor. Nanosci.*, **3**, 378 (2006).
16. H. Deng, *MATCH Commun. Math Comput. Chem.*, **55**, 461 (2006).
17. A. Iranmanesh, Y. Alizadeh and B. Taherkhani, *Int. J. Molecul. Sci.*, **9**, 131 (2008).
18. A. Iranmanesh and N. Gholami, *Micro & Nano Lett.*, **4**, 107 (2007).
19. A. Iranmanesh and Y. Pakravesh, *Utilitas Mathematica*, **75**, 89 (2008).
20. A. Iranmanesh, Y. Pakravesh and A. Mahmiani, *Ars Combinatoria*, **87**, 193 (2008).
21. A. Iranmanesh, B. Soleimani and A. Ahmadi, *J. Comput. Theor. Nanosci.*, **4**, 147 (2007).
22. A. Iranmanesh and B. Soleimani, *MATCH Commun. Math Comput. Chem.*, **57**, 251 (2007).

23. A. Mahmiani, A. Iranmanesh and Y. Pakravesh, *Ars Combinatoria*, **89** (2008).
24. M. Schonert, H.U. Besche, T.H. Breuer, F. Celler, B. Eick, V. Felsch, A. Hulpke, J. Mnich, W. Nickel, G. Pfeiffer, U. Polis and H. Theiben, Niemeyer, GAP, Groups, Algorithms and Programming, Lehrstuhl D fuer Mathematik, RWTH, Aachen (1992).
25. A.R. Ashrafi, *MATCH Commun. Math. Comput. Chem.*, **53**, 161 (2005).
26. N. Trinajstic, Chemical Group Theory, CRC Press, Boca Roton, FL (1992).
27. R.C. Entringer, A.J. Meir, W. Moon and L.A. Székely, *Australas J. Combin.*, **10**, 211 (1994).
28. I. Gutman, Y.N. Yeh, S.L. Lee and Y.L. Luo, *Indian J. Chem.*, **32A**, 651 (1993).
29. I. Gutman, Graph Theory Notes, New York Academy of Science (1994).
30. S. Karmarkar, S. Karmarkar, S. Joshi, A. Das and P.V. Khadikar, *J. Serb. Chem. Soc.*, **62**, 227 (1997).

(*Received*: 16 June 2008; *Accepted*: 17 February 2009) AJC-7241