Parameterized Complexity of Computing Twin Width of a Graph

By

Mayank Gusain

MT23046

Under the supervision of

Dr. Diptapriyo Majumdar

Indraprastha Institute of Information Technology Delhi

June, 2025

Parameterized Complexity of Computing Twin Width of a Graph

By

Mayank Gusain

MT23046

Submitted in partial fulfillment of the requirements for the degree of

Master of Technology

To

Indraprastha Institute of Information Technology Delhi

June, 2025

Certificate

This is to certify that the thesis titled *Parameterized Complexity of Computing Twin Width of a Graph* being submitted by *Mayank Gusain* to the Indraprastha Institute of Information Technology Delhi, for the award of the degree of Master of Technology, is an original research work carried out by him under my supervision. In my opinion, the thesis has reached the standards fulfilling the requirements of the regulations relating to the degree.

The results contained in this thesis have not been submitted in part or full to any other university or institute for the award of any degree/diploma.

December, 2024

Dr. Diptapriyo Majumdar

Indraprastha Institute of Information Technology Delhi

New Delhi 110020

Everything can be taken from a man but the last of the human freedoms—to choose one's attitude in any given set of circumstances, to choose one's own way.

Viktor E. Frankl

The world will ask you who you are, and if you don't know, the world will tell you.

Carl Jung

Meaning is only found in the journey uphill, and never in the fleeting sense of satisfaction awaiting at the next peak.

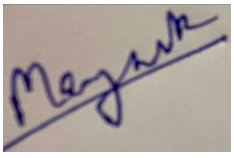
Unknown

Acknowledgements

I would like to take this opportunity to express my sincere thanks with sense of gratitude and indebtedness to my thesis advisor Dr. Diptapriyo Majumdar for his for kind cooperation with constant support and valuable suggestions, encouragement and guidance at the various stages of this thesis work. This thesis would not have reached anywhere without his appropriate suggestions. I must add that it has been a great experience studying at IIITD. Lastly, I acknowledge the role luck and privileges have played to get me going what otherwise would be dead ends.

I must express my gratitude to my parents and family members for being with me at every moment and providing continuous moral boosting and affection during thesis work.

December, 2024



Mayank Gusain
Indraprastha Institute of Information Technology Delhi
New Delhi 110020

Abstract

This thesis explores the design of efficient algorithms for determining the twin-width of a graph from a parameterized complexity perspective. Twin-width is a graph parameter that gives a decomposition of the graph in the lens of a contraction sequence. It is a parameter that measures the similarity of a graph to a cluster graph with smaller values of twin-width corresponding to more regular and structured graphs. However, determining the exact value of twin-width for a given graph is NP-complete. The thesis focuses on fixed-parameter tractable (FPT) algorithms, which can solve instances of the problem efficiently if the parameter remains small, or identifying suitable parameters that lead to meaningful hierarchy results. The thesis presents a detailed analysis of computing twin-width under various structural graph parameters and their relationship to twin-width, including twin cover number, vertex cover number, neighborhood diversity, edge deletion distance to cluster graph, and restricted modular partitions. The thesis makes several contributions to the field, including the development of fixed-parameter tractable (FPT) algorithms for computing twin-width parameterized by these graph parameters. These algorithms are complemented with reduction rules that simplify the input graph, making the algorithms more efficient in practice. The thesis also discusses the implications of these results for the parameterized complexity of other graph problems, and identifies several directions for future research. Overall, the thesis aims to better understand the practical implications of using twin-width in real-world scenarios while acknowledging its computational limitations.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation and Background

Graphs are capable of representing a diverse range of relationships and processes within physical, biological, social, and informational systems. One of the key challenges of graph theory is understanding the structural properties of graphs in order to develop efficient algorithms. In the last few decades, several graph parameters, e.g. treewidth [Cou92] cliquewidth [CMR00] treedepth [BS19] etc have been used as a crucial graph parameters to design parameterized algorithms (see Chapter 2 for detailed definitions). In the most recent times, a new graph parameter called *twin-width* was recently proposed in [Bon+20]. This graph parameter was partially motivated by a fundamental property that a cograph can be contracted into a single vertex by iterating over twins. Initially, there have been a series of results on the structural and algorithmic properties of graphs of bounded twin-width (see [Bon+20; Bon+22c; Bon+22a; Bon+22b; Bon+23; Bon+22d; Bon+24]).

Intuitively, *twin-width* of a graph G (denoted by $tww(G)$) gives a decomposition of the graph in the lens of *contraction sequence*. In particular, if an algorithm has to be designed in graphs of twin-width d, then a contraction sequence $G_n, \ldots, G_1$ has to be given such that G_n is the original input graph G and G_1 is the single vertex graph. The contraction sequence should also have width exactly d (detailed definition is available in Chapter 2). In the prior works, the following

results on FPT algorithms have been developed when the input graph has bounded twin-width. Each of the following algorithmic results have assumed that a

1. Given a graph G and a contraction sequence of width d, there exists an algorithm that can find an independent set of size at least k, a clique of size at least k, a dominating set of size at least k in $2^{O(kf(d))}$-time.

2. Given a graph G, and a contraction sequence of width d, there exists an algorithm that can solve SUBGRAPH ISOMORPHISM, INDUCED SUBGRAPH ISOMORPHISM, MAX-WEIGHTED INDEPENDENT SET with k vertices in $2^{O(k \log kf(d))}$-time.

3. Given an undirected graph $G = (V, E)$, it is NP-hard to determine whether the twin-width of a given graph is at most 4 or not [BBD22].

Observe that the last result makes the problem involving "computing the twin-width" is para-NP-hard when k becomes the parameter itself. Other problems like INDEPENDENT SET, SUBGRAPH ISOMORPHISM etc. are W[1]-hard in general graphs. But, given a graph of twin-width at most d along with a contraction sequence of minimum width d, these problems become FPT and some of which also admit singly-exponential time algorithm.

By definition, smaller values of $tww(G)$ correspond to more regular and structured graphs, making *twin-width* a useful concept for designing efficient algorithms. Furthermore, it is essential to note that determining the exact value of *twin-width* for a given graph is an NP-complete problem [Bon+20]. Consequently, design of algorithms to compute the *twin-width* of a graph from a parameterized complexity perspective is less explored. This includes exploring fixed-parameter tractable (FPT) algorithms, which can solve instances of the problem efficiently if the parameter remains small, or identifying suitable parameters that lead to meaningful hierarchy results. These approaches will help us better understand the practical implications of using twin-width in real-world scenarios, while acknowledging its computational limitations.

While a lot of works have been done on designing FPT algorithms and polynomial kernels for graphs of bounded twin-width, the problem involving "computing the twin-width of a graph" from the parameterized complexity perspective has been mostly unexplored. Balaban et al.

[BGR24] have initiated a systematic study of the parameterized complexity of computing twin-width of the graph when the structural parameters are considered into account. In particular, the authors in the paper developed the following two results.

- Given a graph G, determining whether $tww(G) = 2$ can be computed in time $2^{O(k \log k)} n^{O(1)}$ when k is the feedback edge number of the input graph.
- There is an algorithm that takes an n vertex graph as an input with feedback edge number k, runs in time $f(k)n^{O(1)}$ for a computable function $f : \mathbb{N} \to \mathbb{N}$ and outputs a contraction sequence of width at most $tww(G) + 1$.
- While Balaban et al. [BGR24] did not give concrete FPT algorithms when parameterized by the vertex cover number (vc) and the neighborhood diversity (nd) of the input graph, they provide informal explanations why the problem is fixed-parameter tractable when separately parameterizd by vc as well as when parameterized by nd.

1.2 Our Contributions

In this thesis, we delve relatively deeper into the parameterized complexity of computing the twin-width of the graph under structural parameters. In addition to the traditional structural parameters, e.g. vertex cover number, feedback vertex number, dissociation number, cluster vertex deletion number, feedback edge number, a recent work by Lafond and Luo [LL23] has considered FPT algorithms for domination problems with respect to "restricted modular partitions". A short summary of our results is the following.

1. Our first result considers the "twin-cover number" (tc) of the input graph as the parameter. We prove that if k is the twin-cover number of the input graph G, then we can compute an optimal contraction sequence of G in $2^{O(2^k k)} n^{O(1)}$-time.
2. Our next result considers the "edge-deletion distance to cluster graph".
3. Finally, we consider the parameterized complexity of computing twin-width using restricted modular partitions.

1.3 Organization of the Thesis

We introduce the necessary background in Chapter 2. Specifically, we review basic graph theory concepts. Additionally, we give an overview of parameterized complexity and kernelization, which are essential tools for designing efficient algorithms.

Chapter 3 focuses on the algorithmic aspects of twin-width. The first algorithm, described in Section 3.2, is parameterized by vertex cover number. The second algorithm, presented in Section 3.3, is parameterized by neighborhood diversity $\mathrm{nd}(G)$.

Finally, in Chapter 4, we conclude the thesis by summarizing our contributions and discussing potential future research directions. We believe that the study of twin-width offers many exciting opportunities for advancing the state-of-the-art in graph algorithms. Overall, we hope that this thesis serves as a valuable resource for researchers and practitioners interested in graph algorithms and combinatorial optimization.

Chapter 2

Preliminaries and Notations

In this chapter, we provide the necessary background and notations for understanding the concepts and algorithms presented in this thesis. We begin by reviewing the basic concepts of sets, numbers, and graphs. We then introduce the definition of *twin-width* and discuss its structural properties. Finally, we provide an overview of parameterized complexity and kernelization, which are important tools for developing efficient algorithms.

2.1 Sets, Numbers, and Graphs

In this section, we review some basic definition and notation related to sets, numbers, and graph theory. Throughout the thesis, we use n to denote the number of vertices and m to dentoe the number of edges of a graph. We denote the set of natural numbers by $\mathbb{N} = \{0, 1, 2, \ldots\}$. For any positive integer n, we use the notation $[n] = \{1, \ldots, n\}$. We use the standard asymptotic notation $O()$, $\Omega()$, $\Theta()$ to describe the time complexity of algorithms. A *(simple)graph* $G = (V, E)$ consists of a nonempty finite set $V = V(G)$ of elements called *vertices* and a possibly empty finite set $E = E(G)$ of unordered pairs of distinct vertices, called *edges*. An edge joining vertices u and v is written as uv or vu. If G is clear from the context, we omit it as a subscript. A graph G is *directed* if its edges are ordered pairs; otherwise, it is *undirected*. Directed graphs consist of a set of vertices and a set of ordered pairs of distinct vertices, referred to as *arcs* instead of edges.

A pair of vertices u and v are *true twins* if $N[u] = N[v]$. Informally speaking, a pair of vertices is true twins if they have the same closed neighborhood. Similarly, a pair of vertices are called *false twins* if $N(u) = N(v)$. Informally, two vertices are false twins of each other if they have the same open neighborhood. The following observation is true about true twins and false twins.

Observation 2.1.1. *True twins are adjacent to each other and false twins are nonadjacent to each other.*

We often use *twins* for short in place of true twins or false twins. Two vertices u and v in a graph G are twins if they have the same neighborhood of u, v. In other words, $N(u) \setminus v = N(v) \setminus u$, where N(u) denotes the neighborhood of u. A vertex subset $X \subseteq V$ is a *module* if all vertices of X have the same set of neighbors among the vertices not in X. Formally for $u, v \in X; N_G(u) \setminus X = N_G(v) \setminus X$.

Observation 2.1.2. *Modules respect containment relationships: if $M_1 \subseteq M_2 \subseteq V$, and M_2 is a module, then so is M_1. Also, singletons $\{v\}$ trivially qualify as modules.*

Definition 2.1.1 (Contraction Sequence of a graph)**.** *A contraction sequence, is a sequence of steps, beginning with the given graph, in which each step replaces a pair of vertices by a single vertex. This produces a sequence of graphs, with edges colored red and black; in the given graph, all edges are assumed to be black. When two vertices are replaced by a single vertex, the neighborhood of the new vertex is the union of the neighborhoods of the replaced vertices. In this new neighborhood, an edge that comes from black edges in the neighborhoods of both vertices remains black; all other edges are colored red.*

Definition 2.1.2 (Operation: Contract all twins exhaustively)**.** *The operation of "contracting all twins" is defined as follows:*

1. *Pick a pair of twins $u, v \in G$.*
2. *Contract u, v into a single vertex and replace the edges incident to either u or v with red/black edges incident to the new vertex according to the rules of contraction sequence of twin width.*
3. *Goto Step 1.*

2.2 Graph Parameters

Definition 2.2.1 (Vertex Cover)**.** *A vertex cover of a graph G is a subset of vertices $S \subseteq V(G)$ s.t. for every edge $uv \in E(G), u \in Sorv \in S$. The vertex cover number of G, denoted by $\tau(G)$, is the size of a minimum vertex cover in G.*

Definition 2.2.2 (Twin Cover)**.** *A* twin cover *of a graph $G = (V, E)$ is a subset of vertices $S \subseteq V(G)$ s.t. for every pair of vertices $u, v \in V(G) with uv \in E(G)$, either:*

1. *$u \in S$ or $v \in S$, or*
2. *u and v are true twins*

The twin cover number of G, denoted by $\tau_t(G)$, is the size of a minimum twin cover in G.

Definition 2.2.3 (Neighborhood Diversity)**.** *The* neighborhood diversity *of a graph G, denoted by $\eta(G)$, is the minimum number of distinct neighborhoods in G. The neighborhood diversity of a graph G is the smallest number k such that $V(G)$ has a partition into k modules and every module is either a clique or an independent set.*

These graph parameters will play a crucial role in understanding the structural properties of a graph with bounded *twin-width* and designing efficient algorithms.

2.3 Graphs with Restricted Modular Partitions

Given a graph class $\mathcal{G}$, the *$\mathcal{G}$-modular cardinality* of G is the minimum size of a vertex partitions into modules that each module induces a graph in class $\mathcal{G}$.

Examples of graph classes: Class of disjoint trees, cliques, independent sets

2.4 Twin-Width of a Graph

In this section, we introduce the definition of *twin-width* and discuss its structural properties

Definition 2.4.1 (Trigraph)**.** *A* trigraph $\mathcal{T} = (V, \delta, \lambda)$ *comprises a vertex set* V *accompanied by symmetric irreflexive binary relations* $\delta, \lambda \subseteq V \times V$*, termed* red *and* blue edges*, respectively.*

Definition 2.4.2 (Twin-Width)**.** *A contraction sequence is called a* d*-sequence for some non-negative integer* d *if, throughout the sequence, every vertex touches at most* d *red edges. The* twin-width *of a graph, denoted by* $tww(G)$ *is the smallest value of* d *for which it has a* d*-sequence.*

Examples of graphs with Bounded Twin Width

Lemma 2.4.0.1. *Path graphs with* $|V| <= 3$ *are co-graphs and with* $|V| > 3$ *have* $tww(G) = 1$*.*

Proof: Contraction sequence can be found by repeatedly merging the last two edges of the path. Because only the edge incident to the single merged vertex will be red, so this is a 1-sequence.

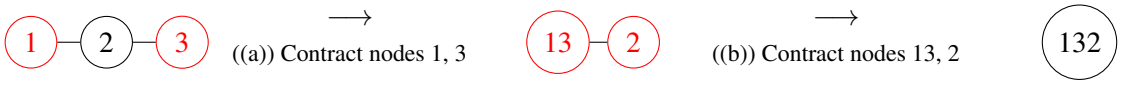


Figure 2.1: Example: 0-Sequence on P3

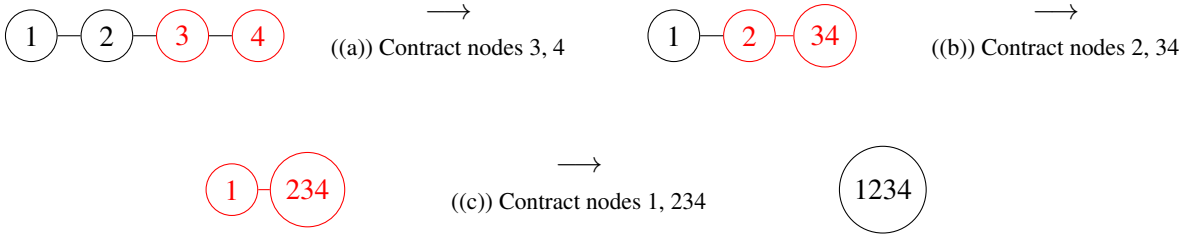


Figure 2.2: Example: 1-Sequence on P4

Lemma 2.4.0.2. *Trees have* $tww(G) <= 2$*.*

Proof: Contraction sequence can be found by choosing a root, then repeatedly merging two leaves that have the same parent if possible, o/w merging the deepest leaf into its parent. The only red edges connect leaves to their parents, and when there are two at the same parent they can be merged, keeping the red degree at most two.

Properties of bounded tww

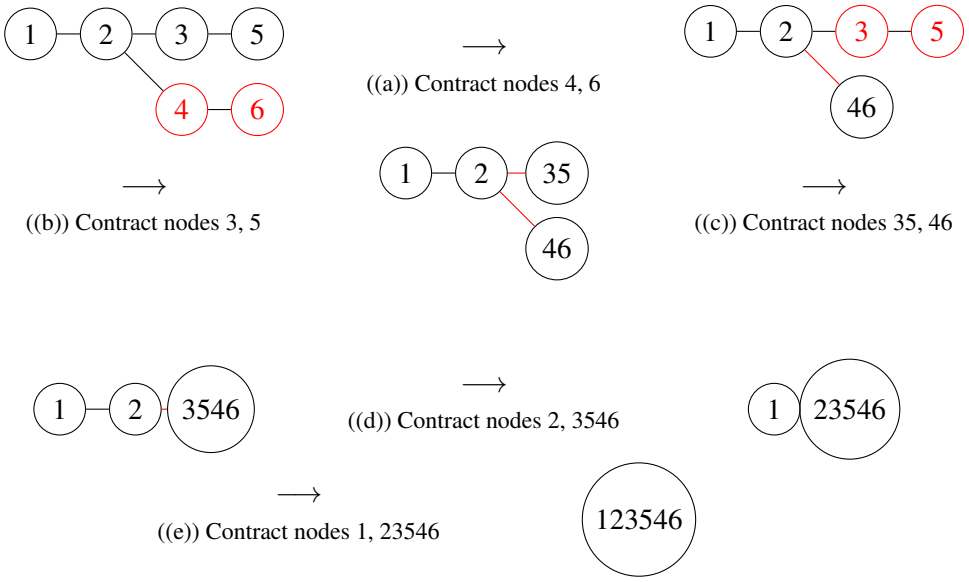


Figure 2.3: Example: 2-Sequence on Tree

In graphs of bounded twin-width, it is possible to perform a breadth-first search, on a graph with n vertices, in time $O(n \cdot \log n)$, even when the graph is dense and has more edges than this time bound. [Bon+21]

Definition 2.4.3 (Descendant)**.** *Node x is called descendant of u if:*

1. *x is u itself*
2. *x is formed by contraction of a node v with a descendant of u*

Definition 2.4.4 (Bag)**.** *Node x is the bag of u if it's a descendant of node u in a contraction sequence*

Definition 2.4.5 (Dangling Vertex)**.** *A dangling vertex is a vertex $u \notin S$ s.t. it's adjacent to a vertex $v \in S$ where $S \subseteq V$*

2.5 Parameterized Complexity and Kernelization

In this section, we provide an overview of parameterized complexity and kernelization, which are important tools for designing efficient algorithms. Parameterized complexity is a framework to analyze the computational complexity of problems by considering not only the input size, but

also additional parameters related to the problem instance. A *parameterized problem* associates an instance I with a numerical parameter $k \in \mathbb{N}$. It is denoted by $L \subseteq \Sigma^\star \times \mathbb{N}$ where Σ is a finite alphabet. The instance to a parameterized problem L is denoted by (I, k) where $I \in \Sigma^*$ and $k \in \mathbb{N}$.

Definition 2.5.1 (Fixed-Parameter Tractability)**.** *A parameterized problem $L \subseteq \Sigma^\star \times \mathbb{N}$ is* fixed-parameter tractable *(FPT) if solubility decisions admit solutions in runtime $f(k)n^c$ for some computable function $f : \mathbb{N} \to \mathbb{N}$ and constant c, independent of n and k.*

Definition 2.5.2 (Kernelization)**.** *A* kernelization algorithm *maps instances (I, k) of a parameterized problem L to equivalent instances (I', k') in polynomial time. Dimensionally constrained reductions mandate that $|I'| + k' \leq g(k)$ hold for some computable function $g : \mathbb{N} \to \mathbb{N}$, rendering compact representations of manageable scale. If g(k) is a polynomial function, then L is said to admit a* polynomial kernel*.*

In the context of *twin-width*, parameterized complexity and kernelization techniques have been successfully applied to design efficient algorithms for computing *twin-width* and solving various graph problems with bounded *twin-width*.

Chapter 3

Our Results on the Computing of Twin-Width

In this chapter, we focus on developing fixed-parameter tractable (FPT) algorithms for computing the twin-width of a graph. We examine different structural parameters of the graph. These are

- twin-cover number,
- neighborhood diversity,
- edge deletion distance to cluster graph, and
- $\mathcal{G}$-modular cardinality for some hereditary graph classes.

Given two vertices x and y of G, the *parity* of x and y is the pair of sets (X_1, X_2) such that X_1 is the set of common neighbors of x and y, excluding x and y themselves) and X_2 is the symmetric difference of neighbors between x and y, excluding x and y themselves. Formally, $X_1 = N(x) \cap N(y) \setminus \{x, y\}$ and $X_2 = (N(x) \setminus N(y)) \cup (N(y) \setminus N(x)) \setminus \{x, y\}$. We prove the following observation that we could not find out in the literature.

Observation 3.0.1. *Let u and v are twins. Then, for any $w \in V(G) \setminus \{u, v\}$, the neighbors of u and w before and after contracting the pair (u, v) have the same parity.*

Proof. Consider u and v that are twins in G and $w \in V(G) \setminus \{u, v\}$. It is natural to assume that after contraction of the pair (u, v), the vertex subset $\{u, v\}$ is identified as u. Consider the neighbors of u and w after contraction of the pair (u, v). What is crucial here is that u and v are twins. Hence, the neighborhood of u remains unchanged both before and after the contraction. Also, this does not change the neighbors of w before and after contraction. If x is a neighbor of u (other than w and u), but not a neighbor of w before contraction, x does not become a neighbor of w after the contraction. Similarly, if x is a neighbor of w (other than u and w) but not a neighbor of u before contraction, since neighborhood of u remains unchanged, x remains a neighbor of w and remains a non-neighbor of u. By using similar arguments, we can prove that the common neighbors of u and w (excluding u and w themselves) remain unchanged after contraction. Therefore, this observation is true. □

Using the above observation, we can prove the following lemma.

Lemma 3.0.0.1. *Let $u, v \in G$ be twins and G' denote the graph by contracting the pair $\{u, v\}$. Then $tww(G') = tww(G)$.*

Proof. The proof is based on the following argument. Given any contraction sequence S of G, there exists another contraction sequence S' in which twins u and v were contracted before all other contractions. Furthermore, the width of S' is at most the width of S. S' can be constructed as follows:

1. Contract all twins u, v in S'. Here, width is 0.
2. Next, follow the same contractions in S' as of S which don't involve bags of u or v. Here, width in S' = width in S with same red edges because rest of graph has same initial structure with same contractions in S' as in S
3. Finally, follow the contractions in S' as of S involving bags of u or v. Here, the width of the sequence S' is at most the width of the sequence S. The reason behind this is that the red edges created in the sequence S' is a (proper) subset of red edges created in the sequence S.

In particular, because of Observation 3.0.1, with possibly less different neighbors of w with u in S' as of u or v in S due to contractions in previous steps. □

3.1 Reduction Rules

1. Use Lemma 3.0.0.1 to contract all twins exhaustively in $O(n^2)$ time, where n is the number of vertices in the graph.

3.2 FPT Algorithm Parameterized by Twin Cover Number and Vertex Cover Number

In this section, we present an FPT algorithm to compute the *twin-width* of a graph where parameterized by the twin cover number. Since twin-cover number is at most the vertex cover number of a graph, our algorithm will automatically imply an FPT algorithm parameterized by the vertex cover number of the input graph. We refer to [Gan11] to highlight that every vertex cover can be transformed into a twin cover. Given a graph G and a twin cover $S \subseteq V(G)$ of size k, our goal is to determine the twin-width of G in time $f(k) \cdot n^c$, where n is the number of vertices in G and f is a computable function. Our algorithm leverages the structural properties of graphs with bounded vertex cover number in G.

3.2.1 Algorithm Description

Our first step is to invoke Lemma 3.0.0.1exhaustively. When we cannot contract twins of the graph, observe that all dangling vertices (see Definition 2.4.5) from S that were part of cliques in $G - S$ are reduced to single vertices in G'. Consequently, all vertices $v \notin S$ have different subset of vertices $u \in S$ as neighbors. Observe that the above procedure ensures us that for every $S' \subseteq S$, there is at most one vertex $u \in V(G) \setminus S$ such that $N_G(u) = S'$. Hence, the number of vertices in G is at most $k + 2^k$ when $|S| = k$.

In the next step, we invoke the exact algorithm by Balaban et al. [BGR24] to find the

twin-width. If the Graph has n vertices, then brute force time complexity to find the *twin-width* is $O(2^{n \cdot \log n})$.a The time complexity of our algorithm is $O(n^2 + 2^{k \cdot 2^k})$, where k is the size of twin cover and n is the number of vertices in the graph. This is because all vertices not in S are now adjacent to different subset of vertices in S. Therefore, the number of vertices not in S are bounded by the number of different subsets of S i.e., $2|S|$.

Using the fact that there are at most 2^k distinct subsets of S, we apply brute force to try all combinations of neighborhoods for vertices outside S. This approach guarantees us a running time of $O(n^2 + 2^{k \cdot 2^k})$. Hence, we have the following theorem.

Theorem 3.2.1. *If the twin-cover number of an n vertex graph is k, then there exists an algorithm that computes the twin-width of G in $O(n^2 + 2^{k \cdot 2^k})$-time.*

3.3 FPT Algorithm Parameterized by Neighborhood Diversity

In this section, we offer an FPT algorithm for computing the twin-width of a graph parameterized by the neighborhood diversity of the input graph. Given a graph $G = (V, E)$ with neighborhood diversity k, we aim to calculate the twin-width of G in time $(f(k) \cdot n^c)$, where n is the number of vertices in G and f is a computable function.

3.3.1 Algorithm Outline

Similarly to Section 3.2, we start by reducing the graph size by an initial preprocessing. As the first step, we invoke Lemma 3.0.0.1 exhaustively.

Observe that G has k modules each of which is either a clique or an independent set. When Lemma 3.0.0.1 cannot be applied anymore, every module is shrunk to a module with exactly one vertex. What is crucial here is that every module contains either true twins or false twins. After that we invoke the exact algorithm provided by Boloban et al. [BGR24]. This strategy results in a running time of $(O(n^2 + 2^{k \cdot \log k}))$. Also, as part of our algorithm, we have in polynomial-time constructed an equivalent instance into $O(k)$ vertices. Hence, we have the following theorem.

Theorem 3.3.1. TWIN-WIDTH *parameterized by the neighborhood diversity,* nd *admits a kernel with $O(k)$ vertices and admits an FPT algorithm with $(O(n^2 + 2^{k \cdot \log k}))$-time.*

3.4 FPT Algorithm Parameterized by Edge Deletion Distance to Cluster Graph

In this section, we design an FPT algorithm for computing the twin-width of a graph, when parameterized by the edge deletion distance to cluster graph. Given a graph G with a minimum of k edge deletions to form a cluster graph, we target obtaining the twin-width of G in time $f(k) \cdot n^c$, where n is the number of vertices in G and $f : \mathbb{N} \to \mathbb{N}$ is a computable function.

3.4.1 Algorithm Outline

To create an FPT algorithm, follow the outlined steps. In the first step, we invoke Lemma 3.0.0.1 exhaustively. When this preprocessing step is not applicable, let us understand the structure of the graph. Let F be the set of k edges the deletion of which makes the graph into a disjoint union of cliques and $V(F)$ be the endpoints of the edges of F. Since $|V(F)| \leq 2k$, after the deletion of the edges of F, the number of clusters (i.e. the connected components) in $G - F$ is at most $2k$. Let $C_1, C_2, \ldots, C_r$ be the connected components of $G - F$. As observed before, $r \leq 2k$. For every C_i, let A_i be the vertices whose neighborhood is contained in C_i and B_i be the vertices that has at least one neighbor outside C_i. Therefore, the following observation holds true.

Observation 3.4.1. *Considering the partition of $C_i = A_i \uplus B_i$ and $|F| \leq k$, the number of vertices in $\cup_{i \in [2k]} B_i$ is at most $2k$.*

Using the above observation, we are ready to prove our result.

Theorem 3.4.1. TWIN-WIDTH *parameterized by the cluster edge deletion number,* ced *admits a kernel with $4k$ vertices, and an FPT algorithm that runs in $O(n^2 + 2^{k \cdot \log k})$-time.*

Proof. First, we invoke Lemma 3.0.0.1 exhaustively. This preprocessing step contracts all twins from A_i for every $i \in [2k]$. Due to Observation 3.4.1, $|\cup_{i \in [2k]} B_i| \leq 2k$. Since after contraction,

there is at most one vertex in A_i for every $i \in [2k]$, it follows that after contraction, the graph has at most $4k$ vertices. This completes the proof that there is a kernel with $4k$ vertices. After this kernelization step is complete, invoke the algorithm by Balaban et al. [BGR24], we get an algorithm that runs in $O(n^2 + 2^{k \cdot \log k})$-time. □

3.5 FPT Algorithm Parameterized by Restricted Modular Partition

In this section, we design an FPT algorithm for computing the twin-width of a graph parameterized by restricted modular partition number. Given a graph G with a minimum of k vertex partitions into modules s.t. each module is in a specific graph class $\mathcal{G}$. In this thesis, we consider the case when $\mathcal{G}$ is hereditary. Our objective is to compute the twin-width of G in time $(f(k) \cdot n^c)$, where n is the number of vertices in G and $f : \mathbb{N} \to \mathbb{N}$ is a computable function.

3.5.1 When every module induces a disjoint union cliques or a component of three vertices

We first describe a generic algorithm, then illustrate its implications for different graph classes. Observe that there are k modules in the input graph G and for every module M, the subgraph $G[M] \in \mathcal{G}$. Here we consider two cases. For both the cases, we invoke Lemma 3.0.0.1 in the first step.

- **Case-1:** Every connected component of $G[M]$ is a clique. In the first step, we invoke Lemma 3.0.0.1 in the first step. Observe that if we carry out this operation exhaustively, then every connected component of $G[M]$ is converted into a single vertex. This converts our case into one where each of the k modules is a clique or an independent set. We invoke Theorem 3.3.1 to solve the problem. Observe that this provides a kernel with $3k$ vertices and an FPT algorithm with running time $2^{O(k \log k)} n^{O(1)}$-time.
- **Case-2:** Every connected component of $G[M]$ has at most 3 vertices. We invoke Lemma 3.0.0.1 exhaustively in the first step. Since every connected component has at most three vertices, it must be that either every connected component is a triangle or induces a P_3. For

a module M, if a connected component of $G[M]$ is an induced P_3, then observe that the corresponding component has two vertices x, y that are pairwise nonadjacent in M. But, observe that they are false twins of each other. They have a unique common neighbor in M, and that is the only neighbor of x and y in M. Additionally, outside M, their neighborhood is exactly the same. We contract the pair (x, y). Since x and y are twins, it follows from Lemma 3.0.0.1 this procedure does not increase the twin width of the modified graph. We call this operation as *contraction of false twins in the module*. If we cannot perform this operation of contracting false twins in the module, then we reach a situation where every connected component of $G[M]$ is a clique with either 2 vertices or three vertices. We continue the process as illustrated in the Case-1 above.

Based on the above two cases, we have the following result.

Theorem 3.5.1. *Given a graph G with a partition of $V(G)$ into k modules such that for every module M, the connected components of $G[M]$ is either a clique or has a path with three vertices, then twin-width of G can be computed in $2^{O(k \log k)} n^{O(1)}$-time.*

3.5.2 When every module is a forest

Now, we consider the case when every module M of G induces a forest.

Algorithm Description:

- As before, in the first step, we invoke Lemma 3.0.0.1.
- The subsequent steps depend on the structure of $G[M]$. Use algorithm in [Bon+22a] to see if $tww(G) \leq 1$:
  - If yes, we are done.
  - If no, we know from now on $tww(G) \geq 2$
- Use contractions in lemma 2.4.0.2 to reduce all trees in modules to single vertex. This is possible because for vertices under contraction u,v, the neighborhood outside the module

is same while inside doesn't affect the twin-width which can be proved by using the fact $tww(G) \geq 2$ and doing similar arguments as of Lemma 3.0.0.1.

After that, use the brute force algorithm to compute the twin-width in $O(2^{k \cdot \log k})$-time.

3.6 Future Directions

Despite proposing FPT algorithms for computing the twin-width parameterized by the twin cover number, neighborhood diversity, dissociation set number, and edge deletion cluster graph number, there are several captivating directions for future research. First, strive to develop even more efficient algorithms with superior time complexities. Second, expand these algorithms to accommodate larger graph classes or incorporate additional parameters, increasing their applicability. Finally, continue working on practical algorithms for computing twin-width on massive scales.

Chapter 4

Conclusions and Future Research

In this thesis, we conducted an extensive examination of twin-width and its ramifications for graph algorithms. Starting with foundational knowledge and nomenclature, we covered essential graph theory, parameterized complexity, and kernelization techniques. Following this, we dove deep into twin-width's algorithmic facets, presenting fixed-parameter tractable (FPT) algorithms for calculating twin-width based on twin cover number and neighborhood diversity.

Our work demonstrates twin-width's potential as a versatile instrument for addressing graph problems. Capitalizing on structural characteristics of graphs with limited twin-width enabled us to engineer efficient algorithms and kernelization techniques. These outcomes bolster the expanding corpus of research dedicated to twin-width and its multifaceted uses.

As for prospective avenues of inquiry, multiple intriguing prospects await discovery:

1. Extending our FPT algorithms to tackle broader graph classes or improving existing algorithms specifically tailored for particular graph families presents an engaging pursuit.
2. Investigating twin-width's correlation with other graph parameters promises fruitful outcomes. Deciphering their interactions may reveal innovative algorithms and theoretical insights.
3. Developing practical algorithms designed for determining twin-width on expansive graphs emerges as a vital objective. Real-world implementations demand reliable performance

metrics, necessitating robust scaling capabilities.

Overall, twin-width displays immense potential for propelling forward-looking developments in graph algorithms. With this thesis paving the way, we anticipate ongoing research efforts and spirited explorations concerning twin-width and its manifold applications.

Bibliography

[Cou92] Bruno Courcelle. “The monadic second-order logic of graphs III: tree-decompositions, minor and complexity issues”. In: *RAIRO Theor. Informatics Appl.* 26 (1992), pp. 257–286.

[CMR00] Bruno Courcelle, Johann A. Makowsky, and Udi Rotics. “Linear Time Solvable Optimization Problems on Graphs of Bounded Clique-Width”. In: *Theory Comput. Syst.* 33.2 (2000), pp. 125–150.

[Gan11] Robert Ganian. “Twin-Cover: Beyond Vertex Cover in Parameterized Algorithmics”. In: *Parameterized and Exact Computation - 6th International Symposium, IPEC 2011, Saarbrücken, Germany, September 6-8, 2011. Revised Selected Papers*. Ed. by Dániel Marx and Peter Rossmanith. Vol. 7112. Lecture Notes in Computer Science. Springer, 2011, pp. 259–271.

[BS19] Marin Bougeret and Ignasi Sau. “How Much Does a Treedepth Modulator Help to Obtain Polynomial Kernels Beyond Sparse Graphs?” In: *Algorithmica* 81.10 (2019), pp. 4043–4068.

[Bon+20] Édouard Bonnet et al. “Twin-width I: tractable FO model checking”. In: *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, Durham, NC, USA, November 16-19, 2020*. Ed. by Sandy Irani. IEEE, 2020, pp. 601–612.

[Bon+21] Édouard Bonnet et al. “Twin-width III: Max Independent Set, Min Dominating Set, and Coloring”. In: *48th International Colloquium on Automata, Languages, and Programming, ICALP 2021, July 12-16, 2021, Glasgow, Scotland (Virtual Conference)*. Ed. by Nikhil Bansal, Emanuela Merelli, and James Worrell. Vol. 198. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021, 35:1–35:20.

[BBD22] Pierre Bergé, Édouard Bonnet, and Hugues Déprés. “Deciding Twin-Width at Most 4 Is NP-Complete”. In: *49th International Colloquium on Automata, Languages, and Programming,*

ICALP 2022, July 4-8, 2022, Paris, France. Ed. by Mikolaj Bojanczyk, Emanuela Merelli, and David P. Woodruff. Vol. 229. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022, 18:1–18:20.

[Bon+22a] Édouard Bonnet et al. "Twin-width and Polynomial Kernels". In: *Algorithmica* 84.11 (2022), pp. 3300–3337.

[Bon+22b] Édouard Bonnet et al. "Twin-width IV: ordered graphs and matrices". In: *STOC '22: 54th Annual ACM SIGACT Symposium on Theory of Computing, Rome, Italy, June 20 - 24, 2022*. Ed. by Stefano Leonardi and Anupam Gupta. ACM, 2022, pp. 924–937.

[Bon+22c] Édouard Bonnet et al. "Twin-width VI: the lens of contraction sequences". In: *Proceedings of the 2022 ACM-SIAM Symposium on Discrete Algorithms, SODA 2022, Virtual Conference / Alexandria, VA, USA, January 9 - 12, 2022*. Ed. by Joseph (Seffi) Naor and Niv Buchbinder. SIAM, 2022, pp. 1036–1056.

[Bon+22d] Édouard Bonnet et al. "Twin-width VII: groups". In: *CoRR* abs/2204.12330 (2022).

[Bon+23] Édouard Bonnet et al. "Twin-Width V: Linear Minors, Modular Counting, and Matrix Multiplication". In: *40th International Symposium on Theoretical Aspects of Computer Science, STACS 2023, March 7-9, 2023, Hamburg, Germany*. Ed. by Petra Berenbrink et al. Vol. 254. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023, 15:1–15:16.

[LL23] Manuel Lafond and Weidong Luo. "Parameterized Complexity of Domination Problems Using Restricted Modular Partitions". In: *48th International Symposium on Mathematical Foundations of Computer Science, MFCS 2023, August 28 to September 1, 2023, Bordeaux, France*. Ed. by Jérôme Leroux, Sylvain Lombardy, and David Peleg. Vol. 272. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023, 61:1–61:14.

[BGR24] Jakub Balabán, Robert Ganian, and Mathis Rocton. "Computing Twin-Width Parameterized by the Feedback Edge Number". In: *41st International Symposium on Theoretical Aspects of Computer Science, STACS 2024, March 12-14, 2024, Clermont-Ferrand, France*. Ed. by Olaf Beyersdorff et al. Vol. 289. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024, 7:1–7:19.

[Bon+24] Édouard Bonnet et al. "Neighbourhood complexity of graphs of bounded twin-width". In: *Eur. J. Comb.* 115 (2024), p. 103772.