COMPUTING THE WEIGHTED WIENER AND SZEGED NUMBER ON WEIGHTED CACTUS GRAPHS IN LINEAR TIME

Blaž Zmazek    Janez Žerovnik

ISSN 1318-4865

April 24, 2002

Ljubljana, April 24, 2002
Computing the weighted Wiener and Szeged number on weighted cactus graphs in linear time

Blaž Zmazek a, b*** and Janez Žerovnik a, b***

a Faculty of Mechanical Engineering, University of Maribor, Smetanova 17, SI-2000 Maribor, Slovenia.
b Department of Theoretical Computer Science, IMFM, Jadranska 19, SI-1111 Ljubljana, Slovenia.

Cactus is a graph in which every edge lies on at most one cycle. Linear algorithms for computing the weighted Wiener and Szeged numbers on weighted cactus graphs are given. Graphs with weighted vertices and edges correspond to molecular graphs with heteroatoms.

---

* Supported in part by the Ministry of Science and Technology of Slovenia under the grant J2-1015-0101.
** E-mail: blaz.zmazek@uni-mb.si.
*** E-mail: janez.zerovnik@imfm.uni-lj.si.
1 Introduction

A topological index is a numerical quantity derived in an unambiguous manner from the structural graph of a molecule. These indices are graph invariants, which usually reflect molecular size and shape.

The first non-trivial topological index in chemistry was introduced by H. Wiener\(^1\) in 1947 to study boiling points of paraffins (for historical data see, for instance Ref. 2). Since then, the Wiener number (also called the Wiener index) has been used to explain various chemical and physical properties of molecules and to correlate the structure of molecules to their biological activity.\(^3\) The research interest in Wiener index and related indices is still considerable.\(^4,5\)

Wiener originally defined his index on trees and studied its use for correlations of physicochemical properties of allanes, alcohols, amines, and other analogous compounds. In arbitrary tree, every edge is a bridge, i.e. after deletion of the edge, the graph is no more connected. The contribution to the Wiener number of an edge was taken to be the product of the numbers of vertices in the two connected components. This number also equals the number of all shortest paths in the tree which go through the edge.\(^6\) Therefore the usual generalization of the Wiener number on arbitrary graphs is defined to be the sum of all distances in a graph. Recalling and generalizing the original definition, edge contributions to the Wiener number were studied in Refs. 7-9.

Another natural generalization was recently put forward and called the Szeged number, $$S_z$$.\(^10\) Now the weights of edges are taken to be the product of the numbers of vertices closer to the two endpoints of the edge. For reasons to introduce the Szeged index and for basic properties of $$S_z$$ see Refs. 10,11. Formulas or special algorithms for Szeged index of several families of graphs were proposed recently.\(^12-14\)

The motivation for study of weighted graphs and their topological indices is the fact that the substitution of carbon atoms in a molecular graph with heteroatoms corresponds to weighting the vertices and edges of the graph.

The main result of this paper are linear algorithms for the weighted Wiener and the weighted Szeged number on weighted cactus graphs. Cactus graphs are generalized trees in which cycles are allowed, but any two cycles have at most one vertex in common.
2 Definitions

A weighted graph $G = (V, E, w, \lambda)$ is a combinatorial object consisting of an arbitrary set $V = V(G)$ of vertices, a set $E = E(G)$ of unordered pairs $\{x, y\} = xy$ of distinct vertices of $G$ called edges, and two weighting functions, $w$ and $\lambda$. $w : V(G) \to \mathbb{R}$ assigns positive real numbers (weights) to vertices and $\lambda : E(G) \to \mathbb{R}$ assigns positive real numbers (lengths) to edges.

A simple path from $x$ to $y$ is a finite sequence of distinct vertices $P = x_0, x_1, \ldots, x_{\ell}$ such that each pair $x_{i-1}, x_i$ is connected by an edge and $x_0 = x$ and $x_{\ell} = y$. The length of the path is the sum of lengths of its edges, $l(P) = \sum_{i=1}^{\ell} \lambda(x_{i-1}, x_i)$. For any pair of vertices $x, y$ we define the distance $d(x, y)$ to be the minimum of lengths over all paths between $x$ and $y$. If there is no such path, we set $d(x, y) = \infty$.

A graph $G$ is connected, if $d(x, y) < \infty$ for any pair of vertices $x, y$. A vertex $v$ is a cut vertex if after removing $v$ and all edges incident to it the resulting graph is not connected. A graph without a cut vertex is called nonseparable. A block is a maximal nonseparable graph. A cycle is an induced subgraph which is connected and in which every vertex is of degree two. A cactus is a graph in which every block of three or more vertices is a cycle. Alternatively, a cactus is a connected graph in which two cycles have at most one vertex in common. For usual graph theoretical terminology not defined here see, for instance, Ref. 15.

The weighted Wiener number of a weighted graph $G$ studied here is

$$W(G) = \sum_{u \neq v} w(u)w(v)d(u, v). \quad (1)$$

It seems that weighted Wiener number has not been studied frequently in the literature. A definition, analogous to (1) was used in Refs. 16 and 17 for vertex weighted graphs. A different definition in which the “weights” of atoms are added to the sum of distances is used in Ref. 18. We find the definition (1) more natural from mathematical point of view and look forward to learn about possible chemical interpretations of this and other definitions. It may be worth to remark that in the definition of Ref. 18 the weighted Wiener number is simply the usual Wiener number plus the sum of weights of vertices and hence it can be computed by standard algorithms.18

The weighted Szeged number of a weighted graph $G$ is defined as

$$Sz(G) = \sum_{e \notin V} s_u(e)s_v(e)\lambda(e), \quad (2)$$

where $s_u(e) = \sum_{x \in V \text{, } d(x, u) < d(x, v)} w(x)$ (and $s_v(e) = \sum_{x \in V \text{, } d(x, u) > d(x, v)} w(x)$).
The definitions used here are clearly generalizations of the usual definitions for (unweighted) Wiener and Szeged numbers. More precisely, if all weights of vertices are 1 and all lengths of edges are 1, then \( W(G) \) and \( Sz(G) \) are the usual Wiener and Szeged numbers.

**Lemma 1** For a weighted graph \( G \), \( W(G) = \sum_{e \in E} \lambda(e) \times \sum_{P_{a,b}^{*} \in \mathcal{P}} \frac{1}{n^{*}(a,b)} w(a)w(b) \), where \( P_{a,b}^{*} \) is a shortest path between \( a \) and \( b \) and \( n^{*}(a,b) \) is the number of shortest paths with endpoints \( a \) and \( b \).

**Proof.** To see this it is enough to sum up the contributions of each edge to \( W \) in two different ways. Each pair \( a, b \) of vertices contributes \( w(a)w(b)d(a,b) \) to the Wiener number. This contribution can be either regarded as a contribution of the pair \( a, b \) or it can be divided to \( n^{*}(a, b) \) path contributions which can furthermore be regarded as a sum of edge contribution along the path. Since \( d(a,b) \) is the sum of edge weights along a shortest path, the contribution of edge \( e \) caused by \( a \) and \( b \) is

\[
\lambda(e) \frac{1}{n^{*}(a,b)} w(a)w(b).
\]

An edge contributes as many times as it appears on various shortest paths. Here the vertices are weighted, so one has to take into account the weights of both terminal vertices. Let us note in passing that the sum of contributions of the edge \( e \),

\[
\omega^{*}(e) := \sum_{P_{a,b}^{*} \in \mathcal{P}} \frac{1}{n^{*}(a,b)} w(a)w(b).
\]

can be called the **shortest path weight distribution**, generalizing the ideas of Refs. 7–9.

Hence one can sum up the lengths of all shortest paths (as in (1)), or, equivalently, sum up the contributions of all edges. \( \square \)

Recall that on a tree, there is a unique shortest path between any pair of vertices. Hence \( n^{*}(a,b) = 1 \) for all \( a, b \) which implies that the definition of the weighted Wiener number and the weighted Szeged number are equivalent on trees.

**Lemma 2** For a weighted tree \( T \), \( W(T) = Sz(T) \).

This lemma is a generalization of a well-known result for unweighted Wiener number, which was already known to Wiener.\(^{20}\)

For a later reference note that on a cactus graph \( n^{*}(a,b) \) can only have value 1 or 2. Clearly, \( n^{*}(a,b) = 2 \) exactly when \( a \) and \( b \) are opposite vertices of a cycle (i.e. \( d(a,b) = d \) and \( a, b \) are vertices of a cycle of girth \( 2d \)).
3 DFS based representation of a cactus

If in a cactus cycles are considered as (super)-vertices, there is a unique shortest path between any two vertices.

Any cactus $G$ can always be represented as a rooted cactus. This means that a vertex $v_0 \in V(G)$ is distinguished and called the root of $G$. Any vertex may be chosen for the root. All other vertices are indexed as $v_1, v_2, \ldots, v_{n-1}$ in a DFS (depth first search) order. DFS is a standard procedure for searching a graph, see for example Ref. 21. It can easily be seen that in each cycle there is exactly one vertex $v_j$, which is the first vertex in the DFS order on the cycle $C$, containing the edge $v_jv_i$. For this reason we call the vertex $v_j$ the root of the cycle $C$. Clearly, each cycle in a cactus has a unique root, determined as a first vertex on the cycle in the DFS order. Moreover, each vertex on a cactus $G$ which is a successor of a vertex with higher index is the root vertex of one or more cycles. Hence, all cycles in a cactus are determined with the root and its successor.

The rooted cactus can be represented with two arrays $T_i$ and $R_i$, $i = 0, 2, \ldots, n-1$, where $T_i$ denotes the unique predecessor of vertex $v_i$ in the rooted tree, constructed with the DFS. The element $R_i$ denotes the root vertex of the cycle containing $v_i$. If $v_i$ does not lie on a cycle then $R_i = v_i$ and if $v_i$ is on a cycle rooted at $v_j$, then $R_i = v_j \neq v_i$. Note that the array $R$ describes all cycles in the cactus.

It is clear that when traversing the graph with a DFS algorithm, one can also obtain array $\text{ind}(v_i) := \{j : v_i = T_j\}$. The meaning of $\text{ind}(v_i)$ is the number of sons of $v_i$. Leaves of the tree have no son directing to them. Hence $\text{ind}(v) = 0$. This can be done within the time complexity of BFS, $O(m)$.

After a DFS run, each vertex $v_i$ can be regarded as a root of a subcactus, which we will denote by $G_i$.

From the discussion above we infer that after a DFS run, one can get a representation of a cactus and, furthermore, one can decide whether a given graph is a cactus or not. More formally,

**Lemma 3** Let $G$ be a cactus graph. In a DFS run, each new vertex $v_i$ can have at most one neighbor among the visited vertices and the neighbor can only be on the direct path from the root $v_0$ to $v_i$.

**Proof.** If, for some vertex two such neighbors existed, then there will be edges on two (or more cycles). Contradiction. A back neighbor which is not on a direct path from the root $v_0$ to $v_i$ was visited already and hence the vertex $v_i$ should have been visited as its son. Contradiction. \(\square\)
Hence, if \( G \) is not a cactus graph, this can be observed while traversing \( G \) in the DFS order.

**Lemma 4**  Cactus graphs can be recognized in linear time.

**Proof.** DFS algorithm takes \( O(m) \) time. \( \square \)

4 The algorithms

We will give the algorithms Sz and W for computing the indices (Szeged and Wiener number) of a rooted subcactus. Algorithms for computing Sz and W differ only in Step 3 while the Steps 1 and 2 are identical for both algorithms. Recall that by Lemma 1 the Wiener number can be computed as a sum of contributions over all edges of a graph.

With each vertex \( v_i \) of the graph \( G \) the value of \( V_i \) will be computed, for which

\[ V_i = \sum_{j \in G_i} w(v_j) \]

will eventually hold. \( V_i \) will thus be the sum of weights of the subcactus \( G_i \) rooted at \( v_i \).

Both algorithms consist of three steps. First, a representation of the given weighted cactus is found. Then we compute \( V_i \) for each vertex. In order to compute the \( V_i \) we start with edges incident to leaves and continue in reversed DFS order up to root. In the third step we compute the contributions of edges (cycles) to the invariant starting from root to leaves.

1. Find a representation of the (rooted) cactus \( G \).
2. Traverse the cactus \( G \) in the reversed DFS order and compute the \( V_i \).
3. Traverse the cactus \( G \) in the DFS order and compute the contributions of edges (cycles) to the invariant.

We now give more details of each step.

**Step 1: Cactus recognition**
WEIGHTED WIENER NUMBER

Run a DFS on \( G \). By discussion above, it is clear that we can compute \( T_i, R_i \) and \( ind(v_i) := \{ j : v_i = T_j \} \) in time \( O(m) \).

**Step 2: Computation of \( V_i \)**

 Traverse the DFS tree in the reversed DFS order. Compute the temporal value \( V_i \) for a leave \( v_i \) and reduce the number of unvisited sons for its father by one. If the father has no more unvisited sons, put it in a FIFO (first in first out) queue. The computation of temporal \( V_i \) is the weight of \( v_i \) plus the sum of \( V_j \) of its sons:

\[
V_i = w(v_i) + \sum_{T_j = v_i} V_j.
\]

**Lemma 5** After Step 2, for every vertex \( v_i \), \( V_i \) is the sum of weights of all vertices of the subactus rooted at \( v_i \). More formally,

\[
V_i = \sum_{v_j \in G_i} w(v_j).
\]

**Proof.** It follows from the definition of \( V_i \). \( \square \)

**Lemma 6** Step 2 can be computed in \( O(m) \) time.

**Proof.** Clear. \( \square \)

**Step 3a: Computation of the edge contributions to \( W(G) \)**

 Traverse the DFS tree in the DFS order. Compute the edge contributions to the \( W(G) \).

 Observe that \( V_0 \) is the sum of weights of all vertices of \( G \).

1. \( W = 0 \)

2. For each vertex \( v_i \) (in the DFS order) do
   - For each son \( v_j \), for which \( v_i v_j \) is not an edge of a cycle of \( G \), (i.e., if \( R_j = v_j \) and \( T_j = v_i \)) set
     \[
     W_e := V_j (V_0 - V_j) \lambda(e)
     \]
     \[
     W := W + W_e.
     \]
WEIGHTED WIENER NUMBER

- For each son $v_j$ of a root vertex $v_i$ ($R_i = v_i$), for which $v_iv_j$ is an edge of a cycle $C_j$ of $G$, (i.e., if $R_j = v_i$ and $T_j = v_i$), compute the contributions of the cycle as follows. Mark the vertices of the cycle $C$ in the DFS order with $u_0, \ldots, u_{e-1}$. For each vertex $u_i$ on the cycle compute $L_i$, the sum of weights of vertices for which the vertex $u_i$ is the nearest vertex on the cycle $C$, using the values $V_j$ and let $|C| = \sum_{i=0}^{e-1} \lambda(u_iu_{i+1})$ be the girth of the cycle.

- $W_C := 0$; (weighted Wiener number of the cycle $C$)
  $f = l := 0$; (loop counters - $f$ is first, $l$ is last vertex - are computed modulo $e$)
  $Cont = 0$; (contribution of the current vertices \{u_f, u_{f+1}, \ldots, u_l\})
  $V := 0$; (sum of weights of \{u_f, u_{f+1}, \ldots, u_l\})
  $d := 0$ (distance from $u_f$ to $u_l$)

- Repeat
  (compute the contributions of edges on the cycle which lies on shortest paths that enter on the cycle through the vertex $v_j$ and continue in a DFS order)

(a) While $d + \lambda(u_lu_{l+1}) \leq \frac{|C|}{2}$ do begin
* $l := l + 1;
* V := V + L_l;
* d := d + \lambda(u_lu_{l+1});
* Cont := Cont + L_l d$
end(while) (Find the new last vertex $u_l$ on the cycle $C_j$ with a shortest path from $u_f$ in a DFS order.)

(b) $W_C := W_C + L_f Cont$

(c) If $d = \frac{|C|}{2}$ then $W_C := W_C - \frac{1}{2} L_f L_l d$

(d) $Cont := Cont - V \lambda(u_ju_{j+1})$

(e) $V := V - L_{j+1}$

(f) $d := d - \lambda(u_ju_{j+1})$

(g) $f := f + 1$

Until $f = 0$.

- $W := W + W_C$. 

Step 3b: Computation of the edge contributions to $S_z(G)$

Traverse the DFS tree in the DFS order. Compute the edge contributions to the $S_z(G)$.

Observe that $V := V_0$ is the sum of weights of all vertices of $G$.

1. $S_z = 0$

2. For each vertex $v_i$ (in the DFS order) do
   - For each son $v_j$, for which $v_iv_j$ is not an edge of a cycle of $G$, (i.e., if $R_j = v_j$ and $T_j = v_i$) set
     \[ S_{ze} := V_j(V - V_j)\lambda(e) \]
     \[ S_z := S_z + S_{ze}. \]
   - For each son $v_j$ of a root vertex $v_i$ ($R_i = v_j$), for which $v_iv_j$ is an edge of a cycle $C_j$ of $G$, (i.e., if $R_j = v_i$ and $T_j = v_i$), compute the contributions of the cycle as follows. Mark the vertices of the cycle $C$ in the DFS order with $u_0, \ldots, u_{\varepsilon-1}$. For each vertex $u_i$ on the cycle compute $L_i$, the sum of weights of vertices for which the vertex $u_i$ is the nearest vertex on the cycle $C$, using the values $V_j$ and let $|C| = \sum_{i=0}^{\varepsilon-1} \lambda(u_iu_{i+1})$ be the girth of the cycle.
     - $S_{zC} := 0$; (weighted Szeged number of the cycle $C$)
     - $f = l := 0$; (loop counters - $f$ is first, $l$ is last vertex - are computed modulo $e$)
     - $V_R := 0$; $V_L := L_0$; (sum of weights of vertices right (R) and left (L) of an edge)
     - $d := \frac{\lambda(u_\varepsilon u_0)}{2}$;
     - Repeat
       (compute the contribution of the edge $u_{f-1}u_f$)
       (a) While $d + \lambda(u_{f-1}u_f) \leq \frac{|C|}{2}$ do begin
          * $l := l + 1$
          * $V_L := V_L + L_l$
          * $d := d + \lambda(u_{l-1}u_l)$
       end (while)
       (b) $V_R := V - V_L$
       (c) If $d = \frac{|C|}{2}$ then $S_{zC} := S_{zC} + V_R(V_L - L_l)\lambda(u_{f-1}u_f)$
       else $S_{zC} := S_{zC} + V_R V_L \lambda(u_{f-1}u_f)$;
WEIGHTED WIENER NUMBER

(d) \( V_L := V_L - V_I \);
(e) \( d := d - \frac{\lambda(u_i u_{i+1}) + \lambda(u_i - u_{i+1})}{2} \);
(f) \( f := f + 1 \)
Until \( f = 0 \).

\[ S_Z := S_Z + S_{Z_C} \]

**Example 1** Compute weighted Wiener and Szeged number of path from Fig. 1.

\[ \begin{array}{ccccccc}
10 & 5 & 1 & 1 & 1 & 1 & 5 & 10 \\
\end{array} \]

*Figure 1 Example 1.*

Traversing the path we get the following table

<table>
<thead>
<tr>
<th>( i )</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>( T_i )</td>
<td>−</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>( \lambda(u_i T_i) )</td>
<td>0</td>
<td>5</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>( V_i )</td>
<td>24</td>
<td>14</td>
<td>13</td>
<td>12</td>
<td>11</td>
<td>10</td>
</tr>
<tr>
<td>( V_i(V_0 - V_i)\lambda(u_i T_i) )</td>
<td>0</td>
<td>700</td>
<td>143</td>
<td>144</td>
<td>143</td>
<td>700</td>
</tr>
</tbody>
</table>

Summarizing the last row of the table we get the Wiener number of the path \( W = 1830 \), which also coincides with the Szeged number.

**Example 2** Compute weighted Wiener and Szeged number on a cycle from Fig. 2. Let the top vertex of the cycle in Fig. 2 be also the root vertex. All steps of computation of Wiener and Szeged numbers are written in the tables below (the entries are computed columnwise from left to right):

<table>
<thead>
<tr>
<th>Step</th>
<th>( f )</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td>( l )</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>0</td>
</tr>
<tr>
<td>( V )</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>6</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>( d )</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>(Cont)</td>
<td>2</td>
<td>4</td>
<td>10</td>
<td>15</td>
<td>10</td>
<td>16</td>
<td>13</td>
<td>13</td>
</tr>
<tr>
<td>(b), (c)</td>
<td>( W_C )</td>
<td>12.5</td>
<td>30.5</td>
<td>38</td>
<td>60</td>
<td>70.5</td>
<td>86</td>
<td>120</td>
</tr>
<tr>
<td>(d)</td>
<td>( Cont )</td>
<td>9</td>
<td>5</td>
<td>6</td>
<td>8</td>
<td>5</td>
<td>12</td>
<td>10</td>
</tr>
<tr>
<td>(e)</td>
<td>( V )</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>(f)</td>
<td>( d )</td>
<td>4</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>3</td>
</tr>
</tbody>
</table>
In Fig. 3 we illustrate the seven steps in the computation of Wiener (Szeged) number of the cycle.

Summarizing, we read from the tables the values $W(C) = 120$ and $S_{z}(C) = 224$.

**Lemma 7** $W_{e}$ and $W_{C}$ are correctly computed. Each edge is considered exactly once.

**Proof.** (sketch) After line (a) in 2. of Step 3a, the value of $V$ is the sum of weights of vertices which are at distance less or equal to $|C|/2$ from the vertex $u_f$. Similarly, the value $Cont$ is the weighted sum

$$Cont = \sum_{i=1}^{f} w(u_i) d(u_f, u_i) = \sum_{i=1}^{f} \sum_{i=1}^{l} w(u_i) \lambda(u_{i-1}, u_i).$$
Hence

\[ w(u_f)\text{Cont} = \sum_{i=1}^{f} \sum_{i=1}^{l} w(u_f)w(u_i)\lambda(u_{i-1}, u_i). \]

is the sum of contributions of all paths with the first vertex \( u_f \) which are oriented coherent with the orientation of the cycle defined by the labeling \( u_0, u_1, \ldots \).

If there are vertices of distance exactly \( |C|/2 \) then there are exactly two shortest paths between them and the contribution has to be divided by 2 (hence step (c)).

Finally observe that in the repeat-until loop each shortest path shorter than \( |C|/2 \) is considered exactly once and that each shortest path of length \( |C|/2 \) is considered exactly twice. We omit the details. \( \square \)

**Lemma 8** Sz and SzC are correctly computed. Each edge is considered exactly once.

**Proof.** (sketch) After line (a) in 2. of Step 3b, the value of \( V_L \) is the sum of weights of vertices which are at distance less or equal to \( |C|/2 \) from the center of edge \( u_{j-1}u_f \). If there is a vertex exactly at distance \( |C|/2 \) it is neither closer to \( u_{j-1} \) nor to \( u_f \) and hence this case has to be treated separately (see (c)). Clearly, in the repeat-until loop each edge is considered exactly once. We omit the details. \( \square \)

**Lemma 9** Step 3a can be computed in \( O(m) \) time. Step 3b can be computed in \( O(m) \) time.

**Proof.** (idea) Each cycle is considered only once (when its root is traversed). Furthermore, the computation on the cycle is linear in the length of the cycle, which easily follows by analysis of the repeat-until loop. \( \square \)

Recalling the complexity of Steps 1 to 3 we conclude that both algorithms run in linear time.

**Proposition 10** The above algorithms compute the weighted Wiener and the weighted Szeged numbers on a weighted cactus in \( O(m) \) time.

**REFERENCES**


5. MATCH 35 (1997). (Special issue on the Wiener index.)


This article was processed using the L\(\text{T}\)e\(\text{X}\) macro package with Croatian style
Figure 3 Seven steps in computation of Wiener and Szeged number of the cycle.