mathematics

MDPI

Article

Linear Algorithms for the Hosoya Index and Hosoya Matrix of a Tree

Aleksander Vesel

Faculty of Natural Sciences and Mathematics, University of Maribor, SI-2000 Maribor, Slovenia; aleksander.vesel@um.si

Abstract: The Hosoya index of a graph is defined as the total number of its independent edge sets. This index is an important example of topological indices, a molecular-graph based structure descriptor that is of significant interest in combinatorial chemistry. The Hosoya index inspires the introduction of a matrix associated with a molecular acyclic graph called the Hosoya matrix. We propose a simple linear-time algorithm, which does not require pre-processing, to compute the Hosoya index of an arbitrary tree. A similar approach allows us to show that the Hosoya matrix can be computed in constant time per entry of the matrix.

Keywords: Hosoya index; Hosoya matrix; optimal algorithm

Citation: Vesel, A. Linear Algorithms for the Hosoya Index and Hosoya Matrix of a Tree. *Mathematics* **2021**, *9*, 142. https://doi.org/10.3390/math9020142

Received: 3 November 2020
Accepted: 8 January 2021
Published: 11 January 2021

1. Introduction

In Haruo Hosoya's seminal paper, a molecular-graph based structure descriptor is proposed that nowadays is known under the name Hosoya index [1]. It is well-known that several physicochemical properties of chemical structures are well correlated with the Hosoya index of the corresponding molecular graphs. In particular, it is used to predict, from the structure of molecules, some of their physico-chemical properties such as boiling points, entropies, heat of vaporization, and π-electron energy [1,2].

The mathematical investigations on the Hosoya index are often related to trees, see for example [3–8]. Note also that there is a strong connection between the energy of a tree and its Hosoya index [9]. It is established that the general problem of determining the Hosoya index is #P-complete even for planar graphs [10]. On the other hand, if the graph is known to be a tree, there are efficient solutions.

In [11], Zhang and co-authors proposed an algorithm that allows computation of the Hosoya index of a tree in linear time. The algorithm uses a relatively complex pre-processing which for a given tree computes the Prüfer code of the corresponding labeled tree. As noted already by the authors, another troublesome aspect of the proposed algorithm is its necessity of computation with fractions. Moreover, it was shown in [12] that the values of the characteristic polynomial of a tree can be computed in linear time, which implies that a computation of the Hosoya index of a tree can be done within the same time-bound; however, an explicit algorithm has not been presented.

It has been demonstrated that the Hosoya index can be computed efficiently for some graphs derived from trees. In particular, for an arbitrary tree T, a closed-form expression is presented for the Hosoya index of the fractal graphs $R(T)$ and $RT(T)$ [13].

Since a graph is entirely determined by specifying either its adjacency structure or its incidence structure, it is natural to state the specification of a graph in matrix form. By associating a matrix with a graph, one can also use selected graph invariants as molecular descriptors. This approach was initiated in a work by Randić, where a matrix based on another well-known molecular-graph structure descriptor, the Wiener index, is proposed [14]. Later, Randić used an analogous approach, based on the Hosoya index, for acyclic systems, which yields the concept of the Hosoya matrix [15].

The paper is organized as follows. In the next section, we give some definitions and concepts needed in this paper. In Section 3, we follow the idea presented in [12] to give a recursive expression for the Hosoya index of a tree. Using this result, we present a linear-time algorithm for computing the Hosoya index of a graph of this class. In Section 4, we apply the previous section's approach to provide a recursive formula for an entry of the Hosoya matrix. The results allow the computation of the Hosoya matrix in constant time per entry of the matrix.

2. Preliminaries

Let $G = (V, E)$ be a graph, possessing n vertices and m edges. The set of edges $X \subseteq E(G)$ is called a matching of G if any two edges of X have no vertex in common. A matching of G with k edges is called a k-matching of G.

A subset of vertices $I \subseteq V(G)$ is independent if no two vertices of I are adjacent. The number of distinct k-element independent vertex sets is denoted by $n(G, k)$. Note that $n(G, 0) = 1$ and $n(G, n) = n$.

The *Hosoya index* of a graph G, denoted by $Z(G)$, is defined as the total number of matchings of G. Let $m(G, k)$ be the number of k-matchings of G and set $m(G, 0) := 1$. The Hosoya index can be formulated as

$$Z(G) := \sum_{k \geq 0} m(G, k).$$

Let $G = (V, E)$ be a graph and $V' \subset V(G)$. Then $G - V'$ denotes the graph obtained from G by removing vertices of V'. If $V' = \{v\}$, i.e., V' is composed of a single vertex, we will write $G - v$ instead of $G - \{v\}$. Analogously, if and $E' \subset E(G)$, then $G - E'$ denotes the graph obtained from G by removing vertices of E'. If $E' = \{e\}$, we will write $G - e$ instead of $G - \{e\}$.

A tree is a connected graph without cycles. If u, v are vertices of a tree T, then T admits exactly one path between u and v. We shall denote this path by P_{uv}.

Let T be a tree with the vertex set $V(T) = \{1, 2, \ldots, n\}$. The Hosoya matrix of a tree T is the $n \times n$ matrix $\mathbf{H}(T)$, where (i, j)-entry of $\mathbf{H}(T)$ is the Hosoya index of the graph obtained from T by removing the edges of the path P_{ij} [15]. More formally, if $i, j \in \{1, 2, \ldots, n\}$, then

$$\mathbf{H}_{i,j}(T) = \begin{cases} Z(T - E(P_{ij})), & i \neq j \\ 0, & i = j \end{cases}.$$

Let v be a vertex of a graph G. Then $N_G(v)$ or simply $N(v)$ denote the set of vertices adjacent to v in G.

A rooted tree is a tree in which one vertex is distinguished from the others and called the root. If r is a root of a tree T, then T_r denotes the corresponding rooted tree. Let $v \neq r$ be a vertex of T_r. As already stated, P_{rv} denotes the path between r and v in T_r. If u is a vertex of P_{rv} adjacent to v, then u is a parent of v and v is a child of u. If u is a parent of v and w parent of u, then w is a grandparent of v. Note that every vertex of a rooted tree admits at most one parent and grandparent. A vertex z is a descendant of v if v is in the path from z to r. A vertex v of a tree T, and all its descendants induce a subgraph of T_r denoted by T_v^r. It is clear that T_v^r is a tree. Note that T_v^r can be seen as a rooted subtree of T_r with the root v.

If v is a vertex of T_r, then $C_r(v)$ denotes the set of children of v. If $v \neq r$, let $p_r(v)$ denote the parent of v. If $C_r(v) = \emptyset$, then v is a *leaf* of T_r.

3. Hosoya Index

3.1. Recursive Formula for Computing the Hosoya Index

The results of this subsection are obtained by using an approach similar to the work of Mohar where the characteristic polynomial of a tree is studied [12].

The following three results are well-known [1,16].

Proposition 1. *Let uv be an edge of a graph G. Then*

$$Z(G) = Z(G - uv) + Z(G - \{u, v\}).$$

Proposition 2. *Let G be a graph composed of two disjoint components G_1 and G_2. Then*

$$Z(G) = Z(G_1)Z(G_2).$$

Proposition 3. *Let v be a vertex of a graph G. Then*

$$Z(G) = Z(G - v) + \sum_{u \in N(v)} Z(G - \{u, v\}).$$

We also need the following

Proposition 4. *Let v and u be vertices of a tree T. Then*

(i)

$$Z(T - v) = \prod_{w \in C_v(v)} Z(T_w^v).$$

(ii) If v is the parent of u in T_v, then

$$Z(T - \{u, v\}) = \frac{Z(T - v)Z(T_u^v - u)}{Z(T_u^v)} = \frac{Z(T_u^v - u) \prod\limits_{w \in C_v(v)} Z(T_w^v)}{Z(T_u^v)},$$

where

$$Z(T_u^v - u) = \begin{cases} 1, & u \text{ is a leaf} \\ \prod\limits_{x \in C_v(u)} Z(T_x^v), & \text{otherwise} \end{cases}.$$

Proof. (i) If T is a single vertex, the claim immediately follows. Otherwise, note that $N_T(v) = C_v(v)$. It follows that $T - v$ is composed of connected components T_w^v, for all $w \in C_v(v)$. Proposition 2 now yields the assertion.

(ii) We can see that $T - \{v, u\}$ is composed of two sets of connected components: one corresponds to components of $T - v$ with the exception of T_u^v, while the other corresponds to $T_u^v - u$. Let us denote the first set of connected components by $(T - v) \setminus T_u^v$. By (i), we have

$$Z((T - v) \setminus T_u^v) = \prod_{w \in C_v(v) \setminus \{u\}} Z(T_w^v) \frac{\prod\limits_{w \in C_v(v)} Z(T_w^v)}{Z(T_u)} = \frac{Z(T - v)}{Z(T_u^v)}.$$

From Proposition 2 now it follows that

$$Z(T - \{u, v\}) = Z((T - v) \setminus T_u^v)Z(T_u^v - u) = \frac{Z(T - v)Z(T_u^v - u)}{Z(T_u^v)}.$$

If u is a leaf, then $T - \{v, u\}$ is the graph composed of connected components T_w^v, for all $w \in C_v(v) \setminus \{u\}$. Since $Z(T_u^v - u) = 1$, the case is settled.

If u is not a leaf, then $T - \{v, u\}$ is the graph composed of connected components T_w^v and T_x^v, for all $w \in C_v(v) \setminus \{u\}$ and $x \in C_v(u)$. From Proposition 2 and (i) then it follows

$$Z(T - \{u, v\}) = \frac{\prod\limits_{x \in C_v(u)} Z(T_x^v) \prod\limits_{w \in C_v(v)} Z(T_w^v)}{Z(T_u^v)}.$$

This assertion concludes the proof. □

Theorem 1. *Let v and r be (not necessarily distinct) vertices of a tree T. Then*

$$Z(T_v^r) = \begin{cases} 1, & v \text{ is a leaf} \\ \prod\limits_{w \in C_r(v)} Z(T_w^r) + \sum\limits_{u \in C_r(v)} \dfrac{Z(T_u^r - u) \prod\limits_{w \in C_r(v)} Z(T_w^r)}{Z(T_u^r)}, & \text{otherwise} \end{cases}$$

where

$$Z(T_u^r - u) = \begin{cases} 1, & u \text{ is a leaf of } T_r \\ \prod\limits_{x \in C_r(u)} Z(T_x^r), & \text{otherwise} \end{cases}.$$

Proof. If v is a leaf, then T_v^r is a single vertex and the case is settled.

If v is not a leaf, then note that $N_{T_v^r}(v) = C_r(v)$. By Proposition 3, we have $Z(T_v^r) = Z(T_v^r - v) + \sum\limits_{u \in C_r(v)} Z(T_v^r - \{u, v\})$. Proposition 4 now completes the proof. □

3.2. Algorithm for Computing the Hosoya Index

Let v be a vertex of a nonempty tree T. The data structure of the rooted tree T_v is given by Algorithm 1, where C_v and p_v represent the list of children of v and the parent of v in T_v, respectively.

Algorithm 1. Rooted(T, v, C, p)

1. C_v := the list comprised of all vertices of $N(v)$;
2. **For all** $w \in N(v)$ **do**
begin
2.1 Remove v from $N(w)$;
2.2 Rooted(T, w, C);
2.3 $p_w := v$;
end;
end.

Let v be a vertex of T and $w \in N(v)$. If $N(w)$ is represented by a (doubly connected) adjacency list, then we may remove v from $N(w)$ in constant time. It is easy to see now that the time complexity of Rooted is linear.

Algorithm 2 computes the Hosoya index Z_v of a tree T with respect to a root v. Before the algorithm is applied, the algorithm Rooted is called for the vertex v. Thus, for every vertex u in T_v, the rooted tree is represented by the list of children C_u.

Theorem 2. *Let T be tree of order n. Then algorithm Hosoya computes the Hosoya index of T in $O(n)$ time and space.*

Algorithm 2. Hosoya(v, C, Z, $\bar{Z}$)

1. $\bar{Z}_v := 1$;
2. **if** $|C_v| = 0$ **then begin** $Z_v := 1$; EXIT; **end**;
3. **for all** $w \in C_v$ **do**
   Hosoya(w, C, Z, $\bar{Z}$);
4. **for all** $w \in C_v$ **do**
   $\bar{Z}_v := \bar{Z}_v \cdot Z_w$;
5. $Z_v := \bar{Z}_v$;
6. **for all** $u \in C_v$ **do**
   begin
   6.1 $zu := 1$;
   6.2 **if** $|C_u| > 0$ **then**
   6.2.1 **for all** $x \in C_u$ **do**
   $zu := zu \cdot Z_x$;
   6.3 $Z_v := Z_v + zu \cdot \bar{Z}_v / Z_u$;
   end;
end.

Proof. The validity of the algorithm follows from Theorem 1. The algorithm computes vectors Z and $\bar{Z}$, such that the entry Z_v (resp. $\bar{Z}_v$) represents $Z(T)$ (resp. $Z(T-v)$). The algorithm also maintains the variable zu, which for $u \in C_v(u)$ represents the value of $\prod_{x \in C_v(u)} Z(T_x^v)$.

If v is a leaf in T, then $|C_v| = 0$ and Z_v is set to one. Otherwise, the algorithm is recursively called for each child of v in Step 3. Note that the call of the algorithm for a vertex w computes the Hosoya index of the corresponding subtree T_w^v. Thus, after Step 3 is executed, the Hosoya index is already computed for all subtrees of v. Next, $\prod_{w \in C_v(v)} Z(T_w^v)$ is computed in Step 4. This gives the initial value of Z_v assigned in Step 5, while the sum $\sum_{u \in C_v(v)} Z(T_u^v - u) \frac{\prod_{w \in C_v(v)} Z(T_w^v)}{Z(T_u^v)}$ is computed in Step 6. Steps 6.1 and 6.2 compute the value of $Z(T_u^v - u)$. If u is a leaf, then $|C_u| = 0$ and the value of zu remains equal to one. Otherwise, the value of zu is equal to $\prod_{x \in C_v(u)} Z(T_x^v)$ after Step 6.2 is executed. In Step 6.3, the value of zu is multiplied by $\bar{Z}_v$ and divided by Z_u; thus, the value added to Z_v in this step is equal to $Z(T_u^v - u)/Z_u \prod_{w \in C_v(v)} Z(T_w^v)$. This assertion completes the proof of the correctness of the algorithm.

For the time complexity first note that the time complexity of a single recursive call neglecting recursive calls (in Step 3) and neglecting the Steps 4 and 6 is clearly constant. Since the number of recursive calls of the algorithm equals the number of vertices of the tree, we have to show that the total number of operations executed in Steps 4 and 6 is linear in the number of vertices. This number can bounded above by the number of entries of the vector Z used by the algorithm. In Step 4, the values of Z_w, $w \in C_v$ are used, while in Step 6.2.1, for $u \in C_v$, the entries Z_x, $x \in C_u$ are used. Since every vertex v of a rooted tree admits at most one parent and grandparent, the corresponding entry Z_v is used at most three times: twice when Hosoya is called for the parent of v (Steps 4 and 6.3) and once when Hosoya is called for the grandparent of v (Step 6.2.1).

It follows that the total number of operations executed in Steps 4 and 6 is bounded above with some constant multiplied by the number of vertices of T.

Since the space complexity of the algorithm is clearly linear, the proof is complete. □

Note that the parameter p of the algorithm Rooted as well as the parameter $\bar{Z}$ of the algorithm Hosoya are not used for the computation of the Hosoya index. These parameters will be needed in the next section.

3.3. Example

As an example of the execution of the algorithm Hosoya observe the three T with the vertex set $\{1, 2, \ldots, 11\}$ depicted in Figure 1. We demonstrate the algorithm for the root 4. Note that before the algorithm is applied, the algorithm Rooted is called for the vertex 4.

Since 4 is not a leaf, the algorithm is recursively called for the vertices 3, 8, and 11 (the neighbors of 4) in Step 3. The recursion stops when a leaf is reached and the corresponding entry of the vector Z is set to one. We therefore obtain $Z_1 = Z_5 = Z_6 = Z_7 = Z_9 = Z_{10} = 1$. For the vertex 2 we then obtain $\bar{Z}_2 = Z_1 \cdot Z_5 = 1 \cdot 1 = 1$ in Step 4. Since C_1 and C_5 are both empty, Step 6.2.1 is never executed and zu remains equal to one. The final value $Z_2 = 3$ is therefore obtained in Step 6.3, where for the vertices 1 and 5 the value one is added to the current value of Z_2. Analogously, for the vertex 3 we obtain $\bar{Z}_3 = Z_2 \cdot Z_6 = 3 \cdot 1 = 3$ in Step 4. Since $C_2 = \{1, 5\}$, we have $zu = Z_1 \cdot Z_5 = 1 \cdot 1 = 1$ for $u = 2$ and $zu = 1$ for $u = 6$ in Step 6.2. Finally, for $u = 2$ and $u = 6$ we add $zu \cdot \bar{Z}_3 / Z_2 = 1 \cdot 3/3 = 1$ and $zu \cdot \bar{Z}_3 / Z_6 = 1 \cdot 3/3 = 3$ to Z_3 in Step 6.3. These operations give $Z_3 = 7$. The other values are obtained in an analogous manner. Thus, we obtain $Z_8 = 2$, $Z_{11} = 3$, and $Z_4 = 95$. The Hosoya index of a (sub)tree T_v^4 (resp. the value of $Z(T_v^4 - v)$) is shown at the upper left-hand side (resp. the right hand side) of the vertex v in Figure 1.

An intuitive explanation of the algorithm Hosoya for a tree T rooted at r is as follows.

1. All the entries of the vector Z that correspond to leaves of T_r are set to one.
2. If v is a vertex of T_r such that the Hosoya indices are already computed for all descendants of v, the computation of Z_v goes as follows:
   (a) we multiply the Hosoya indices of all children of v and get the value of $Z(T_v^r - v)$ which is also the initial value of Z_v;
   (b) for every child u of v we multiply the Hosoya indices of all children of u and the Hosoya indices of all children of v with the exception of u. The obtained product is added to the current value of Z_v.

Consider again the rooted tree T_4 in Figure 1 and suppose that the Hosoya indices are already computed for all descendants of 4. For Step a., we obtain the initial value of Z_4, which equals $Z_3 \cdot Z_8 \cdot Z_{11} = 7 \cdot 2 \cdot 3 = 42$. For Step b., we have

$$Z_4 = 42 + Z_2 \cdot Z_6 \cdot Z_8 \cdot Z_{11} + Z_3 \cdot Z_7 \cdot Z_{11} + Z_3 \cdot Z_8 \cdot Z_9 \cdot Z_{10} = 95.$$

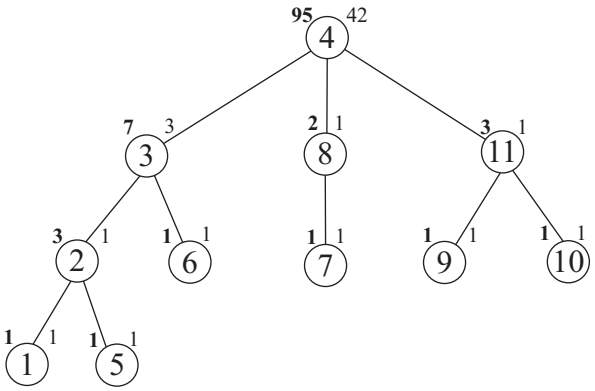

Figure 1. Hosoya indices in T_4.

4. Hosoya Matrix

We will show that the Hosoya matrix of a tree of order n can be computed in $O(n^2)$ time.

4.1. Recursive Formula for Computing the Hosoya Matrix

Proposition 5. *If uv is an edge of a tree T, then*

$$\begin{aligned} Z(T-uv) &= Z(T-V(T_u^v))Z(T_u^v) \\ &= Z(T) - \frac{Z(T-v)Z(T_u^v-u)}{T_u^v}. \end{aligned}$$

Proof. Note first that $T-uv$ is composed of two connected components: $T-V(T_u)$ and T_u^v. Proposition 2 yields

$$Z(T-uv) = Z(T-V(T_u^v))Z(T_u^v).$$

By Proposition 1, we have

$$Z(T-uv) = Z(T) - Z(T-\{u,v\}).$$

The assertion now follows immediately from Proposition 4. □

Theorem 3. *If $P = v_1, v_2, \ldots, v_k$ is a path in a tree T, then*

$$Z(T-E(P)) = (Z(T_{v_{k-1}}^{v_1}) - \frac{Z(T_{v_{k-1}}^{v_1}-v_{k-1})Z(T_{v_k}^{v_1}-v_k)}{Z(T_{v_k}^{v_1})}) \prod_{i=1}^{k-2} \frac{Z(T_{v_i}^{v_1}-v_iv_{i+1})}{Z(T_{v_{i+1}}^{v_1})}.$$

Proof. If $k=2$, then this is Proposition 5.

If $k>2$, we first show that $T-E(P)$ is composed of connected components: $T-V(T_{v_2})$, $T_{v_2}^{v_1}-V(T_{v_3}^{v_1}), \ldots, T_{v_{k-2}}^{v_1}-V(T_{v_{k-1}}^{v_1})$, and the graph $T_{v_{k-1}}^{v_1}-v_{k-1}v_k$. We use induction on k. If $k=3$, then $T_{v_1}^{v_1}-E(P) = T_{v_1}-\{v_1v_2, v_2v_3\}$ is a graph composed of components: $T-V(T_{v_2}^{v_1})$ and $T_{v_2}^{v_1}-v_1v_2$. Let us assume that the proposition holds for all paths of length less than k, and let $P' = v_1, v_2, \ldots, v_{k-1}$ and $P = v_1, v_2, \ldots, v_k$. By the induction hypothesis, $T-E(P')$ is composed of components $T-V(T_{v_2}^{v_1})$, $T_{v_2}^{v_1}-V(T_{v_3}^{v_1}), \ldots, T_{v_{k-3}}^{v_1}-V(T_{v_{k-2}}^{v_1})$, and $T_{v_{k-2}}^{v_1}-v_{k-2}v_{k-1}$. Since $T_{v_{k-2}}^{v_1}-v_{k-2}v_{k-1}$ is composed of two connected components, $T_{v_{k-2}}^{v_1}-V(T_{v_{k-1}}^{v_1})$ and $T_{v_{k-1}}^{v_1}$, and since the edge $v_{k-1}v_k$ belongs to the tree $T_{v_{k-1}}^{v_1}$, the assertion easily follows.

By Proposition 5, we have

$$Z(T_{v_i}^{v_1} - V(T_{v_{i+1}}^{v_1})) = \frac{Z(T_{v_i}^{v_1}-v_iv_{i+1})}{Z(T_{v_{i+1}}^{v_1})}$$

and

$$Z(T_{v_{k-1}}^{v_1} - v_{k-1}v_k) = Z(T_{v_{k-1}}^{v_1}) - \frac{Z(T_{v_{k-1}}^{v_1}-v_{k-1})Z(T_{v_k}^{v_1}-v_k)}{Z(T_{v_k}^{v_1})}.$$

Since $T-E(P)$ is composed of connected components: $T-V(T_{v_2})$, $T_{v_2}^{v_1}-V(T_{v_3}^{v_1}), \ldots,$ $T_{v_{k-2}}^{v_1}-V(T_{v_{k-1}}^{v_1})$, and the graph $T_{v_{k-1}}^{v_1}-v_{k-1}v_k$, Proposition 2 completes the proof. □

Corollary 1. *Let $P = v_1, v_2, \ldots, v_k$ be a path in a tree T. If*

$$Z_k := Z(T_{v_{k-1}}^{v_1}) - \frac{Z(T_{v_{k-1}}^{v_1}-v_{k-1})Z(T_{v_k}^{v_1}-v_k)}{Z(T_{v_k}^{v_1})}$$

and

$$P_k := \begin{cases} 1, & k=2 \\ \frac{Z_{k-1}}{Z(T_{v_{k-1}}^{v_1})}P_{k-1}, & k \geq 3 \end{cases},$$

then

$$Z(T - E(P)) = Z_k P_k.$$

Proof. By Theorem 3, we have to show that $P_k = \prod_{i=1}^{k-2} \frac{Z(T_{v_i}^{v_1} - v_i v_{i+1})}{Z(T_{v_{i+1}}^{v_1})}$. We use induction on k. If $k = 2$, then $P_k = 1$ and the assertion is trivial. Let $k \geq 3$ and assume that the claim holds for $k - 1$. By Proposition 5,

$$Z(T_{v_{k-2}}^{v_1} - v_{k-2}v_{k-1}) = Z(T_{v_{k-2}}^{v_1}) - \frac{Z(T_{v_{k-2}}^{v_1} - v_{k-2})Z(T_{v_{k-1}}^{v_1} - v_{k-1})}{Z(T_{v_{k-1}}^{v_1})} = Z_{k-1}.$$

It follows

$$P_k = \frac{Z_{k-1}}{Z(T_{v_{k-1}}^{v_1})} P_{k-1} = \frac{Z(T_{v_{k-2}}^{v_1} - v_{k-2}v_{k-1})}{Z(T_{v_{k-1}}^{v_1})} P_{k-1}.$$

Since by the induction hypothesis we have

$$P_{k-1} = \prod_{i=1}^{k-3} \frac{Z(T_{v_i}^{v_1} - v_i v_{i+1})}{Z(T_{v_{i+1}}^{v_1})},$$

the assertion follows. □

4.2. Algorithm for Computing the Hosoya Matrix

Algorithm 3 computes the entries of the r-th row of the Hosoya matrix for the tree T. In other words, the algorithm for the rooted tree T_r computes the value $\mathbf{H}_{r,v}$ for every $v \in V(T)$. If v is a vertex of T_r, the list C_v represents the children of v, while p_v represent the parent of v. Vectors Z and $\bar{Z}$ for entries Z_v and $\bar{Z}$ represent $Z(T_v)$ and $Z(T_v - v)$ in T_r, respectively.

Algorithm 3. Row(v, r, C, p, Z, $\bar{Z}$, **H**)

1. **if** $v = r$ **then**
   begin
   1.1 $Pk := 1$;
   1.2 $Zk := Z_v$;
   1.3 $\mathbf{H}_{r,v} := 0$;
   end
2. **else**
   begin
   2.1 $Pk := Pk \cdot \frac{Zk}{Z_{p_v}}$;
   2.2 $Zk := Z_{p_v} - \frac{Z_{p_v}\bar{Z}_v}{Z_v}$;
   2.3 $\mathbf{H}_{r,v} := Zk \cdot Pk$;
   end;
3. **for all** $w \in C_v$ **do**
   Row(w, r, C, p, Z, $\bar{Z}$, **H**);

end.

Proposition 6. *Let $r \in \{1, 2, \ldots, n\}$ be a vertex of a tree T. Then algorithm Row computes the r-th row of the Hosoya matrix of T in linear time.*

Proof. The correctness of the algorithm is based on Corollary 1. Note that p_v and C_v represent the parent of v and the children of v, respectively. The algorithm maintains

variables Zk and Pk representing the values Z_k and P_k of Corollary 1, respectively. We first consider the following two cases.

A. If $v = r$, the correctness easily follows. The algorithm sets the values of Pk and Zk to one and Z_v, respectively.

B. If r is the parent of v, the value of Pk remains equal to one after Step 2.1 is executed, since Zk is set to $Z(T_r)$ in case A. The new value of Z_k is established in Step 2.2., while Step 2.3 provides the correct value of $\mathbf{H}_{r,v}$.

Since in the above cases the correct value of Pk is computed, the correctness for other cases follows from Corollary 1.

For the time complexity, note that the number of recursive calls of the algorithm equals the number of vertices of T_r. Since the time complexity of a single call of the algorithm (neglecting the recursive calls) is constant, the assertion follows. □

Algorithm 4 computes the Hosoya matrix $\mathbf{H}$ for the tree T with the vertex set $V(T) = \{1, 2, \ldots, n\}$.

Algorithm 4. Hosoya matrix(T, $\mathbf{H}$)

```
1. for i := 1 to n do
    begin
      1.1 Rooted(T, i, C, p);
      1.2 Hosoya(i, Z, C, Z̄);
      1.3 Row(i, i, C, p, Z, Z̄, H);
    end;
end.
```

Theorem 4. *Let T be tree of order n. Then algorithm Hosoya matrix computes the Hosoya matrix of T in $O(n^2)$ time and space.*

Proof. For every vertex $i \in V(T) = \{1, 2, \ldots, n\}$, the algorithm creates the rooted tree T_i in Step 1.1. The needed values $Z(T_v)$ and $Z(T_v - v)$ are then computed for every vertex v of T_i in Step 1.2. Finally, all entries of the i-th row of the Hosoya matrix are computed in Step 1.3.

Since the time complexity of the algorithms Rooted, Hosoya, and Row is linear, the time and space bound of the algorithm Hosoya matrix easily follow. □

4.3. Example

Observe again the three T with the vertex set $\{1, 2, \ldots, 11\}$ depicted in Figure 1. We demonstrate the execution of the algorithm Row for the vertex 4 in the sequel. Note that before the algorithm is applied, the algorithm Hosoya, which computes the vectors Z and $\bar{Z}$, is called for the vertex 4.

1. $v = 4$: $Pk = 1$, $Zk = Z_4 = 96$, $\mathbf{H}_{4,4} = 0$.
2. $v = 3$: $Pk = \frac{1 \cdot 95}{95} = 1$, $Zk = 95 - \frac{42 \cdot 3}{7} = 77$, $\mathbf{H}_{4,3} = 77$.
3. $v = 2$: $Pk = \frac{1 \cdot 77}{7} = 11$, $Zk = 7 - \frac{3 \cdot 1}{3} = 6$, $\mathbf{H}_{4,2} = 66$.
4. $v = 1$: $Pk = \frac{11 \cdot 6}{3} = 22$, $Zk = 3 - \frac{1 \cdot 1}{1} = 2$, $\mathbf{H}_{4,1} = 44$.
5. $v = 5$: $Pk = \frac{11 \cdot 6}{3} = 22$, $Zk = 3 - \frac{1 \cdot 1}{1} = 2$, $\mathbf{Z}_{4,5} = 44$.
6. $v = 6$: $Pk = \frac{1 \cdot 77}{7} = 11$, $Zk = 7 - \frac{3 \cdot 1}{1} = 4$, $\mathbf{H}_{4,6} = 44$.
7. $v = 8$: $Pk = \frac{1 \cdot 95}{95} = 1$, $Zk = 95 - \frac{42 \cdot 1}{2} = 74$, $\mathbf{H}_{4,8} = 74$.
8. $v = 7$: $Pk = \frac{1 \cdot 74}{2} = 37$, $Zk = 2 - \frac{1 \cdot 1}{1} = 1$, $\mathbf{H}_{4,7} = 37$.
9. $v = 11$: $Pk = \frac{1 \cdot 95}{95} = 1$, $Zk = 95 - \frac{42 \cdot 1}{3} = 81$, $\mathbf{H}_{4,11} = 81$.
10. $v = 9$: $Pk = \frac{1 \cdot 81}{3} = 27$, $Zk = 3 - \frac{1 \cdot 1}{1} = 2$, $\mathbf{H}_{4,9} = 54$.
11. $v = 10$: $Pk = \frac{1 \cdot 81}{3} = 27$, $Zk = 3 - \frac{1 \cdot 1}{1} = 2$, $\mathbf{H}_{4,10} = 54$.

5. Conclusions

Quantitative structure–property and structure–activity relationships of molecular compounds are frequently modeled using the molecular topological features of these compounds. In that regard, topological indices are crucial for investigating chemical compounds to grasp chemical structures' fundamental topology.

In this paper, we have obtained efficient algorithms for computing the Hosoya index and the Hosoya matrix on an arbitrary acyclic graph. Both algorithms are optimal in the sense that the running time of an algorithm is constant per a fundamental essential item of the input: a vertex (for the computation of the Hosoya index) and an entry of the matrix (for the computation of the Hosoya matrix). The complexity of the presented algorithm for computing the Hosoya index of a tree is within the same time bound as some previously presented procedures. That said, the algorithm presented in this paper is much simpler than its predecessor given in [11] and does not require pre-processing.

It is worth noticing that both presented algorithms exploit the recursive nature of a tree. Thus, a similar approach could be applied for computing the Hosoya index and the Hosoya matrix of tree-like graphs, e.g., cactus graphs, and of graphs derived from trees. Moreover, the presented concepts could initiate studies of efficient methods of computation for other topological indices, especially the ones that are closely connected to the Hosoya index, particularly the Merrifield–Simons index [17,18], the energy of a graph [3], and the matching energy of a graph [19].

Funding: This work was supported by the Slovenian Research Agency under the grants P1-0297, J1-9109, J1-2452, and J1-1693.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The author declares no conflict of interest.

References

1. Hosoya, H. A newly proposed quantity characterizing the topological nature of structural isomers of saturated hydrocarbons. *Bull. Chem. Soc. Jpn.* **1971**, *44*, 2332–2339. [CrossRef]
2. Hosoya, H. The topological index Z before and after 1971. *Internet Electron. J. Mol. Des.* **2002**, *1*, 428–442. [CrossRef]
3. Andriantiana, E.O.D. Energy, Hosoya index and Merrifield-Simmons index of trees with prescribed degree sequence. *Discret. Appl. Math.* **2013**, *161*, 724–741. [CrossRef]
4. Hua, H. On maximal energy and hosoya index of trees without perfect matching. *Bull. Aust. Math. Soc.* **2010**, *81*, 47–57. [CrossRef]
5. Kazemi, R.; Behtoei, A. Hosoya index of tree structures. *Trans. Comb.* **2020**, *9*, 161–169.
6. Tian, W.; Zhao, F.; Sun, Z.; Mei, X.; Chen, G. Orderings of a class of trees with respect to the Merrifield–Simmons index and the Hosoya index. *J. Comb. Optim.* **2019**, *38*, 1286–1295. [CrossRef]
7. Chen, X.; Zhang, J.; Sun, W. On the Hosoya index of a family of deterministic recursive trees. *J. Phys. A Stat. Mech. Appl.* **2017**, *465*, 449–453. [CrossRef]
8. Xiao, C.; Chen, H. Kekulé structures of square–hexagonal chains and the Hosoya index of caterpillar trees. *Discret. Math.* **2016**, *339*, 506–510. [CrossRef]
9. Heuberger, C.; Wagner, S.G. Chemical Trees Minimizing Energy and Hosoya Index. *J. Math. Chem.* **2009**, *46*, 214–230. [CrossRef]
10. Jerrum, M. Two-dimensional monomer-dimer systems are computationally intractable. *J. Stat. Phys.* **1987**, *48*, 121–134. [CrossRef]
11. Zhang, J.; Chen, X.; Sun, W. A Linear-Time Algorithm for the Hosoya Index of an Arbitrary Tree. *MATCH Commun. Math. Comput. Chem.* **2016**, *75*, 703–714.
12. Mohar, B. Computing the characteristic polynomial of a tree. *J. Math. Chem.* **1989**, *3*, 403–406. [CrossRef]
13. Liu, J.-B.; Zhao, J.; Min, J.; Cao, J. The Hosoya index of graphs formed by a fractal graph. *Fractals* **2019**, *27*, 1950135. [CrossRef]
14. Randić, M. Novel molecular descriptor for structure–property studies. *Chem. Phys. Lett.* **1993** , *211*, 478–483. [CrossRef]
15. Randić, M. Hosoya matrix—A source of new molecular descriptors. *Croat. Chem. Acta* **1994**, *67*, 415–429.
16. Gutman, I.; Plavšić, D.; Šoškić, M.; Landeka, I.; Graovac, A. On the Calculation of the Path Numbers ${}^1Z$, ${}^2Z$ and the Hosoya Z Index. *Croat. Chem. Acta* **1997**, *70*, 941–954.
17. Wagner, S.; Gutman, I. Maxima and minima of the Hosoya index and the Merrifeld-Simmons index. *Acta. Appl. Math.* **2010**, *112*, 323–346. [CrossRef]

18. Huang, Y.; Shi, L.; Xu, X. The Hosoya index and the Merrifield–Simmons index. *J. Math. Chem.* **2018**, *56*, 3136–3146. [CrossRef]
19. Chen, H.; Deng, H. Extremal bipartite graphs of given connectivity with respect to matching energy. *Discret. Appl. Math.* **2018**, *239*, 200–205. [CrossRef]