Indian Journal of Fundamental and Applied Life Sciences ISSN: 2231– 6345 (Online)
An Open Access, Online International Journal Available at www.cibtech.org/sp.ed/jls/2015/02/jls.htm
2015 Vol. 5 (S2), pp. 2234-2238/Bahrami and Attarzadeh
Review Article

COMPUTING TOPOLOGICAL INDEX OF SOME TYPE OF NANOTUBES AND NANOTORI BY ONE PROGRAM

[*]Amir Bahrami[1] and Nima Attarzadeh[2]
[1] Young Researcher and Elite Club, Garmsar Branch, Islamic Azad University, Garmsar, Iran
[2] Department of Computer Engineering, Mahshahr Branch, Islamic Azad University, Mahshahr, Iran
**Author for Correspondence*

ABSTRACT
The Wiener index *W* is the sum of distances between all pairs of vertices of a (connected) graph. In this paper, we will explained one program (by Visual Basic language), for Computing Wiener index of nanotubes and nanotori which have heptagons.

Keywords: *Topological Index, Wiener Index, Molecular Graphs, Nanotubes, Nanotori*

INTRODUCTION

In this article, all graphs are finite, undirected, connected, without loops and multiple edges. The vertex and edge sets of a graph *G* are *V (G)* and *E(G)*. The numbers of vertices and edges of *G* are denoted by *p* = *pG* and *q* = *qG*, respectively.
Under *distance dG(u, v)* between vertices *u, v* ∈ *V (G)* we mean the standard distance of the simple graph *G*, i.e., the number of edges on a shortest path connecting these vertices in *G* (Buckley *et al.*, 1990). The *distance of a vertex v* ∈ *V (G)*, $d_G(v)$, is the sum of distances between *v* and all other vertices of *G*. The *Wiener index* is a graph invariant based on distances in a graph. It is denoted by *W(G)* and defined as the sum of distances between all pairs of vertices in *G*:

$$W(G) = \sum_{u,v \in V(G)} d(u,v) = \sum_{v \in V(G)} d_G(v) \qquad (1)$$

The name *Wiener index* or *Wiener number* for the quantity defined in Equation (1) is usual in chemical literature, since Harold Wiener (Wiener, (1947), in 1947, seemed to be the first to consider it. Wiener himself used the name *path number*, but denoted his quantity by *w*. Wiener's original definition was slightly different – yet equivalent – to (1). The definition of the Wiener index in terms of distances between vertices of a graph, such as in Equation (1), was first given by Hosoya (1971). 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 see references: (Hosoya, 1988), (Huang *et al.*, 1996), (Huang *et al.*, 1997), (Imrich *et al.*, 2000) and (Juvan *et al.*, 1995).
In this paper, we refer to one program for computing the wiener index of nanotubes and nanotori. Through this paper, our notation is standard and is similar to (Buckley *et al.*, 1990).

RESULTS AND DISCUSSION

In this section we will explained one program (by Visual Basic language), for Computing Wiener index of nanotubes and nanotori which have heptagons.

```
Imports System
Friend Java As A
Public Class Hepta
()Public Sub New
For i As Integer = 0 To layers
()LayersInfo(i) = New layer
Next i
For i As Integer = 0 To num_vertices
()vertices(i) = New node
```

```
Next i
End Sub
Public layers As Integer=14
Public num_vertices As Integer=2000
Public edges As Integer=0
,num_vertices+1)Return Rectangular Integer Array  .RectangularArrays = Public adj()() As Integer
(num_vertices+1
,num_vertices+1)Return Rectangular Integer Array.RectangularArrays = Public dist()() As Integer
(num_vertices+1
Public vertices(num_vertices) As node
Public LayersInfo(layers) As layer
(String Public Overridable Sub CalculateFloyd(ByVal S As
Dim i, j, k As Integer
Me.LayersInfo(Me.layers).tail For i = 0 To
Me.LayersInfo(Me.layers).tail For j = 0 To
If Me.adj(i)(j)=0 Then
Me.dist(i)(j)=9999
Else
(Me.dist(i)(j)=Me.adj(i)(j
End If
Me.dist(i)(i)=0
Next j
Next i
For k = 1 To Me.LayersInfo(Me.layers).tail
For i = 1 To Me.LayersInfo(Me.layers).tail
Me.LayersInfo(Me.layers).tail For j = 1 To
Me.dist(i)(j) Then>(Me.dist(i)(k)+Me.dist(k)(j If
(Me.dist(k)(j+(Me.dist(i)(j)=Me.dist(i)(k
End If
Next j
Next i
Next k
Try
(Dim o As New FileOutputStream(S
(Dim bs As New BufferedOutputStream(o
(Dim ps As New DataOutputStream(bs
Me.LayersInfo(Me.layers).tail For i = 1 To
Me.LayersInfo(Me.layers).tail For j = 1 To
((ps.writeInt(Me.dist(i)(j
Next j
Next i
()ps.close
()o.close
Catch e As Exception
End Try
End Sub
(Public Overridable Sub LoadFloyd(ByVal S As String
Dim i, j As Integer
Try
(Dim fr As New FileInputStream(S
```

(Dim br As New DataInputStream(fr
Me.LayersInfo(Me.layers).tail For i = 1 To
Me.LayersInfo(Me.layers).tail For j = 1 To
()Me.dist(i)(j)=br.readInt
Next j
Next i
()br.close
()fr.close
Catch e As Exception
End Try
End Sub
()Friend Overridable Sub initnodes
For i As Integer = 1 To num_vertices
vertices(i).degree=0
vertices(i).number=i
Next i
End Sub
(As Integer, ByVal value As Integer Friend Overridable Sub [set](ByVal i As Integer, ByVal j
edges += 1
adj(i)(j)=value
adj(j)(i)=value
vertices(i).degree += 1
vertices(j).degree += 1
End Sub
()Friend Overridable Sub calculatelayers
LayersInfo(1).number=7
LayersInfo(1).head=1
LayersInfo(1).tail=7
For i As Integer = 2 To layers
LayersInfo(i-1).number+14=LayersInfo(i).number
i-1).tail+1)LayersInfo(i).head=LayersInfo
i).head+LayersInfo(i).number-1)LayersInfo(i).tail=LayersInfo
Next i
End Sub
()Friend Overridable Sub initlayer
()calculatelayers
For i As Integer = 1 To layers - 1
LayersInfo(i).tail - 1 For j As Integer = LayersInfo(i).head To
(set](j,j+1,1]
Next j
(LayersInfo(i).tail,1,set](LayersInfo(i).head]
Next i
End Sub
Integer) As Integer Friend Overridable Function findfirstdegree2(ByVal l As
LayersInfo(l).tail For i As Integer = LayersInfo(l).head To
If vertices(i).degree=2 Then
Return i
End If
Next i
Return -1

```
End Function
()Public Overridable Sub initnetwork
()initlayer
Dim temp, [iterator] As Integer
For i As Integer = 2 To layers
(temp= findfirstdegree2(i-1
(head,1.(set](temp, LayersInfo(i]
iterator]=LayersInfo(i).head]
For k As Integer = 1 To 7
iterator]=[iterator]+3]
(temp= findfirstdegree2(i-1
Then 0<If temp
(set](temp,[iterator],1]
End If
For l As Integer = 1 To i-2
iterator]=[iterator]+2]
(temp=findfirstdegree2(i-1
Then 0<If temp
(set](temp,[iterator],1]
End If
Next l
Next k
Next i
End Sub
End Class
Partial Friend Class Rectangular Arrays
ByVal Size1 As Integer, ByVal Size2 As)Return Rectangular Integer Array Friend Shared Function
()()Integer) As Integer
{} ()(1 - Dim Array As Integer()() = New Integer(Size1
For Array1 As Integer = 0 To Size1 - 1
{} (Array(Array1) = New Integer(Size2 - 1
Next Array1
Return Array
End Function
End Class
```

ACKNOWLEDGEMENT

The first author is supported from "Young Researchers and Elite Club, Garmsar Branch, Islamic Azad University, Garmsar, Iran", for research project entitled:" Computing Topological Indices of some type of Nanotubes and Nanotori". The authors would like to thank for supporting this project.

REFRENCES

Buckley F and Harary F (1990). *Distance in Graphs* (Addison-Wesley) Redwood.

Hosoya H (1971). Topological index. A newly proposed quantity characterizing the topological nature of structural isomers of saturated hydrocarbons. *Bulletin of Chemical Society of Japan* **4** 2332–2339.

Hosoya H (1988). On some counting polynomials in chemistry. *Discrete Applied Mathematics* **19** 239–257.

Huang IW, Yang BY and Yeh YN (1996). Wiener indices of hex carpets – from hexagon models to square grids. *SEA Bulletin of Mathematics* **20** 81–102.

Huang WC, Yang BY and Yeh YN (1997). From ternary strings to Wiener indices of benzenoid chains. *Discrete Applied Mathematics* **73** 113–131.

Imrich W and Klavžar S (2000). *Product Graphs: Structure and Recognition* (Wiley) New York.
Juvan M, Mohar B, Graovac A, Klavžar S and Žerovnik J (1995). Fast computation of the Wiener index of fasciagraphs and rotagraphs. *Journal of Chemical Information and Computer Sciences* **35** 834–840.
Wiener H (1947). Structural determination of paraffin boiling points. *Journal of American Chemical Society* **69** 17–20.