





Computation of Topological Indices of Binary and Ternary Trees using Algorithmic Approach

Kashif Elahi¹ , Ali Ahmad² , Muhammad Ahsan Asim³  and Roslan Hasni^{4*} 

¹Deanship of Human Resources and Information Technology, Jazan University, Jazan, Saudi Arabia

²Department of Computer Science, College of Engineering and Computer Science, Jazan University, Jazan, Saudi Arabia

³Division of Computing, Analytics and Mathematics School of Science and Engineering University of Missouri-Kansas City, MO 64110, USA

⁴Special Interest Group of Modeling and Data Analytics (SIGMDA), Faculty of Computer Science and Mathematics, Universiti Malaysia, Terengganu 21030 Kuala Nerus, Terengganu, Malaysia

Keywords:

Algorithms,
Distance,
Binary tree,
Ternary tree,
Topological indices

AMS Subject Classification (2020):

05C10; 05C12; 05C90

Article History:

Received: 22 March 2023

Accepted: 18 January 2024

Abstract

In this paper, algorithms are used to compute distance-based topological indices for the Complete Binary Tree (CBT) and the Complete Ternary Tree (CTT). Computation of distance-based topological indices is complex for varied heights of CBT and CTT. Hence designed algorithms to compute distance between any-to-any vertex made this possible to compute the required topological indices for CBT and CTT. The distance calculator algorithm designed for this study can also be customized in digital chemical structures, mathematical chemistry, network traffic control in wireless networks, search applications, high bandwidth routing, parse construction in compilers, and memory management.

© 2024 University of Kashan Press. All rights reserved.

1 Introduction

In the concepts of applied mathematics, graphs are one of the powerful tools for representing and understanding objects and their relationship in computer science, robotics, chemistry, physics, biology, human sciences, data sciences, and several other fields [1, 2]. Calculation of distance-based topological indices for dynamic graphs is more interesting and close to real-life problems. Topological index is one of the numerical descriptors that represent the characteristics of a graph with a single numeric value [3-5]. They are important in chemical graph theory to develop quantitative structure-activity relationships (QSARs), and to define and understand

*Corresponding author

E-mail addresses: kelahi@jazanu.edu.sa (K. Elahi), ahmadms@gmail.com (A. Ahmad),

maaym7@umkc.edu (M. A. Asim), hroslan@umt.edu.my (R. Hasni)

Academic Editor: Abbas Saadatmandi

the correlation between the molecules [3, 6–8]. Topological indices that manage calculations based on the vertices are "degree-based topological indices" and those based on distance are "distance-based topological indices" [9, 10].

In a rooted tree, the level of any vertex v is the length of the unique path from the root to this vertex. A rooted tree is called an m -ary tree if every internal vertex has no more than m children. The tree is called a *full m -ary tree* if every internal vertex has exactly m children [2]. A full m -ary tree is a complete m -ary tree where all leaf vertices are at the same level. A complete m -ary tree with $m = 2$ and $m = 3$ is called CBT (Complete Binary Tree) and CTT (Complete Ternary Tree) respectively. Trees have wide applications in computer networks, coding techniques, memory management, decision support systems, cryptography, and parallel processing. A lot of mathematical properties of complete m -ary trees have been evolved but still, there is no property that can calculate the distance between any-to-any vertex for all vertices of a tree one by one. For this purpose, a generalized algorithm is designed and implemented to compute distance-based topological indices for graphs of any height. The outcomes of this algorithm are outstanding and can be used in many other applications later on.

Calculation of distance-based topological indices for dynamic graphs is more interesting and closer to real-life problems. In this article, CBT and CTT are considered to be dynamic in nature due to their varied height. They are important in mathematical chemistry, by handling chemical structures digitally and mathematically [11, 12]. In 1947, Harry Wiener introduced the first distance-based topological index [13, 14]. To calculate the Wiener index, counts of unique cycle paths are required [10]. By varying the height of the tree ultimately the size and order of the tree also change thus the distance-based topological indices for each case need to be re-calculated. In this paper, we produced a generalized dynamic system for the calculation of distance-based topological indices including Wiener index, Hosoya polynomial, Schultz index and generalized Schultz index [9, 10]. The distance calculator is designed for CBT and CTT only for any height H . But this algorithm can be customized easily for any other rooted trees or graphs that can be combinatorially mapped as rooted trees.

2 Distance-based topological indices

Let G be a graph with vertex set $V(G)$ and edge set $E(G)$, respectively. The distance between vertex u and vertex v of graph G is $d(u, v)$ and it is the shortest path between vertex u and v . The degree of vertex v is represented by $d(v)$. Wiener index [10] is defined as:

$$W = \sum_{uv \in V(G)} d(u, v). \quad (1)$$

Hosoya polynomial of G is defined as [10]:

$$H = \sum_{uv \in V(G)} x^{d(u, v)}. \quad (2)$$

Schultz index is defined as [15]:

$$S = \sum_{uv \in V(G)} \{d(u) + d(v)\}d(u, v). \quad (3)$$

Modified Schultz index is defined as [15]:

$$S^* = \sum_{uv \in V(G)} \{d(u) \times d(v)\}d(u, v). \quad (4)$$

Table 1: Output of Algorithm 1 for $m = 2$ and $H = 2$.

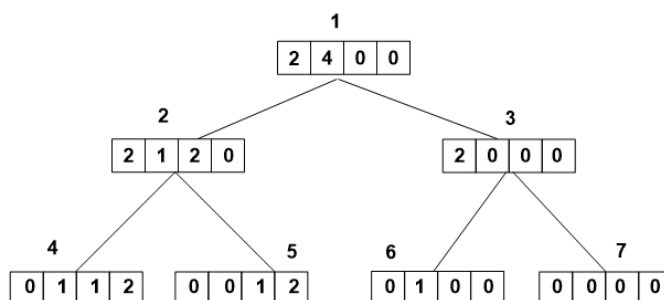
Vertices	Count of Distances			
	1	2	3	4
1	2	4	0	0
2	2	1	2	0
3	2	0	0	0
4	0	1	1	2
5	0	0	1	2
6	0	1	0	0
No. of Paths	6	7	4	4

3 Algorithmic solution

Algorithmic solutions are alternate techniques to solve complex mathematical problems, in almost all branches of the sciences [16–19]. They are like roadmaps for accomplishing a given task in a well-defined and efficient way. Some well-known graph algorithms like breadth-first, depth-first, spanning-tree, Dijkstra and Bellman-Ford are related to distance from a single source to other vertices. Whereas the Floyd-Warshall algorithm considers all pair's shortest distance that deals with weighted graphs [20]. Similarly, the algorithm designed in this article computes the distance between vertices $u \in V(G)$ and $v \in V(G)$ where $u \neq v$ for CBT and CTT. All the algorithms devised in this article can be easily modified for other types of graphs and trees for calculating distance-based topological indices.

In the design phase, Tree-of-Arrays (classical data structures) are used for clear depiction. Whereas in the implementation phase 2D array is used for the sake of efficiency. To handle the levels of trees, both iterative and recursive approaches are used in parallel. For the sake of traversing, backtracking is used as an algorithm design strategy.

To speed up the traversing between the levels of CBT and CTT, inline functions are used to probe the values of Left-Child, Right-Child, Mid-Child and Parent vertices. In CBT $\text{Left}(u)$ returns location $2 \times u$, $\text{Right}(u)$ returns location $2 \times u + 1$ and $\text{Parent}(u)$ returns location $\lfloor \frac{u}{2} \rfloor$. Similarly, in CTT, $\text{Left}(u)$ returns location $3 \times u - 1$, $\text{Mid}(u)$ returns location $3 \times u$, $\text{Right}(u)$ returns location $3 \times u + 1$ and $\text{Parent}(u)$ returns location $\lfloor \frac{u}{3} + 0.5 \rfloor$.

Figure 1: Output of Algorithm 1 for $m = 2$ and $H = 2$.

Description of Algorithm 1: The algorithm takes two positive integers m and H as input parameters. The output of the Algorithm 1 is a 2D array that stores the count of all vertices that are at distance j for a vertex u . These distances are used to compute topological indices

Algorithm 1 Distance-Calculator(m,H)

```

1:  $V \leftarrow \frac{(m^{H+1}-1)}{(m-1)}$ 
2:  $Distance \leftarrow 2 \times H$ 
3:  $Array[V][Distance] \leftarrow 0$ 
4: for  $u \leftarrow 1$  to  $V$ 
5:    $j \leftarrow 1$ 
6:   Fill(Array, u, root, j)
7:   Back-Track(Array, u, root, j)
8: return Array
Fill(Array, u, v, j)
1: if ( $m = 2$ )
2:   if ( $Right(u) \leq V$ )
3:      $Array[u][j] \leftarrow Array[u][j] + 2$ 
4:   if ( $Right(Right(u)) \leq V$ )
5:     Apply recursive calls for all sub-child of u
6: if ( $m = 3$ )
7:   if ( $Right(u) \leq V$ )
8:      $Array[u][j] \leftarrow Array[u][j] + 3$ 
9:   if ( $Right(Right(u)) \leq V$ )
10:    Apply recursive calls for all sub-child of u
Back-Track(Array, u, v, j)
1:  $prev \leftarrow u$ 
2: if ( $m = 2$ )
3:   while ( $Parent(u) \neq Right-Nodes()$ )
4:     if ( $prev = Left(Parent(u))$ )
5:        $Array[u][j] \leftarrow Array[u][j] + 1$ 
6:       Call Fill for right sibling for j+1 distance
7:        $prev, u \leftarrow Parent(u)$ 
8:        $j \leftarrow j + 1$ 
9:        $Array[u][j] \leftarrow Array[u][j] + 1$ 
10:      Call Fill for right sibling for j+1 distance
11: if ( $m = 3$ )
12:   while ( $Parent(u) \neq Right-Nodes()$ )
13:     if ( $prev = Left(Parent(u))$ )
14:        $Array[u][j] \leftarrow Array[u][j] + 1$ 
15:     Call Fill for mid and right siblings for j+1 distance
16:      $Array[u][j] \leftarrow Array[u][j] + 1$ 
17:     if ( $prev = Mid(Parent(u))$ )
18:        $Array[u][j] \leftarrow Array[u][j] + 1$ 
19:     Call Fill for right sibling for j+1 distance
20:      $prev, u \leftarrow Parent(u)$ 
21:      $j \leftarrow j + 1$ 
22:     if ( $prev = Left(Parent(u))$ )
23:        $Array[u][j] \leftarrow Array[u][j] + 1$ 

```

```

24: Call Fill for mid and right siblings for j+1 distance
25:   Array[u][j] ← Array[u][j] + 1
26:   else
27:     Array[u][j] ← Array[u][j] + 1
28:   Call Fill for right sibling for j+1 distance

```

Table 2: The results of algorithms for $m = 2$ and $H = 8$ obtained, by implementing them in computers.

H	Weiner	Hosoya x=1.1	Schultz	Modified Schultz
1	3	3.41	10	6
2	21	26.25	150	114
3	105	148.41	1262	1066
4	465	760.51	8286	7386
5	1953	3759.66	47550	43706
6	8001	18337.43	251262	235386
7	32385	88992.56	1257214	1192698
8	130305	431105.97	6055422	5795322

in Algorithms 2 and 3. tree of arrays given in Figure 1 depicts the algorithmic results. Results of the computer-generated in Algorithm 1 are tabulated in Table 1 for a particular example.

Description of Algorithm 2: This algorithm takes two positive integers m and H as input

Algorithm 2 Wiener-Hosoya($m,H,Array$)

```

1:  $V \leftarrow \frac{(m^{H+1}-1)}{(m-1)}$ 
2:  $WeinerIndex \leftarrow 0$ 
3: for  $j \leftarrow 1$  to  $Distance$ 
4:    $HosoyaExp \leftarrow 0$ 
5:   for  $i \leftarrow 1$  to  $V$ 
6:      $WeinerIndex \leftarrow WeinerIndex + Array[i][j]$ 
7:      $HosoyaExp \leftarrow HosoyaExp + Array[i][j]$ 
8:    $HosoyaPol \leftarrow HosoyaPol + HosoyaExp \times x^j$ 
9: return  $WeinerIndex, HosoyaPol$ 

```

parameters, and a 2D array (output of Algorithm 1) that has a count of all vertices which are at distance j for a vertex u . Output of the algorithm is Wiener index and Hosoya polynomial for CBT ($m = 2$) or CTT ($m = 3$). After experimenting with the computer by implementing the Algorithms 1 and 2. The example of the computer-generated Algorithm 2 is tabulated in Table 2.

Description of Algorithm 3: The algorithm takes two positive integers m and H as input parameters. Output of the Algorithm 3 is the Schultz index and the modified Schultz index for CBT and CTT.

The computer-generated Algorithm 3 is tabulated in Table 3. The topological indices are compared and their comparison is important in this field of graph theory [21].

Algorithm 3 Schultz-Indices(m,H)

```

1:  $V \leftarrow \frac{(m^{H+1}-1)}{(m-1)}$ 
2:  $Distance \leftarrow 2 \times H$ 
3:  $Array[V][Distance] \leftarrow 0$ 
4:  $Sltz, MSltz \leftarrow 0$ 
5: for  $u \leftarrow 1$  to  $V$ 
6:    $j \leftarrow 1$ 
7:   Fill-Schultz(Array, u, root, j)
8:   Back-Track-Schultz(Array, u, root, j)
9: return  $Sltz, MSltz$ 

Fill-Schultz(Array, u, v, j)
1: if ( $m = 2$ )
2:   if ( $Right(u) \leq V$ )
3:     Calculate Schultz and Modified Schultz of all sub-child for j distance
4:   if ( $Right(Right(u)) \leq V$ )
5:     Apply recursive calls for all sub-child of u as v
6:   if ( $m = 3$ )
7:     if ( $Right(u) \leq V$ )
8:       Calculate Schultz and Modified Schultz of all sub-child for j distance
9:     if ( $Right(Right(u)) \leq V$ )
10:      Apply recursive calls for all sub-child of u as v

Back-Track-Schultz(Array, u, v, j)
11:  $prev \leftarrow u$ 
12: if ( $m = 2$ )
13:   while ( $Parent(u) \neq Right-Nodes()$ )
14:     if ( $prev = Left(Parent(u))$ )
15:       Calculate Schultz and Modified Schultz of right sibling for j+1 distance
16:       Call Fill-Schultz on right sibling for j+1 distance
17:        $prev, u \leftarrow Parent(u)$ 
18:        $j \leftarrow j + 1$ 
19:     Calculate Schultz and Modified Schultz of right sibling for j+1 distance
20:     Call Fill-Schultz on right sibling for j+1 distance
21:   if ( $m = 3$ )
22:     while ( $Parent(u) \neq Right-Nodes()$ )
23:       if ( $prev = Left(Parent(u))$ )
24:         Calculate Schultz and Modified Schultz of mid and right sibling for j+1 distance
25:         Call Fill-Schultz on mid and right sibling for j+1 distance
26:       if ( $prev = Mid(Parent(u))$ )
27:         Calculate Schultz and Modified Schultz of right sibling for j+1 distance
28:         Call Fill-Schultz on right sibling for j+1 distance
29:        $prev, u \leftarrow Parent(u)$ 
30:        $j \leftarrow j + 1$ 
31:     if ( $prev = Left(Parent(u))$ )
32:       Calculate Schultz and Modified Schultz of mid and right sibling for j+1 distance
33:       Call Fill-Schultz on mid and right sibling for j+1 distance
34:     else
35:       Calculate Schultz and Modified Schultz of right sibling for j+1 distance
36:       Call Fill-Schultz on right sibling for j+1 distance

```

Table 3: The results of algorithms for $m = 3$ and $H = 6$, obtained by implementing them in computers.

H	Weiner	Hosoya $x=1.1$	Schultz	Modified Schultz
1	6	6.93	24	15
2	78	102.1	708	564
3	780	1193.9	12048	10527
4	7260	13239.42	164328	149928
5	66066	144838.92	1999464	1867695
6	596778	1579102.99	45557385	35320593

4 Conclusion

Algorithm 1 provides us the analysis of all possible paths in CBT and CTT. It can help in analyzing wireless networks [22], for better network traffic control, as these network algorithms are practical applications of m -ary trees. Algorithms 1 to 3 calculated distance-based topological indices for both CBT and CTT, which are the representation of distance-based characteristics of the CBT, and the CTT. These results will enhance the research for optimization in the applications of these trees, including search applications, digital chemical structures, mathematical chemistry, high bandwidth routing, parse construction in compilers, and memory management.

The outcomes of this research are helpful for the fields of mathematics, mathematical chemistry, digital structures and computer science. At first, it computes distance-based topological indices of CBT and CTT for any height. These results are quite significant due to dynamic size of CBT and CTT, which is not possible without algorithmic approach. Results give a comparison between different indices on CBT and CTT, also the results can be compared with other graph families. Secondly distance calculator algorithm for CBT and CTT can be further modified to find the shortest path between any-to-any nodes of network graphs. Given algorithm is valid for any height as long as resources of computer supports, and its output can be used in plenty of computer applications like high-bandwidth routers, chemical structure digital analysis, search applications, grammar checking applications, p2p programs, specialized image-signatures, video games, implementing efficient priority-queues, scheduling processes, cryptographic applications, and memory management.

Conflicts of interest. The authors declare that they have no conflicts of interest regarding the publication of this article.

Acknowledgement. The authors wish to thank the referee for his/her valuable and constructive comments which improved the paper.

References

- [1] H. Hasanzadeh Bashir and K. Ahmadidelir, Some structural graph properties of the non-commuting graph of a class of finite Moufang loops, *Electron. J. Graph Theory Appl.* **8** (2020) 319–337, <https://doi.org/10.5614/ejgta.2020.8.2.9>.
- [2] K. H. Rosen, *Discrete Mathematics and its Applications*, McGraw Hill, seventh edition, 2011.

- [3] A. U. Khan, Descriptors and their selection methods in QSAR analysis: paradigm for drug design, *Drug Discov. Today* **21** (2016) 1291–1302, <https://doi.org/10.1016/j.drudis.2016.06.013>.
- [4] I. Duman and B. Tutmez, A topological index-based new smoother for spatial interpolation, *Earth Sci. Inform.* **13** (2020) 555–564, <https://doi.org/10.1007/s12145-020-00447-8>.
- [5] X. Zuo, J. B. Liu, H. Iqbal, K. Ali and S. T. R. Rizvi, Topological indices of certain transformed chemical structures, *J. Chem.* **2020** (2020) Article ID 3045646, <https://doi.org/10.1155/2020/3045646>.
- [6] M. Arockiaraj, J. Clement, D. Paul and K. Balasubramanian, Quantitative structural descriptors of sodalite materials, *J. Mol. Struct.* **1223** (2021) p. 128766, <https://doi.org/10.1016/j.molstruc.2020.128766>.
- [7] S. Ahmadi, S. Lotfi and P. Kumar, A Monte Carlo method based QSPR model for prediction of reaction rate constants of hydrated electrons with organic contaminants, *SAR QSAR Environ. Res.* **31** (2020) 935–950, <https://doi.org/10.1080/1062936X.2020.1842495>.
- [8] S. Shirakol, M. Kalyanshetti and S. M. Hosamani, QSPR analysis of certain distance based topological indices, *Appl. Math. Nonlinear Sci.* **4** (2019) 371–385, <https://doi.org/10.2478/AMNS.2019.2.00032>.
- [9] K. Pattabiraman, Degree and distance based topological indices of graphs, *Electron. Notes Discrete Math.* **63** (2017) 145–159, <https://doi.org/10.1016/j.endm.2017.11.009>.
- [10] F. Ali, M. Salman, A. Hafeez and S. Huang, On computation of some distance-based topological indices on circulant networks-II, *J. Inf. Optim. Sci.* **39** (2018) 759–782, <https://doi.org/10.1080/02522667.2016.1223588>.
- [11] Z. Chen, Z. Yin, J. Cui, J. Yang and Z. Tang, DNA switching circuits based on binary tree, *IEEE* **9** (2021) 94033 – 94039, <https://doi.org/10.1109/ACCESS.2021.3091010>.
- [12] Z. Huang, Q. Shi, J. Guo, F. Meng, Y. Zhang, Y. Lu, Z. Qian, X. Li, N. Zhou, Z. Zhang and X. Zhu, Binary tree-inspired digital dendrimer, *Nat. Commun.* **10** (2019) p. 1918, <https://doi.org/10.1038/s41467-019-09957-6>.
- [13] A. A. Dobrynin and A. Iranmanesh, Wiener index of edge thorny graphs of catacondensed benzenoids, *Mathematics* **8**(2020) p. 467, <https://doi.org/10.3390/math8040467>.
- [14] H. Wiener, Structural determination of paraffin boiling points, *J. Am. Chem. Soc.* **69** (1947) 17–20, <https://doi.org/10.1021/ja01193a005>.
- [15] M. R. Farahani, R. Kanna and W. Gao, The Schultz, modified Schultz indices and their polynomials of the Jahangir graphs $J_{n,m}$ for integer numbers $n = 3, m \geq 3$, *Asian J. Appl. Sci.* **3** (2015) 823–827.
- [16] F. Es-sabery and A. Hair, A MapReduce C4.5 decision tree algorithm based on fuzzy rule-based system, *Fuzzy Inf. Eng.* **11** (2019) 446–473, <https://doi.org/10.1080/16168658.2020.1756099>.
- [17] S. Katoch, S. S. Chauhan and V. Kumar, A review on genetic algorithm: past, present, and future, *Multimed. Tools Appl.* **80** (2021) 8091–8126, <https://doi.org/10.1007/s11042-020-10139-6>.

-
- [18] A. Slowik and H. Kwasnicka, Evolutionary algorithms and their applications to engineering problems, *Neural Comput. & Applic.* **32** (2020) 12363–12379, <https://doi.org/10.1007/s00521-020-04832-8>.
- [19] G. Xin and D. Fei-qi, Complete ternary tree-based data aggregation routing algorithm for wireless sensor networks, *In: Proceedings 2013 International Conference on Mechatronic Sciences, Electric Engineering and Computer (MEC). Shengyang, China: IEEE*, (2013) 578–581, <https://doi.org/10.1109/MEC.2013.6885129>.
- [20] T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein, *Introduction to Algorithms*, The MIT Press, Cambridge, Massachusetts London, England, 3rd Edition, 2009.
- [21] T. Réti, A. Ali and I. Gutman, On bond-additive and atoms-pair-additive indices of graphs, *Electron. J. Math.* **2** (2021) 52–61, <https://doi.org/10.47443/ejm.2021.0033>.
- [22] E. Goldoni and P. Gamba, PRISM: a novel protocol for real-time synchronous acquisitions in WSNs, *International Journal of Sensors, Wireless Communications and Control* **1** (2011) 1–8.