COMPUTING WIENER INDEX OF FIBONACCI WEIGHTED TREES

K. R. Udaya Kumar Reddy[1] and Ranjana S. Chakrasali[2]

[1, 2]Department of Computer Science and Engineering, BNM Institute of Technology, Bangalore-560070, Karnataka, India

{krudaykumar@yahoo.com, ranjanagirish@gmail.com}

ABSTRACT

Given a simple connected undirected graph $G = (V, E)$ with $|V| = n$ and $|E| = m$, the Wiener index $W(G)$ of G is defined as half the sum of the distances of the form $d(u, v)$ between all pairs of vertices u, v of G. If (G, w_E) is an edge-weighted graph, then the Wiener index $W(G, w_E)$ of (G, w_E) is defined as the usual Wiener index but the distances is now computed in (G, w_E). The paper proposes a new algorithm for computing the Wiener index of a Fibonacci weighted trees with Fibonacci branching in place of the available naive algorithm for the same. It is found that the time complexity of the algorithm is logarithmic.

KEYWORDS

Algorithms, Distance in graphs, Fibonacci weighted tree, Wiener index.

1. INTRODUCTION

Let $G = (V(G), E(G))$ be a connected unweighted undirected graph without self-loops and multiple edges. Let $|V(G)| = n$ and $|E(G)| = m$.

The Wiener index $W(G)$ of G is defined as half the sum of the distances between all pairs of vertices of a graph G. Wiener index is a distance based graph invariant which is one of the most popular topological indices in mathematical chemistry. It is named after the chemist Harold Wiener, who first introduced it in 1947 to study chemical properties of alkanes. It is not recognized that there are good correlations between $W(G)$ and physico-chemical properties of the organic compound from which G is derived, especially when G is a tree. Wiener index have been studied quite extensively in both the mathematical and chemical literature. For chemical applications of Wiener index, see [7, 9]. The Wiener index is also studied to investigate a related quantity the average distance (defined as $2W(G)/n(n-1)$)of a graph, which is frequently done in pure mathematics [3].

In this paper we are concerned with a tree called Fibonacci weighted tree with Fibonacci branching. Let $\eta = 1 + F_1 + F_2 + F_3 + \prod_{i=1}^{4} F_i + \ldots + \prod_{i=1}^{k} F_i$ be the number of vertices in T_k, where $F_i = i$-th Fibonacci number. One way to compute the Wiener index of Fibonacci weighted tree with Fibonacci branching is to compute the distances between all pairs of vertices of a graph. It is known [2] that the straightforward approach for solving the distances on a weighted graph between all pairs of vertices of G is to run Floyd-Warshall algorithm which takes a time $O(n^3)$; thus for Fibonacci weighted tree with Fibonacci branching of order k with η vertices, such an algorithm can compute the Wiener index in time $O(\eta^3)$ and requires as an input a description of Fibonacci weighted tree with Fibonacci branching of order k, e.g., an adjacency matrix. In this note, we propose a new algorithm for computing the Wiener index of Fibonacci weighted tree

Natarajan Meghanathan, et al. (Eds): SIPM, FCST, ITCA, WSE, ACSIT, CS & IT 06, pp. 471–478, 2012.

DOI : 10.5121/csit.2012.2346

with Fibonacci branching in time O(log η), assuming that the input is only the order k of the Fibonacci weighted tree with Fibonacci branching.

2. PRELIMINARIES

The Wiener index $W(G)$ of G is defined as

$$W(G) = \frac{1}{2}\sum_{u\in V(G)}\sum_{v\in V(G)} d(u,v)\,, \tag{1}$$

where $d(u,v)$ denotes the distance (the number of edges on a shortest path between u and v between u, v in G.

Wiener index $W(G)$ comes under different names such as sum of all distances [5, 10], total status [1], gross status [6], graph distance [4], and transmission [8]. A related quantity is the average distance $\mu(G)$ defined as

$$\mu(G) = \frac{2W(G)}{n(n-1)}.$$

Let $w(i,j)$ denote the edge weight on the edge {i, j}. Then

$$w(i,j) = \begin{cases} weight\ of\ edge\ (i,j)\ if\ (i,j) \in E(G), \\ +\infty \qquad\qquad\qquad if\ (i,j) \in E(G). \end{cases}$$

Consider an edge-weighted graph G with weight function $w_E : E(G) \rightarrow R^+$ denoted as (G, w_E). Then the *weight* of a path is the sum of the weights of its edges on that path. A *shortest path* between two vertices u and v is a path of minimum weight. The *shortest-path distance* $d_{(G,w_E)}(u,v)$ (or simply $d(u,v)$) is the sum of the weights of the edges along the shortest path connecting u and v. For $u \in V(G)$ and $H \subseteq V(G)$, let $d^+(u,H) = \sum_{v\in H} d(u,v)$. The *Wiener index* $W(G,w_E)$ of (G,w_E) is defined as the usual Wiener index, that is, $W\,(G,w_E) = \frac{1}{2}\sum_{u\in v(G)}\sum_{v\in V(G)} d(u,v)$ where $d(u,v)$ is now computed in (G,w_E). Clearly if all the edges have weight one, then $W(G,w_E) = W(G)$. In the sequel, for notational convenience we assume that $W(T) = W(T,w_E)$.

It is well known that the Fibonacci numbers are defined recursively as follows: (i) The Fibonacci numbers $F_0 = 0$ and $F_1 = 1$, and (ii) For $k \geq 2$, the Fibonacci number $F_k = F_{k-1} + F_{k-2}$.

We define Fibonacci weighted path P_{f_n} of order n, as a path on n + 1 vertices, where the consecutive edges are assigned weights $F_1, \ldots, F_n$ starting from an edge incident on a pendent vertex.

Let k be a positive integer. The *Fibonacci weighted tree with Fibonacci branching* T_k of order k, is defined recursively in the following way:

i. $T_1 = (V_1, E_1)$is a rooted tree, where $V_1 = \{v_1^0, v_1^1\}$ and $E_1 = \{(v_1^0, v_1^1)\}$, with $w(v_1^0, v_1^1) = F_1$.

ii. $T_2 = (V_1 \cup V_2, E_1 \cup E_2)$ is a rooted tree, where $V_2 = \{v_1^2\}$ and $E_2 = \{(v_1^1, v_1^2)\}$, with $w(v_1^0, v_1^1) = F_1$ and $w(v_1^1, v_1^2) = F_2$.

iii. For $k \geq 3$, the rooted tree T_k is constructed as follows:

Let $p = \prod_{i=1}^{k-2} F_i$, $q = pF_{k-1}$ and $r = qF_k$. Let $V = (V_1 \cup \ldots \cup V_{k-1})$ and $E = (E_1 \cup \ldots \cup E_{k-1})$, where $V_{k-1} = \{v_1^{k-1}, \ldots, v_q^{k-1}\}$ and $E_{k-1} = \{(v_i^{k-2}, v_j^{k-1}) : 1 \leq i \leq p, 1 \leq j \leq q$ and $(i-1)F_{k-1} + 1 \leq j \leq iF_{k-1}\}$. If $T_{k-1} = (V, E)$ is a rooted tree, then $T_k = (V \cup V_k, E \cup E_k)$, where $V_k = \{v_1^k, \ldots, v_r^k\}$ and $E_k = \{(v_i^{k-1}, v_j^k): 1 \leq i \leq q, 1 \leq j \leq r$ and $(i-1)F_k + 1 \leq j \leq iF_k\}$ and $\forall\,(u,v) \in E_k, w(u,v) = F_k$.

Figure 1 shows the Fibonacci weighted trees with Fibonacci branching T_1 through T_4.

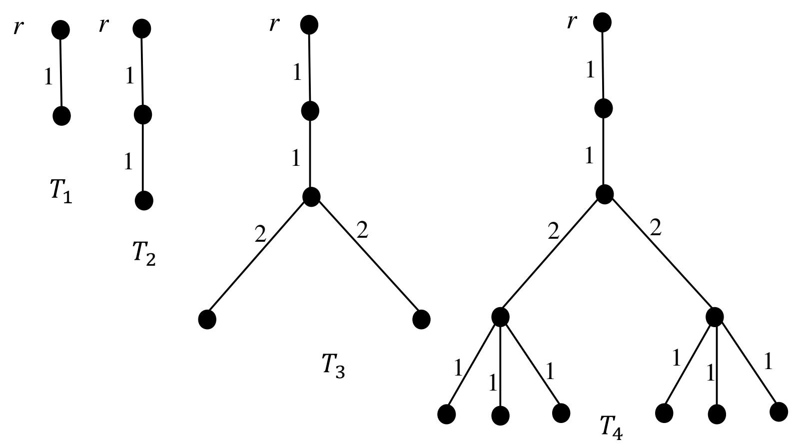

Figure 1: Fibonacci weighted trees with Fibonacci branching T_1 through T_4.

3. COMPUTING WIENER INDEX OF FIBONACCI WEIGHTED TREES WITH FIBONACCI BRANCHING

We begin with the following lemma which gives a closed-form expression for $W(P_{f_n})$.

Lemma 1:
Let P_{f_n} be a Fibonacci weighted path with n + 1 vertices. Then for n ≥ 2, the Wiener index $W(P_{f_n})$ is given by

$$W(P_{f_n}) = n(F_{n+4} + 2) - 2F_{n+5} + 10 . \quad (2)$$

Proof. From the Fibonacci weighted path P_{f_n}, it is clear that

$$W(P_{f_n}) = W(P_{f_{n-1}}) + \sum_{j=1}^{n} j\, F_j . \quad (3)$$

with initial condition $W(P_{f_1}) = 1$. Equation (3) can be simplified to

$$W(P_{f_n}) = W(P_{f_{n-1}}) + nF_{n+2} + 2 - F_{n+3} . \quad (4)$$

Simplifying (4) gives the desired expression for $W(P_{f_n})$ as given in (2).

Lemma 2:
Let $r = \prod_{i=1}^{k} F_i$ and $Z = \{ v_i^k \mid 1 \le i \le r \}$. For $1 \le j \le r$, let $X_{1,j-1} = v_1^k, \dots v_{j-1}^k$ and $Y = V(T_k) \setminus X_{1,j-1}$. For a positive integer $k \ge 5$, let T_k be the Fibonacci weighted tree with Fibonacci branching of order *k*. Then $DIST(T_k)$ is given by

$$DIST(T_k) = \sum_{v_j^k \in Z} d^+(v_j^k, Y) = 2DIST'(T_k)$$

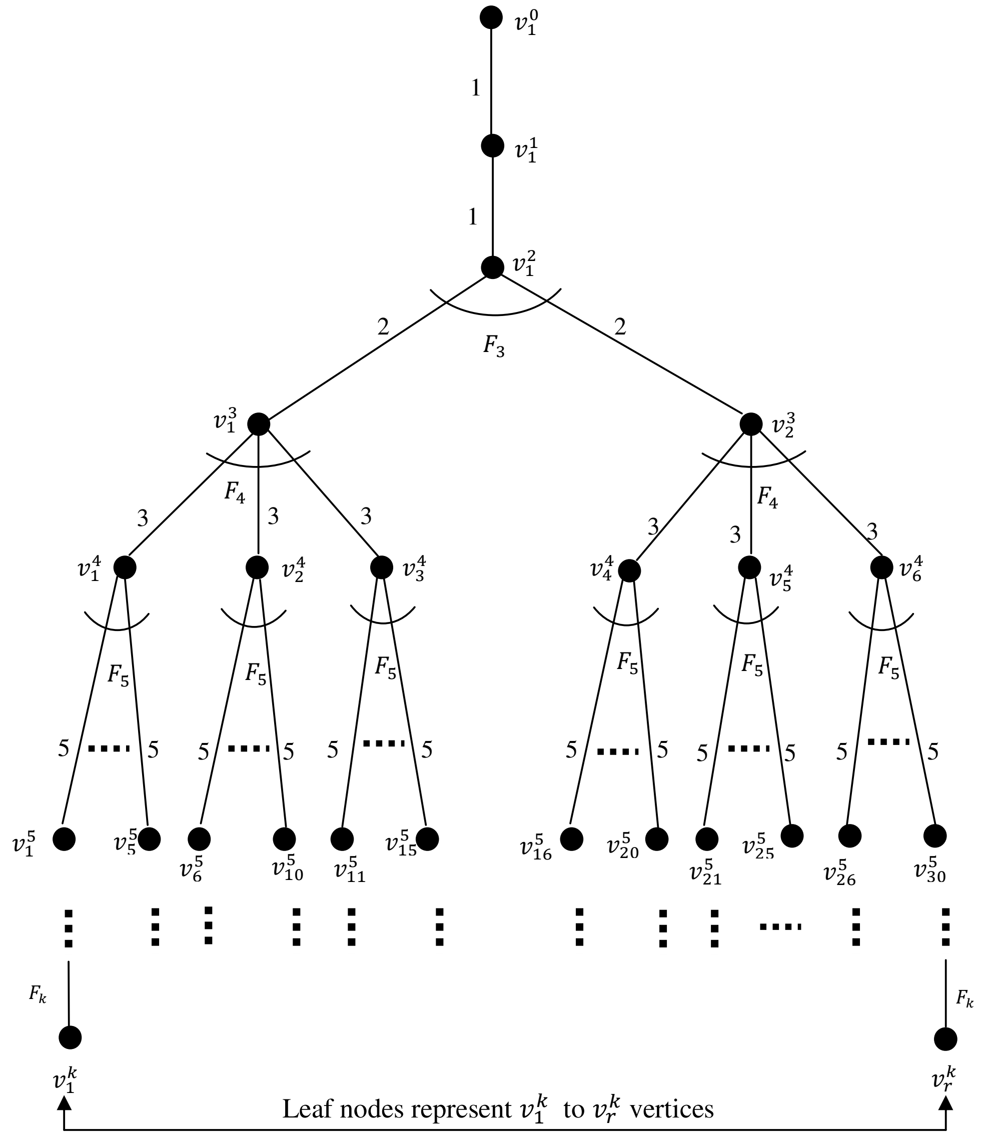

Figure 2: Fibonacci weighted tree with Fibonacci branching T_k.

where,

$$
\begin{aligned}
DIST'(T_k) \;=\; & \prod_{i=4}^{k} F_i \left[\sum_{\substack{i=1 \\ j=i-1}}^{k-1} (F_{k+2} - 1) - (F_{j+2} - 1) \right] \\
& + \prod_{i=4}^{k} F_i \left[(F_{k+2} - 1) \sum_{i=4}^{k-1} \left(\prod_{j=4}^{i} F_j \left((F_{k+2} - 3) + (F_{i+2} - 3) \right) \right) \right] \\
& + F_4\, 2 \prod_{i=5}^{k} F_i \left[(F_{k+2} - 2) \sum_{i=5}^{k-1} \left(\prod_{j=5}^{i} F_j \left((F_{k+2} - 5) + (F_{i+2} - 5) \right) \right) \right] \\
& + 6 \prod_{i=5}^{k} F_i^2 (F_{k+2} - 5) + \prod_{i=4}^{k-1} F_i F_k^3 + \prod_{i=4}^{k} F_i^2 (F_{k+2} - 3) \,. \qquad (5)
\end{aligned}
$$

Proof. Consider the Fibonacci weighted tree with Fibonacci branching shown in Fig. 2. Let the leftmost path $P_1 = v_1^0 v_1^1 v_1^2 \ldots v_1^{k-2} v_1^{k-1} v_1^k$. We begin by finding the shortest-path distance from v_1^0 to v_1^k. That is,

$$d(v_1^0, v_1^k) = \sum_{i=0}^{k} w(v_1^i, v_1^{i+1}) . \\ = \sum_{i=1}^{k} F_i . \tag{6}$$

Let the Wiener index on path P_1 denote D_{P_1}. Then similar to (6), we can write as D_{P_1} as

$$D_{P_1} = \sum_{i=1}^{k} F_i + \sum_{i=2}^{k} F_i + \ldots \ldots + \sum_{i=k-2}^{k} F_i + \sum_{i=k-1}^{k} F_i .$$

Since in the left subtree there are $\frac{r}{2} = \prod_{i=4}^{k} F_i$ paths originating from v_1^0, the Wiener index on all such paths is

$$\prod_{i=4}^{k} F_i \ D_{P_1} . \tag{7}$$

Consider the path $P_2 = v_1^k v_1^{k-1} \ldots v_1^3 v_1^2$ followed by v_2^3 . Then

$$d(v_1^k, v_2^3) = \sum_{i=3}^{k} F_i + F_3 . \tag{8}$$

In Fig. 2 we can see that the node v_2^3 has F_4 branches. Similar to (8), for a path P_2 followed by $v_2^3 v_l^4$, $4 \le l \le 6$, we get

$$F_4(d(v_1^k, v_2^3) + F_4) = F_4 \left(\sum_{i=3}^{k} F_i + \sum_{i=3}^{4} F_i\right) . \tag{9}$$

Let $S = \{v_j^{k-1} : \frac{q}{2} + 1 \le j \le q\}$. Now consider the paths from v_2^3 to all vertices $u \in S$ preceded by the path P_2 . Then similar to (9), the shortest-path distances on such paths is

$$d(v_1^k, v_1^2) + d^+(v_1^2, S) = F_4 \left(\sum_{i=3}^{k} F_i + \sum_{i=3}^{4} F_i\right) + F_4 F_5 \left(\sum_{i=3}^{k} F_i + \sum_{i=3}^{5} F_i\right) + \\ \ldots F_4 F_5 \ldots F_{k-1}\left(\sum_{i=3}^{k} F_i + \sum_{i=3}^{k-1} F_i\right) . \tag{10}$$

Let $D_{P_2} = d(v_1^k, v_2^3) + d(v_1^k, v_1^2) + d^+(v_1^2, S)$. Since there are $r/2$ paths in the subtree originating from v_1^2 down the tree, the shortest-path distances on all such paths and on the paths considered in (8) and (10), we get

$$\prod_{i=4}^{k} F_i \ D_{P_2} . \tag{11}$$

Let the path $P_3 = v_1^k v_1^{k-1} \ldots v_1^3$. Let $x = \prod_{i=5}^{k} F_i$ and $y = \prod_{i=4}^{k-1} F_i$. Let $P_3^4 = P_3 v_2^4$, $P_3^5 = P_3^4 v_i^5$ $(6 \le i \le 10)$, $P_3^6 = P_3^5 v_j^6$ $(6 \le i \le 10,\ 41 \le j \le 80)$,…,$P_3^{k-1} = P_3 v_2^4 v_i^5 v_j^6 \ldots v_z^{k-1}$ $(6 \le i \le 10,\ 41 \le j \le 80$ and $y+1 \le z \le 2y)$. Then similar to (8)-(10), the shortest-path distances on paths $P_3^4, P_3^5, \ldots, P_3^{k-1}$ denoted D_{P_3} is

$$D_{P_3} = \left(\sum_{i=4}^{k} F_i + F_4\right) + F_5\left(\sum_{i=4}^{k} F_i + \sum_{i=4}^{5} F_i\right) + \ldots + F_5 F_6 \ldots F_{k-1}\left(\sum_{i=4}^{k} F_i + \sum_{i=4}^{k-1} F_i\right) . \tag{12}$$

Since there are x paths that originates from v_1^3 followed by v_1^4 down the tree, the shortest-path distances on all such x paths and on paths $P_3^3, P_3^4, \ldots, P_3^{k-1}$, (12) now becomes

$$\prod_{i=5}^{k} F_i \ D_{P_3} . \tag{13}$$

Since there F_4 branches at node v_1^3 and since each branch down the tree from v_1^3 (3rd-level to kth-level of the tree) has t paths, the shortest-path distances on all such x paths of one branch and on the paths of the other two branches up to k – 1 th-level of the tree, (13) now becomes

$$2F_4 \prod_{i=5}^{k} F_i \ D_{P_3} . \tag{14}$$

Let $P_3^k = P_3^{k-1} v_{z'}^k$, where $x+1 \le z' \le 2x$. Now consider three branches down the tree at node v_1^3. Since at v_1^3 each branch down the tree (up to kth-level) has x paths, we compute the shortest-path distances on paths mentioned below:

- From the leftmost branch consisting of x paths starting from v_j^k, $1 \le j \le x$, to the other two branches each consisting of x paths ending with v_l^k, $x+1 \le l \le 3x$, we get

$$2x^2\left(\sum_{i=4}^{k} F_i + \sum_{i=4}^{k} F_i\right) = 4x^2 \sum_{i=4}^{k} F_i \,. \tag{15}$$

- From the middle branch consisting of x paths starting from v_j^k, $x+1 \le j \le 2x$, to the rightmost branch consisting of x paths ending with v_l^k, $2x+1 \le l \le 3x$, similar to (15), we get

$$2x^2 \sum_{i=4}^{k} F_i \,. \tag{16}$$

Since $x = \prod_{i=5}^{k} F_i$, adding equation (15) and (16) yields

$$6 \prod_{i=5}^{k} F_i^2 \sum_{i=4}^{k} F_i \,. \tag{17}$$

Let D_1 denote addition of (7), (11), (14) and (17). That is

$$D_1 = \prod_{i=4}^{k} F_i \, D_{P_1} + \prod_{i=4}^{k} F_i \, D_{P_2} + 2F_4 \prod_{i=5}^{k} F_i \, D_{P_3} + 6 \prod_{i=5}^{k} F_i^2 \sum_{i=4}^{k} F_i \,. \tag{18}$$

Observe that all the paths considered in deriving (18) begin with a vertex labelled in the left subtree of T_k. Similarly, if we consider all such similar paths that begin with a vertex in the right subtree of T_k we get an expression identical to (18). Thus considering paths that begin with a vertex both in the left subtree and in the right subtree, we get

$$2D_1 \,. \tag{19}$$

Finally, the following paths are considered:
Let the leftmost path $P_l = P_2$ and rightmost path $P_r = v_2^3 v_6^4 \ldots v_p^{k-2} v_q^{k-1} v_r^k$. Let $P_{lr} = P_l P_r$. Clearly the shortest-path distances on P_{lr} leads to the formula

$$d\left(v_1^2, v_1^k\right) + d\left(v_1^2, v_r^k\right) = 2 \sum_{i=3}^{k} F_i \,. \tag{20}$$

Since each of the left subtree and the right subtree of T_k at k-th level consists of $\frac{r}{2} = \prod_{i=5}^{k} F_i$ nodes, implies that there exist r/2 paths on either side of T_k. Thus the shortest-path distances on such paths starting from vertex v_i^k, $1 \le i \le r/2$, to vertex $v_j^k, \frac{r}{2}+1 \le j \le r$, (20) can be extended by taking $(r/2)^2$ paths as

$$2 \prod_{i=4}^{k} F_i^2 \sum_{i=3}^{k} F_i \,. \tag{21}$$

Let $Q = \left\{v_1^k, v_2^k, \ldots, v_{F_{k-1}}^k, v_{F_k}^k\right\}$ be a subset of leaf nodes at k-th level of the left subtree. Clearly $|Q| = F_k$, and the shortest-path distances between all vertex pairs in Q is F_k^3. Since there are $q = 2 \prod_{i=4}^{k-1} F_i$ nodes at k-1th-level of T_k. Thus the shortest-path distances between all vertex pairs in q vertex sets, we get

$$qF_k^3 = 2 \prod_{i=4}^{k-1} F_i F_k^3 \,. \tag{22}$$

Therefore, adding (19), (21) and (22) yields $DIST(T_k)$ as

$$DIST(T_k) = 2\left(\mathrm{D}_1 + \prod_{i=4}^{k} \mathrm{F}_i^2 \sum_{i=3}^{k} \mathrm{F}_i + \prod_{i=4}^{k-1} F_i F_k^3\right). \tag{23}$$

It is well known that $\sum_{i=1}^{k} F_i = F_{k+2} - 1$. Then simplifying (23), we get

$$
\begin{aligned}
DIST'(T_k) = {} & \prod_{i=4}^{k} F_i \left[\sum_{\substack{i=1 \\ j=i-1}}^{k-1} (F_{k+2} - 1) - \left(F_{j+2} - 1\right) \right] \\
& + \prod_{i=4}^{k} F_i \left[(F_{k+2} - 1) + \sum_{i=4}^{k-1} \left(\prod_{j=4}^{i} F_j \left((F_{k+2} - 3) + (F_{i+2} - 3) \right) \right) \right] \\
& + F_4 2 \prod_{i=5}^{k} F_i \left[(F_{k+2} - 2) + \sum_{i=5}^{k-1} \left(\prod_{j=5}^{i} F_j \left((F_{k+2} - 5) + (F_{i+2} - 5) \right) \right) \right] \\
& + 6 \prod_{i=5}^{k} F_i^2 (F_{k+2} - 5) + \prod_{i=4}^{k-1} F_i F_k^3 + \prod_{i=4}^{k} F_i^2 (F_{k+2} - 3) .
\end{aligned} \tag{24}
$$

Therefore,

$$DIST(T_k) = 2\, DIST'(T_k) . \tag{25}$$

We now give a simple formula for computing $W(T_k)$ as given Lemma 3.

Lemma 3:

For a positive integer k, let T_k be the Fibonacci weighted tree with Fibonacci branching of order k. Then the Wiener index $W(T_k)$ is given by

$$
W(T_k) = \begin{cases} 1 & k = 1, \\ 4 & k = 2, \\ 26 & k = 3, \\ 320 & k = 4, \\ W(T_{k-1}) + DIST(\mathrm{T_k}) & k \geq 5. \end{cases} \tag{26}
$$

Proof. The result of $W(T_k)$ follows from Lemma 2 and a simple combinatorial argument.

Theorem 1:
For a tree T_k ($k > 0$), we can algorithmically compute $W(T_k)$ in time O(*k*). *The input to the algorithm requires only the order k of the tree T_k.*

Proof. We know that $V|T_k| = \eta\eta$. Clearly F_{k+2} can be computed in time O(*k*). Thus (25) and (26) can be computed in time O(*k*) = O(log η) which computes the operations such as additions and multiplications.

4. Conclusion

We have presented a new algorithm for computing the Wiener index of a Fibonaccci weighted trees with Fibonacci branching in place of the available naïve algorithm for the same. The running time of this algorithm is logarithmic assuming that the input is only the order *k* of the tree.

Acknowledgements

The authors would like to thank the higher authorities of B.N.M. Institute of Technology, Bangalore, India, for supporting funding to this article.

REFERENCES

[1] F. Buckley and F. Harary, (1990) "Distance in Graphs" (Addison-Wesley, Redwood, Vol. 42.

[2] T. H. Cormen, C.E. Leiserson, R.L. Rivest, and C. Stein, (2001) *Introduction to Algorithms*, McGraw-Hill, 2nd edition.

[3] P. Dankelmann, S. Mukwembi, and H. C. Swart, (2009) "Average distance and vertex connectivity", J. *Graph Theory*, Vol. 62(2), pp 157-177.

[4] R.C. Entringer, D.E. Jackson and D. A. Snyder, (1976) "Distance in graphs", *Czech. Math. J.*, Vol. 26, pp 283-296.

[5] I. Gutman, (1988) "On distances in some bipartite graphs", *Publ. Inst. Math.*, (Beograd), Vol. 43, pp 3-8.

[6] F. Harary, (1959) "Status and contrastatus", *Sociometry*, Vol. 22, pp 23-43.

[7] S. Klavzar, and I. Gutman, (1997) "Wiener number of vertex-weighted graphs and chemical applications", *Discrete Appl. Math.*, Vol. 80, pp 73-81.

[8] J. Plesnik, (1984) "On the sum of distances in graphs or digraph", *J. Graph Theory*, Vol. 8, pp 1-21.

[9] S.G. Wagner, H. Wang, and G. Yu, (2009) "Molecular Graphs and the Inverse Wiener Index Problem", *Discrete Appl. Math.*, Vol. 157, pp 1544-1554.

[10] Y. N. Yeh and I. Gutman, (1994) "On the sum of all distances in composite graphs", *Discrete Math.*, Vol. 135, pp 359-365.

AUTHORS

K. R. Udaya Kumar Reddy completed his Diploma in Computer Science and Engineering from Siddaganga Polytechnic, Tumkur, Bangalore University in 1993. In 1998 he completed his Bachelor of Engineering in Computer Science and Engineering from Golden Valley Institute of Technology (now Dr. TTIT), K.G.F, Bangalore University, India. In 2004 he completed his Master of Engineering in Computer Science and Engineering from University Visvesvaraya College of Engineering, Bangalore, India. In 2012, he completed his Ph.D in the area of Graph Algorithms in Computer Science and Engineering, National Institute of Technology, Trichy, India (formerly Regional Engineering College). He held various positions at B.N.M. Institute of Technology, Bangalore, India, before joining Ph.D course and is currently a Professor at B.N.M. Institute of Technology, Bangalore, India. His fields of interests are Algorithmic graph theory and Theory of computation.

Ranjana S. Chakrasali received her graduation in Computer Science & Engineering from Tontadarya College of Engineering, Gadag, Visvesvaraya Technological University (VTU), Belgaum, India in 2002. Thereafter entered into teaching profession as a Lecturer and worked for 6 years. Currently pursuing post graduate at BNM Institute of Technology, Bangalore, India, affiliated to VTU. Her areas of interests are Graph Theory, Computer Graphics and Computer Networks.