Neutrosophic Sets and Systems, Vol. 77, 2025

University of New Mexico

Short note of supertree-width and n-Superhypertree-width

Takaaki Fujita[1]∗

[1]Independece Researcher; Shinjuku, Shinjuku-ku, Tokyo, Japan; t171d603@gunma-u.ac.jp

Abstract. This paper investigates the properties of tree-width and related graph width parameters for n-SuperHyperGraphs, a broader generalization of hypergraphs. By exploring concepts such as SuperHyperTree-width and Hypertree-width, we aim to understand how these structures resemble tree-like formations. We also analyze the relevance of these width parameters in computational contexts, following extensive research in graph theory and hypergraph applications.

Keywords: hypergraph, supergraph, superhypergraph, treewidth, hypertree-width

1. Introduction

1.1. *Graph theory*

Graph theory, a fundamental branch of mathematics, focuses on the study of networks made up of nodes and edges, examining their paths, structures, and properties [22]. Many concepts in the world can be modeled using graphs [19, 55]. While graphs are highly useful tools, in algorithm development and problem-solving, they are often utilized after determining whether specific structures, such as tree or path structures, are present [14, 22]. Alternatively, they are used in cases where constraints are imposed by particular structures. A graph parameter is a numerical value assigned to a graph that describes certain structural properties, such as size or connectivity [80].

1.2. *Supergraph and Hypergraph*

A *supergraph* is a graph G that contains another graph H, where $V(G) \supseteq V(H)$ and $E(G) \supseteq E(H)$. Several studies have been conducted from the perspective of efficient algorithm development.

And a hypergraph is a generalization of a conventional graph, abstracting and extending concepts from graph theory [15, 46, 51, 52, 56]. Hypergraphs have broad applications in fields such as machine learning and network analysis [16, 44, 65, 69].

1.3. *SuperHyperGraph*

The concept of the *SuperHyperGraph* was introduced by Florentin Smarandache in 2019 [81] as a broad extension of both hypergraphs and supergraphs. This framework has since generated significant research interest, akin to the study of hypergraphs. Recently, an advanced structure called the *n-SuperHyperGraph* was introduced, where vertices and edges are defined using n-power sets [81]. Both SuperHyperGraphs and n-SuperHyperGraphs have become key areas of exploration [47, 57–59, 81, 82, 84, 86, 89], with each of these structures based on the n-th power set of a set H, reflecting the complexity of real-world systems.

Alongside SuperHyperGraph, related structures have been defined, such as the SuperHyperStructure and Neutrosophic SuperHyperStructure, including specific instances like SuperHyperAlgebra [89, 91]. Furthermore, concepts like SuperHyperSoft Set [87], SuperHyperFunction [85], and SuperHyperTopology [88] have been proposed. These structures, like SuperHyperGraph, provide abstract frameworks for modeling functions, algebra, and topology.

Tree graphs have been widely applied across various fields. Similarly, in the context of SuperHyperGraphs, the concept of a Tree graph has been defined, and more recently, the notion of SuperHyperTree-width has been introduced [32, 38, 43]. The explanation of width parameters and Tree-width will follow. Additionally, related width parameters have also been established in the literature [43].

1.4. *Graph Width Parameter*

A "Graph width parameter" measures the maximum width across all cuts or layers within a hierarchical decomposition of a graph [1, 5–7, 21, 23–25, 29–33, 36–38, 41–43, 45, 74, 77–79]. This parameter is essential for analyzing the complexity and structure of a graph, playing a pivotal role in transforming computationally challenging graph problems into more tractable ones, particularly when the graph class has bounded width.

Tree-width is one of the most prominent graph width parameters [7–13, 68, 77–79]. It measures how closely a graph resembles a tree by determining the minimum width of a tree decomposition, thus reflecting the graph's degree of tree-likeness.

This has spurred extensive research on Hypertree-width [3, 50, 52, 70, 96] and Hyperpath-width [2, 72, 73], which quantify how much a hypergraph approximates a tree or a path. Hypertree-width, in particular, has found significant applications in fields like database systems [17, 27, 28, 46, 49, 51–54, 93].

1.5. *Our Contribution*

Building on the historical development of various graph classes, the concept of *superhypertree-width* has been recently introduced [38]. However, its concrete characteristics remain largely unexplored, presenting ample opportunities for further research. Additionally, the study of even more abstract structures, such as n-superhypergraphs, has emerged, yet the corresponding notions of tree-width for these structures have not been fully defined [82].

In this paper, we investigate the characteristics of tree-width and related graph width parameters in the context of supergraphs and n-superhypergraphs. Through this exploration, we aim to stimulate more active research in the areas of graph width parameters and superhypergraphs, thereby advancing the field of graph theory.

2. **Preliminaries and definitions**

In this section, we briefly explain the definitions and notations used in this paper.

2.1. *Basic Graph Concepts*

A graph G is a mathematical structure consisting of nodes (vertices) connected by edges, representing relationships or connections. In a graph G, $V(G)$ denotes the set of vertices, and $E(G)$ denotes the set of edges. The notation $G = (V, E)$ indicates that the graph G is defined by the pair of sets V (vertices) and E (edges).

Definition 2.1. A subgraph is formed by selecting specific vertices and edges from a graph.

Definition 2.2. A path is a walk with no repeated vertices, a cycle is a closed path, and a tree is a connected acyclic graph.

Example 2.3. Consider the following graphs:

- Path: A path graph P_4 consists of 4 vertices: v_1, v_2, v_3, v_4, and 3 edges:

$$\{v_1, v_2\}, \{v_2, v_3\}, \{v_3, v_4\}.$$

  This is a walk with no repeated vertices.
- Cycle: A cycle graph C_4 consists of 4 vertices: v_1, v_2, v_3, v_4, and 4 edges:

$$\{v_1, v_2\}, \{v_2, v_3\}, \{v_3, v_4\}, \{v_4, v_1\}.$$

  This forms a closed loop.
- Tree: A tree graph with 5 vertices: v_1, v_2, v_3, v_4, v_5, and 4 edges:

$$\{v_1, v_2\}, \{v_1, v_3\}, \{v_2, v_4\}, \{v_3, v_5\}.$$

  This graph is connected and acyclic, satisfying the conditions of a tree.

Definition 2.4. A *star* in a graph is a tree consisting of one central vertex, called the *center*, and a set of leaves, which are vertices that are connected directly to the center by edges. Formally, a star with n leaves is a tree with $n+1$ vertices, where one vertex has degree n (the center) and the remaining n vertices have degree 1 (the leaves).

Example 2.5. Consider a star graph S_4, where $n = 4$. The graph consists of a central vertex v_1 and four leaves v_2, v_3, v_4, v_5, each connected directly to v_1 by edges. The set of vertices is $V(S_4) = \{v_1, v_2, v_3, v_4, v_5\}$, and the set of edges is $E(S_4) = \{\{v_1, v_2\}, \{v_1, v_3\}, \{v_1, v_4\}, \{v_1, v_5\}\}$. The degree of the central vertex v_1 is 4, and each leaf vertex v_2, v_3, v_4, v_5 has a degree of 1.

Definition 2.6. [61] An empty graph is a graph with a set of vertices but no edges, meaning no connections between any vertices. A *null graph (order-zero graph)* is a graph that contains no edges. It can have zero or more vertices, but none of its vertex pairs are connected by an edge.

For additional basic graph notation and concepts, please refer to [22, 55, 55, 95].

2.2. *Tree-width of basic graph*

In graph theory, there is a concept known as graph minor theory. A graph minor is created by performing operations such as deleting edges, deleting vertices, or contracting edges from a larger graph. Tree-width is a measure of how close a graph is to resembling a tree, by representing it with a tree-like structure that has minimal width [7–9, 66, 78, 79, 79]. Below are the formal definitions of Tree-width.

Definition 2.7. [79] A tree-decomposition of an undirected graph G is a pair (T, W), where T is a tree, and $W = (W_t \mid t \in V(T))$ is a family of subsets that associates with every node t of T a subset W_t of vertices of G such that:

(T1) $\bigcup_{t \in V(T)} W_t = V(G)$,
(T2) For each edge $(u, v) \in E(G)$, there exists some node t of T such that $\{u, v\} \subseteq W_t$, and
(T3) For all nodes r, s, t in T, if s is on the unique path from r to t then $W_r \cap W_t \subseteq W_s$.

The width of a tree-decomposition (T, W) is the maximum of $|W_t| - 1$ over all nodes t of T. The tree-width of G is the minimum width over all tree-decompositions of G.

Additionally, path-width [20, 67, 92] and cycle-width [63, 94] are the path and cycle versions of tree-width, respectively.

Example 2.8. Consider the following graph G:

$$V(G) = \{v_1, v_2, v_3, v_4\}$$

$$E(G) = \{(v_1, v_2), (v_1, v_3), (v_2, v_4), (v_3, v_4)\}$$

We aim to construct a tree-decomposition for this graph G. Let T be a tree with three nodes t_1, t_2, t_3. The corresponding bags W_t are defined as:

$$W_{t_1} = \{v_1, v_2, v_3\}, \quad W_{t_2} = \{v_2, v_3, v_4\}, \quad W_{t_3} = \{v_4\}$$

- *(T1):* The union of all the bags covers the entire vertex set of G:

$$W_{t_1} \cup W_{t_2} \cup W_{t_3} = \{v_1, v_2, v_3, v_4\} = V(G)$$

- *(T2):* For each edge $(u, v) \in E(G)$, there exists a bag that contains both u and v:
  - (v_1, v_2) is in W_{t_1},
  - (v_1, v_3) is in W_{t_1},
  - (v_2, v_4) is in W_{t_2},
  - (v_3, v_4) is in W_{t_2}.
- *(T3):* For all nodes r, s, t in T, if s lies on the unique path from r to t, then:

$$W_r \cap W_t \subseteq W_s$$

  This holds, as $W_{t_1} \cap W_{t_3} = \{v_4\} \subseteq W_{t_2}$.

The size of each bag is:

$$|W_{t_1}| = 3, \quad |W_{t_2}| = 3, \quad |W_{t_3}| = 1$$

Thus, the width of this tree-decomposition is the maximum bag size minus 1:

$$\text{width} = \max(3 - 1, 3 - 1, 1 - 1) = 2$$

For information on tree-width, there are numerous surveys and lecture notes available. Please refer to those as needed [7, 9, 62].

2.3. *Hypergraph Concepts*

A hypergraph is a generalization of a graph where edges, called hyperedges, can connect any number of vertices, not just two. This structure is useful for modeling complex relationships in various fields like computer science and biology [15, 26, 48, 75]. The definition is provided below.

Definition 2.9. [4] A hypergraph is a pair $H = (V(H), E(H))$, consisting of a nonempty set $V(H)$ of vertices and a set $E(H)$ of subsets of $V(H)$, called the hyperedges of H. In this paper, we consider only finite hypergraphs.

Example 2.10. Consider the hypergraph $H = (V(H), E(H))$, where:

$$V(H) = \{1, 2, 3, 4, 5\}$$

The hyperedges are subsets of $V(H)$, defined as:

$$E(H) = \{\{1, 2, 3\}, \{2, 4\}, \{3, 5\}\}$$

In this case, $V(H)$ consists of 5 vertices, and $E(H)$ consists of 3 hyperedges, where each hyperedge is a subset of $V(H)$. For instance, the first hyperedge $\{1, 2, 3\}$ connects the vertices 1, 2, and 3, while the second hyperedge $\{2, 4\}$ connects vertices 2 and 4.

Definition 2.11. [15] For a hypergraph H and a subset $X \subseteq V(H)$, the subhypergraph induced by X is defined as $H[X] = (X, \{e \cap X \mid e \in E(H)\})$. We denote the hypergraph obtained by removing X from H as $H \setminus X := H[V(H) \setminus X]$.

Example 2.12. Consider the hypergraph $H = (V(H), E(H))$ where:

$$V(H) = \{1, 2, 3, 4, 5\}$$

The hyperedges are:

$$E(H) = \{\{1, 2, 3\}, \{2, 4\}, \{3, 5\}\}$$

Now, let $X = \{2, 3, 4\}$ be a subset of $V(H)$. The subhypergraph induced by X is:

$$H[X] = (\{2, 3, 4\}, \{\{2, 3\}, \{2, 4\}\})$$

This subhypergraph contains only the vertices in X and the hyperedges that connect these vertices. The first hyperedge $\{1, 2, 3\}$ from the original hypergraph becomes $\{2, 3\}$ after intersecting with X, and the hyperedge $\{2, 4\}$ remains unchanged as it is already a subset of X.

For more basic hypergraph notation and concepts, please refer to [15, 18].

2.4. *Hypertree decomposition*

Hypertree-width measures how well a hypergraph can be decomposed into a tree-like structure, minimizing the maximum size of edge subsets (guards) at each node. Hypertree-width has a wide range of applications and is extensively studied in various fields [28, 53].

Definition 2.13. [3] A generalized hypertree decomposition of H is a triple (T, B, C), where (T, B) is a tree-decomposition of H and $C = (C_t)_{t \in V(T)}$ is a family of subsets of $E(H)$ such that for every $t \in V(T)$ we have $B_t \subseteq \bigcup C_t$. Here $\bigcup C_t$ denotes the union of the sets (hyperedges) in C_t, that is, the set $\{v \in V(H) \mid \exists e \in C_t : v \in e\}$. The sets C_t are called the guards of the decomposition. The width of the decomposition (T, B, C) is $\max\{|C_t| \mid t \in V(T)\}$. The generalized hypertree width of H, denoted by $\mathrm{ghw}(H)$, is the minimum of the widths of the generalized hypertree decompositions of H.

A hypertree decomposition of H is a generalized hypertree decomposition (T, B, C) that satisfies the following special condition: $(\bigcup C_t) \cap \bigcup_{u \in V(T_t)} B_u \subseteq B_t$ for all $t \in V(T)$. Recall that T_t denotes the subtree of the T with root t. The hypertree width of H, denoted by $\mathrm{hw}(H)$, is the minimum of the widths of all hypertree decompositions of H.

Definition 2.14. (cf. [71]) A *hypertree* T is a connected hypergraph in which the removal of any hyperedge from T results in a disconnected hypergraph. Specifically:

- For any hyperedge $e \in E(T)$, if e is removed, the resulting subhypergraph is no longer connected.
- A hypertree can contain cycles, as long as removing any hyperedge disconnects the hypergraph.

2.5. *Supergraph*

A *supergraph* is a graph G that contains another graph H, where $V(G) \supseteq V(H)$ and $E(G) \supseteq E(H)$. The following provides the definition and an example of a supergraph [60, 76].

Definition 2.15. (cf. [60, 76]) Given two graphs G and H:

- G is a *supergraph* of H if $V(G) \supseteq V(H)$ and $E(G) \supseteq E(H)$, meaning that the vertex set and edge set of G contain those of H.
- G is an *induced supergraph* of H if G is a supergraph of H and $E(G) = E(H) \cap (V(G) \times V(G))$. This means G includes all edges of H that exist between vertices in $V(G)$.

Example 2.16. Let H be a graph with vertex set $V(H) = \{1, 2, 3\}$ and edge set $E(H) = \{\{1, 2\}, \{2, 3\}\}$.

Now, consider a graph G with vertex set $V(G) = \{1, 2, 3, 4\}$ and edge set $E(G) = \{\{1, 2\}, \{2, 3\}, \{3, 4\}, \{1, 4\}\}$.

In this case:

- G is a *supergraph* of H because $V(G) \supseteq V(H)$ and $E(G) \supseteq E(H)$.
- However, G is not an *induced supergraph* of H because it contains additional edges, such as $\{1, 4\}$ and $\{3, 4\}$, which do not exist in H.

Alternatively, if we define a new graph G' with $V(G') = \{1, 2, 3\}$ and $E(G') = \{\{1, 2\}, \{2, 3\}\}$, then G' is an *induced supergraph* of H, as the vertices and edges of H are preserved without any additional edges.

Definition 2.17. (cf. [76]) A supergraph that is a forest or tree is called a *superforest* or *supertree*, respectively.

Example 2.18. Consider the graph H which is a tree with vertices $V(H) = \{1, 2, 3\}$ and edges $E(H) = \{\{1, 2\}, \{2, 3\}\}$.

Now, let G be a supergraph of H with vertex set $V(G) = \{1, 2, 3, 4\}$ and edge set $E(G) = \{\{1, 2\}, \{2, 3\}, \{3, 4\}\}$.

In this case:

- G is a *supertree* of H because G is a supergraph of H and both G and H are trees.
- If G had multiple trees (i.e., was a forest), it would be called a *superforest* of H.

2.6. *SuperHyperGraph*

A SuperHyperGraph is an advanced structure extending hypergraphs and supergraphs by allowing vertices and edges to be sets. The concept of the SuperHyperGraph was introduced by Florentin Smarandache in 2019 in [81]. The definition is provided below [81].

Definition 2.19. [81] A *SuperHyperGraph (SHG)* is an ordered pair SHG $= (G \subseteq P(V), E \subseteq P(V))$, where:

(1) $V = \{V_1, V_2, \ldots, V_m\}$ is a finite set of $m \geq 0$ vertices, or an infinite set.
(2) $P(V)$ is the power set of V (all subsets of V). Therefore, an SHG-vertex may be a single (classical) vertex, a super-vertex (a subset of many vertices) that represents a group (organization), or even an indeterminate-vertex (unclear, unknown vertex); $\emptyset$ represents the null-vertex (a vertex that has no element).
(3) $E = \{E_1, E_2, \ldots, E_m\}$, for $m \geq 1$, is a family of subsets of V, and each E_j is an SHG-edge, $E_i \in P(V)$. An SHG-edge may be a (classical) edge, a super-edge (an edge between super-vertices) that represents connections between two groups (organizations), a hyper-super-edge that represents connections between three or more groups (organizations), a multi-edge, or even an indeterminate-edge (unclear, unknown edge); $\emptyset$ represents the null-edge (an edge that means there is no connection between the given vertices).

Example 2.20. We define a finite set of vertices:

$$V = \{v_1, v_2, v_3, v_4, v_5\}$$

In a SuperHypergraph, vertices can be single vertices, super-vertices (subsets of vertices), or even the empty set. Here are some examples:

- Single Vertices:

$$V_1 = \{v_1\}$$
$$V_2 = \{v_2\}$$
$$V_3 = \{v_3\}$$
$$V_4 = \{v_4\}$$
$$V_5 = \{v_5\}$$

- Super-Vertices (Subset Vertices):

$$\mathrm{SV}_{1,2} = \{v_1, v_2\}$$
$$\mathrm{SV}_{3,4,5} = \{v_3, v_4, v_5\}$$

- Null Vertex:

$$\emptyset_V = \emptyset$$

Edges in a SuperHypergraph can be single edges, super-edges, hyper-edges, hyper-super-edges, or the null edge. Below are definitions of each type:

- Single Edges:
  - $E_{1,2} = \{V_1, V_2\}$: An edge connecting v_1 and v_2
  - $E_{4,5} = \{V_4, V_5\}$: An edge connecting v_4 and v_5
- Hyper-Edges:

$$\mathrm{HE}_{1,3,5} = \{V_1, V_3, V_5\}$$

  An edge connecting v_1, v_3, and v_5
- Super-Edges (Subset Edges):

$$\mathrm{SE}_{(1,2),(3)} = \{\mathrm{SV}_{1,2}, V_3\}$$

  An edge connecting super-vertex $\mathrm{SV}_{1,2}$ and vertex v_3
- Hyper-Super-Edges (Hyper Subset Edges):

$$\mathrm{HSE}_{(1,2),(3,4,5)} = \{\mathrm{SV}_{1,2}, \mathrm{SV}_{3,4,5}\}$$

  An edge connecting super-vertices $\mathrm{SV}_{1,2}$ and $\mathrm{SV}_{3,4,5}$
- Null Edge:

$$\emptyset_E = \emptyset$$

  Represents the absence of a connection

Definition 2.21. [38,43] A SuperHyperTree (SHT) is a SuperHyperGraph SHT that satisfies the following conditions:

(1) Host Graph Condition: There exists a host graph T, which is a tree, such that:
   - T has the same vertex set V as SHT.
   - T has edges corresponding to the connections between vertices in V.

(2) SuperHyperTree Condition: Every hyperedge E_i in E of the SuperHyperGraph SHT corresponds to a connected subtree of the host tree T. Specifically:
   - Each hyperedge E_i can be a single edge (connecting two vertices), a super-edge (connecting subsets of vertices, where each subset is a connected subtree in T), or a hyper-edge (connecting more than two vertices, with the condition that they all form a connected subtree in T).
   - If E_i is an indeterminate edge, it must satisfy the condition that for any realization of E_i, the vertices involved still form a connected subtree in T.

(3) Acyclic Condition: The host graph T must be acyclic, which is a fundamental property of trees. Therefore, SHT inherits this acyclic nature through the structure of its hyperedges.

Properties of a SuperHyperTree:

- Connectedness: A SuperHyperTree is connected, meaning that there is a path between any two vertices through a series of hyperedges.
- No Cycles: Since T is a tree, SHT does not contain any SuperHypercycles.
- Generalization of Trees: A SuperHyperTree generalizes the concept of a tree by allowing super-vertices and super-edges while maintaining the acyclic and connected properties of a tree.

2.7. *SuperHyperTree-width*

SuperHyperTree-width is an abstraction of Hypertree-width, extending the concept of Tree-width. The definition is presented as follows [32, 38, 43].

Definition 2.22. [32, 38, 43] Let SHT $= (V, E)$ be a SuperHyperGraph (SHG), where V is the set of vertices and E is the set of SuperEdges.

A *SuperHyperTree decomposition* of SHT is a tuple $(T, \mathcal{B}, \mathcal{C})$ where:

- $T = (V_T, E_T)$ is a tree.
- $\mathcal{B} = \{B_t \mid t \in V_T\}$ is a family of subsets of V (called bags) associated with the nodes of the tree T, such that:

(1) Coverage Condition for SuperEdges: For each SuperEdge $e \in E$, there exists a node $t \in V_T$ such that the entire SuperEdge e is contained within the corresponding bag B_t, i.e., $e \subseteq B_t$.

(2) Vertex Connectivity Condition: For each vertex $v \in V$, the set of nodes $\{t \in V_T \mid v \in B_t\}$ forms a connected subtree of T.

- $\mathcal{C} = \{C_t \mid t \in V_T\}$ is a family of subsets of E (called guards) associated with the nodes of the tree T, such that:

  (1) Guard Condition for SuperEdges: For each $t \in V_T$, $B_t \subseteq \bigcup C_t$, where $\bigcup C_t$ denotes the union of all SuperEdges in C_t, i.e., $\bigcup C_t = \{v \in V \mid \exists e \in C_t : v \in e\}$.

  (2) SuperHyperTree Condition: For each $t \in V_T$, $(\bigcup C_t) \cap \bigcup_{u \in V(T_t)} B_u \subseteq B_t$, where T_t denotes the subtree of T rooted at t.

The *width* of the SuperHyperTree decomposition $(T, \mathcal{B}, \mathcal{C})$ is defined as:

$$\text{width}(T, \mathcal{B}, \mathcal{C}) = \max_{t \in V_T} |C_t|,$$

where $|C_t|$ is the cardinality of the guard C_t.

The *SuperHyperTree-width* (SHT-width) of the SuperHyperGraph SHT, denoted by SHT-width(SHG), is the minimum width over all possible SuperHyperTree decompositions of SHG:

$$\text{SHT-width(SHT)} = \min_{(T,\mathcal{B},\mathcal{C})} \text{width}(T, \mathcal{B}, \mathcal{C}).$$

- The SuperHyperTree-width is a measure of how close the SuperHyperGraph is to being a SuperHyperTree.
- For a classical graph, the SuperHyperTree-width coincides with the treewidth.

A *SuperHyperPath decomposition* of SHG is a path version of SuperHyperTree-decomposition.

2.8. *n-SuperHyperGraph*

We define the generalization of a SuperHyperGraph, called an n-SuperHyperGraph, as follows [82, 83, 90].

Definition 2.23. (cf. [82, 83, 90]) The n-SuperHyperGraph (n-SHG) is a generalization of classical graph theory, extending to higher-order structures. Let $V = \{v_1, v_2, \ldots, v_m\}$ be a set of vertices, where $1 \leq m \leq \infty$. The set V includes:

- Single Vertices: The classical vertices;
- Indeterminate Vertices: Vertices that are unclear, vague, or partially known;
- Null Vertices: Vertices that are totally unknown or empty.

Let $\mathcal{P}(V)$ denote the power set of V, including the empty set $\varnothing$. The n-power set $\mathcal{P}_n(V)$ is defined recursively as:

$$\mathcal{P}_1(V) = \mathcal{P}(V), \quad \mathcal{P}_2(V) = \mathcal{P}(\mathcal{P}(V)), \quad \ldots, \quad \mathcal{P}_n(V) = \mathcal{P}(\mathcal{P}_{n-1}(V)), \quad 1 \leq n \leq \infty,$$

with the base case $\mathcal{P}_0(V) = V$.

An n-SuperHyperGraph is an ordered pair:

$$\text{n-SHG} = (G_n, E_n),$$

where $G_n \subseteq \mathcal{P}_n(V)$ and $E_n \subseteq \mathcal{P}_n(V)$, for $1 \leq n \leq \infty$. Here, G_n represents the set of vertices, and E_n represents the set of edges.

The vertex set G_n contains the following types of vertices:

- Single Vertices: The classical vertices;
- Indeterminate Vertices: Vertices that are unclear, vague, or partially unknown;
- Null Vertices: Vertices that are totally unknown or empty;
- SuperVertex: A collection of two or more vertices (single, indeterminate, or null) grouped together;
- n-SuperVertex: A collection of vertices, where at least one is an $(n-1)$-SuperVertex, and all other vertices are r-SuperVertices with $r \leq n-1$.

The edge set E_n contains the following types of edges:

- Single Edges: The classical edges;
- Indeterminate Edges: Edges that are unclear, vague, or partially unknown;
- Null Edges: Edges that are totally unknown or empty;
- HyperEdge: An edge connecting three or more single vertices;
- SuperEdge: An edge connecting two vertices, where at least one is a SuperVertex;
- n-SuperEdge: An edge connecting two vertices, where at least one is an n-SuperVertex and the other is an r-SuperVertex, with $r \leq n$;
- SuperHyperEdge: An edge connecting three or more vertices, where at least one is a SuperVertex;
- n-SuperHyperEdge: An edge connecting three or more vertices, where at least one is an n-SuperVertex and the others are r-SuperVertices, with $r \leq n$;
- MultiEdges: Two or more edges connecting the same pair of vertices;
- Loop: An edge that connects a vertex to itself.

Additionally, an n-SuperHyperGraph may contain the following graph types:

- Directed Graph: A classical directed graph;
- Undirected Graph: A classical undirected graph;

- Neutrosophic Directed Graph: A graph that is partially directed, partially undirected, and has indeterminate directionality.

3. New definition and Result of this paper

The results in this paper are presented as follows.

3.1. *Comparison of Graph Classes*

In this subsection, we compare each graph class. The theorem is presented below.

Theorem 3.1. *Let SHG $= (V, E)$ be a SuperHyperGraph, where V is the set of vertices and E is the set of hyperedges (subsets of V). Then, SHG is a supergraph of the base graph $H = (V, E_H)$, where $E_H \subseteq E$ and each hyperedge in E_H is a pair of vertices from V (i.e., classical edges). In other words, every SuperHyperGraph is a supergraph.*

Proof. Let SHG $= (V, E)$ be a SuperHyperGraph, where V is the set of vertices, and E is the set of hyperedges.

We define the base graph $H = (V, E_H)$ as a classical graph where:

- $V(H) = V(\text{SHG})$ (the vertex set of H is the same as the vertex set of SHG),
- $E_H \subseteq \{\{u, v\} \mid u, v \in V(H), u \neq v\}$ consists only of edges between pairs of vertices.

In a SuperHyperGraph, the edge set E can contain not only classical edges between two vertices, but also hyperedges that connect multiple vertices. This means that for each edge in E_H, there may be a corresponding hyperedge in E, such that:

$$E_H \subseteq E$$

Thus, SHG contains all the edges of H, and possibly more hyperedges that involve three or more vertices. Therefore, SHG satisfies the condition that $E(\text{SHG}) \supseteq E(H)$.

By the definition of a supergraph, SHG is a supergraph of H if:

- $V(\text{SHG}) = V(H)$,
- $E(\text{SHG}) \supseteq E(H)$.

Since both conditions are satisfied, SHG is a supergraph of H. □

Theorem 3.2. *(cf. [82]) Let $n = 1$. An n-SuperHyperGraph (n-SHG) is equivalent to a supergraph. Specifically, for $n = 1$, the n-SuperHyperGraph $SHG_1 = (G_1, E_1)$, where $G_1 \subseteq \mathcal{P}(V)$ and $E_1 \subseteq \mathcal{P}(V)$, forms a supergraph of a base graph H, which is constructed by interpreting each vertex in G_1 as a single vertex in V and each edge in E_1 as a set of edges that contain classical edges.*

Proof. By definition, an n-SuperHyperGraph (n-SHG) for any $n \geq 1$ is an ordered pair (SHG_n, E_n), where $G_n \subseteq \mathcal{P}_n(V)$ and $E_n \subseteq \mathcal{P}_n(V)$, allowing for vertices and edges to be subsets of the power set $\mathcal{P}_n(V)$.

For $n = 1$, the structure simplifies. In this case, the vertex set $G_1 \subseteq \mathcal{P}(V)$ consists of single vertices and subsets of V, while the edge set $E_1 \subseteq \mathcal{P}(V)$ consists of subsets of these vertices. This means G_1 represents either single vertices or groups of vertices, and E_1 represents edges connecting these vertices or groups.

Now, consider a supergraph G that contains another graph H, where $V(G) \supseteq V(H)$ and $E(G) \supseteq E(H)$. Since in $n = 1$-SuperHyperGraph, the vertices are directly from the set V (without further power set iterations), it follows that the 1-SuperHyperGraph is a graph that contains subsets of vertices and their corresponding edges, just like a supergraph. Thus, for $n = 1$, the n-SuperHyperGraph satisfies the conditions of being a supergraph. □

Theorem 3.3. *Let SHT $= (V, E)$ be a SuperHyperTree, where V is the set of vertices and E is the set of hyperedges (subsets of V). Then, SHT is a Hypertree.*

Proof. A Hypertree is defined as a hypergraph in which every hyperedge corresponds to a connected subtree of a host tree T.

By definition, a SuperHyperTree satisfies the following conditions:

- There exists a host tree T, with the same vertex set V as SHT, and each hyperedge $E_i \in E$ corresponds to a connected subtree of T.
- Each hyperedge E_i can be either a single edge or a hyperedge (connecting more than two vertices), but all vertices in E_i must form a connected subtree in T.

Since every hyperedge in a SuperHyperTree corresponds to a connected subtree of the host tree T, SHT satisfies the necessary condition to be a Hypertree. Therefore, SHT is a Hypertree. □

Theorem 3.4. *Let SHT $= (V, E)$ be a SuperHyperTree. Then, SHT is a Supertree.*

Proof. A Supertree is defined as a supergraph that is a tree, meaning it must satisfy the following conditions:

- The vertex set $V(G)$ of the supergraph G contains the vertex set $V(H)$ of the base tree H.
- The edge set $E(G)$ contains the edge set $E(H)$ of H, and the structure of G must form a tree.

A SuperHyperTree SHT $= (V, E)$ includes vertices and hyperedges where each hyperedge corresponds to a connected subtree of a host tree T, and no cycles are allowed, as the host

tree T is acyclic by definition. Moreover, the supergraph structure includes all vertices and edges of the base tree T, and thus SHT satisfies the conditions of a Supertree.

Therefore, SHT is a Supertree. □

3.2. *Supertree-width*

Supertree-width is an extension of the tree-width concept for supergraphs, designed to measure how closely a supergraph resembles a tree-like structure while considering the additional vertices and edges introduced in the supergraph. It generalizes tree-width for more complex structures, such as supergraphs and superhypergraphs, which may contain extra vertices and edges not present in the original graph.

Definition 3.5. Let G be a supergraph of another graph H, where $V(G) \supseteq V(H)$ and $E(G) \supseteq E(H)$. The *supertree-width* of G is defined in terms of a tree-decomposition of the supergraph G.

A *supertree-decomposition* of a supergraph G is a pair (T, W), where T is a tree, and $W = (W_t \mid t \in V(T))$ is a family of subsets (called *bags*) of the vertices of G, satisfying the following conditions:

(1) Vertex Coverage: The union of all bags covers all vertices of G:

$$\bigcup_{t \in V(T)} W_t = V(G)$$

(2) Edge Containment: For each edge $(u, v) \in E(G)$, there exists a bag W_t such that both u and v are in W_t:

$$\exists t \in V(T) : \{u, v\} \subseteq W_t$$

(3) Tree Structure Condition: For every three nodes r, s, t in T, if s lies on the unique path from r to t, then:

$$W_r \cap W_t \subseteq W_s$$

The *width* of a supertree-decomposition is defined as the maximum size of any bag minus 1:

$$\text{width}(T, W) = \max_{t \in V(T)} (|W_t| - 1)$$

The *supertree-width* of the supergraph G is the minimum width over all possible supertree-decompositions of G.

Theorem 3.6. *Let G be a supergraph of another graph H. The supertree-width of an empty graph can be characterized as follows:*

(1) *For an empty graph G, the supertree-width is* 0.

Proof. To prove this theorem, we analyze the supertree-width of empty graphs.

An empty graph is a graph G with vertices but no edges, i.e., $V(G) = \{v_1, v_2, \ldots, v_n\}$ and $E(G) = \emptyset$. For an empty graph, a tree-decomposition can be constructed by creating a bag for each vertex in $V(G)$, where each bag contains only one vertex. Formally, for each vertex $v_i \in V(G)$, we define a bag $W_{t_i} = \{v_i\}$.

Since there are no edges to satisfy the edge containment condition, and each vertex is in its own bag, the tree-decomposition is valid. The size of each bag is 1, and thus the width of the supertree-decomposition is calculated as:

$$\text{width} = 1 - 1 = 0.$$

Therefore, the supertree-width of an empty graph is 0. □

Theorem 3.7. *Let SHT $= (V, E)$ be a SuperHyperTree. The supertree-width of a SuperHyperTree is equal to the tree-width of the underlying tree. Specifically, the supertree-width of a SuperHyperTree SHT is 1 if SHT is a simple tree structure.*

Proof. A SuperHyperTree (SHT) is defined as a SuperHyperGraph where each hyperedge corresponds to a connected subtree of a host tree T.

To compute the supertree-width of SHT, we follow the definition of supertree-width, which is an extension of the tree-width for supergraphs. A supertree-decomposition (T, W) involves creating a tree T and assigning bags W_t to each node of T, such that:

- Every vertex of SHT is included in at least one bag.
- For each hyperedge (or simple edge) in SHT, there is a bag containing all the vertices of that hyperedge.
- For any vertex, the nodes containing that vertex form a connected subtree in T.

In a SuperHyperTree, the structure is inherently tree-like because it is based on a host tree T. The hyperedges of the SuperHyperTree are subtrees of T, and the tree decomposition reflects this structure.

Now, consider the tree-decomposition of T itself. Since T is a tree, the standard tree-width of T is 1, because every edge in T can be placed into a bag containing the two vertices it connects, and this satisfies the tree-decomposition conditions. Since every hyperedge in SHT corresponds to a subtree of T, the supertree-width will also be equal to 1 for a simple tree structure.

Thus, the supertree-width of SHT is 1, as it inherits the tree structure of T. □

Theorem 3.8. *Let SHT $= (V, E)$ be a SuperHyperTree. Then, SHT is a Supertree, and the supertree-width of SHT is 1.*

Proof. A Supertree is defined as a supergraph that is a tree. A SuperHyperTree (SHT) has the structure of a hypergraph, but its hyperedges correspond to connected subtrees of a host tree T, which ensures that SHT has a tree-like structure.

To verify that SHT is a Supertree, we need to check the following:

- The vertex set $V(\text{SHT})$ contains the vertex set of the host tree T, as each hyperedge of SHT corresponds to a subtree of T.
- The edge set $E(\text{SHT})$ contains edges that form subtrees of T, so $E(\text{SHT})$ inherits the tree-like structure of T, ensuring that SHT is acyclic.

Since SHT satisfies the conditions of being both a supergraph and a tree, it follows that SHT is a Supertree.

Furthermore, the supertree-width of SHT is determined by the tree-decomposition of the host tree T. As discussed in the first theorem, the tree-width of T is 1, and since SHT inherits the tree structure, the supertree-width of SHT is also 1.

Thus, we conclude that SHT is a Supertree, and its supertree-width is 1. □

Theorem 3.9. *Let G be a supergraph of itself. The supertree-width of G is less than or equal to the tree-width of G, i.e.,*

$$\textit{supertree-width}(G) \leq \textit{tree-width}(G).$$

Proof. By definition, the supertree-decomposition follows the same structure and constraints as the tree-decomposition for a graph G, meaning the supertree-width must obey the same conditions as tree-width. However, the concept of supertree-width applies to supergraphs, where additional vertices and edges could exist. Since we are considering the same graph G in both cases (i.e., G is a supergraph of itself), there are no additional vertices or edges to account for in the supertree-decomposition.

Therefore, the supertree-decomposition for G will yield at least the same set of decompositions as the tree-decomposition. Since the supertree-width is defined over the same set of decompositions, we have:

$$\text{supertree-width}(G) \leq \text{tree-width}(G),$$

because every valid tree-decomposition is also a valid supertree-decomposition. □

Theorem 3.10. *Let G and H be the same graph, i.e., $G = H$. Then, the supertree-width of H is equal to the tree-width of H:*

$$\textit{supertree-width}(H) = \textit{tree-width}(H).$$

Proof. Since $G = H$, the supertree-decomposition of G is equivalent to the tree-decomposition of H. Both decompositions must satisfy the same three conditions:

(1) The union of all bags covers the entire vertex set: $\bigcup_{t \in V(T)} W_t = V(H)$,
(2) For every edge $(u, v) \in E(H)$, there exists a bag W_t containing both u and v,
(3) For any three nodes $r, s, t \in T$, if s lies on the path between r and t, then $W_r \cap W_t \subseteq W_s$.

Since the definitions and requirements for both tree-decomposition and supertree-decomposition are identical when $G = H$, the supertree-width and tree-width are exactly the same. Both are based on decompositions of the same graph H. Thus, in this case, the inequality holds as an equality. □

3.3. *Definition of n-SuperHyperTree-width*

The n-SuperHyperTree is defined as follows. It is a tree on an n-SuperHyperGraph and serves as a generalization of a SuperHyperTree.

Definition 3.11. An n-SuperHyperTree (n-SHT) is a special type of n-SuperHyperGraph. Let $G_n = (V_n, E_n)$ be an n-SuperHyperGraph, where $V_n \subseteq \mathcal{P}_n(V)$ is the set of vertices and $E_n \subseteq \mathcal{P}_n(V)$ is the set of edges. The graph G_n is called an *n-SuperHyperTree* if the following conditions are satisfied:

- G_n is *connected*: there exists a path between any two vertices in G_n, where the path consists of edges and vertices of G_n;
- G_n contains no *n-cycles*: no subset of vertices and edges in G_n forms a closed loop of any neutrosophic, indeterminate, or n-SuperEdge structures, ensuring that G_n is acyclic;
- G_n has at least one n-SuperVertex: at least one vertex in G_n is a collection of vertices, where at least one is an $(n-1)$-SuperVertex and the others are r-SuperVertices, with $r \leq n-1$.

Definition 3.12. A subtree of an n-SuperHyperTree (n-SHT) is defined as follows: Let $H_n = (A_n, B_n)$ be an n-SuperHyperGraph. H_n is called a *subtree* if there exists a tree T with the same vertex set V_n, and for each edge $e \in E_n$, the edge induces a subtree in T. That is, for each e (which could be an n-SuperEdge or n-SuperHyperEdge), there exists a corresponding subtree structure in T.

Based on the above definition, the n-SuperHyperTree-width on an n-SuperHyperGraph is defined as follows.

Definition 3.13. Let n-SHT $= (V_n, E_n)$ be an n-SuperHyperGraph, where $V_n \subseteq \mathcal{P}_n(V)$ is the set of vertices and $E_n \subseteq \mathcal{P}_n(V)$ is the set of edges, including n-SuperEdges, n-SuperHyperEdges, and other higher-order structures.

A *n-SuperHyperTree decomposition* of n-SHT is a tuple $(T, \mathcal{B}, \mathcal{C})$ where:

- $T = (V_T, E_T)$ is a tree.
- $\mathcal{B} = \{B_t \mid t \in V_T\}$ is a family of subsets of V_n (called bags) associated with the nodes of the tree T, such that:
  (1) Vertex Coverage: For each vertex $v \in V_n$, the set of nodes $\{t \in V_T \mid v \in B_t\}$ forms a connected subtree of T.
  (2) SuperEdge Coverage: For each edge $e \in E_n$, there exists a node $t \in V_T$ such that the entire superedge e is contained within the corresponding bag B_t, i.e., $e \subseteq B_t$.
- $\mathcal{C} = \{C_t \mid t \in V_T\}$ is a family of subsets of E_n (called guards) associated with the nodes of the tree T, such that:
  (1) Guard Coverage: For each node $t \in V_T$, $B_t \subseteq \bigcup C_t$, where $\bigcup C_t$ denotes the union of all edges in C_t.
  (2) Subtree Condition: For each $t \in V_T$, the set $(\bigcup C_t) \cap \bigcup_{u \in V(T_t)} B_u \subseteq B_t$, where T_t denotes the subtree of T rooted at t.

The *width* of the n-SuperHyperTree decomposition $(T, \mathcal{B}, \mathcal{C})$ is defined as:

$$\text{width}(T, \mathcal{B}, \mathcal{C}) = \max_{t \in V_T} |C_t|,$$

where $|C_t|$ is the cardinality of the guard C_t.

The *n-SuperHyperTree-width* (n-SHT-width) of the n-SuperHyperGraph n-SHT, denoted by n-SHT-width(n-SHT), is the minimum width over all possible n-SuperHyperTree decompositions:

$$\text{n-SHT-width(n-SHT)} = \min_{(T, \mathcal{B}, \mathcal{C})} \text{width}(T, \mathcal{B}, \mathcal{C}).$$

- The n-SuperHyperTree-width is a measure of how close the n-SuperHyperGraph is to being a tree.
- For a classical graph, the n-SuperHyperTree-width coincides with the treewidth.

The width parameters restricted to Tree, Path, Cycle, and Star are defined as follows.

Definition 3.14. Let integer $n \geq 1$.

- n-SuperHyperPath-width measures how well an n-SuperHyperTree can be decomposed along a path-like structure.
- n-SuperHyperCycle-width measures how well an n-SuperHyperTree can be decomposed along a cycle-like structure.

- n-SuperHyperStar-width measures how well an n-SuperHyperTree can be decomposed along a star-like structure.

Next, the properties of SuperHyperTree-width are analyzed as follows. The theorem is stated below.

Theorem 3.15. *Let n-SHT* $= (V_n, E_n)$ *be an n-SuperHyperTree. The n-SuperHyperTree-width of n-SHT, denoted n-SHT-width(n-SHT), is 1.*

Proof. We will show that for any n-SuperHyperTree n-SHT $= (V_n, E_n)$, there exists an n-SuperHyperTree decomposition $(T, \mathcal{B}, \mathcal{C})$ with width 1, thereby proving that the minimal width over all decompositions is 1.

By definition, an n-SuperHyperTree is an acyclic and connected n-SuperHyperGraph. Specifically:

- n-SHT is connected: there exists a path between any two vertices in n-SHT.
- n-SHT contains no cycles: no subset of vertices and edges forms a cycle.

We construct $T = (V_T, E_T)$, where each node $t \in V_T$ corresponds to a vertex $v \in V_n$. The bags $\mathcal{B} = \{B_t \mid t \in V_T\}$ are defined such that each bag contains a single vertex, and the guards $\mathcal{C} = \{C_t \mid t \in V_T\}$ contain only the n-SuperEdge or n-SuperHyperEdge that connects the vertex.

The width of the decomposition is $\text{width}(T, \mathcal{B}, \mathcal{C}) = \max_{t \in V_T} |C_t|$. Since each guard contains only one edge, we have $|C_t| = 1$. Therefore, the width of the decomposition is 1.

Since we have constructed an n-SuperHyperTree decomposition with width 1, the n-SuperHyperTree-width of n-SHT is n-SHT-width(n-SHT) = 1. □

Theorem 3.16. *Let n-SHT* $= (V_n, E_n)$ *be an n-SuperHyperTree, where* $n \geq 1$. *The n-SuperHyperTree-width of n-SHT is less than or equal to the SuperHyperTree-width of n-SHT, that is,*

$$\textit{n-SHT-width}(\textit{n-SHT}) \leq \textit{SHT-width}(\textit{n-SHT}).$$

Proof. We will prove this theorem by distinguishing between two cases: $n = 1$ and $n > 1$.

Case 1: $n = 1$

When $n = 1$, the n-SuperHyperTree is a classical SuperHyperTree, so the widths are equivalent:

$$\text{1-SHT-width(1-SHT)} = \text{SHT-width(1-SHT)}.$$

Case 2: $n > 1$

When $n > 1$, n-SuperVertices and n-SuperEdges extend the structure. By constructing the

SuperHyperTree decomposition from the n-SuperHyperTree decomposition (by replacing n-SuperVertices and n-SuperEdges with their lower-order equivalents), we have:

$$|C_t| \leq |C_t^n| \quad \text{for all} \quad t \in V_T,$$

implying that:

$$\text{n-SHT-width(n-SHT)} \leq \text{SHT-width(n-SHT)}.$$

Thus, the inequality holds in both cases. □

4. Future tasks

We plan to investigate width parameters in Crisp, Fuzzy, Intuitionistic Fuzzy, Picture Fuzzy, and Neutrosophic n-SuperHyperGraphs (cf. [34, 39, 40, 82]). Additionally, we will explore the concepts of hypermatroids (cf. [64]) and superhypermatroids. Furthermore, we aim to examine Directed SuperHyperGraphs and other related structures.

Funding: This research received no external funding.
Acknowledgments: I humbly express my sincere gratitude to all those who have extended their invaluable support, enabling me to successfully accomplish this paper.
Conflicts of Interest: The authors declare no conflict of interest.
Disclaimer: The preprint of this paper can be found in [35, 38].

References

1. Isolde Adler. Directed tree-width examples. *Journal of Combinatorial Theory, Series B*, 97(5):718–725, 2007.
2. Isolde Adler, Tomáš Gavenčiak, and Tereza Klimošová. Hypertree-depth and minors in hypergraphs. *Theoretical Computer Science*, 463:84–95, 2012.
3. Isolde Adler, Georg Gottlob, and Martin Grohe. Hypertree width and related hypergraph invariants. *European Journal of Combinatorics*, 28(8):2167–2181, 2007.
4. Claude Berge. *Hypergraphs: combinatorics of finite sets*, volume 45. Elsevier, 1984.
5. Daniel Bienstock. Graph searching, path-width, tree-width and related problems (a survey). *Reliability of computer and communication networks*, 5:33–50, 1989.
6. Daniel Bienstock, Neil Robertson, Paul D Seymour, and Robin Thomas. Quickly excluding a forest. *J. Comb. Theory, Ser. B*, 52(2):274–283, 1991.
7. Hans L Bodlaender. A tourist guide through treewidth. *Acta cybernetica*, 11(1-2):1–21, 1993.
8. Hans L Bodlaender. Treewidth: Algorithmic techniques and results. In *International Symposium on Mathematical Foundations of Computer Science*, pages 19–36. Springer, 1997.
9. Hans L Bodlaender. A partial k-arboretum of graphs with bounded treewidth. *Theoretical computer science*, 209(1-2):1–45, 1998.
10. Hans L Bodlaender, Pål Grønås Drange, Markus S Dregi, Fedor V Fomin, Daniel Lokshtanov, and Michał Pilipczuk. A c^kn 5-approximation algorithm for treewidth. *SIAM Journal on Computing*, 45(2):317–378, 2016.

11. Hans L Bodlaender, Alexander Grigoriev, and Arie MCA Koster. Treewidth lower bounds with brambles. In *Algorithms–ESA 2005: 13th Annual European Symposium, Palma de Mallorca, Spain, October 3-6, 2005. Proceedings 13*, pages 391–402. Springer, 2005.
12. Hans L Bodlaender, Alexander Grigoriev, and Arie MCA Koster. Treewidth lower bounds with brambles. *Algorithmica*, 51(1):81–98, 2008.
13. Hans L Bodlaender and Arie MCA Koster. Treewidth computations i. upper bounds. *Information and Computation*, 208(3):259–275, 2010.
14. Andreas Brandstädt, Van Bang Le, and Jeremy P Spinrad. *Graph classes: a survey*. SIAM, 1999.
15. Alain Bretto. Hypergraph theory. *An introduction. Mathematical Engineering. Cham: Springer*, 1, 2013.
16. Derun Cai, Moxian Song, Chenxi Sun, Baofeng Zhang, Shenda Hong, and Hongyan Li. Hypergraph structure learning for hypergraph neural networks. In *IJCAI*, pages 1923–1929, 2022.
17. Florent Capelli, Arnaud Durand, and Stefan Mengel. Hypergraph acyclicity and propositional model counting. In *International Conference on Theory and Applications of Satisfiability Testing*, pages 399–414. Springer, 2014.
18. Qionghai Dai and Yue Gao. Mathematical foundations of hypergraph. In *Hypergraph Computation*, pages 19–40. Springer, 2023.
19. Narsingh Deo. *Graph theory with applications to engineering and computer science*. Courier Dover Publications, 2016.
20. Dariusz Dereniowski. From pathwidth to connected pathwidth. *SIAM Journal on Discrete Mathematics*, 26(4):1709–1732, 2012.
21. Reinhard Diestel. Ends and tangles. In *Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg*, volume 87, pages 223–244. Springer, 2017.
22. Reinhard Diestel. *Graph theory*. Springer (print edition); Reinhard Diestel (eBooks), 2024.
23. Reinhard Diestel, Fabian Hundertmark, and Sahar Lemanczyk. Profiles of separations: in graphs, matroids, and beyond. *Combinatorica*, 39:37–75, 2019.
24. Reinhard Diestel and Sang il Oum. Tangle-tree duality: In graphs, matroids and beyond. *Combinatorica*, 39:879 – 910, 2017.
25. Reinhard Diestel and Sang-il Oum. Tangle-tree duality in abstract separation systems. *Advances in Mathematics*, 377:107470, 2021.
26. Song Feng, Emily Heath, Brett Jefferson, Cliff Joslyn, Henry Kvinge, Hugh D Mitchell, Brenda Praggastis, Amie J Eisfeld, Amy C Sims, Larissa B Thackray, et al. Hypergraph models of biological networks to identify genes critical to pathogenic viral response. *BMC bioinformatics*, 22(1):287, 2021.
27. Johannes K Fichte, Markus Hecher, Neha Lodha, and Stefan Szeider. An smt approach to fractional hypertree width. In *Principles and Practice of Constraint Programming: 24th International Conference, CP 2018, Lille, France, August 27-31, 2018, Proceedings 24*, pages 109–127. Springer, 2018.
28. Wolfgang Fischl, Georg Gottlob, Davide Mario Longo, and Reinhard Pichler. Hyperbench: A benchmark and tool for hypergraphs and empirical findings. *Journal of Experimental Algorithmics (JEA)*, 26:1–40, 2021.
29. Takaaki Fujita. Reconsideration of tangle and ultrafilter using separation and partition. *arXiv preprint arXiv:2305.04306*, 2023.
30. Takaaki Fujita. Matroid, ideal, ultrafilter, tangle, and so on: Reconsideration of obstruction to linear decomposition. *International Journal of Mathematics Trends and Technology (IJMTT)*, 70:18–29, 2024.
31. Takaaki Fujita. Novel idea on edge-ultrafilter and edge-tangle. *Asian Research Journal of Mathematics*, 20(4):18–22, 2024.
32. Takaaki Fujita. Obstruction for hypertree width and superhypertree width. *preprint(researchgate)*, 2024.

33. Takaaki Fujita. Quasi-ultrafilter on the connectivity system: Its relationship to branch-decomposition. *International Journal of Mathematics Trends and Technology-IJMTT*, 70, 2024.
34. Takaaki Fujita. A review of the hierarchy of plithogenic, neutrosophic, and fuzzy graphs: Survey and applications. *ResearchGate(Preprint)*, 2024.
35. Takaaki Fujita. Short note of n-superhypertree-width. *preprint (researchgate)*, 2024.
36. Takaaki Fujita. Short note of rough tree-width from an information system. *ResearchGate(Preprint)*, 2024.
37. Takaaki Fujita. A short note of the relationship between loose tangles and filters. *International Journal of Mathematics Trends and Technology-IJMTT*, 70, 2024.
38. Takaaki Fujita. Superhypertree-width and neutrosophictree-width. *preprint(researchgate)*, 2024.
39. Takaaki Fujita. Survey of intersection graphs, fuzzy graphs and neutrosophic graphs. *ResearchGate*, July 2024.
40. Takaaki Fujita. Survey of planar and outerplanar graphs in fuzzy and neutrosophic graphs. *ResearchGate*, July 2024.
41. Takaaki Fujita. Tree-decomposition on fuzzy graph. *preprint(researchgate)*, 2024.
42. Takaaki Fujita. Ultrafilter in digraph: Directed tangle and directed ultrafilter. *Journal of Advances in Mathematics and Computer Science*, 39(3):37–42, 2024.
43. Takaaki Fujita. Various properties of various ultrafilters, various graph width parameters, and various connectivity systems. *arXiv preprint arXiv:2408.02299*, 2024.
44. Yue Gao, Yifan Feng, Shuyi Ji, and Rongrong Ji. Hgnn+: General hypergraph neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(3):3181–3199, 2022.
45. Jim Geelen, Bert Gerards, and Geoff Whittle. Obstructions to branch-decomposition of matroids. *Journal of Combinatorial Theory, Series B*, 96(4):560–570, 2006.
46. Lucantonio Ghionna, Luigi Granata, Gianluigi Greco, and Francesco Scarcello. Hypertree decompositions for query optimization. In *2007 IEEE 23rd International Conference on Data Engineering*, pages 36–45. IEEE, 2006.
47. Masoud Ghods, Zahra Rostami, and Florentin Smarandache. Introduction to neutrosophic restricted superhypergraphs and neutrosophic restricted superhypertrees and several of their properties. *Neutrosophic Sets and Systems*, 50:480–487, 2022.
48. Sathyanarayanan Gopalakrishnan, Supriya Sridharan, Soumya Ranjan Nayak, Janmenjoy Nayak, and Swaminathan Venkataraman. Central hubs prediction for bio networks by directed hypergraph-ga with validation to covid-19 ppi. *Pattern Recognition Letters*, 153:246–253, 2022.
49. Georg Gottlob, Gianluigi Greco, Nicola Leone, and Francesco Scarcello. Hypertree decompositions: Questions and answers. In *Proceedings of the 35th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*, pages 57–74, 2016.
50. Georg Gottlob, Gianluigi Greco, Francesco Scarcello, et al. Treewidth and hypertree width. *Tractability: Practical Approaches to Hard Problems*, 1:20, 2014.
51. Georg Gottlob, Nicola Leone, and Francesco Scarcello. Hypertree decompositions and tractable queries. In *Proceedings of the eighteenth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 21–32, 1999.
52. Georg Gottlob, Nicola Leone, and Francesco Scarcello. Hypertree decompositions: A survey. In *Mathematical Foundations of Computer Science 2001: 26th International Symposium, MFCS 2001 Mariánské Lázne, Czech Republic, August 27–31, 2001 Proceedings 26*, pages 37–57. Springer, 2001.
53. Georg Gottlob and Reinhard Pichler. Hypergraphs in model checking: Acyclicity and hypertree-width versus clique-width. *SIAM Journal on Computing*, 33(2):351–378, 2004.

54. Georg Gottlob, Reinhard Pichler, and Fang Wei. Tractable database design through bounded treewidth. In *Proceedings of the twenty-fifth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 124–133, 2006.
55. Jonathan L Gross, Jay Yellen, and Mark Anderson. *Graph theory and its applications*. Chapman and Hall/CRC, 2018.
56. Tom Gur, Noam Lifshitz, and Siqi Liu. Hypercontractivity on high dimensional expanders. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, pages 176–184, 2022.
57. Mohammad Hamidi, Florentin Smarandache, and Elham Davneshvar. Spectrum of superhypergraphs via flows. *Journal of Mathematics*, 2022(1):9158912, 2022.
58. Mohammad Hamidi, Florentin Smarandache, and Mohadeseh Taghinezhad. *Decision Making Based on Valued Fuzzy Superhypergraphs*. Infinite Study, 2023.
59. Mohammad Hamidi and Mohadeseh Taghinezhad. *Application of Superhypergraphs-Based Domination Number in Real World*. Infinite Study, 2023.
60. Asma Hamzeh and Ali Reza Ashrafi. The order supergraph of the power graph of a finite group. *Turkish Journal of Mathematics*, 42(4):1978–1989, 2018.
61. Frank Harary and Ronald C Read. Is the null-graph a pointless concept? In *Graphs and Combinatorics: Proceedings of the Capital Conference on Graph Theory and Combinatorics at the George Washington University June 18–22, 1973*, pages 37–44. Springer, 2006.
62. Daniel J Harvey and David R Wood. Parameters tied to treewidth. *Journal of Graph Theory*, 84(4):364–385, 2017.
63. Meike Hatzel, Roman Rabinovich, and Sebastian Wiederrecht. Cyclewidth and the grid theorem for perfect matching width of bipartite graphs. In *Graph-Theoretic Concepts in Computer Science: 45th International Workshop, WG 2019, Vall de Núria, Spain, June 19–21, 2019, Revised Papers 45*, pages 53–65. Springer, 2019.
64. Thorkell Helgason. Aspects of the theory of hypermatroids. In *Hypergraph Seminar: Ohio State University 1972*, pages 191–213. Springer, 2006.
65. Jing Huang and Jie Yang. Unignn: a unified framework for graph and hypergraph neural networks. *arXiv preprint arXiv:2105.00956*, 2021.
66. Remie Janssen, Mark Jones, Steven Kelk, Georgios Stamoulis, and Taoyang Wu. Treewidth of display graphs: bounds, brambles and applications. *arXiv preprint arXiv:1809.00907*, 2018.
67. Haim Kaplan and Ron Shamir. Pathwidth, bandwidth, and completion problems to proper interval graphs with small cliques. *SIAM Journal on Computing*, 25(3):540–561, 1996.
68. Ton Kloks. *Treewidth: computations and approximations*. Springer, 1994.
69. Xiaowei Liao, Yong Xu, and Haibin Ling. Hypergraph neural networks for hypergraph matching. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1266–1275, 2021.
70. Dániel Marx. Approximating fractional hypertree width. *ACM Transactions on Algorithms (TALG)*, 6(2):1–17, 2010.
71. Rogers Mathew, Ilan Newman, Yuri Rabinovich, and Deepak Rajendraprasad. Boundaries of hypertrees, and hamiltonian cycles in simplicial complexes. *arXiv preprint arXiv:1507.04471*, 2015.
72. Zoltán Miklós. *Understanding Tractable Decompositions for Constraint Satisfaction*. PhD thesis, University of Oxford, 2008.
73. Dan Olteanu and Jakub Závodnỳ. Factorised representations of query results: size bounds and readability. In *Proceedings of the 15th International Conference on Database Theory*, pages 285–298, 2012.
74. Sang-il Oum and Paul Seymour. Testing branch-width. *Journal of Combinatorial Theory, Series B*, 97(3):385–393, 2007.

75. Nicole Pearcy, Jonathan J Crofts, and Nadia Chuzhanova. Hypergraph models of metabolism. *International Journal of Biological, Veterinary, Agricultural and Food Engineering*, 8(8):752–756, 2014.
76. Dieter Rautenbach and Florian Werner. Induced subforests and superforests. *arXiv preprint arXiv:2403.14492*, 2024.
77. Neil Robertson and Paul D. Seymour. Graph minors. i. excluding a forest. *Journal of Combinatorial Theory, Series B*, 35(1):39–61, 1983.
78. Neil Robertson and Paul D. Seymour. Graph minors. iii. planar tree-width. *Journal of Combinatorial Theory, Series B*, 36(1):49–64, 1984.
79. Neil Robertson and Paul D. Seymour. Graph minors. x. obstructions to tree-decomposition. *Journal of Combinatorial Theory, Series B*, 52(2):153–190, 1991.
80. Robert Sasak. Comparing 17 graph parameters. Master's thesis, The University of Bergen, 2010.
81. Florentin Smarandache. n-superhypergraph and plithogenic n-superhypergraph. *Nidus Idearum*, 7:107–113, 2019.
82. Florentin Smarandache. *Extension of HyperGraph to n-SuperHyperGraph and to Plithogenic n-SuperHyperGraph, and Extension of HyperAlgebra to n-ary (Classical-/Neutro-/Anti-) HyperAlgebra*. Infinite Study, 2020.
83. Florentin Smarandache. *Introduction to SuperHyperAlgebra and Neutrosophic SuperHyperAlgebra*. Infinite Study, 2022.
84. Florentin Smarandache. *Introduction to the n-SuperHyperGraph-the most general form of graph today*. Infinite Study, 2022.
85. Florentin Smarandache. *The SuperHyperFunction and the Neutrosophic SuperHyperFunction (revisited again)*, volume 3. Infinite Study, 2022.
86. Florentin Smarandache. Decision making based on valued fuzzy superhypergraphs. 2023.
87. Florentin Smarandache. Foundation of the superhypersoft set and the fuzzy extension superhypersoft set: A new vision. *Neutrosophic Systems with Applications*, 11:48–51, 2023.
88. Florentin Smarandache. New types of topologies and neutrosophic topologies (improved version). *Neutrosophic Sets and Systems*, 57(1):14, 2023.
89. Florentin Smarandache. *SuperHyperFunction, SuperHyperStructure, Neutrosophic SuperHyperFunction and Neutrosophic SuperHyperStructure: Current understanding and future directions*. Infinite Study, 2023.
90. Florentin Smarandache and Nivetha Martin. *Plithogenic n-super hypergraph in novel multi-attribute decision making*. Infinite Study, 2020.
91. Florentin Smarandache, Memet Şahin, Derya Bakbak, Vakkas Uluçay, and Abdullah Kargın. *Neutrosophic SuperHyperAlgebra and New Types of Topologies*. Infinite Study, 2023.
92. Atsushi Takahashi, Shuichi Ueno, and Yoji Kajitani. Minimal acyclic forbidden minors for the family of graphs with bounded path-width. *Discrete Mathematics*, 127(1-3):293–304, 1994.
93. Susan Tu and Christopher Ré. Duncecap: Query plans using generalized hypertree decompositions. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, pages 2077–2078, 2015.
94. NICOLE WEIN. Cyclewidth: An analogy of treewidth.
95. Douglas Brent West et al. *Introduction to graph theory*, volume 2. Prentice hall Upper Saddle River, 2001.
96. Nikola Yolov. Minor-matching hypertree width. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 219–233. SIAM, 2018.

Received: Aug 15, 2024. Accepted: Nov 6, 2024