TU WIEN Informatics

Shining Light on Periodic Dominating Sets in Bounded-Treewidth Graphs

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Logic and Computation

eingereicht von

Jakob Greilhuber, BSc
Matrikelnummer 11902063

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Associate Prof. Robert Ganian, PhD
Mitwirkung: Assistant Prof. Dr. rer. nat. Philip Wellnitz
Philipp Schepper, M.Sc.

Wien, 30. August 2024 [redacted] [redacted]

Jakob Greilhuber Robert Ganian

Technische Universität Wien
A-1040 Wien ▪ Karlsplatz 13 ▪ Tel. +43-1-58801-0 ▪ www.tuwien.at

TU WIEN Informatics

Shining Light on Periodic Dominating Sets in Bounded-Treewidth Graphs

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieur

in

Logic and Computation

by

Jakob Greilhuber, BSc
Registration Number 11902063

to the Faculty of Informatics

at the TU Wien

Advisor: Associate Prof. Robert Ganian, PhD
Assistance: Assistant Prof. Dr. rer. nat. Philip Wellnitz
Philipp Schepper, M.Sc.

Vienna, August 30, 2024

[redacted] Jakob Greilhuber

[redacted] Robert Ganian

Erklärung zur Verfassung der Arbeit

Jakob Greilhuber, BSc

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.
Ich erkläre weiters, dass ich mich generativer KI-Tools lediglich als Hilfsmittel bedient habe und in der vorliegenden Arbeit mein gestalterischer Einfluss überwiegt. Im Anhang „Übersicht verwendeter Hilfsmittel“ habe ich alle generativen KI-Tools gelistet, die verwendet wurden, und angegeben, wo und wie sie verwendet wurden. Für Textpassagen, die ohne substantielle Änderungen übernommen wurden, haben ich jeweils die von mir formulierten Eingaben (Prompts) und die verwendete IT- Anwendung mit ihrem Produktnamen und Versionsnummer/Datum angegeben.

Wien, 30. August 2024

[redacted]

Jakob Greilhuber

Acknowledgements

My deepest gratitude goes out to everyone who has helped in the creation of this thesis. In particular, I want to thank Philip and Philipp, who took me under their wing at the Max Planck Institute of Informatics and the months following the internship, and Robert Ganian for making the thesis possible here at TU Wien. I also want to thank Evi for always supporting and lifting me up, and my cat Kiwi who is always by my side.

Kurzfassung

Sei ein Graph G gegeben, dann ist eine Menge $S \subseteq V(G)$ eine (σ, ρ)-Menge genau dann, wenn

- für alle $v \in S$ $|N(v) \cap S| \in \sigma$ gilt, und
- für alle $v \notin S$ $|N(v) \cap S| \in \rho$ gilt.

Das Problem zu entscheiden, ob für einen Graph eine (σ, ρ)-Menge S existiert, ist das Entscheidungsproblem von (σ, ρ)-GenDomSet. Natürlicherweise gibt es auch das entsprechende Minimierungs und Maximierungsproblem, in dem S möglichts klein bzw. groß sein soll.

Das (σ, ρ)-GenDomSet Framework verallgemeinert viele klassische Graphenprobleme. Zum Beispiel erhalten wir Dominating Set, wenn $\sigma = \mathbb{N}$ und $\rho = \mathbb{N} \setminus \{0\}$ ist, und Independent Set, wenn wir $\sigma = \{0\}, \rho = \mathbb{N}$ setzen.

In dieser Diplomarbeit werden die Resultate von Focke et al. [SODA 2023] für den Fall, dass σ und ρ periodische Mengen, genauer gesagt Restklassen modulo m sind, erweitert.

Konkret wird gezeigt, dass wenn $0 \notin \rho$ und $\mathrm{m} \geq 3$ gilt, für jedes $\varepsilon > 0$ das Entscheidungsproblem nicht in der Laufzeit $(\mathrm{m} - \varepsilon)^{\mathsf{tw}} \cdot |G|^{O(1)}$ gelöst werden kann, selbst wenn der Eingabegraph G zusammen mit einer Baumzerlegung der Weite tw gegeben ist, außer die *Strong Exponential Time Hypothesis* ist falsch. Wenn $\mathrm{m} = 2$ gilt, kann das Entscheidungsproblem in Polynomialzeit gelöst werden. Für diesen Fall erweitern wir die Resultate für das Minimierungsproblem unter der Annahme $0 \notin \rho$, sowie das Maximierungsproblem für alle Möglichkeiten von σ, ρ. Alle präsentierten unteren Schranken sind scharf, da man das Entscheidungsproblem und beide Optimierungsprobleme in der Zeit $\mathrm{m}^{\mathsf{tw}} \cdot |G|^{O(1)}$ lösen kann.

Diese Arbeit stellt die erste dar, die scharfe konditionelle untere Schranken für (σ, ρ)-GenDomSet mit der Baumweite als Parameter gibt, wenn weder σ, ρ noch $\mathbb{N} \setminus \sigma$, $\mathbb{N} \setminus \rho$ endlich sind und repräsentiert somit einen ersten Schritt in diese Richtung.

Große Teile der Arbeit wurden während eines Sommerpraktikums beim Max-Planck-Institut für Informatik unter Betreuung von Philip Wellnitz und Philipp Schepper durchgeführt.

Abstract

Given a graph G, a set $S \subseteq V(G)$ is a (σ, ρ)-set of G if and only if

- for all $v \in S$ we have $|N(v) \cap S| \in \sigma$, and
- for all $v \notin S$ we have $|N(v) \cap S| \in \rho$.

The problem of deciding whether a graph has a (σ, ρ)-set is the decision problem of (σ, ρ)-GenDomSet. Naturally, one can also consider the minimization and maximization problems, in which S is supposed to be as small, respectively as large as possible.

The framework of (σ, ρ)-GenDomSet captures numerous classical graph problems. For instance, the problem corresponds to Dominating Set when $\sigma = \mathbb{N}$ and $\rho = \mathbb{N} \setminus \{0\}$, and we obtain Independent Set by setting $\sigma = \{0\}, \rho = \mathbb{N}$.

In this thesis, the work by Focke et al. [SODA 2023] is extended for the case where σ and ρ are periodic sets, specifically residue classes modulo m.

We show that when $0 \notin \rho$ and $\mathrm{m} \geq 3$, for any $\varepsilon > 0$, the decision problem cannot be solved in time $(\mathrm{m} - \varepsilon)^{\mathsf{tw}} \cdot |G|^{O(1)}$, even when the input graph is provided together with a tree decomposition of width tw, unless the *Strong Exponential Time Hypothesis* is false. If $\mathrm{m} = 2$, the decision problem can be solved in polynomial time. For this case, we extend the lower bound to the minimization problem assuming $0 \notin \rho$, and the maximization problem in all settings. In all cases, the obtained lower bounds are tight, as one can solve the decision problem and both optimization problems in time $\mathrm{m}^{\mathsf{tw}} \cdot |G|^{O(1)}$.

The work in this thesis represents the first work that provides tight conditional lower bounds for (σ, ρ)-GenDomSet parameterized by treewidth when σ, ρ are neither finite nor cofinite, making it a first step into this domain.

Large parts of this work were done during a summer internship at the Max Planck Institute of informatics supervised by Philip Wellnitz and Philipp Schepper.

Contents

CHAPTER 1

Introduction

In the *Lights Out* game (see [FY13] for a literature survey), the player is presented with 25 lightbulbs arranged on a 5×5 grid. Each lightbulb has an associated switch, and a lightbulb can be either on or off. Using the switch of a bulb does not only change the state of the lightbulb, but also the state of all non-diagonal neighboring lightbulbs. Given some starting configuration of lightbulbs being on or off, the goal is to turn off all lights.

A simple observation that was, for example, presented in [Sut89] and [AF98], is that using the same switch twice has the same effect as not using it at all. Moreover, the order in which the switches are pressed does not matter. Thus, a solution can be described by a *set* of switches (or equivalently, lamps those switches belong to) that need to be used in order to turn off all lights.

We can now formulate the problem as a *graph-theoretic* problem, in which the solution is a subset of the vertices of the graph. This formulation also immediately yields a reasonable generalization of the original Lights Out game, in which the size of the board game was fixed. For this purpose, we model the game on an undirected graph G. Each vertex of G corresponds to a lamp, and the switch of the lamp/vertex changes the state of the vertex itself, and of the neighboring vertices in the graph. We are furthermore provided a $|V(G)|$-dimensional binary vector $\vec{c}$, which describes the initial state of the vertices. Our task is finding a set $S \subseteq V(G)$, such that

$$|N[v] \cap S| \equiv_2 \vec{c}[v] \quad \text{for all } v \in V(G).$$

That is, if a vertex is initially on, it should have an *odd* number of vertices in its closed neighborhood. If a vertex is initially turned off, it should have an *even* number of vertices in its closed neighborhood. This Lights Out problem, and variants thereof, have been studied by numerous researchers in the past [Sut88, Sut89, GKTZ95, GKT97, AF98, HKT00b, DW01, GK07, GH08, FY13, BBH21].

A natural emerging questions is how one can efficiently solve the described problem. It turns out that this is actually quite easy. Indeed, we must only solve the system of equations $(A+I)\cdot\vec{x}=\vec{c}$ over the field $\mathbb{F}_2$ for the binary vector $\vec{x}$, where A is the adjacency matrix of G (see e.g. [Sut89, GKTZ95, GKT97, AF98, HKT00b]). This can be done in polynomial time, for example by using Gaussian Elimination. However, a player may not simply want *any* solution to the game, but a solution in which *as few switches as possible* must be triggered. Hence, a natural extension of the problem is to not ask for any solution, but for a solution of small size. As is often the case, the additional size requirement makes the problem NP-hard [Sut88, HKT00b, CGK01].

Naturally, one can also formulate the problem in which open neighborhoods instead of closed neighborhoods are used. This problem has also been explored by researchers in the past [Sut88, HKT00b, GK07, GH08]. Similarly to the case in which the switches are reflexive, the minimization variant of the problem is NP-hard, and the decision problem can be solved in polynomial-time [Sut88, HKT00b].

In this thesis, a generalization of the Lights Out problem, based on the framework by Telle [Tel94, TP93] is explored. In the framework, the problem is defined relative to two non-empty sets of non-negative integers σ and ρ. When given an input graph G, a set $S\subseteq V(G)$ is a (σ,ρ)-set if for all $v\in S$, we have $|N(v)\cap S|\in\sigma$, and for all $v\in V(G)\setminus S$, we have $|N(v)\cap S|\in\rho$. Hence, σ described the allowed number of selected neighbors of selected vertices, whereas ρ described the allowed number of selected neighbors for unselected vertices.

The problem of deciding whether any (σ,ρ)-set exists for G is referred to as the *decision* problem, and the problem of deciding whether a (σ,ρ)-set that has at most (at least) a certain size exists is the *minimization (maximization)* problem. When setting $\sigma=\{0,1,\dots\}$ and $\rho=\{1,2,\dots\}$, the classical Dominating Set problem appears, by choosing $\sigma=\{0\}$ and $\rho=\{0,1,\dots\}$ we obtain Independent Set. These were just a few examples of well-known problems that are covered by the framework, refer to [Tel94] for a more extensive list. As the framework generalizes Dominating Set, we refer to the problem as (σ,ρ)-GenDomSet. This problem and special cases of it have been studied both in the classical and, due to the NP-hardness for many choices of σ,ρ, also in the parameterized setting, by many researchers [TP93, Tel94, vBR09, Cha10, ABR+10, BvvV10, GKS12, JKST19, JRS19, MFMP20, van20, van21, FMI+23a, FMI+23b, FMI+23c].

Within the framework of (σ,ρ)-GenDomSet, we do not consider closed, but open neighborhoods, which must be taken into account when modeling Lights Out in it. We will assume for now that the starting configuration of Lights Out is the configuration in which all lights are on, and note that this restriction can easily be circumvented later. As a selected vertex requires an odd number of selected neighbors in the closed neighborhood, and it is selected itself, it requires an even number of selected neighbors in its open neighborhood. An unselected vertex also requires an odd number of selected neighbors in the closed neighborhood, which means that it needs an odd number of selected neighbors in its open neighborhood. This problem is called Refl-AllOff by us, and it is obtained by setting $\sigma = \{0, 2, \dots\}$ and $\rho = \{1, 3, \dots\}$. Similarly, by choosing $\sigma = \rho = \{1, 3, \dots\}$, we can also express the problem variant denoted as AllOff that uses open instead of closed neighborhoods for the switches.

It is noteworthy that, when the AllOff problem is formulated as a graph in the framework, both σ and ρ are residue classes modulo 2. This gives rise to the variant of the problem studied in this thesis. A set $\tau \subseteq \mathbb{N}$ is called *periodic* if it is a residue class modulo m, for some positive integer m, that is, $\tau = \{n \in \mathbb{N} \mid n \equiv_{\mathrm{m}} k\}$ for some integer k with $0 \leq k < \mathrm{m}$. Moreover, we call m the period of the set in this case. Observe that the even and odd integers are both periodic sets with period 2.

Given two periodic sets σ and ρ with the same period m and an input graph G, we study both the decision and minimization variants of (σ, ρ)-GenDomSet. The sets σ and ρ are treated as fixed constant sets in this setting. As both the decision and the minimization problems are NP-hard for most choices of σ and ρ, we explore the question with regard to the predominant parameter *treewidth*. Intuitively, the treewidth is a measure of how tree-like the input graph is. Trees, clearly the most tree-like graphs, have a treewidth of one, whereas cliques, the graphs that appear to be the furthest away from trees, have a treewidth linear in their number of vertices. We refer to [CFK+15, Chapter 7] for more information about the concept of treewidth.

The ultimate goal for (σ, ρ)-GenDomSet parameterized by treewidth is obtaining an algorithm with a running time bounded by $c^{\mathsf{tw}} \cdot |V(G)|^{O(1)}$, where tw is the treewidth of the input graph G, and c a constant that is as small as possible. Algorithms with a running time of this type exist for the case where σ and ρ are finite or cofinite sets [TP93, vBR09, Cha10, van20, van21, FMI+23a, FMI+23b]. A natural question that appears is how small the constant c can get. To answer questions of this type, lower bounds based on the *Strong Exponential Time Hypothesis* (SETH) [IP01, CIP09] can be used. The SETH is the assumption that, for any $\varepsilon > 0$, there is a k such that n-variable, m clause k-SAT[1] cannot be solved in time $(2 - \varepsilon)^n \cdot (n + m)^{O(1)}$. For the case where σ and ρ are finite, Focke et al. [FMI+23a, FMI+23c] consider the decision problem, and provide a lower bound conditioned on SETH matching the running time of the best known algorithms for this case.

For the considered problem of (σ, ρ)-GenDomSet with sets having the same period no explicit running times for any problem variant were known in general. Concretely, while it is known that the decision problem admits a running time of the form $c^{\mathsf{tw}} \cdot |G|^{O(1)}$ for some constant c [Cha10], no concrete values for c are known in most cases. The only exception to this is for the cases where $\mathrm{m} = 2$, for which Gassner and Hatzl [GH08] provide an algorithm for all problem variants that runs in time $2^{3\mathsf{tw}} \cdot |V(G)|$. Hence, before our work, there either existed no concrete information on what the integer c should be, or only an algorithm which is presumably far from optimal.

[1]The problem k-SAT is the classical SAT problem in which clauses are given in conjunctive normal form, and each clause contains at most k literals.

When carefully analyzing the algorithm by Focke et al. [FMI+23b], it becomes clear that many of their ideas should be applicable to this setting as well. Their results suggest that a running time of $\mathrm{m}^{\mathsf{tw}} \cdot |V(G)|^{O(1)}$ could be achievable for all problem variants, which would represent a drastic reduction in the exponential dependency on the treewidth compared to the algorithm by Gassner and Hatzl for the case $\mathrm{m} = 2$.

The question on what the optimal constant c is was investigated by me for this master thesis, and in a summer internship at the Max Planck Institute of Informatics, where I was supervised by Philip Wellnitz and Philipp Schepper. The work on this topic is also presented in a preprint [GSW24]. We showed that the algorithm by Focke et al. [FMI+23b] can indeed be adapted to the setting of periodic sets to solve the decision and optimization problems in time $\mathrm{m}^{\mathsf{tw}} \cdot |V(G)|^{O(1)}$. Note that this running time is exponentially smaller in the treewidth than a naive algorithm would be.

In this thesis, the focus is on the lower bounds, which are, except for lower bounds for the maximization problems with period 2, also presented in [GSW24]. As [GSW24] also includes a matching upper bound, the main results presented in this thesis are a subset of those presented in [GSW24]. Note, however, that the construction given in Chapter 3 differs slightly from the one given in [GSW24, Section 5]. The reason for this is that the aim of the thesis is to be more self-contained, whereas the approach in [GSW24] utilizes some theorems in a black-box fashion. This also allows for the simplification of some parts of the construction. The content in Section 5.3 is not presented in [GSW24].

Concretely, we provide a matching lower bound conditioned on the SETH, showing that the running time of the algorithm is presumably optimal, even for the decision problem, as long as $\mathrm{m} \geq 3$ and $0 \notin \rho$. To exclude these cases, we define the notion of *easy* and *difficult* pairs of (σ, ρ).

Definition 1 (Easy and difficult cases; [GSW24, Definition 2.7])**.** *Let σ and ρ be two periodic sets. We say that this pair is* easy *if $0 \in \rho$ or*

- $\sigma = \{0, 2, 4, \dots\}$ *and* $\rho = \{1, 3, 5, \dots\}$, *or*
- $\sigma = \rho = \{1, 3, 5, \dots\}$.

Otherwise, the pair is said to be difficult*.*

When 0 is part of the set ρ, the empty set is a solution of minimum size. The remaining easy cases are when $\mathrm{m} = 2$, and correspond exactly the Refl-AllOff and AllOff problems, for which the decision problem can be solved in polynomial time. We now state the first main result of the thesis. Note that the pathwidth is at least as large as the treewidth of any graph, and hence stating the result for pathwidth is even stronger than for treewidth.

Main Theorem 1 ([GSW24, Main Theorem 2])**.** *Write $\sigma, \rho \subseteq \mathbb{N}$ for difficult periodic sets that both have the same period* $\mathrm{m} \geq 2$*. Unless SETH fails, for all $\varepsilon > 0$, there is no algorithm that can decide in time $(\mathrm{m}-\varepsilon)^{\mathsf{pw}} \cdot |G|^{O(1)}$ whether the input graph G has a (σ, ρ)-set, when a path decomposition of width* pw *is given with the input.*

For Refl-AllOff and AllOff, we show that, while deciding whether any solution exists can be done in polynomial time, the problem of deciding whether a solution smaller than a given integer k exists can not be done quicker than $(2-\varepsilon)^{\mathsf{tw}} \cdot |G|^{O(1)}$ under the SETH. Moreover, we show that also the problem of deciding whether a solution of size at least k exists cannot be solved in time $(2-\varepsilon)^{\mathsf{tw}} \cdot |G|^{O(1)}$ for any $\varepsilon > 0$ when σ, ρ have period 2, unless the SETH is false.

Main Theorem 2 ([GSW24, Main Theorem 3])**.** *Unless SETH fails, for all $\varepsilon > 0$, there is no algorithm for each of the problems Refl-AllOff and AllOff deciding in time $(2-\varepsilon)^{\mathsf{pw}} \cdot |G|^{O(1)}$ whether there exists a solution of size at most k for a graph G that is given with a path decomposition of width* pw*.*

Main Theorem 3. *Let σ, ρ be periodic sets with period 2. Unless SETH fails, for all $\varepsilon > 0$, there is no algorithm for (σ, ρ)-GenDomSet deciding in time $(2-\varepsilon)^{\mathsf{pw}} \cdot |G|^{O(1)}$ whether there exists a solution of size at least k for a graph G that is given with a path decomposition of width* pw*.*

Hence, we conclusively settle the complexity of the minimization problem in all cases, and even show that the algorithm is optimal under SETH for the maximization problem when the period is 2. The only question that remains open is the complexity of the maximization problem when $0 \in \rho$. We expect that one would need to craft many components from scratch to deal with this situation, and hence, we have chosen to not focus on it.

In Section 2.1 we define the graph notation we use. In Chapter 2 existing literature about the topics (σ, ρ)-GenDomSet, treewidth, parameterized complexity, and lower bounds conditioned on the SETH will also be examined. In particular, the key ideas behind the quick algorithm leading to the upper bound matching the lower bound will be presented. The main scientific contributions of the thesis are the lower bounds presented in Chapters 3 to 5.

CHAPTER 2

Preliminaries and Related Work

In this chapter, the notation used will be introduced briefly, and literature related to the topics of the thesis will be examined. Compared to the introduction, the coverage of the papers will be more in-depth, that is, for some papers concrete results and ideas behind the proofs will be presented. The aim is not only to cover papers that are directly related to the problem at hand, but also literature related to the central concepts of the scientific contribution. Especially Sections 2.2 and 2.3 cover standard material, and the experienced reader might want to directly skip to the subsequent sections. The notation is covered in Section 2.1. We then introduce the notion of *parameterized complexity* and the parameter *treewidth* in Section 2.2. Then, we present the lower bound machinery in Section 2.3, and finish by covering *generalized dominating set* in Section 2.4.

2.1 Notation

A graph $G = (V, E)$ is a structure where V is a finite set of vertices, and $E \subseteq \binom{V}{2}$ a set of edges. Hence, all considered graphs are finite and undirected. Given a graph $G = (V, E)$, we often refer to the set V as $V(G)$ and the set of edges E as $E(G)$. Given an edge $\{u, v\}$ we may also simply refer to it as uv. The (open) neighborhood of a vertex v in a graph G is defined as $N_G(v) = \{u \mid uv \in E(G)\}$. The closed neighborhood is $N_G[v] = N_G(v) \cup \{v\}$. These concepts can be extended to sets of vertices. When $S \subseteq V(G)$, then $N_G(S) = \bigcup_{v \in S} N_G(v) \setminus S$, and $N_G[S] = \bigcup_{v \in S} N_G[v]$. The degree of a vertex v is $d_G(v) = |N_G(v)|$. We may drop the subscript G from all of these concepts when the graph G is clear from the context. The symbol $\mathbb{N}$ refers to the set of natural numbers and $0 \in \mathbb{N}$. We write $[n, m]$ for the set $\{x \in \mathbb{Z} \mid n \leq x \leq m\}$.

2.2 Parameterized Complexity and Treewidth

The intractability of many algorithmic problems, such as SAT, Dominating Set and Independent Set is one of the most fundamental and widespread beliefs in theoretical computer science. This leaves only a handful of possibilities left for dealing with problems that are NP-complete or even more difficult. One can turn to approximation or heuristic algorithms, if obtaining the optimum solution is not a requirement. Otherwise, the problem may still be solvable in polynomial-time if the input belongs to a certain graph class. For example, it is well-known that Independent Set can be solved in polynomial time on tree graphs. A related way of dealing with intractability is provided by the area of *parameterized complexity*. See [AEFM89, DF92b, DF92a, DF95a, DF95b] for early work in the area and [DF13, CFK+15] for some relatively recent textbooks about the topic. The content of this section will largely follow the well-known textbook on parameterized complexity by Cygan et al. [CFK+15], as the textbook exactly covers the fundamental notions of the area, albeit at greater detail than this thesis.

2.2.1 Parameterized Complexity

The central idea of parameterized complexity is that input instances with a certain structure can allow for quick algorithms, even if the considered problem is intractable in general. More concretely, a parameterized problem is a set $Q \subseteq \Sigma^* \times \mathbb{N}$, where Σ is a finite alphabet, e.g. $\{0, 1\}$ (see [CFK+15, Definition 1.1]). Given an instance $I = (x, k)$, the task is to decide whether I is a part of Q. Usually, the integer (parameter) k of the instance will be some measurement of the remaining instance string x, such as the size of the solution the instance asks for, or a structural measurement.

If there is an algorithm that can decide whether $I \in Q$ in time $f(k) \cdot |x|^{O(1)}$, the problem is said to be *fixed parameter tractable (FPT)* [CFK+15, Definition 1.2]. We also denote the class of all problems that are fixed parameter tractable as FPT. As running times of this type effectively separate the running time dependency on the size of the instance and the parameter, achieving running times of this type is one of the main goals. Another often obtainable, potentially worse running time would be $|x|^{f(k)}$. Problems that can be decided in that time are called *slice-wise polynomial (XP)* [CFK+15, Definition 1.3].

To illustrate the concept, let us take a look at the famous Vertex Cover problem parameterized by the solution size k. It is easy to see that an algorithm with running time $|V(G)|^{k+O(1)}$ exists that simply enumerates all subsets of the vertices of size at most k. Hence, the problem is Vertex Cover parameterized by the solution size k is in XP. But, an even better running time can also be achieved, as illustrated e.g. by the very quick algorithm due to Chen et al. [CKX10] that runs in time $1.2738^k + |V(G)|^{O(1)}$. This FPT algorithm can evidently be massively quicker than the naive XP algorithm.

The complexity of parameterized problems that are XP but not FPT may still be classifiable according to the W-hierarchy [CFK+15, Section 13.3]. A parameterized problem that is $W[i]$-hard for any $i \geq 1$ can be in XP, but it is believed to not be in FPT.

For instance, Independent Set parameterized by solution size is $W[1]$-hard[DF95b] and Dominating Set parameterized by solution size is $W[2]$-hard [DF95a].

The main problem considered in this paper is FPT, and hence, concrete details about the W-hierarchy is of no importance for the thesis. To understand the remaining parts of the literature review, it will be sufficient to keep in mind that techniques exist to provide evidence for the fact that a problem is not FPT.

2.2.2 Treewidth

It is well-known that many NP-hard graph problems can be solved in polynomial-time when the input graph is a tree. For instance, Vertex Cover, Independent Set, Dominating Set, 3-Coloring, Feedback Vertex Set, and Hamiltonian Cycle admit linear-time algorithms on tree graphs (see e.g. [CFK⁺15, Theorem 7.9 and Theorem 7.10]). A natural next step is asking whether such problems can also be solved quickly on graphs that are structurally close to tree graphs.

One way of capturing how close a graph is to a tree is the notion of *treewidth*, which is tightly bound to the notion of *tree decomposition*. Refer to [CFK⁺15, Chapter 7] for a recent introduction to the topic given in a textbook. Roughly speaking, a tree decomposition is obtained by mapping the vertices of a graph to subsets of vertices of a tree, while ensuring that the mapping fulfills certain properties that show that the original graph is structurally similar to the tree the vertices are mapped onto. The width of a tree decomposition is a numerical measurement of it, and the treewidth of G will simply be the minimum width over all tree decompositions. The notions of treewidth and pathwidth in their current forms were introduced by Robertson and Seymour [RS86, RS83], but many equivalent characterizations are known (see e.g. [Bod98] or [CFK⁺15, Section 7.5]).

Formally, tree decompositions and treewidth are defined as follows.

Definition 2 (Tree decompositions and treewidth; [CFK⁺15, Section 7.2]). *A tree decomposition of a graph G is a pair $\mathcal{T} = (T, \{X_t\}_{t\in V(T)})$, where T is a tree graph. Every vertex of T is assigned a set $X_t \subseteq V(G)$, often called a bag, such that the following conditions are fulfilled:*

1. $\bigcup_{t\in V(T)} X_t = V(G)$, *that is, every vertex of G is present in some bag of the decomposition,*
2. *for every edge $uv \in E(G)$ there exists a t such that $u \in X_t$ and $v \in X_t$, that is, there is a bag which contains both endpoints of the edge,*
3. *for every $v \in V(G)$ the graph $T[T_v]$ is connected, where $T_v = \{t \in V(T) \mid v \in X_t\}$.*

The width of tree decomposition $\mathcal{T}$ is $\max\{|X_t| - 1 \mid t \in V(T)\}$, *that is, the size of a smallest bag of $\mathcal{T}$ minus one.*

The treewidth of G is the minimum width of any tree decomposition of G.

We can similarly define the standard notion of a *path decomposition*.

Definition 3 (Path decomposition; [CFK+15, Section 7.2])**.** *A path decomposition of a graph G is a tree decomposition $(T, \{X_t\}_{t \in V(T)})$ of G in which the decomposition tree T is a path. The pathwidth of G is the minimum width of any path decomposition of G.*

Given a graph G, we denote the treewidth of the graph by $\mathsf{tw}(G)$, and the pathwidth by $\mathsf{pw}(G)$. We may drop the G from this quantity if it is clear which graph is meant. The reason why one subtracts one from the largest bag size in the definitions is that this way, it is ensured that trees with at least two vertices have a treewidth of one, and paths with at least two vertices have a pathwidth of one [CFK+15, Section 7.2].

Often, problems that are solvable in polynomial-time on trees are also solvable in polynomial time if the input graph G is provided together with a tree decomposition of small width. Hence, these problems tend to be FPT when the parameter is the treewidth of the input graph. This is true for the problems Vertex Cover, Independent Set, Dominating Set, 3-Coloring, Feedback Vertex Set, and Hamiltonian Cycle mentioned at the beginning of the section (see e.g. [CFK+15, Theorem 7.9 and Theorem 7.10]).

Unfortunately, computing the treewidth of a graph is NP-hard in general [ACP87]. However, the problem is FPT parameterized by the treewidth, and it can be solved in time $2^{O(\mathsf{tw}^2)} \cdot |V(G)|^{O(1)}$ [KL23]. Many other exact and approximation algorithms are also known, see, for example, Table 1 in [KL23] for a good overview. Hence, the assumption that a tree decomposition is provided together with the input is not a strong restriction. In particular, if an FPT algorithm is obtainable when a tree decomposition is provided together with the input graph, one can solve the problem in FPT time even when no tree decomposition is provided by first computing a decomposition of minimum width.

Given that so many problems are FPT parameterized by treewidth, a perhaps surprising result is given by Fellows et al. [FFL+11]. They show that the problem List Coloring, which is a well-known extension of the classical Coloring problem, is $W[1]$-hard parameterized by treewidth. Contrasting this result, they prove that the List Chromatic Number problem is FPT parameterized by the same parameter.

Algorithms that show tractability parameterized by treewidth typically operate by *dynamic programming* on the provided tree decomposition. To make such algorithms easier to define, one usually also assumes that the provided tree decomposition is a *nice tree decomposition* [CFK+15, Section 7.2]. Nice tree decompositions are rooted tree decompositions, in which every node of the decomposition tree is one of a constant number of types. This makes writing dynamic programs that operate bottom up, starting from the leaves, significantly easier. One can convert any tree decomposition into a nice one sufficiently quickly without increasing the width [CFK+15, Lemma 7.4].

One characterization of pathwidth which is at times quite convenient is that of the *node search number* (see [KP86, KP85, FT08] or [CFK+15, Section 7.5]). Imagine that the graph is a network of tunnels, and each edge is initially contaminated by dangerous gas. The graph has to be cleaned by using some number of *searchers*, which are either on a vertex of the graph, or in a pool of free searchers. An edge uv is cleaned when a searcher is present on vertex u, and another searcher is present on vertex v at the same time. Unfortunately, the gas instantaneously spreads through vertices on which no searchers are placed, and hence, even edges that were already cleaned can be recontaminated if there is a path from the edge to another edge that is not clean, and no searcher is placed on a vertex of the path. The goal of cleaning the entire graph has to be achieved by using a *node search strategy*. The possible moves for a strategy are either placing a searcher from the pool of free searchers on a vertex of the graph, or removing a searcher from the graph and placing it into the pool of free searchers. The node search number is then the minimum number of searchers required to clean the graph using a node search strategy. Moreover, if a graph has pathwidth at most pw, then its node search number is at most $\mathsf{pw} + 1$. Finally, if the graph can be cleaned using k searchers, it can be cleaned by a search strategy using k searchers that never recontaminates an edge, and such a search strategy can be transformed into a path decomposition of width $k - 1$ in polynomial time. [1]

2.3 Lower Bounds based on the (Strong) Exponential Time Hypothesis

The *Strong Exponential Time Hypothesis (SETH)* is a complexity-theoretic assumption about the minimum amount of time that is needed to decide the k-SAT problem as k goes towards infinity. As it assumes a concrete running time lower bound, the hypothesis can be used to also infer tight lower bounds for other problems, by providing carefully crafted reductions from k-SAT. The hypothesis is strongly related to the *Exponential Time Hypothesis (ETH)* introduced by Impagliazzo and Paturi [IP01], and proving the SETH assuming ETH was stated as an open problem by them. The name Strong Exponential Time Hypothesis for the conjecture was introduced by Calabro et al. [CIP09]. Refer to [CFK+15, Chapter 14] and [LMS11] for good introductions to the topics.

We now proceed to the formal definition of the ETH and SETH as given in [CFK+15, Section 14.1].

Let δ_k be the infimum of the set

$$\{\delta \mid \text{there is an algorithm for } k\text{-SAT with running time in } 2^{\delta \cdot n} \cdot (n+m)^{O(1)}\},$$

where n denotes the number of variables and m the number of clauses of an instance. The ETH is the hypothesis that $\delta_3 > 0$. The SETH assumes that $\lim_{k\to\infty} \delta_k = 1$. Note that the SETH is equivalent to saying that for any $\varepsilon > 0$, there is a k such that k-SAT

[1] The only exception to this are graphs without edges, which require zero searchers to clean, yet have pathwidth 0. However, this exception is of no relevance in this thesis.

cannot be solved in time $(2-\varepsilon)^n \cdot (n+m)^{O(1)}$ on instances with n variables and m clauses [CFK+15, Section 14.5].

The ETH implies that SAT with arbitrarily sized clauses cannot be solved in subexponential time, whereas SETH implies that SAT cannot be solved in time $(2-\varepsilon)^n$ for any $\varepsilon > 0$ [CFK+15, Section 14.1]. According to Cygan et al. [CFK+15, Section 14.1] the ETH is a widely-believed assumption, whereas the SETH is not as universally accepted. However, the bounds obtainable from SETH are considerably more precise than those obtainable via ETH, and an algorithm that disproves the SETH would still be considered ground-breaking.

2.3.1 Lower Bounds under the Exponential Time Hypothesis

Classical polynomial time reductions from 3-SAT that show the NP-hardness of problems immediately imply lower bounds under ETH (see, for example, the survey [LMS11]). Typical reductions from 3-SAT to the problems Vertex Cover or Dominating Set result in instances with size in $O(n+m)$, if the input instance contained n variables and m clauses. Moreover, the number of clauses m can be in $\Theta(n^3)$. This is quite unfortunate, as this means that the size of the output instance is in $O(n^3)$, and not necessarily linear in n, which weakens the lower bound. If there existed an algorithm that can solve Vertex Cover or Dominating Set in time $2^{o(\sqrt[3]{|G|})}$, we could solve 3-SAT in time $2^{o(n)}$, which would contradict the ETH (see also [CFK+15, Section 14.1]).

The fact that we only get a lower bound having the third-root in the exponent rather than n directly is quite unfortunate, but can luckily be circumvented to obtain a lower bound of $2^{o(|G|)}$ [CFK+15, Theorem 14.6]. The required tool is the so-called *sparsification lemma* [IPZ01]. The sparsification lemma is not relevant for the remainder of this thesis, and hence omitted. However, the issue illustrates that reductions for proving tight lower bounds must be crafted very carefully in order to not worsen the bound too much.

Similarly to the lower bound for Dominating Set and Vertex Cover conditioned on ETH, the well-known problem Hamiltonian Path cannot be solved in time $2^{o(|V(G)|)}$ if the ETH is true [LMS11]. Lokshtanov et al. [LMS11] also note that the reduction from 3-SAT to Hamiltonian Path on planar graphs due to Garey et al. [GJT76] shows a lower bound of the form $2^{o(\sqrt{|V(G)|})}$ under ETH, and remark that this bound is asymptotically tight. Observe that in the latter case, the "decrease" from a bound in $2^{o(|V(G)|)}$ to $2^{o(\sqrt{|V(G)|})}$, stemming from the fact that the number of vertices of the output instance is quadratic in the number of variables of the input 3-SAT instance, is vital to obtain the correct bound.

When crafting lower bounds for FPT problems under ETH, the requirement shifts slightly. Since we now seek lower bounds for the function of the parameter of the FPT running time, the size of the output instance itself is generally no longer a concern. Instead, it must be ensured that the value of the parameter of the output instance has the correct size, for example that it is linear in the number of variables of the input instance.

Cai and Juedes [CJ03] build upon the work of Impagliazzo et al. [IPZ01], and show that problems such as Max SAT and Vertex Cover parameterized by the solution size k cannot be solved in time $2^{o(k)} \cdot |I|^{O(1)}$ under ETH ($|I|$ is the size of the instance). Moreover, they consider problems on planar graphs, showing that Dominating Set on planar graphs, Vertex Cover on planar graphs and Independent Set on planar graphs cannot be solved in time $2^{o(\sqrt{k})} \cdot |V(G)|^{O(1)}$ unless ETH fails.

Often, FPT algorithms admit a running time in $c^k \cdot |I|^{O(1)}$ (for parameter k), in which c is a constant. A natural question is whether some FPT problems are "inherently harder" and only admit running times that are at least "slightly superexponential". For instance, Lokshtanov et al. [LMS18b] tackle this problem, and show the following array of lower bounds assuming ETH:

- The problem Closest String does not admit an algorithm with running time $2^{o(d \log d)} \cdot |I|^{O(1)}$ or $2^{o(d \log |\Sigma|)} \cdot |I|^{O(1)}$, where Σ is the alphabet the strings are over, and d a bound on the distance.
- The problem Distortion cannot be solved in time $2^{o(d \log d)} \cdot |V(G)|^{O(1)}$, where d is a bound on the distortion.
- There is no algorithm for the Disjoint Paths problem that has a running time in $2^{o(\mathsf{tw} \log \mathsf{tw})} \cdot |V(G)|^{O(1)}$.
- Similarly, there is no algorithm for Chromatic Number that runs in time $2^{o(\mathsf{vc} \log \mathsf{vc})} \cdot |V(G)|^{O(1)}$, where vc is the size of a vertex cover of the input graph.

For all of these problems, the lower bounds match the best known algorithms [LMS18b].

Cygan et al. [CPP16] show that the problem Edge Clique Cover does not have a $2^{2^{o(k)}} \cdot |V(G)|^{O(1)}$ algorithm if the ETH is true, here, k is the solution size. This result is complemented by a preprocessing strategy due to Gramm et al. [GGHN08], that can reduce the number of vertices of any instance to at most 2^k in polynomial time. The resulting instance then has at most 4^k edges, and a dynamic programming algorithm on the subset of edges of the reduced instance can solve the problem in time $2^{O(4^k)} \cdot |V(G)|^{O(1)}$ overall [CPP16], matching the lower bound.

Continuing with lower bounds for problems with running times that are superexponential, Marx and Mitsou [MM16] provide lower bounds for the k-Choosability problem, showing that it cannot be solved in time $2^{2^{o(\mathsf{tw})}} \cdot |V(G)|^{O(1)}$ for any fixed integer $k \geq 3$, unless the ETH is false. The k-Choosability Deletion problem admits an even more extreme running time. For $k \geq 4$, the problem can be solved in time $2^{2^{2^{O(\mathsf{tw})}}} \cdot |V(G)|^{O(1)}$ and an algorithm running in time $2^{2^{2^{o(\mathsf{tw})}}} \cdot |V(G)|^{O(1)}$ would contradict the ETH.

We now briefly shift our attention to the parameter treewidth and consider the infamous Constraint Satisfaction Problem. Marx [Mar10] shows that, unless the ETH is false, there cannot be an algorithm for the problem and a class $\mathcal{G}$ containing graphs of unbounded treewidth, such that the algorithm can solve the problem in time $f(G) \cdot |I|^{o(\mathsf{tw}/\log \mathsf{tw})}$ when the primal graph G of the input instance is in $\mathcal{G}$, and tw is the treewidth of G. This in turn indicates that it is not possible to do much better than already known algorithms.

Cygan et al. [CFG+16] deal with the Graph Homomorphism and Subgraph Isomorphism problems, and prove lower bounds under ETH that match the running time of brute-force algorithms for both problems.

Let us now take a look at problems that are $W[1]$-hard, and hence most likely not FPT. For these problems, lower bounds based on the ETH should clearly not just rule out algorithms that have a subexponential running time in the parameter, but instead rule out algorithms that have a specific non-FPT running time. For instance, just because a problem is most likely not FPT, that does not mean that we cannot solve in, say, time $n^{\log k}$, which is still significantly better than trivial algorithms with running time n^k. Here, the ETH can be used to obtain evidence that such algorithms do not exist. And ideally, also in this case the obtained lower bounds should match the best known algorithm. Chen et al. [CCF+05] show that the problems Hitting Set and Set Cover do not admit algorithms with running time $n^{o(k)} \cdot m^{O(1)}$, unless $W[1] = \mathsf{FPT}$. Here, n refers to the size of the universe, m to the instance size and k to the solution size. Further, the fundamental problems Clique and Independent Set do not have algorithms with running time $f(k) \cdot |G|^{o(k)}$ if the ETH holds [CHKX06]. A simple corollary of this is that the ETH implies that $\mathsf{FPT} \neq W[1]$.

Fomin et al. [FGLS14] show that the problems Max-Cut and Edge Dominating Set cannot be solved in time $f(\mathsf{cw}) \cdot |V(G)|^{o(\mathsf{cw})}$ on graphs with cliquewidth cw under the ETH. Moreover, they provide asymptotically matching upper bounds for the problems.

Lastly, it is worth mentioning that lower bounds under ETH were also achieved for algorithms that are not exact. For instance, Marx [Mar07] showed lower bounds for *polynomial-time approximation schemes* of many problems, including Maximum Independent Set on planar graphs and Minimum Dominating Set on planar graphs under the hypothesis.

2.3.2 Strong Exponential Time Hypothesis

As illustrated in the last section, lower bounds under the ETH often provide evidence that algorithms with running time $2^{o(n)}$ are unlikely to exist. While this information rules out substantial improvements, it is still quite fuzzy in the sense that it does not differentiate between running times like 1.001^n or $1000!^n$. In particular, when having an algorithm with running time c^n at hand, the ETH cannot be used to determine whether the constant c is as low as possible.

That is where the SETH comes in. Assuming the SETH holds, we know that for any $\varepsilon > 0$, there is a k such that k-SAT cannot be solved in time $(2-\varepsilon)^n \cdot (n+m)^{O(1)}$. By showing that an improved algorithm to a specific problem would yield a faster than allowed algorithm for k-SAT, much more precise lower bounds can be given.

To illustrate this, we show that a constraint satisfaction problem called q-CSP-B, introduced by Lampis [Lam20], cannot be solved in time $(B-\varepsilon)^n \cdot (n+m)^{O(1)}$ for instances with n variables and m constraints, unless the SETH is false. The reason for choosing this problem as the exemplary problem is that this problem will be used as the starting point of the reduction in the main contribution of the thesis.

In the q-CSP-B problem [Lam20], one is given a set of n variables $\mathcal{X} = \{x_1, \ldots, x_n\}$, and a set of ℓ constraints $\mathcal{C} = \{C_1, \ldots, C_\ell\}$. Each constraint C_i consists of an ordered q-tuple of variables $\mathsf{scp}(C_i) = (x_{\lambda_1}, \ldots, x_{\lambda_q})$, and a set of satisfying assignments $\mathsf{acc}(C_i) \subseteq [1, B]^q$. The goal is to find a variable assignment $\pi : \mathcal{X} \to [1, B]$ such that, for each constraint C_i of the instance, with $\mathsf{scp}(C_i) = (x_{\lambda_1}, \ldots, x_{\lambda_q})$ we have $(\pi(x_{\lambda_1}), \ldots, \pi(x_{\lambda_q})) \in \mathsf{acc}(C_i)$. When such an assignment π exists, we say that the instance is satisfiable. We now restate and prove the lower bound.

Theorem 1 ([Lam20, Theorem 3.1])**.** *For any integer $B \geq 2$ and real number $\varepsilon > 0$ there is a q such that n variable m constraint q-CSP-B cannot be solved in time $(B-\varepsilon)^n \cdot (n+m)^{O(1)}$, unless the SETH is false.*

Proof. The proof is due to Lampis [Lam20].

For the sake of simplicity, instances of q-CSP-B are allowed to contain constraints with less than q variables, which can easily be handled by introducing dummy variables. Fix arbitrary integer $B \geq 2$ and real $\varepsilon > 0$, and assume that there is an algorithm that can solve q-CSP-B instances with n variables and m constraints in time $(B-\varepsilon)^n \cdot (n+m)^{O(1)}$ for any q.

Our first goal is to choose p large enough such that $2 \cdot (B-\varepsilon)^p \leq B^p$, which is possible since B^x grows exponentially quicker than $(B-\varepsilon)^x$. So, let p be the smallest integer such that $2 \cdot (B-\varepsilon)^p \leq B^p$. Then, we want to choose an integer t such that $2^t \leq B^p$ while being as large as possible. Now, observe that we have $2^t > \frac{B^p}{2}$; otherwise, t could be increased by at least one. So, we have $(B-\varepsilon)^p \leq \frac{B^p}{2} < 2^t \leq B^p$. Finally, we must choose a $\delta > 0$ such that $(B-\varepsilon)^p \leq (2-\delta)^t$. Given that $2^t > (B-\varepsilon)^p$, and the fact that $(B-\varepsilon)^p$ is already fixed, we must only choose δ small enough, which is always possible since $\lim_{x \to 0}(2-x)^t = 2^t$. Note that p, t only depend on B and ε, and are hence constants independent of the input size.

After having established these values and properties, we can finally proceed to the actual reduction. Let the input k-SAT instance contain the variables $x_1, \ldots, x_n$ and clauses $C_1, \ldots, C_m$. Next, partition the variables of the k-SAT instance into $\gamma = \lceil \frac{n}{t} \rceil$ groups, such that every group, except for potentially the last one, contains t variables. Denote these groups as $V_1, \ldots, V_\gamma$. For each of these γ groups, create a group of p variables for the q-CSP-B instance, and call these groups $X_1, \ldots, X_\gamma$. Recall that we have $2^t \leq B^p$. Hence, for all $i \in [1, \gamma]$, each possible assignment of the variables in group V_i can be injectively mapped to an assignment to the variables in group X_i.

The final task of the reductions is mapping the clauses to constraints of the q-CSP-B instance. Consider an arbitrary input clause. Then, we create a constraint for the

q-CSP-B instance, that has the union of all variable groups that correspond to variables of the clause as the scope. That is, if the variables of the clause appear in the groups $V_{\lambda_1}, \ldots, V_{\lambda_r}$, then the scope of the constraint is $X_{\lambda_1} \cup \cdots \cup X_{\lambda_r}$. The satisfying assignment of the constraint are exactly the assignments corresponding to satisfying assignments of the clause. As $r \leq k$, each constraint we create contains at most $k \cdot p$ variables, and hence, the output instance is an instance of q-CSP-B where $q = k \cdot p$. Recall that k, p and B are constants. Hence, the number of variables of the constraint is constant, and so is the overall size of the constraint. This means that we can perform the above operation for every clause of the input instance of polynomial time.

It is easy to see that the instances are equivalent, as the constraints of the q-CSP-B instance ensure that each input clause is satisfied, and a satisfying assignment of the k-SAT instance can directly be mapped to a satisfying assignment of the constraint satisfaction problem.

Finally, we need to validate that our algorithm indeed breaks the SETH. For that purpose, apply the algorithm to the output instance. It runs in time $(B-\varepsilon)^{\gamma \cdot p} \cdot (\gamma \cdot p + m)^{O(1)}$, since the output instance has $\gamma \cdot p$ variables and m constraints. We can rewrite and obtain

$$(B-\varepsilon)^{\gamma \cdot p} \leq (B-\varepsilon)^{(1+\frac{n}{t}) \cdot p} = ((B-\varepsilon)^p)^{1+\frac{n}{t}} \leq \left((2-\delta)^t\right)^{1+\frac{n}{t}} = (2-\delta)^{n+t}.$$

Moreover, t is a constant, and hence $(2-\delta)^{n+t} = O((2-\delta)^n)$. Also, $\gamma \cdot p \leq p + \frac{n \cdot p}{t}$ for constants p and t, showing that $\gamma \cdot p$ is in $O(n)$.

Hence, we can solve k-SAT in time $(2-\delta)^n \cdot (n+m)^{O(1)}$, which contradicts the SETH since k was arbitrary, that is, we would have $\lim_{x \to \infty} \delta_x \leq 2 - \delta < 2$ if a q-CSP-B algorithm with the assumed running time were to exist. □

The usefulness of the q-CSP-B problem is evident: Oftentimes, one does not want to prove a lower bound of the form $(2-\varepsilon)^n$ under SETH, but a lower bound of the form $(c-\varepsilon)^n$, where c is an integer different from 2. Instead of having to deal with the shenanigans used in the proof above, one can directly use q-CSP-c as the starting problem, which provides the correct base right away.

Next, we cover lower bounds under the SETH that scientists previously came up with. In general, lower bounds under the SETH are even more sensitive to changes in the size of the output instance or the parameter (in the parameterized setting) than lower bounds under the ETH. Moreover, the lack of a theorem similar to the sparsification lemma appears to be a major obstruction. In particular, if the output size/parameter is linear in the size of the number of *clauses* of the input instance, that is simply not good enough to provide tight bounds under SETH (see e.g. the discussion in [CFK+15, Section 14.5]).

Pătraşcu and Williams [PW10] show lower bounds for a variety of problems. Concretely, they show that Dominating Set parameterized by the solution size k does not have an $O(n^{k-\varepsilon})$ algorithm for any $\varepsilon > 0$ and $k \geq 3$, unless the SETH fails. They also provide conditional lower bounds for the problems d-SUM, 2-SAT and HornSat.

Abboud et al. [ABW15] consider the problem of computing the longest common subsequence and dynamic time warping distance of two strings of length n. They show that both problems do not admit algorithms with running time in $O(n^{2-\varepsilon})$ for any ε, unless the SETH is not true.

Abboud, Bringman, Hermelin and Shabtay [ABHS22] show that a known algorithm with pseudo-polynomial running time for the SUBSET SUM problem can most likely not be improved upon, that is, there is no algorithm with running time $T^{1-\varepsilon} \cdot 2^{o(n)}$ for any $\varepsilon > 0$ under the SETH. In the used notation, T is the target value and n the count of numbers of the input instance.

Backurs et al. [BIS17] consider problems arising in machine learning, and provide lower bounds under SETH for them. Bennet et al. [BGS17] consider special cases of the CLOSEST VECTOR PROBLEM and prove that a running time of $2^{(1-\varepsilon)\cdot n}$ is not obtainable under the SETH, where n is the lattice rank.

Now, we once again point our attention to our beloved parameter treewidth. The first major results in this area are due to Lokshtanov et al. [LMS18a], who show the following assortment of results under the SETH:

- INDEPENDENT SET cannot be solved in time $(2-\varepsilon)^{\mathsf{tw}} \cdot |V(G)|^{O(1)}$,
- DOMINATING SET cannot be solved in time $(3-\varepsilon)^{\mathsf{tw}} \cdot |V(G)|^{O(1)}$,
- MAX CUT cannot be solved in time $(2-\varepsilon)^{\mathsf{tw}} \cdot |V(G)|^{O(1)}$,
- ODD CYCLE TRANSVERSAL cannot be solved in time $(3-\varepsilon)^{\mathsf{tw}} \cdot |V(G)|^{O(1)}$,
- q-COLORING has no $(q-\varepsilon)^{\mathsf{tw}} \cdot |V(G)|^{O(1)}$ algorithm for any integer $q \geq 3$,
- PARTITION INTO TRIANGLES cannot be solved in time $(2-\varepsilon)^{\mathsf{tw}} \cdot |V(G)|^{O(1)}$.

The authors mention that each of the lower bounds listed above matches the running time of the best known algorithm, and hence, they settle the question of the precise complexity of these problems.

Borradaile and Le [BL16] provide an algorithm that can solve the r-DOMINATING SET problem in time $O((2r+1)^{\mathsf{tw}} \cdot |V(G)|)$. Moreover, they show that the connected variant of the problem can be solved in time $O((2r+2)^{\mathsf{tw}} \cdot |V(G)|^{O(1)})$ by a randomized algorithm. By providing lower bounds conditioned on the SETH, they provide evidence that these running times are in fact the best possible running times for the problems.

Cygan et al. [CNP+22] provide a breakthrough result by showing that the problems HAMILTONIAN PATH, STEINER TREE, FEEDBACK VERTEX SET and CONNECTED DOMINATING SET admit randomized FPT algorithms with running time $c^{\mathsf{tw}} \cdot |G|^{O(1)}$, where c is a constant. Lower bounds for the parameter *pathwidth*, providing evidence that the constant c is as small as possible, are given by them for the problems STEINER TREE, FEEDBACK VERTEX SET and CONNECTED DOMINATING SET.

Besides the result about q-CSP-B we have already covered, Lampis [Lam20] shows that, for all integers $k \geq 3$, the k-Coloring problem can be solved in time $(2^k - 2)^{\mathsf{cw}} \cdot |G|^{O(1)}$, and cannot be solved in time $(2^k - 2 - \varepsilon)^{\mathsf{cw}} \cdot |G|^{O(1)}$ on graphs with cliquewidth cw, for any $\varepsilon > 0$, unless the SETH is not correct.

Cygan et al. [CDL+16] provide a variety of results conditioned on the SETH and a similar conjecture. One major contribution due to them is that a variety of conjectures are *equivalent* to the SETH, which increases the credibility of the assumption. They show that the following assumptions are equivalent to the SETH:

- For all $\varepsilon < 1$, there is a k such that k-Hitting Set cannot be solved in time $O(2^{\varepsilon \cdot n})$ for a universe of size n and sets of size at most k,
- For all $\varepsilon < 1$, there is a k such that k-Set Splitting cannot be solved in time $O(2^{\varepsilon \cdot n})$ for a universe of size n and sets of size at most k,
- For all $\varepsilon < 1$, there is a k such that n variable k-NAE-SAT cannot be solved in time $O(2^{\varepsilon \cdot n})$,
- For all $\varepsilon < 1$, there exists a k such that n variable k-VSP-Circuit-SAT cannot be solved in time $O(2^{\varepsilon \cdot n})$.

The Strong Exponential Time Hypothesis can also be used to rule out faster algorithms for counting problems, in which determining the number of solutions to a problem is the goal. However, the nature of these problems suggest that a variant of the hypothesis that is tuned to counting problems should be used. This hypothesis, called the *counting Strong Exponential Time Hypothesis (#SETH)* (see [CM16], and also [DHM+14] for the ETH equivalent) is simply the same as SETH, only that deciding whether a solution exists is replaced with counting the number of solutions. Clearly, if one can count the number of solutions quickly, one can also decide whether a solution exists quickly. Hence, the SETH implies the #SETH, and the latter is the more plausible assumption. Curticapean and Marx [CM16] show that counting the number of perfect matchings of a graph has no $(2-\varepsilon)^{\mathsf{tw}} \cdot |V(G)|^{O(1)}$ algorithm for any $\varepsilon > 0$, unless the #SETH is false. They also show that the same problem does not have an $O(|V(G)|^{(1-\varepsilon)\cdot \mathsf{cw}})$ algorithm where cw is the cliquewidth of the input graph under the same conjecture.

Marx et al. [MSS22] consider the AntiFactor problem, and provide a lower bound under #SETH for the counting version of a special parameterization of the problem, among other results. A similar result was also obtained by Max et al. [MSS21] for the General Factor problem.

Roditty and Williams [RW13] improve upon a known algorithm for approximating the diameter of a graph, and show that no algorithm with running time $O(|E(G)|^{2-\varepsilon})$ exists for the $(\frac{3}{2} - \varepsilon)$-approximation of the diameter under SETH.

In conclusion, we see that the ETH and the SETH are useful hypotheses with solid foundations. Using them, lower bounds can be obtained for classical NP-hard problems, but they are also extremely useful in the parameterized setting, and can even be used to obtain lower bounds for problems that can be solved in polynomial-time.

2.4 Generalized Dominating Set

We now proceed to the family of graph problems that is considered in this work, namely (σ, ρ)-GenDomSet. The problem was introduced by Telle [Tel94, TP93] and its decision version is defined below. Given a graph G, a set $S \subseteq V(G)$ is a (σ, ρ)-set if and only if we have $|N(v) \cap S| \in \sigma$ for all $v \in S$ and $|N(v) \cap S| \in \rho$ for all $v \in V(G) \setminus S$.

(σ, ρ)-GenDomSet (Decision problem)
Input: Graph G
Question: Is there a set $S \subseteq V(G)$ such that S is a (σ, ρ)-set of G?

Note that σ, ρ are fixed non-empty subsets of the natural numbers and not part of the problem input. Naturally, one may not only want to decide if any solution exists, but instead ask for solutions of specific sizes. We refer to the problem of deciding whether a (σ, ρ)-set of size at most k exists as the minimization version, and to the problem of deciding whether a (σ, ρ)-set of size at least k exists as the maximization version.

The strength of the (σ, ρ)-GenDomSet framework is that it captures a huge variety of different domination-like problems. For illustrative purposes, we provide a selection of the problems listed by Telle [Tel94, Table I]. It is understood that some problems referred to in the list may typically not be framed as decision problems, but as minimization or maximization problems.

- When $\sigma = \mathbb{N}$, $\rho = \{1, 2, \dots\}$, we obtain Dominating Set.
- When $\sigma = \{0\}, \rho = \mathbb{N}$, we obtain Independent Set.
- When $\sigma = \{0\}, \rho = \{1\}$ we obtain Perfect Code.
- When $\sigma = \{0\}, \rho = \{1, 2, \dots\}$ we obtain Independent Dominating Set.
- When $\sigma = \mathbb{N}$, $\rho = \{1\}$ we obtain Perfect Dominating Set.
- When $\sigma = \rho = \{1, 2, \dots\}$ we obtain Total Dominating Set.
- When $\sigma = \mathbb{N}, \rho = \{q, q + 1, \dots\}$, where $q \geq 1$, we obtain q-Dominating Set.
- When $\sigma = \{q\}, \rho = \mathbb{N}$, where $q \geq 0$, we obtain Induced q-Regular Subgraph, which is Induced Matching when $q = 1$.

Evidently, the problem family is *extremely expressive*, and obtaining fast algorithms for problems in it is of high priority. The problem is clearly decidable if σ, ρ are recursive sets, but not a lot more can be immediately deduced. In particular, the complexity of the different types of problems can differ completely depending on the structure of σ and ρ. On one hand, the framework captures problems like Dominating Set, where the maximization and decision problems are trivial, and only the minimization problem is hard. On the other hand, we have problems like Independent Set, where the maximization problem is difficult, while the other problems are again trivial. And then, there are also problems like Perfect Code, for which the decision problem is NP-hard even if the input graph is planar and bipartite [LT02].

Telle [Tel94] examines the problem for specific sets, and shows NP-hardness as well as polynomial-time solvability for many cases. For instance, [Tel94, Theorem 1] states that the decision problem is NP-complete when $\sigma = \{0\}$ and $\rho = \{q, q+1, q+2, \dots\}$ for all integers $q \geq 2$. Moreover, the NP-completeness of Dominating Induced Matching is shown. Another interesting result due to them is that the decision problem is NP-complete when $0 \notin \rho$, and when both σ and ρ are finite. The last result of Telle [Tel94] that we want to highlight is that the maximization problem is solvable in polynomial time when $\sigma = \{k, k+1, k+2, \dots\}$ and ρ has a specific structure.

Another special case of the problem was studied by Halldórsson et al. [HKT00a] under the name *independent sets with domination constraints*. There, σ is fixed to be $\{0\}$, and hence, any solution is an independent set. The set ρ is not restricted, except that it is non-empty. Note that they assume that an oracle for deciding whether an element is in ρ is given. Under this assumption, they fully settle the question of the classical complexity of the decision problem, by showing that it is NP-complete if $\rho \neq \mathbb{N}^+$ and there is a $k \geq 0$ with $k \notin \rho$ but $k+1 \in \rho$. In the remaining cases, the problem can be solved in polynomial time. Moreover, results in the realm of approximation algorithms are provided for optimization variants.

Jacob et al. [JRS19] consider a problem called Fair Set that is a special case of (σ, ρ)-GenDomSet. Meybodi et al. [MFMP20] consider the special cases $[1, j]$-Dominating Set, where $\sigma = \mathbb{N}$ and $\rho = [1, j]$, and $[1, j]$-Total Dominating Set that is expressible by setting $\sigma = \rho = [1, j]$.

Fomin et al. [FGK+11] consider the problem where σ does not contain two consecutive integers and either (1) σ and ρ are finite or (2) at least one of σ, ρ is finite and $\sigma \cap \rho = \emptyset$. They present novel branching algorithms for the considered problem, and show that all (σ, ρ)-sets can be enumerated in time $c^n \cdot |G|^{O(1)}$ for some constant $c < 2$ (depending on σ and ρ).

Another algorithm due to Fomin et al. [FGK+09] is for the case when $\sigma = \{p\}$ and $\rho = \{q\}$. In that case, there is an exact algorithm that can solve the decision and optimization problems as well as the counting problem in time $2^{|V(G)|/2} \cdot |V(G)|^{O(1)}$. They extend their algorithm so that it also works for the case where $\sigma = p + m \cdot \mathbb{N}$ and $\rho = q + m \cdot \mathbb{N}$ and m is an integer that is at least two. It should be noted that the latter case is a generalization of the case we consider in this thesis for the parameter treewidth.

Due to the hardness that can be observed for many cases of (σ, ρ)-GenDomSet, researchers commonly view problems of this type under the lens of parameterized complexity. Golovach et al. [GKS12] initiated the study of the problem using the parameter solution size. For finite sets σ and ρ with $0 \notin \rho$ and the parameter solution size, the minimization problem turns out to be $W[1]$-complete. On the other hand, when σ or ρ are finite or cofinite, then the problem of finding a solution of size at least $|V(G)| - k$ is FPT parameterized by k. We would also like to point out their result for the case of even and odd sets. In Theorem 16, they show that, when $\sigma, \rho \in \{\{0, 2, 4, \dots\}, \{1, 3, 5, \dots\}\}$, then the problem of deciding whether a (σ, ρ)-set of size at least $|V(G)| - k$ exists is $W[1]$-hard parameterized by k. Note that this covers problems strongly related to Refl-AllOff and AllOff.

Parameterization by treewidth, the main topic of this thesis, has also gained considerable attention for the problem. The observations given in this paragraph are based on the thoughts provided in van Rooij [vBR09] and Focke et al. [FMI+23a]. Let σ be finite or cofinite set. Define $\sigma_{\max} = \max \sigma$ if σ is finite, and $\sigma_{\max} = 1 + \max \mathbb{N} \setminus \sigma$ if σ is cofinite and $\sigma \neq \mathbb{N}$, if σ is cofinite and $\sigma = \mathbb{N}$, then $\sigma_{\max} = 0$. Similarly, let ρ be finite or cofinite and define $\rho_{\max}$. Then, a classical dynamic programming algorithm on tree decomposition will need to track two properties for each vertex of a bag. First, it needs to track whether the vertex is selected or not in the current (partial) solution, and secondly, it needs to track how many selected neighbors the vertex has. This naturally leads to the σ states $\sigma_0, \dots, \sigma_{\sigma_{\max}}$ and the ρ states $\rho_0, \dots, \rho_{\rho_{\max}}$. Observe that, when σ is finite, we must not track the case that a selected vertex has more than $\sigma_{\max}$ neighbors, such a partial solution is never feasible. When σ is cofinite, any selected vertex that has at least $\sigma_{\max}$ neighbors will remain "happy", as it can always receive more selected neighbors. Analogous properties hold for the set ρ. Thus, these states perfectly suffice. A naive dynamic programming algorithm (for the decision or optimization problems) would then have a running time of $(\sigma_{\max} + \rho_{\max} + 2)^{2\mathsf{tw}} \cdot |G|^{O(1)}$. Although not explicitly stated, the algorithm by Telle and Proskurowski [TP93] works for these cases and yields precisely this running time.

Van Rooij et al. [vBR09] dramatically improve upon this naive bound, showing that the problems can actually be solved in time $(\sigma_{\max} + \rho_{\max} + 2)^{\mathsf{tw}} \cdot |G|^{O(1)}$ by optimizing the computations in the join-node of the tree decomposition. The algorithm was subsequently improved [van20, van21] by optimizing polynomial factors in the running time. We do not go into the details regarding these improvements, but note that one of the introduced techniques is vital for the quick algorithm for the case where σ, ρ are periodic with the same period.

Focke et al. [FMI+23a, FMI+23b, FMI+23c] consider the problem of (σ, ρ)-GenDomSet for finite or cofinite sets σ and ρ parameterized by treewidth. In particular, one of their main goals initially was showing that the previously known algorithms are optimal under the SETH[2], which turned out to be only partially true. More concretely, they call a set τ m-structured if there exists a constant C such that for all $x \in \tau$ the property $x \equiv_{\mathrm{m}} C$

[2] personal communication with the authors

holds. The pair of sets (σ, ρ) is m-structured if both σ and ρ are m-structured. Observe that any subset of the integers is trivially 1-structured. Partially building upon the algorithm by van Rooij [van20], they show that the decision, optimization and counting problem can be solved in time

1. $(\sigma_{\max} + \rho_{\max} + 2)^{\mathsf{tw}} \cdot |G|^{O(1)}$ if (σ, ρ) is not m-structured for any integer $m \geq 2$,
2. $(\sigma_{\max} + 2)^{\mathsf{tw}} \cdot |G|^{O(1)}$ if $\sigma_{\max} = \rho_{\max}$ is even and (σ, ρ) is 2-structured but not m-structured for any integer $m \geq 3$, and
3. $(\max(\sigma_{\max}, \rho_{\max}) + 1)^{\mathsf{tw}} \cdot |G|^{O(1)}$ in all other cases, that is, when (σ, ρ) is m-structured for some integer $m \geq 3$, or (σ, ρ) is 2-structured with $\sigma_{\max} \neq \rho_{\max}$, or (σ, ρ) is 2-structured with $\sigma_{\max} = \rho_{\max}$ being an odd integer.

Compared to the previously known algorithms, this is sometimes an exponential improvement in the running time. Continuing with their original goal, they show that these running times are essentially optimal for the decision problem under the SETH when σ and ρ are finite (and $0 \notin \rho$). Moreover, the running times are proven to be tight under the SETH for the counting problem, even if cofinite sets are allowed. Lastly, it should be mentioned that they provide an algorithm based on representative sets that can, in certain cases, solve the decision and optimization problems even quicker when cofinite sets are in play.

Chapelle [Cha10] considers the problem for sets that are not necessarily finite or cofinite. They show that the decision problem is $W[1]$-hard for infinitely many cases of recursive sets σ and ρ. Considering that domination-like problems of this type tend to be FPT parameterized by the treewidth, this is certainly an interesting and relevant result that gets us closer to a full dichotomy of the complexity of the problem. It is also shown that the problem admits an algorithm running in time $s^{\mathsf{tw}} \cdot |G|^{O(1)}$ when σ or ρ are ultimately periodic, where s is a constant that depends on the number of states of minimal automata that enumerate σ and ρ. Hence, no explicit running time bound is provided in this case, and the question of how low the constant in the base of the running time can go is still open in most cases.

The (σ, ρ)-GenDomSet problem has also been studied for other parameterizations. For instance, the parameter *boolean-width* was considered in the work of Adler et al. [ABR+10]. They provide algorithms for the case of σ and ρ being finite or cofinite sets. Bodlaender et al. [BvvV10] consider the problem for finite or cofinite sets σ and ρ and the parameter *branchwidth*.

Finally, we would like to point out that also extensions and variations of the already extremely expressive (σ, ρ)-GenDomSet problem were studied in the past. For instance, Jaffke et al. [JKST19] study the problem of (σ, ρ)-GenDomSet in which a different notion of neighborhood is used. Concretely, for any vertex v in a graph G, they define the ball of radius r around v as the set $\{w \in V(G) \setminus \{v\} \mid dist_G(v, w) \leq r\}$, where $dist_G(v, w)$

is the length of the shortest path from v to w. Using the concept, the definition of (σ, ρ)-sets can naturally be extended by simply replacing the standard neighborhood with the ball of radius r around v. The authors prove that the decision and optimization problems of this generalization are in XP parameterized by the *mim-width* of the input graph when σ, ρ are finite or cofinite. Complementing these results, they prove that certain problem variants are $W[1]$-hard parameterized by the solution size plus the mim-width for specific cases of σ, ρ.

Heggernes and Telle [HT98] study the (k, σ, ρ)-PARTITION problem. In this problem, a solution consists of a partition of the vertex set of the input graph, such that the partition consists of k (possibly empty) blocks, and each block is itself a (σ, ρ)-set. They consider a variety of these problems, and provide cutoff values for k. That is, they show that, for certain fixed cases of σ, ρ, the problem (k, σ, ρ)-PARTITION is NP-hard, but the problem (k', σ, ρ)-PARTITION can be solved in polynomial time for all non-negative integers $k' < k$.

2.4.1 Lights Out

We now point our attention to two specific cases of (σ, ρ)-GENDOMSET that served as one of the motivations for the thesis. In the Lights Out game, originally distributed by Tiger Electronics, the player is given a board containing 25 lightbulbs arranged on a 5×5 grid (see e.g. [DW01]). Each lightbulb can either be off, or on, and each lightbulb has an associated switch. Unlike a usual switch, toggling it does not only change the state of the associated lightbulb, but also changes the state of all non-diagonal neighboring lightbulbs. Initially, the player is presented with a board in which some (potentially all) lightbulbs are turned on, and cleverly using the switches to turn off all lights is the goal.

As illustrated in Chapter 1 and mentioned within [AF98], it does not make sense to use a switch more than once, and the order in which the switches are used does also not matter. A solution to the problem can thus be represented by a set of switches that need to be pressed. A lamp that is initially turned on must be turned off after all switches of the solution are pressed, thus, its state must be changed an odd number of times. Hence, the solution must select an odd number of switches from the neighborhood of the switch, where the neighborhood also includes the switch itself. Similarly, a lamp that is initially turned off requires an even number of neighboring selected switches in order to stay off in the end.

It is easy to see that we can model the game in a graph-theoretic manner, such that it is played on a 5×5 grid graph, where the vertices correspond to the lamps. When the initial configuration is so that every lamp is turned on, we are exactly looking for a (σ, ρ)-set of the graph for $\sigma = \{0, 2, \dots\}$ and $\rho = \{1, 3, \dots\}$. Naturally, the game can be played on arbitrary graphs. We call the resulting problem REFL-ALLOFF since each switch also changes the state of the associated lamp. When using the open neighborhood instead of the closed neighborhood, we call the problem ALLOFF, and solutions to this problem are (σ, ρ)-sets where $\sigma = \rho = \{1, 3, \dots\}$. Let us stick to the problem in which all lights are initially turned on for now, we will later see that a proposed algorithm can easily also handle the other cases.

Sutner [Sut89] showed that any graph has a solution for the Refl-AllOff problem. Hence, the decision problem is trivial for this case. For the AllOff problem it is easily observed that the same property does not hold. For instance, a graph that consists of a single vertex does not have a solution. Still, it is easy to see and mentioned multiple times in the literature (see e.g. [Sut89, GKTZ95, GKT97, AF98, HKT00b]) that one must only solve the set of equations $A \cdot \vec{x} = \vec{1}$ for $\vec{x}$, where $\vec{1}$ is the 1-vector of length n, $\vec{x}$ a vector of n variables, A the adjacency matrix of the graph, and the set of equations is over the field $\mathbb{F}_2$. As one can determine a solution in cubic time, the decision problem of AllOff is also not too difficult.

Naturally, the player wants to press as few switches as possible to solve the problem, and hence, solving the minimization problem is a next natural goal. In contrast to the tractability of the decision problem, the minimization problem and maximization problem of AllOff and Refl-AllOff are NP-hard [Sut88, CGK01, HKT00b].

With regard to parameterization by treewidth, the first algorithm for the optimization problems we know of is due to Gassner and Hatzl [GH08], and has a running time of $2^{3\mathsf{tw}} \cdot |G|$. This was subsequently improved by Greilhuber et al. [GSW24, Section 4]. Their results utilized ideas by Focke et al. [FMI$^+$23b] to show that the minimization and maximization problems can be solved in time $2^{\mathsf{tw}} \cdot |G|^{O(1)}$, which is a large improvement.

2.4.2 Algorithm for (σ, ρ)-GenDomSet and Periodic Sets

The problems AllOff and Refl-AllOff show that there are natural, interesting problems of (σ, ρ)-GenDomSet where σ and ρ are neither finite nor cofinite. This suggests that taking a closer look at such a setting is reasonable. Concretely, Greilhuber et al. [GSW24] consider the problem of (σ, ρ)-GenDomSet where σ and ρ are both periodic sets with the same period $\mathrm{m} \geq 2$. A set τ is periodic with period m if it is a residue class modulo m, that is, $\tau = \{n \in \mathbb{N} \mid n \equiv_{\mathrm{m}} k\}$ for some positive integer m and some integer k with $0 \leq k < \mathrm{m}$. This setting covers AllOff and Refl-AllOff by choosing specific residue classes modulo two. Their algorithm admits a running time of $\mathrm{m}^{\mathsf{tw}} \cdot |V(G)|^{O(1)}$, and utilizes multiple non-trivial improvement techniques first used by Focke et al. [FMI$^+$23b]. As this is the algorithm the lower bound part of the thesis matches, we will now cover its central ideas. It is understood that the remainder of this section presents work that was already presented in [GSW24], and in particular builds upon ideas of [FMI$^+$23b].

At its core, the algorithm is simply a classical dynamic programming algorithm operating on tree decompositions. However, several major obstructions need to be overcome. Since we consider the problem of (σ, ρ)-GenDomSet where both σ and ρ are residue classes modulo m for some $\mathrm{m} \geq 2$, we naturally have m states for selected vertices, and m states for unselected vertices. This suffices since a vertex that has x selected neighbors behaves exactly the same as a vertex that has $x + \mathrm{m}$ selected neighbors. To make this more concrete, we formally define the states we keep track of in the algorithm.

Definition 4 (Vertex states; See [GSW24, Definition 3.3])**.** *We define the ρ states to be $\mathbb{R} = \{\rho_0, \ldots, \rho_{\mathrm{m}-1}\}$ and the σ states to be $\mathbb{S} = \{\sigma_0, \ldots, \sigma_{\mathrm{m}-1}\}$. The set of all states is $\mathbb{A} = \mathbb{R} \cup \mathbb{S}$.*

Initially, one might be tempted to think that the fact that we have 2m states available for each vertex should lead to a lower bound of $(2\mathrm{m} - \varepsilon)^{\mathsf{tw}} \cdot |V(G)|^{O(1)}$. Surprisingly, this is not the case.

For our proof sketch, we require the notion of graphs with portals and partial solutions.

Definition 5 (Graph with portals; [FMI+23b, Section 3.2])**.** *A graph with portals is a pair (G, U) consisting of a graph G and a set $U \subseteq V(G)$.*

Intuitively, for a graph with portals (G, U), the set U will usually be some form of "interface" to a larger graph, of which G is a subgraph. For instance, it may be the case that we have a node of a nice tree decomposition t at hand, and consider the graph with portals $(G[V_t], X_t)$, where V_t is the set of all vertices introduced in and below node t. Given a graph with portals (G, U), we are then interested in vertex subsets that already fulfill the constraints for the vertices in $V(G) \setminus U$, but not necessarily the constraints for the vertices in U. For the case of a tree decomposition, this corresponds to the fact that, during the dynamic program, all vertices in $V_t \setminus X_t$ must already have their constraints satisfied, while the vertices in the bag might receive more selected neighbors as we proceed.

Definition 6 (Partial solution; [FMI+23b, Definition 3.3])**.** *A partial solution with regard to a graph with portals (G, U) (and sets σ, ρ) is a set $S \subseteq V(G)$ such that*

1. *For all $v \in S \setminus U$ we have $|N(v) \cap S| \in \sigma$,*
2. *for all $v \in V(G) \setminus (S \cup U)$ we have $|N(v) \cap S| \in \rho$.*

That is, the set S already respects the degree constraints of σ and ρ for all vertices not in U.

Now, given a graph with portals (G, U), we want to associate partial solutions with strings that describe the vertex states within U.

Definition 7 (Compatible strings; Modification of [FMI+23b, Definition 3.4], [GSW24, Definition 3.4])**.** *Let (G, U) be a graph with portals, and σ, ρ be two periodic sets with period* m. *String $x \in \mathbb{A}^U$ is compatible with (G, U) if there is a partial solution $S_x \subseteq V(G)$ such that, for all $v \in U \cap S$ we have $x[v] = \sigma_i$ and $i = |N_G(v) \cap S_x| \bmod \mathrm{m}$, and for all $v \in U \setminus S$ we have $x[v] = \rho_i$ and $i = |N_G(v) \cap S_x| \bmod \mathrm{m}$.*

Partial solution S_x is also said to be a witness or witnessing solution for x.

Essentially, any $x \in \mathbb{A}^U$ describes a potential partial solution via the states of the vertices of U, and x is compatible if a matching partial solution actually exists.

Within the dynamic programming algorithm, we naturally want to keep track of strings that are compatible with $(G[V_t], X_t)$ for all nodes t of the tree decomposition. We also call this set of strings the *realized language*. The related notions *providers* and *realizers* are of special importance for the lower bound.

Definition 8 (Realized language, L-realizer and L-provider; [FMI+23c, Definition 3.9], [GSW24, Definition 3.5])**.** *Given a graph with portals (G, U), its realized language is defined as*

$$L(G, U) = \{x \in \mathbb{A}^U \mid x \text{ is compatible with } (G, U)\}.$$

For a language $L \subseteq \mathbb{A}^U$, (G, U) is a L-realizer if $L = L(G, U)$, and a L-provider if $L \subseteq L(G, U)$.

After this heap of formal definitions, we can finally explain the intuition behind why we do not have to track as many states as expected. It turns out that, for a graph with portals (G, U) even though we have $|\mathbb{A}^U| = (2\mathrm{m})^{|U|}$, at most $\mathrm{m}^{|U|+1}$ of those strings can actually be compatible with the graph. That is, the realized language of (G, U) is significantly smaller than the naive bound would suggest. Hence, one does not need to keep track of as many solutions as expected at first.

We continue by formally proving the bound on the size of the realized language of a graph with portals. For this purpose, we decompose strings over $\mathbb{A}^U$ into a selection vector, and a weight vector.

Definition 9 (Selection and weight vectors; See [FMI+23b, Definition 4.2], [GSW24, Definition 4.1])**.** *Given a string $x \in \mathbb{A}^n$, the selection vector $\vec{\sigma}(x) \in \{0, 1\}^n$ is defined as the vector where $\vec{\sigma}(x)[i]$ is 1 if and only if $x[i] \in \mathbb{S}$.*

The weight vector $\vec{w}(x) \in [0, \mathrm{m} - 1]^n$ is defined as the vector with $\vec{w}[i] = c$, where $x[i] \in \{\sigma_c, \rho_c\}$.

Next, we prove an important property of strings of realized languages, that will eventually allow us to prove the size bound.

Lemma 1 ([GSW24, Lemma 4.3])**.** *Let σ and ρ be two periodic sets with the same period $\mathrm{m} \geq 2$, and let (G, U) be a graph with portals. Consider two strings $x, y \in \mathbb{A}^U$, such that there are witnessing solutions S_x, S_y for x, respectively y, with $|S_x \setminus U| \equiv_{\mathrm{m}} |S_y \setminus U|$. Then, $\vec{\sigma}(x) \cdot \vec{w}(y) \equiv_{\mathrm{m}} \vec{\sigma}(y) \cdot \vec{w}(x)$.*

Proof. The central ideas behind the lemma and proof should be attributed to Focke et al. [FMI+23b, Lemma 4.3].

Let (G, U), x, y and S_x, S_y be as in the statement of the lemma. We prove the statement by counting the edges from S_x to S_y in two different ways. For this purpose, define

$E(X, Y) = \{uv \in E(G) \mid u \in X, v \in Y\}$ for $X, Y \subseteq V(G)$. We examine $E(S_x, S_y)$. A vertex in S_x is either

1. part of U, or
2. part of $S_y \setminus U$, or
3. not part of U and not part of S_y.

If the vertex is part of $S_y \setminus U$, then the number of neighbors it has that are in the set S_y must be in σ, because S_y is a partial solution. In particular, this number must be congruent to $\min \sigma$ modulo m.

If the vertex is neither part of U nor of S_y, then its number of selected neighbors in S_y must be in ρ, and in particular congruent to $\min \rho$ modulo m.

Finally, if the vertex v is part of U, the weight vector entry $\vec{w}(y)[v]$ describes precisely how many neighbors v has in S_y.

Combining these observations, we obtain

$$\begin{aligned}|E(S_x, S_y)| \equiv_{\mathrm{m}} &\min \rho \cdot |S_x \setminus (U \cup S_y)| \\ &+ \min \sigma \cdot |(S_x \cap S_y) \setminus U| \\ &+ \vec{\sigma}(x) \cdot \vec{w}(y) \\ &= \min \rho \cdot (|(S_x \setminus U)| - |S_x \cap S_y| + |S_x \cap S_y \cap U|) \\ &+ \min \sigma \cdot |(S_x \cap S_y) \setminus U| \\ &+ \vec{\sigma}(x) \cdot \vec{w}(y).\end{aligned}$$

Using the same strategy for $E(S_y, S_x)$, we get

$$\begin{aligned}|E(S_y, S_x)| \equiv_{\mathrm{m}} &\min \rho \cdot (|(S_y \setminus U)| - |S_y \cap S_x| + |S_y \cap S_x \cap U|) \\ &+ \min \sigma \cdot |(S_y \cap S_x) \setminus U| \\ &+ \vec{\sigma}(y) \cdot \vec{w}(x).\end{aligned}$$

Finally, by utilizing our assumption that $|S_x \setminus U| \equiv_{\mathrm{m}} |S_y \setminus U|$, we see that

$$\vec{\sigma}(x) \cdot \vec{w}(y) \equiv_{\mathrm{m}} \vec{\sigma}(y) \cdot \vec{w}(x)$$

as desired. □

We can later show that a language $L \subseteq \mathbb{A}^n$ has bounded size whenever

$$\vec{\sigma}(x) \cdot \vec{w}(y) \equiv_{\mathrm{m}} \vec{\sigma}(y) \cdot \vec{w}(x)$$

holds for all $x, y \in L$. We call such languages L *sparse* (see also [GSW24, Definition 4.2] and [FMI+23b, Page 16]). The lemma above allows us to utilize this information to bound the size of any realized language. Note that we can decompose any realized language into m bins of strings having witness solutions of a specific size modulo m, which only leads to an overhead of factor m when considering the full language.

To proof the size bound on sparse languages, we need the concept of σ-defining sets which was also used by Focke et al. [FMI+23b].

Definition 10 (σ-defining set; See [GSW24, Definition 4.5],[FMI+23b, Definition 4.7]). *Let $X \subseteq \{0,1\}^n$. Then, $S \subseteq [1,n]$ is a σ-defining set for X if S is an inclusion-wise minimal set such that for all $u, v \in X$ we have $u[S] = v[S]$ implies $u = v$.*

Greilhuber et al. [GSW24, Lemma 4.7] show that such a set S can be computed in time $O(|X| \cdot n^3)$, which is actually a relatively easy task and requires no advanced techniques.

The crucial property of σ-defining sets $S \subseteq [1,n]$ is that their complement $\overline{S} = [1,n] \setminus S$ characterizes the *weight* vectors uniquely. This is made more precise in the following lemma.

Lemma 2 ([GSW24, Lemma 4.8]). *Let σ and ρ be two periodic sets with the same period $\mathrm{m} \geq 2$. Let $L \subseteq \mathbb{A}^n$ be a sparse language with a σ-defining set S for $\vec{\sigma}(L) = \{\vec{\sigma}(x) \mid x \in L\}$. Then, for any two strings $x, y \in L$ with $\vec{\sigma}(x) = \vec{\sigma}(y)$ we have*

$$\vec{w}(x)[\overline{S}] = \vec{w}(y)[\overline{S}] \quad \textit{implies} \quad x = y.$$

Proof. The central ideas of the lemma and proof are attributed to Focke et al. [FMI+23b, Lemma 4.9].

Let L and S be like in the statement of the lemma, and $x, y \in L$ with $\vec{\sigma}(x) = \vec{\sigma}(y)$ and $\vec{w}(x)[\overline{S}] = \vec{w}(y)[\overline{S}]$. Consider an arbitrary position $i \in S$. As i is a position in S and S is a σ-defining set, we know that there must exist vectors $w_{1,i}$ and $w_{0,i} \in \vec{\sigma}(L)$ such that $w_{1,i}[S \setminus i] = w_{0,i}[S \setminus i]$ and $w_{1,i}[i] = 1, w_{0,i}[i] = 0$.

First, we show that $(\vec{w}(x) - \vec{w}(y)) \cdot w_{1,i} \equiv_{\mathrm{m}} 0$. For this purpose, consider that $w_{1,i} \in \vec{\sigma}(L)$, and hence, there is a $z \in L$ such that $\vec{\sigma}(z) = w_{1,i}$. Using the property that L is a sparse language, we see that $\vec{\sigma}(z) \cdot \vec{w}(x) \equiv_{\mathrm{m}} \vec{\sigma}(x) \cdot \vec{w}(z)$ and $\vec{\sigma}(z) \cdot \vec{w}(y) \equiv_{\mathrm{m}} \vec{\sigma}(y) \cdot \vec{w}(z)$. Using the assumed property that $\vec{\sigma}(x) = \vec{\sigma}(y)$, we obtain $\vec{\sigma}(z) \cdot \vec{w}(x) \equiv_{\mathrm{m}} \vec{\sigma}(z) \cdot \vec{w}(y)$ which yields the claim after simple rearrangements. We can show that $(\vec{w}(x) - \vec{w}(y)) \cdot w_{0,i} \equiv_{\mathrm{m}} 0$ the same way.

Using these properties, we now see that

$$\begin{aligned}(\vec{w}(x) - \vec{w}(y)) \cdot (w_{1,i} - w_{0,i}) &= (\vec{w}(x) - \vec{w}(y)) \cdot w_{1,i} - (\vec{w}(x) - \vec{w}(y)) \cdot w_{0,i} \\ &\equiv_{\mathrm{m}} 0 - 0 \\ &= 0.\end{aligned}$$

Now, we have $0 \equiv_{\mathrm{m}} (\vec{w}(x) - \vec{w}(y)) \cdot (w_{1,i} - w_{0,i})$. Furthermore, by $\vec{w}(x)[\overline{S}] = \vec{w}(y)[\overline{S}]$ and $w_{1,i}[S \setminus i] = w_{0,i}[S \setminus i]$, we also have

$$(\vec{w}(x) - \vec{w}(y)) \cdot (w_{1,i} - w_{0,i}) = (\vec{w}(x)[i] - \vec{w}(y)[i]) \cdot (w_{1,i}[i] - w_{0,i}[i]).$$

Altogether we now have

$$\begin{aligned}0 &\equiv_{\mathrm{m}} (\vec{w}(x) - \vec{w}(y)) \cdot (w_{1,i} - w_{0,i}) \\ &= (\vec{w}(x)[i] - \vec{w}(y)[i]) \cdot (w_{1,i}[i] - w_{0,i}[i]) \\ &= (\vec{w}(x)[i] - \vec{w}(y)[i]) \cdot (1 - 0) \\ &= (\vec{w}(x)[i] - \vec{w}(y)[i]).\end{aligned}$$

And as this implies that $\vec{w}(x)[i] \equiv_{\mathrm{m}} \vec{w}(y)[i]$, and entries of the weight vectors range in $[0, \mathrm{m} - 1]$, we actually have $\vec{w}(x)[i] = \vec{w}(y)[i]$, concluding the proof. □

Using these facts, we can already bound the size of sparse languages, and therefore the size of any realized language.

Lemma 3 ([GSW24, Lemma 4.9])**.** *Let σ and ρ denote two periodic sets with the same period $\mathrm{m} \geq 2$. Every sparse language $L \subseteq \mathbb{A}^n$ satisfies $|L| \leq \mathrm{m}^n$.*

Proof. We restate the proof of Greilhuber et al. [GSW24].

Let $L \subseteq \mathbb{A}^n$ be a sparse language, and S a σ-defining set for $\vec{\sigma}(L) = \{\vec{\sigma}(x) \mid x \in L\}$. A string $x \in L$ is uniquely determined by its σ vector and its weight-vector.

Count the elements of L by first fixing a σ-vector. We know that $|\vec{\sigma}(L)| \leq 2^{|S|}$ per the definition of σ-defining sets. Then, we know that the elements in $\overline{S}$ uniquely determine the weight-vector of a string per Lemma 2, and as we have already fixed the σ-vector, the full string. We conclude that

$$|L| \leq 2^{|S|} \cdot \mathrm{m}^{|\overline{S}|} \leq \mathrm{m}^n.$$

□

The lemma above can be seen as the key property that enables the quick algorithm. The only remaining challenge is that also the join operation of the dynamic program must be done efficiently. For the join operation, the algorithm iterates over the intersection of the σ-vectors of the languages that should be joined, and then joins them by combining the weight vectors of strings with the same σ-vector. For this purpose, a quick convolution algorithm due to van Rooij [van20] is employed. It is not sufficient to directly employ it, though. While the size of a sparse language is small, each weight-vector can stem from a fairly large space, concretely of size m^n. To make the convolution quick enough, it is necessary to move to a significantly smaller space.

Luckily, computing the convolution sufficiently fast is possible by using a *compression* of the weight vectors. Concretely, given a σ-defining set S, the positions of $\overline{S}$ uniquely determine the remaining positions of a weight-vector in such a way, that they can be fully recovered even when they are completely dropped. Thus, the compression of weight-vectors is simply the projection to the positions on $\overline{S}$, and these then stem from a space of size $\mathrm{m}^{|\overline{S}|}$, which is small enough.

The final algorithm is then a simple dynamic program which additionally utilizes this fast join operation. The quick running time is obtained because (1) it could be proven that the number of strings tracked at each bag of the tree decomposition is small due to Lemma 3 and (2) using the compression, the join-nodes can be handled in the required running time. The final Theorem expressing the obtained running time is repeated here.

Theorem 2 ([GSW24, Main Theorem 1])**.** *Let σ, ρ be periodic sets with the same period $\mathrm{m} \geq 2$. Then, in time $\mathrm{m}^{\mathsf{tw}} \cdot |G|^{O(1)}$ we can decide simultaneously for all s if a given input graph G has a (σ, ρ)-set of size s when a tree decomposition of width tw is provided with the input.*

As we can actually decide whether a solution of a specific size exists for all possible sizes at the same time, this algorithm not only solves the decision, but also the minimization and maximization versions of (σ, ρ)-GenDomSet.

With a simple modification of the dynamic program, one can also handle the case where an additional shift-vector is part of the input. The shift-vector essentially provides starting values for the number of selected neighbors of the vertices, that is, even though a vertex has, say, 0 selected neighbors in a solution, its count of selected neighbors might be 1, because a shift of 1 for this particular vertex is defined in the input. Using this notion of shift-vectors, one can solve the Lights Out problem (both for closed and open neighborhoods) for arbitrary starting configurations [GSW24].

CHAPTER 3

Intermediate Lower Bound

We can now finally proceed to the main contribution of the thesis, that is, establishing the lower bound for (σ, ρ)-GenDomSet with sets σ, ρ having the same period m. The original content of this section is also presented in [GSW24, Section 5] in a similar manner.

The central idea behind the reduction from k-SAT to (σ, ρ)-GenDomSet is that we first reduce to q-CSP-B (this reduction due to Lampis [Lam20] was covered in Theorem 1), then reduce to a hybrid between q-CSP-B and (σ, ρ)-GenDomSet, and then finally to (σ, ρ)-GenDomSet. The general strategy was previously used by Curticapean and Marx [CM16], Marx et al. [MSS21, MSS22], and in particular also utilized by Focke et al. [FMI$^+$23c] for showing their lower bounds for (σ, ρ)-GenDomSet (for finite and cofinite sets σ, ρ). In this chapter, we cover the reduction to the intermediate hybrid problem.

Given that Focke et al. [FMI$^+$23c] present a lower bound for the case where σ and ρ are finite, it is tempting to think that one can easily reduce from the problem with finite sets σ and ρ, or at least use the construction by Focke et al. as it is. Unfortunately, this turns out to not be the case, as the problem with periodic sets sometimes behaves quite different from the problem with finite sets. For instance, the approach by Focke et al. heavily exploits the fact that σ and ρ have a largest element, which is no longer the case for periodic sets. Thus, while the general strategy for proving the lower bound remains the same and, especially in this section, a good deal of concepts and ideas were reused by us, also many non-trivial adaptions need to be made, and new gadgets have to be created to make it work for the considered case.

The Hybrid Problem

Before getting into the details of the reduction to the hybrid problem, we must first define it. The problem is a natural extension of (σ, ρ)-GenDomSet to *graphs with relations*, which is the concept defined next.

Definition 11 (Graph with relations; [FMI+23c, Definition 4.1])**.** *A graph with relations $G = (V, E, \mathcal{C})$ is a triple, where (V, E) is a graph, and $\mathcal{C}$ a set of constraints. Concretely, each $C \in \mathcal{C}$ is a pair $(\mathsf{scp}(C), \mathsf{acc}(C))$, where $\mathsf{scp}(C)$ is a subset of V, and $\mathsf{acc}(C) \subseteq 2^{\mathsf{scp}(C)}$ is a $|\mathsf{scp}(C)|$-ary relation that specifies the allowed selections within $\mathsf{scp}(C)$. We also say that C observes $\mathsf{scp}(C)$.*

The size of G is defined to be $|G| = |V| + \sum_{c \in \mathcal{C}} |\mathsf{acc}(C)|$. Slightly abusing notation, we may not distinguish between G and the graph (V, E), and refer to both objects as G depending on the context.

The definition of graph with relations already suggests how the hybrid problem, called (σ, ρ)-GenDomSet$^{\text{Rel}}$, will extend (σ, ρ)-GenDomSet, namely by taking a graph with relation as input, and respecting its constraints. For this purpose, we first define the notion of (σ, ρ)-sets on such graphs.

Definition 12 ((σ, ρ)-set of a graph with relations; [FMI+23c, Definition 4.3])**.** *Let $G = (V, E, \mathcal{C})$ be a graph with relations. A set $S \subseteq V$ is a (σ, ρ) set of G if it is a (σ, ρ)-set of (V, E), and $S \cap \mathsf{scp}(C) \in \mathsf{acc}(C)$ for all $C \in \mathcal{C}$.*

The definition of the hybrid problem (σ, ρ)-GenDomSet$^{\text{Rel}}$ naturally follows. Note that the definition stems from [FMI+23c, Definition 4.8].

> (σ, ρ)-GenDomSet$^{\text{Rel}}$
> **Input:** Graph with relations $G = (V, E, \mathcal{C})$
> **Question:** Is there a (σ, ρ)-set of G?

As we parameterize by structural parameters which are only defined on graphs thus far, we must also define them on graphs with relations. The definitions are chosen in a way that allows us to later move from a graph with relations to a regular graph without increasing the parameter too much.

Definition 13 (Width measures for graphs with relations; [FMI+23c, Definition 4.4])**.** *Let $G = (V, E, \mathcal{C})$ be a graph with relations. Let $\hat{G}$ be the graph we obtain from (V, E) by introducing a complete set of edges on $\mathsf{scp}(C)$ for all $C \in \mathcal{C}$. The treewidth of a graph with relations G is the treewidth of the graph $\hat{G}$. Analogously, we define tree decomposition, path decomposition, and pathwidth of G as the corresponding concepts in the graph $\hat{G}$.*

An important property of this definition is that for any relation and any tree or path decomposition, there must be a bag of the decomposition that contains all vertices of the relation scope. The reason for this is that for any clique of a graph, any such decomposition must contain a bag which contains all vertices of the clique (see e.g. [BM93]).

3.1 The Idea

We now present the main ideas behind the reduction from q-CSP-B to the hybrid problem (σ, ρ)-GenDomSet$^{\text{Rel}}$. We are looking for a lower bound for the problem parameterized by treewidth, however, we can actually prove the bound for the parameter pathwidth, which implies the bound for treewidth. As we are aiming for a lower bound of the form $(\mathrm{m} - \varepsilon)^{\mathsf{pw}} \cdot |G|^{O(1)}$, we reduce from q-CSP-B where $B = \mathrm{m}$, that is, from q-CSP-m.

There are four main challenges that the reduction must overcome, we will list these challenges and sketch how they are solved next.

Low Pathwidth

When provided with a q-CSP-m instance on n variables and ℓ constraints, it must be ensured that the pathwidth of the output graph is only marginally larger than n. To achieve this, the output graph will follow the general structure of an $n \times \ell$ grid. The output graph contains one row for each variable and one column for each constraint. Note that, since the output graph is a graph with relations, we must also ensure that each relational constraint does not observe too many vertices.

Encoding the Assignment

One of the central tasks of the reduction is bridging the gap between a constraint satisfaction problem and a graph problem. Each variable of the input instance can take up to m values, and we must somehow encode these values by states of vertices in the output graph. For this purpose, so-called *information* vertices are introduced, and we have one such vertex for every row and every column of the output graph. The state of an information vertex (in a solution) is determined by the number of selected neighbors it has. The construction ensures that each information vertex can have m different states, and hence, we achieve a direct correspondence between the states of information vertices and the values that the variables can take. The neighbors are provided to the information vertices through specific gadgets which we call *managers*. Their creation will be described in the first step of the reduction.

Consistency of the Assignment

To keep the pathwidth low, we have ℓ different information vertices for each variable of the input q-CSP-m instance. It must be guaranteed that all of these different information vertices have the same state, otherwise, a solution to the q-CSP-m instance might not correspond to a variable-assignment at all. Luckily, this property can be achieved by introduction *consistency relations* with a sufficiently small scope.

Encoding the Constraints

Finally, we must also ensure that a solution to the (σ, ρ)-GenDomSet$^{\text{Rel}}$ problem not only corresponds to a variable assignment of the input instance, but actually to a satisfying assignment. For this purpose, we must introduce the constraints of the q-CSP-m instance into the output instance in some manner. To do this, we can add ℓ additional relations to the output instance, one relation for each constraint of the input instance. Each such *constraint relation* will observe vertices of a different column, and ensure that the states of the information vertices correspond to assignments that satisfy the constraint of the column.

3.2 Managers

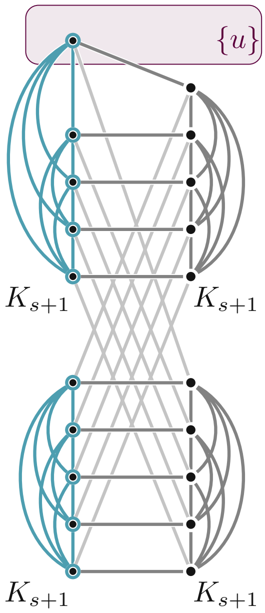
Now, we will proceed with the actual construction. One important module that is used within it are the managers, which are the subgraphs of the output instance that provide neighbors to the information vertices. In [GSW24], the managers provided by Focke et al. [FMI$^+$23c] are used in a black-box fashion, which is perfectly sufficient. To keep this thesis self-contained, we will present how they can be created here. This choice also simplifies some definitions and proofs. We first provide the formal definition of a manager. This definition is a simplified version of Definition 4.7 in Focke et al. [FMI$^+$23c] adapted to periodic sets, which is sufficient for our purposes.

Definition 14 (A-manager; Compare [FMI$^+$23c, Definition 4.7])**.** *Consider two sets σ and ρ that are both periodic with period* m*. For a set $A \subseteq \mathbb{A}$, an* A-manager *is an infinite family $((G_\ell, U_\ell))_{\ell \geq 1}$ of pairs (G_ℓ, U_ℓ) such that*

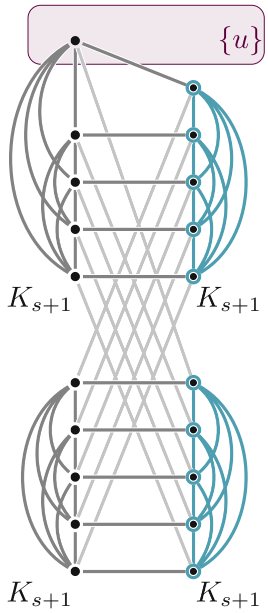
- *G_ℓ is a graph and*
- *$U_\ell = \{u_1, \dots, u_\ell\} \subseteq V(G_\ell)$ is a set of ℓ distinguished vertices.*

Moreover, there is a non-negative integer b (that depends only on σ and ρ) such that the following holds for every $\ell \geq 1$:

- *The vertices from $V(G_\ell) \setminus U_\ell$ can be partitioned into 2ℓ vertex-disjoint sets $B_1, \dots, B_\ell$ and $\overline{B}_1, \dots, \overline{B}_\ell$ (called* blocks*), such that*
  - *$|B_i| \leq b$ and $|\overline{B}_i| \leq b$ for all $i \in [1, \ell]$,*
  - *$N(u_i) \subseteq B_i \cup \overline{B}_i$ for all $i \in [1, \ell]$,*
  - *there are no edges between two distinct blocks.*
- *Each $x \in A^\ell \subseteq \mathbb{A}^\ell$ is* managed *in the sense that there is a (σ, ρ)-set S_x of G_ℓ such that for all $i \in [1, \ell]$:*
  - *If $x[i] = \sigma_s$, then $u_i \in S_x$. Moreover, u_i has exactly s neighbors in $B_i \cap S_x$ and exactly $(\min \sigma - s) \bmod \mathrm{m}$ neighbors in $\overline{B}_i \cap S_x$.*
  - *If $x[i] = \rho_r$, then $u_i \notin S_x$. Moreover, u_i has exactly r neighbors in $B_i \cap S_x$ and exactly $(\min \rho - r) \bmod \mathrm{m}$ neighbors in $\overline{B}_i \cap S_x$.*



(a) The partial solution compatible with $\sigma_{s \bmod \mathrm{m}}$.



(b) The partial solution compatible with $\rho_{r \bmod \mathrm{m}}$.

Figure 3.1: The $\{\rho_{r \bmod \mathrm{m}}, \sigma_{s \bmod \mathrm{m}}\}$-provider from Lemma 4 sketched for the case $r = 2$ and $s = 4$. The figures are adaptions of [FMI+23c, Figure 5.1 (a)].

We refer to (G_ℓ, U_ℓ) *as the* A-manager *of rank* ℓ*.*

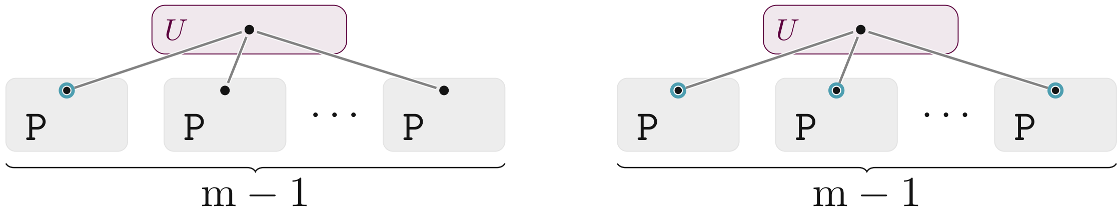
Given a manager and distinguished vertex u_i, we sometimes refer to B_i as the left block of vertex u_i and to $\overline{B}_i$ as the right block of vertex u_i.

The manager we will use will itself consist of several smaller gadgets, called providers. Recall the notion of compatible strings and providers from Definitions 7 and 8. We first show that certain providers exist. The proof is a direct consequence of a Lemma provided by Focke et al. [FMI+23c], and restated here for completeness.

Lemma 4 (Follows from [FMI+23c, Lemma 5.1])**.** *Let* σ *and* ρ *be periodic sets with period* m*, and* $r \in \rho$ *with* $r > 0$*,* $s \in \sigma$*. Then, there is a* $\{\rho_{r \bmod \mathrm{m}}, \sigma_{s \bmod \mathrm{m}}\}$*-provider* $(G, \{u\})$*.*

Proof. The proof is due to Focke et al. [FMI+23c]. The graph G consists of $2r$ cliques $X_1, \ldots, X_r, Y_1, \ldots, Y_r$ on $s+1$ vertices each. Denote the i-th vertex of clique X_j as $x_i^{(j)}$, and the i-th vertex of clique Y_j as $y_i^{(j)}$. Connect vertex $x_i^{(j)}$ to vertex $y_i^{(j)}$ for all $i \in [1, s+1]$ and all $j \in [1, r]$. This concludes the construction of G, for the portal vertex u, simply set $u = x_1^{(1)}$. The graph is visualized in Fig. 3.1.

We now argue that $(G, \{u\})$ is indeed a $\{\rho_{r \bmod \mathrm{m}}, \sigma_{s \bmod \mathrm{m}}\}$-provider. As we deal with periodic sets, we count neighbors modulo m, and in particular, it would be sufficient to



(a) The provider with an exemplary selection that leads to the state ρ_1.

(b) The provider with an exemplary selection that leads to the state ρ_{m-1}.

Figure 3.2: The provider from Lemma 5. The figures are adaptions of [FMI+23c, Figure 5.2]. The graphs P represent the copies of the provider from Lemma 4, from each copy only the portal vertex is shown.

show that there exists a solution in which u is selected and has s selected neighbors, and a solution in which u is not selected and has r selected neighbors.

Consider the set that consists of the vertices of all cliques Y_i. Then, each vertex of such a clique has exactly s neighbors that are selected. Moreover, each vertex of a clique X_i is adjacent to exactly r vertices of the Y-cliques, which is also sufficient. Hence, the described set is a (σ, ρ)-set in which u is not selected and has r selected neighbors, which is exactly what we need.

Similarly, when we select all vertices of the X-cliques, then each vertex of those cliques has s neighbors that are selected, and each vertex of the Y-cliques has r selected neighbors. We once again have a (σ, ρ)-set of G, and u is selected and has s selected neighbors itself, as required. □

This provider (G, u) is very useful, because vertex u can be both selected or unselected. Hence, if we add a vertex v to G that is unselected and make it adjacent to u, we can switch between vertex u providing a selected neighbor to v, and vertex u not providing a selected neighbor to v. Using this idea, we can immediately create a $\{\rho_0, \rho_1, \ldots, \rho_{m-1}\}$-provider, which is an extremely versatile gadget. Once again, this was already shown by Focke et al. [FMI+23c], and we restate their statement and proof in slightly different words here.

Lemma 5 (Compare [FMI+23c, Lemma 5.3])**.** *Let σ and ρ be periodic sets with the same period* m. *Then, there is a $\{\rho_0, \rho_1, \ldots, \rho_{m-1}\}$-provider.*

Proof. The proof is attributed to Focke et al. [FMI+23c].

Take the disjoint union of $m-1$ copies of the provider of Lemma 4 for arbitrary $s \in \sigma$ and $r \in \rho$ with $r > 0$. A good choice would, for instance, be $s = \min \sigma$ and $r = \min \rho + m$. Add another vertex u to the graph, and connect this vertex to the portal vertex of each of the $m-1$ copies. The provider is sketched in Fig. 3.2.

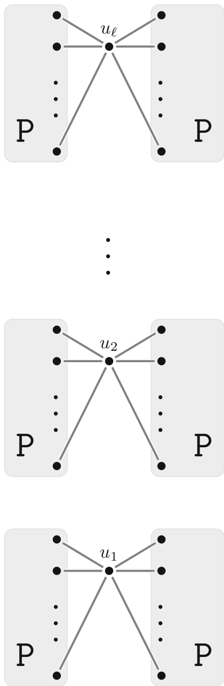


Figure 3.3: The manager from Lemma 6 of rank ℓ. The graphs P in the figure represent the copies of the $\{\rho_0, \ldots, \rho_{\mathrm{m}-1}\}$-provider from Lemma 5 (excluding their portal vertex).

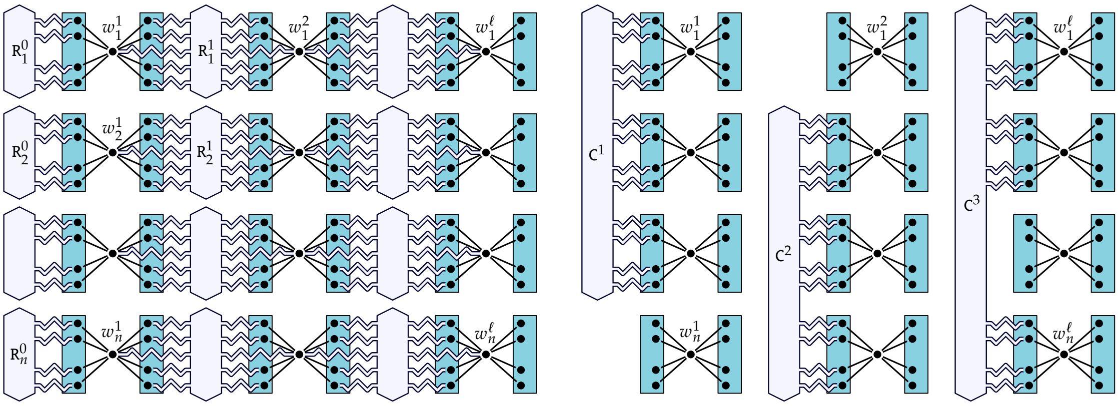
Since each of the copies has a partial solution in which the respective portal vertex is selected and a partial solution in which the respective portal vertex is not selected, we can easily see that for any $i \in [0, \mathrm{m}-1]$, there is at least one partial solution in which vertex u is unselected and receives i selected neighbors. □

Using this powerful provider, we can now construct the manager we will use in the reduction.

Lemma 6 (Compare [FMI$^+$23c, Lemma 6.1])**.** *Let σ and ρ be periodic sets with the same period* m*. Then, there is a* $\mathbb{R}$*-manager.*

Proof. The proof is due to Focke et al. [FMI$^+$23c].

Fix any $\ell \geq 1$. Add the provider from Lemma 5 to the graph ℓ times, and denote the providers as $P_1, P_2, \ldots, P_\ell$. Add the same provider ℓ more times and denote the providers as $\overline{P}_1, \overline{P}_2, \ldots, \overline{P}_\ell$. Identify the portal vertex of P_i with the portal vertex of $\overline{P}_i$ and denote the resulting vertex as u_i for all $i \in [1, \ell]$. Then, $(G_\ell, \{u_1, \ldots, u_\ell\})$ forms the desired manager of rank ℓ. The blocks of the manager are provided by setting $B_i = V(P_i) \setminus \{u_i\}$ and $\overline{B}_i = V(\overline{P}_i) \setminus \{u_i\}$ for all $i \in [1, \ell]$. Look at Fig. 3.3 for an illustration of the manager of rank ℓ.



(a) The instance G_I, of the relations only the consistency relations are shown.

(b) The instance G_I, of the relations only the constraint relations are shown.

Figure 3.4: A sketch of the output instance G_I where $\mathrm{m} = 5, n = 4$ and $\ell = 3$. A similar illustration is presented in [GSW24, Figure 2].

It is easy to see that the constructed graph is indeed the $\mathbb{R}$-manager of rank ℓ. In particular, the distinguished vertices only have neighbors in two blocks each, and the blocks are not directly connected to each other. Moreover, the size of each used provider depends only on σ and ρ, and is thus constant as these sets are fixed. Furthermore, the neighborhood of the distinguished vertices lies entirely within the associated blocks.

Then, for any $x \in \mathbb{R}^\ell$, we can manage x by selecting vertices according to a (σ, ρ)-set that does not select any distinguished vertex, and for each i, r neighbors in B_i where $x[i] = \rho_r$, and $(\min \rho - r) \bmod \mathrm{m}$ neighbors in $\overline{B}_i$. Such a selection possible because each block essentially forms a $\{\rho_0, \rho_1, \ldots, \rho_{\mathrm{m}-1}\}$-provider. □

3.3 The Reduction

Now, we have all components available that we will require for the actual reduction. Since it is relatively intricate, we will first formally define the output instance, and then prove properties about it in a second step.

3.3.1 Construction of the Graph

Let the input instance of q-CSP-m be $I = (X, \mathcal{C})$, where $X = \{x_1, \ldots, x_n\}$ and $\mathcal{C} = \{C_1, \ldots, C_\ell\}$. We will output an equivalent instance G_I of (σ, ρ)-GenDomSet$^{\text{Rel}}$, which has sufficiently low pathwidth. As mentioned earlier, the structure of the graph is grid-like. To make our notation easier to understand, we will often refer to objects by row and column, and we use the convention that the row is written as subscript index, while the column is written as superscript index.

We can now describe the construction of G_I:

- For all $i \in [1, n]$ and all $j \in [1, \ell]$, the output graph G_i contains a vertex w_i^j, which is called an *information vertex.*
- For all $j \in [1, \ell]$ there is a $\mathbb{R}$-manager M^j of rank n (provided by Lemma 6), where $\{w_i^j \mid i \in [1, n]\}$ is the set of distinguished vertices of M^j. We write B_i^j and $\overline{B}_i^j$ for the vertices of the corresponding blocks of the manager.
- For all $i \in [1, n], j \in [1, \ell - 1]$ we create a *consistency relation* $\mathtt{R}_i^j$ with scope $\overline{B}_i^j \cup B_i^{j+1} \cup \{w_i^j, w_i^{j+1}\}$, which we define in the following paragraphs.
- For all $i \in [1, n]$ we similarly create a relation $\mathtt{R}_i^0$ with scope B_i^1, which we also define in the following paragraphs.
- For all $j \in [1, \ell]$ we finally create a *constraint relation* $\mathtt{C}^j$ with scope $\bigcup_{x_i \in \mathtt{scp}(C_j)} B_i^j$, whose definition we give in the following paragraphs.

Since we have not yet defined the allowed assignments of our relations, we proceed to formally define them now. Let S be a subset of $V(G_I)$, and set $S_i^j = S \cap \mathtt{scp}(\mathtt{R}_i^j)$ for all $i \in [1, n], j \in [1, \ell - 1]$. Moreover, we denote the number of selected neighbors that vertex w_i^j has in its left block by b_i^j, that is, we define $b_i^j = |N(w_i^j) \cap B_i^j \cap S|$. We similarly define the selected neighbors in the right block as $\overline{b}_i^j = |N(w_i^j) \cap \overline{B}_i^j \cap S|$.

Relation $\mathtt{R}_i^j$ accepts set S_i^j if and only if

- w_i^j and w_i^{j+1} are both not in S, i.e., they are unselected, and
- $b_i^{j+1} = (\min \rho - \overline{b}_i^j) \bmod \mathrm{m}$.

Due to the modulo reduction the second condition implies that $b_i^{j+1} \in [0, \mathrm{m} - 1]$.

Relation $\mathtt{R}_i^0$ accepts $S_i^0 = S \cap \mathtt{scp}(\mathtt{R}_i^0)$ if and only if $b_i^1 \in [0, \mathrm{m} - 1]$. While this is anyway the case for the managers we use, we still use this relation to stay more agnostic about how the manager looks concretely.

The only definition that is still missing is that of the constraint relations. For each $j \in [1, \ell]$, denote the variables of the constraint C_j as $\mathtt{scp}(C_j) = (x_{\lambda_1}, \ldots, x_{\lambda_q})$. Then, $S \cap \mathtt{scp}(\mathtt{C}^j)$ is accepted by $\mathtt{C}^j$ if and only if $(b_{\lambda_1}^j + 1, \ldots, b_{\lambda_q}^j + 1) \in \mathtt{acc}(C_j)$. That is, the states given to the information vertices corresponding to the variables of the j-th constraint must correspond to an assignment satisfying the constraint. The only reason why we add the constant one to each selected neighbor count is that the original definition of q-CSP-m, which we use, uses variables from the range $[1, \mathrm{m}]$, whereas the selected neighbor counts range in $[0, \mathrm{m} - 1]$. See Fig. 3.4 for an illustration of the output instance G_I.

3.3.2 Properties of G_I

We will now prove the crucial properties of G_I. Concretely, we will show that the instance G_I is equivalent to the input instance, that it is sufficiently small, and that it has low pathwidth.

Lemma 7 ([GSW24, Lemma 5.5])**.** *If I is a satisfiable instance of q-CSP-*m*, then the (σ, ρ)-*GenDomSet$^{\text{Rel}}$ *instance G_I has a solution.*

Proof. We restate a slightly adapted variant of the proof of [GSW24].

Recall that the input q-CSP-m instance I contains the variables $x_1, \ldots, x_n$ and constraints $C_1, \ldots, C_\ell$. Now, assume that I is a yes-instance. Then, there exists a variable assignment π that satisfies all constraints.

We select vertices of the graph according to a (σ, ρ)-set that ensures that we have $b_i^j = \pi(x_i) - 1$ for all $i \in [1, n], j \in [1, \ell]$. Note that this is possible because $\rho_{\pi(x_i)-1} \in \mathbb{R}$, and then such a solution must exist by Definition 14. Since this selection S already provides a (σ, ρ)-set, all that remains is arguing that also all relational constraints of the (σ, ρ)-GenDomSet$^{\text{Rel}}$ instance are fulfilled.

We first examine the consistency relation $\mathtt{R}_i^j$ for $i \in [1, n], j \in [1, \ell - 1]$. We can quickly see that all information vertices are unselected, and hence w_i^j and w_i^{j+1} are unselected, as required. It is also required that $b_i^{j+1} = (\min \rho - \overline{b}_i^j) \bmod \mathrm{m}$. To see that this is indeed fulfilled, first notice that w_i^j must have a selected neighbor count that is congruent to $\min \rho$ modulo m, as otherwise, we would not have a (σ, ρ)-set at all. Moreover, we have $b_i^{j+1} = b_i^j$ by construction. Hence, we see that $b_i^{j+1} + \overline{b}_i^j \equiv_{\mathrm{m}} \min \rho$, which can be rearranged to $b_i^{j+1} \equiv_{\mathrm{m}} \min \rho - \overline{b}_i^j$. Since $b_i^{j+1} \in [0, \mathrm{m} - 1]$, this finally yields $b_i^j = (\min \rho - \overline{b}_i^j) \bmod \mathrm{m}$, as desired.

The relations $\mathtt{R}_i^0$ are fulfilled for any $i \in [1, n]$ since they only constrain the number of selected vertices of the left block of some information vertices, and we select at most $\mathrm{m} - 1$ vertices of any such block.

It remains to argue that also the constraint relations are fulfilled. This follows directly from the fact that we select vertices such that $b_i^j = \pi(x_i) - 1$ for all $i \in [1, n]$ and $j \in [1, \ell]$, and that π is a satisfying assignment for the input instance. Thus, all relations are fulfilled, and G_I is indeed a yes-instance of (σ, ρ)-GenDomSet$^{\text{Rel}}$. □

We now proceed to the second direction of the correctness.

Lemma 8 ([GSW24, Lemma 5.6])**.** *If the (σ, ρ)-*GenDomSet$^{\text{Rel}}$ *instance G_I has a solution, then the q-CSP-*m *instance I is satisfiable.*

Proof. A similar proof is given in [GSW24].

Let S be a (σ, ρ)-set of G_I (which also satisfies all relations). We first show that S implies a consistent variable assignment, that is, we show that $b_i^j = b_i^{j+1}$ for all $i \in [1, n]$ and $j \in [1, \ell - 1]$.

For this purpose, fix arbitrary $i \in [1, n]$ and $j \in [1, \ell - 1]$. Per definition of $\mathtt{R}_i^j$, we have $b_i^{j+1} = (\min \rho - \overline{b}_i^j) \bmod \mathrm{m}$. Moreover, the relation guarantees that w_i^j and w_i^{j+1} are both unselected. Since w_i^j must have a neighbor count in the set ρ, and we are dealing with periodic sets with period m, we have $b_i^j + \overline{b}_i^j \equiv_{\mathrm{m}} \min \rho$. Rearranging this expression yields $\overline{b}_i^j \equiv_{\mathrm{m}} \min \rho - b_i^j$. Combining this with the previous observation, we obtain $b_i^{j+1} \equiv_{\mathrm{m}} \min \rho - (\min \rho - b_i^j) \equiv_{\mathrm{m}} b_i^j$. The relations $\mathtt{R}_i^{j-1}$ and $\mathtt{R}_i^j$ also ensure that $b_i^j, b_i^{j+1} \in [0, \mathrm{m} - 1]$, and hence, we obtain $b_i^j = b_i^{j+1}$ overall.

Having ensured consistency across the rows of the construction, we can now proceed to define a variable assignment π by setting $\pi(x_i) = b_i^1 + 1$ for all $i \in [1, n]$.

Now, consider an arbitrary constraint C_j for some $j \in [1, \ell]$, and let the scope of the constraint be $\mathsf{scp}(C_j) = (x_{\lambda_1}, \ldots, x_{\lambda_q})$. Relation $\mathtt{C}^j$ ensures that $(b_{\lambda_1}^j + 1, \ldots, b_{\lambda_q}^j + 1) \in \mathsf{acc}(C_j)$, which directly gives that π satisfies constraint C_j. Thus, π is indeed a satisfying assignment of I, and I is satisfiable. □

We have now shown that the instances I and G_I are equivalent. Of course, this alone is not sufficient for our purposes, we must also ensure that the size of G_I is not too large, and in particular, that the pathwidth of G_I is low.

Lemma 9 ([GSW24, Lemma 5.8])**.** *There is a function f depending only on the sets σ and ρ such that G_I has size at most $n \cdot \ell \cdot f(q)$, pathwidth at most $n + f(q)$, and arity at most $f(q)$.*

Proof. We restate the proof of [GSW24] with slight adaptions.

We first elaborate on the number of vertices of G_I. Graph G_I contains ℓ copies of an $\mathbb{R}$-manager of rank n. By Definition 14, the size of each block of an $\mathbb{R}$-manager is at most b, for some constant b depending only on σ and ρ. Since each manager consists of exactly $2n$ blocks as well as n information vertices, we see that G_I consists of at most $\ell \cdot (2n \cdot b + n)$ vertices. Observe that

$$\ell \cdot (2n \cdot b + n) = n \cdot \ell \cdot c_{\mathsf{v}}$$

for an appropriately chosen constant c_{v} only depending on σ and ρ.

For the arity of the relations, we notice that each consistency relations has arity at most $2 \cdot (b + 1)$, each constraint relation has arity at most $q \cdot b$. Hence, the maximum arity of each relation is bounded by $O(q \cdot b)$.

Recall from Definition 11, that the size of a graph with relations is defined as the number of vertices plus the size of each relation (which might be exponential in the arity). The graph G_I contains exactly $n \cdot \ell$ consistency relations and ℓ constraint relations.

Combining our knowledge of the number of vertices, with the maximal arity of any relation, and the number of relations, we can conclude that the size of G_I is bounded by

$$n \cdot \ell \cdot c_{\mathsf{v}} + n \cdot \ell \cdot 2^{O(q \cdot b)} = n \cdot \ell \cdot f_1(q, \sigma, \rho),$$

for an appropriately chosen function $f_1(q, \sigma, \rho)$.

It remains to bound the pathwidth of G_I. We do this by the standard approach of providing a node search-strategy (see [CFK+15, Section 7.5] for example). From Definition 13, we know that the pathwidth of a graph G with relations is defined as the pathwidth of the graph we obtain when making the vertices in the scope of each relation a clique. Let $\hat{G}_I$ be the graph obtained from G_I by this modification (while keeping all indexed vertices/sets the same).

The graph is cleaned in $\ell + 1$ stages, where each stage consists of n rounds. Intuitively, each stage is responsible for cleaning the left side of one column of the construction, and each round for cleaning a block of the column.

For each round, we list the vertices on which searchers are placed. This makes it clear that one can go from one stage to the next without recontaminating already cleaned parts and without the use of additional searchers. For notational convenience, we define

- w_i^j as a dummy vertex that is not part of the graph $\hat{G}_I$ whenever $i \notin [1, n]$ or $j \notin [1, \ell]$,
- B_i^j and $\overline{B}_i^j$ to be the empty set whenever $i \notin [1, n]$ or $j \notin [1, \ell]$,
- $\mathsf{scp}(\mathtt{C}^j)$ to be the empty set when $j \notin [1, \ell]$.

Let $\mathcal{S}_i^j$ denote the set of vertices on which searchers are placed in round i of stage j. We define this set as

$$\begin{aligned} \mathcal{S}_i^j = &\{w_x^{j-1} \mid i \leq x \leq n\} \cup \{w_x^j \mid 1 \leq x \leq i\} \\ &\cup \overline{B}_i^{j-1} \cup B_i^j \\ &\cup \mathsf{scp}(\mathtt{C}^j). \end{aligned}$$

First observe that every vertex and both endpoints of every edge are contained in some set $\mathcal{S}_i^j$. It remains to argue that the graph does not get recontaminated. Consider the intersection of the vertices from two consecutive rounds of the same stage, that is,

$$\mathcal{S}_i^j \cap \mathcal{S}_{i+1}^j = \{w_x^{j-1} \mid i+1 \leq x \leq n\} \cup \{w_x^j \mid 1 \leq x \leq i\} \cup \mathsf{scp}(\mathtt{C}^j).$$

As these vertices form a separator of the graph, the cleaned part of the graph does not get recontaminated.

When moving from one stage to the next one, we can use the same technique by observing that

$$\mathcal{S}_n^j \cap \mathcal{S}_1^{j+1} = \{w_x^j \mid 1 \leq x \leq n\}$$

separates the graph. We conclude that the node search number of $\hat{G}_I$ is at most

$$\max_{i\in[1,n],j\in[1,\ell+1]} \left|\mathcal{S}_i^j\right| = n + 1 + 2b + q \cdot b,$$

which means that the pathwidth of $\hat{G}_I$ and thus, of G_I is at most

$$n + f_2(q, \sigma, \rho),$$

for an appropriately chosen function $f_2(q, \sigma, \rho)$. This concludes the proof by choosing the function f from the statement as the maximum of f_1 and f_2. □

3.4 Combining the Results

Having proven the correctness of the reduction and bounds on the size of the output graph, the arity, and the pathwidth, we can now combine these to proof the lower bound under the Strong Exponential Time Hypothesis.

Theorem 3 ([GSW24, Lemma 5.9])**.** *Let σ and ρ be two periodic sets with the same period* $\mathrm{m} \geq 2$*.*

Then, for all $\varepsilon > 0$, there is a constant d such that (σ, ρ)-$\textsc{GenDomSet}^{\textsc{Rel}}$ on instances of arity at most d cannot be solved in time $(\mathrm{m} - \varepsilon)^{\mathsf{pw}} \cdot |G|^{O(1)}$, where pw is the width of a path composition provided with the input graph G, unless the SETH fails.

Proof. We restate the proof provided in [GSW24].

We assume for contradiction's sake, that, for some $\varepsilon > 0$, there exists an algorithm that can solve (σ, ρ)-$\textsc{GenDomSet}^{\textsc{Rel}}$ in time $(\mathrm{m} - \varepsilon)^{\mathsf{pw}} \cdot |G|^{O(1)}$ when the input contains a graph G and a path decomposition of G of width pw.

Let q be the arity from Theorem 1 such that there is no algorithm that can solve q-CSP-m in time $(\mathrm{m} - \varepsilon)^n \cdot (n + \ell)^{O(1)}$ for an instance with n variables and ℓ constraints.

Given an arbitrary q-CSP-m instance I as input, let G_I denote the corresponding (σ, ρ)-$\textsc{GenDomSet}^{\textsc{Rel}}$ instance from Section 3.3 together with its provided path decomposition. Then, we run the hypothetical algorithm for (σ, ρ)-$\textsc{GenDomSet}^{\textsc{Rel}}$ on this instance and return the output as the output of the q-CSP-m instance.

Since, by Lemmas 7 and 8, the instance G_I has a solution if and only if I is satisfiable, this algorithm correctly decides if the q-CSP-m instance I is satisfiable.

It remains to analyze the running time of the algorithm. The construction of G_I (which bounds the size) takes time polynomial in the size of I (and thus, also in G_I). By

Lemma 9, the width of the given path decomposition is at most $n + f(q)$. Hence, the final algorithm runs in time

$$(\mathrm{m} - \varepsilon)^{n+f(q)} \cdot (n+\ell)^{O(1)} = (\mathrm{m} - \varepsilon)^{n} \cdot (n+\ell)^{O(1)}$$

since q is a constant that only depends on σ and ρ, which are fixed sets and thus, not part of the input. Therefore, this directly contradicts SETH by Theorem 1 and concludes the proof. □

CHAPTER 4

Realizing Relations

The lower bound established in Theorem 3 for the problem (σ, ρ)-GenDomSet$^{\text{Rel}}$ is not yet sufficient for our purpose; we must get rid of the relational constraints. That is, we are looking for a reduction from (σ, ρ)-GenDomSet$^{\text{Rel}}$ instances of bounded arity and bounded pathwidth to (σ, ρ)-GenDomSet instances with a similar pathwidth. This chapter provides such a reduction, and as the idea behind the reduction is the realization of the relations as graph gadgets, we fittingly call it "Realizing Relations". The original content presented here is also presented in [GSW24, Section 6].

The first main goal is proving Theorem 4, which then easily allows us to prove the main result of the thesis.

Theorem 4 ([GSW24, Lemma 6.8])**.** *Let σ and ρ be two difficult periodic sets with the same period* m*. For all constants d, there is a polynomial-time reduction from (σ, ρ)-GenDomSet$^{\text{Rel}}$ on instances with arity d given with a path decomposition of width* pw *to (σ, ρ)-GenDomSet on instances given with a path decomposition of width $\mathsf{pw} + O(2^d)$.*

4.1 Replacing Arbitrary Relations With Simple Relations

Following the ideas of [CM16, MSS21, MSS22, FMI+23c], we first replace the arbitrary relations by relations that are much simpler, which can then be handled more easily. These easier relations are hamming-weight and equality relations, which are defined next.

Definition 15 (Hamming-weight and equality relations; [FMI+23c, Definition 4.2])**.** *Let U be a set of vertices with $|U| = d$. The hamming-weight 1 relation of arity d, denoted by $\mathtt{HW}_{=1}^{(d)}$, with relation scope U is the relation that accepts $S \subseteq U$ if and only if $|S| = 1$, that is, exactly one vertex of the relation scope is selected.*

The equality relation of arity d, denoted by $\mathtt{EQ}^{(d)}$, with relation scope U accepts $S \subseteq U$ if and only if $|S| = 0$ or $|S| = d$, that is, either all vertices in the relation scope are selected, or none of them.

Note that we may drop the arity from the notation of a relation if it is clear from the context which arity is meant. The overall goal of this section is to show that we can actually replace arbitrary relations by hamming-weight 1 relations without dramatically increasing the pathwidth of the graph. This result was shown by Focke et al. [FMI+23c], and although they use the result for finite and cofinite sets, we can directly use it for our scenario as well. To keep the thesis self-contained, we also present their result in detail.

Lemma 10 ([FMI+23c, Corollary 8.8])**.** *Let σ, ρ be non-empty sets and $\rho \neq \{0\}$. For all constants d there is a polynomial-time reduction from (σ,ρ)-GenDomSet^{Rel} on instances of arity at most d and pathwidth pw to (σ,ρ)-GenDomSet^{Rel} on instances of arity at most 2^d+1 and pathwidth $\mathsf{pw}+O(1)$, where each relation is a hamming-weight 1 relation.*

In the proofs of this chapter, the central idea is that we can replace a relation by a graph gadget (which itself might use other, simpler relations). We formally define this notion of realizing relations, and the required notion of graphs with relations and portals, next.

Definition 16 (Graph with relations and portals; [FMI+23c, Definition 4.6])**.** *Let $G = (V, E, \mathcal{C})$ be a graph with relations, and $U \subseteq V$. Then, (G, U) is a* graph with relations and portals. *A set $X \subseteq V$ is a partial solution of (G, U) if*

- *X is a partial solution of $((V, E), U)$, and*
- *$X \cap \mathit{scp}(C) \in \mathit{acc}(C)$ for all $C \in \mathcal{C}$.*

The notions of compatible strings*, L-*providers*, and L-*realizers*, given in Definitions 7 and 8 generalize in the natural way.*

Definition 17 (Realization of a relation; [FMI+23c, Definition 8.2])**.** *For a set of vertices S with $d = |S|$ let $R \subseteq 2^S$ denote a d-ary relation. For an element $r \in R$ we write x_r for the length d string that is σ_0 at every position $v \in r$ and ρ_0 at the remaining positions, i.e.,*

$$x_r[v] = \begin{cases} \sigma_0 & \text{if } v \in r, \\ \rho_0 & \text{otherwise.} \end{cases}$$

We set $L_R = \{x_r \mid r \in R\}$. Let $G = (V, E, \mathcal{C})$ be a graph with relations, and let $U = \{u_1, \ldots, u_d\} \subseteq V$ be a set of portals of G. Slightly overloading notation, we say that (G, U) realizes R if (G, U) realizes L_R. We say that R is realizable if there is a graph with d portals that realizes R.

In a first step, we replace arbitrary relations with equality and hamming-weight relations.

Lemma 11 ([FMI+23c, Lemma 8.3])**.** *Let σ, ρ be non-empty sets with $\rho \neq \{0\}$, S a set of vertices with $|S| = d$, and let $R \subseteq 2^S$ denote an arbitrary d-ary relation. Then R is realizable by a graph G with relations and portals with the following properties:*

- *All relations used in G are $\texttt{HW}_{=1}$ or $\texttt{EQ}$ relations with arity at most $2^d + 1$.*
- *The size of G is in $O(2^d \cdot d)$.*

Proof. We present the proof by Focke et al. [FMI+23c] with slight changes.

Let $U = \{u_1, \ldots, u_d\}$ be the set of d vertices in the relation scope. Let $R = \{r_1, \ldots, r_x\}$ (where $r_i \subseteq [1, d]$) be a d-ary relation that, slightly abusing the notation, describes the allowed selections from U via the indexes of the vertices in U. Our output graph (G, U), where G is a graph with portals and relations and $U \subseteq V(G)$ will consist of many copies of the provider from Lemma 4, and d singleton vertices, as well as many relations that ensure the correct behavior.

Start from the graph that contains only the vertices in U. For each $i \in [1, x]$ and all $j \in [1, d] \setminus r_i$ add a copy $(H_i^j, \{s_i^j\})$ of the provider from Lemma 4 (for some suitable constants $r \in \rho, s \in \sigma$) to the graph. Add one more such provider $(T_i, \{t_i\})$ to the graph for all $i \in [1, x]$. Next, we proceed to add the necessary relations to the graph. For every $i \in [1, x]$, add an $\texttt{EQ}^{(|[1,d] \setminus r_i|+1)}$ relation with relation scope $\{t_i\} \cup \{s_i^j \mid j \in [1, d] \setminus r_i\}$. Moreover, add a $\texttt{HW}_{=1}^{(x)}$-relation to the graph that with relation scope $\{t_i \mid i \in [1, x]\}$. Finally, for each $j \in [1, d]$, add a $\texttt{HW}_{=1}$ relation with scope $\{u_j\} \cup \{s_i^j \mid i \in [1, x], j \notin r_i\}$ to the graph with relations. The gadget is visualized in [FMI+23c, Figure 8.1] using slightly different notation than in this proof.

Then, (G, U) realizes the relation R. We proof this by first showing that there is a (σ, ρ)-set S_i of G such that $S_i \cap U = \{u_j \mid j \in r_i\}$ for all $i \in [1, x]$. Fix an r_i, and set S_i to be the set that selects the vertices u_j for all $j \in r_i$, the vertex t_i, and the vertices s_i^j for all $j \in [1, d] \setminus r_i$. From within the providers, choose a vertex selection that is feasible, which is always possible. It remains to argue that all relations of G are satisfied by this selection. It is easy to see that the equality relations are satisfied, because the s-vertices with subscript index i and the t-vertex with index i are all selected, whereas the s-vertices with an index different from i and the remaining t-vertices are all unselected. Moreover, the hamming-weight 1 relation between the t-vertices is satisfied because exactly one such vertex is selected. Finally, we need to argue about the hamming-weight 1 relation of the remaining vertices. For this purpose, consider a vertex u_j such that $j \notin r_i$. Then, u_j is not selected. However, the $\texttt{HW}_{=1}$ relation that has u_j in its scope also has the vertices $\{s_\ell^j \mid \ell \in [1, x], j \notin r_i\}$ within its scope. As we have $j \notin r_i$, we know that s_i^j is selected, and it is the only selected vertex from the set, fulfilling the relation. Now, consider a vertex u_j with $j \in r_i$. The $\texttt{HW}_{=1}$ relation that has u_j in its scope has no s-vertex with subscript index i in its scope. Hence, vertex u_j is the only selected vertex of the relation scope.

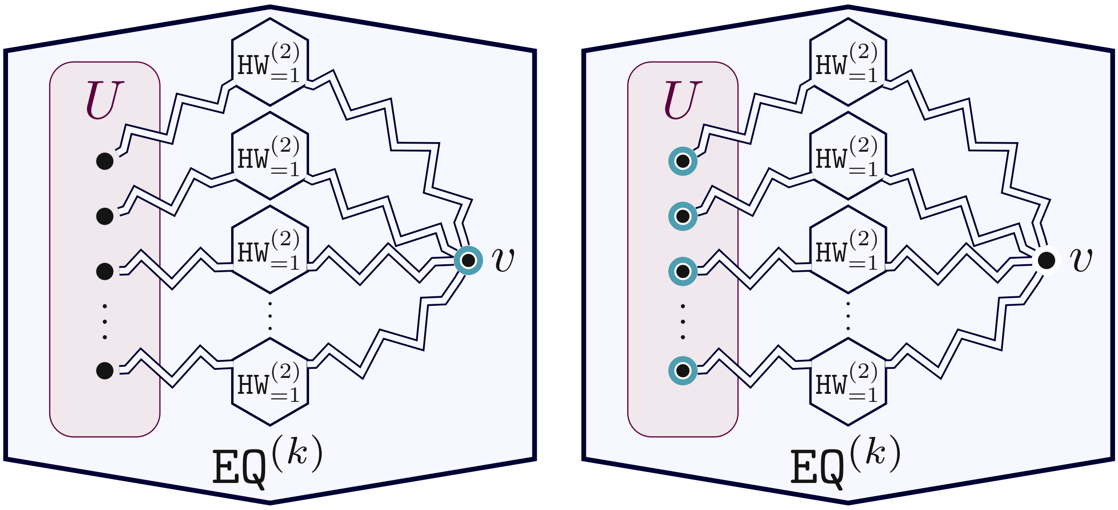


Figure 4.1: The gadget from Lemma 12. Two different solutions corresponding to the two allowed selections of the portal vertices are sketched. From the provider, only the portal vertex v is drawn for the sake of visibility. The figure is originally from [FMI$^+$23c, Figure 8.2].

Now, we show that any (σ, ρ)-set S of G must select vertices of U corresponding to some set r_i. First, notice that the hamming-weight 1 relation with the t-vertices in its relation scope forces exactly one such vertex to be selected. Let t_i be the single selected t-vertex. Then, the equality relations force the vertices $\{s_i^j \mid j \in [1, d] \setminus r_i\}$ to be selected, and all other s-vertices to be unselected. Consider an arbitrary vertex u_j for $j \notin r_i$. Vertex s_i^j is selected, and hence, u_j is not selected due to the hamming-weight 1 relation that has both s_i^j and u_j in its scope. On the other hand, consider u_j for $j \in r_i$. Then, u_j must be selected because all the vertices together in the hamming-weight 1 relation with u_j are not selected.

We have thus established that (G, U) indeed realizes the relation. To see that bound on the size, notice that the x equality relations have arity at most $d + 1$, and the relation between the t-vertices has arity at most 2^d. Moreover, the d hamming-weight 1 relations containing vertices of U have arity at most $1 + 2^d$. Hence, the maximum arity is bounded by $2^d + 1$.

For the number of vertices of the output graph, notice that it contains d singleton vertices, at most $2^d \cdot d$ providers for the s-vertices, and at most 2^d providers for the t-vertices. As the size of each provider is constant, we see that the number of vertices is in $O(2^d \cdot d)$, and, together with the information in the previous paragraph and our knowledge of the size of hamming-weight 1 and equality relations, so is the size of the whole graph. □

In a next step, we want to get rid of the equality relations by replacing them with $\mathtt{HW}_{=1}^{(2)}$-relations.

Lemma 12 ([FMI+23c, Lemma 8.6])**.** *Let σ and ρ be non-empty sets with $\rho \neq \{0\}$. Then, for any $k \geq 1$, there is a graph G with relations and portals that realizes $\texttt{EQ}^{(k)}$ such that G has the following properties:*

- *All relations used in G are $\texttt{HW}^{(2)}_{=1}$ relations.*
- *The size of G is in $O(k)$.*
- *G has pathwidth $O(1)$.*

Proof. The proof is due to Focke et al. [FMI+23c].

When $k = 1$, we can trivially realize the relation by taking the single vertex in the relation scope as the whole graph. For $k \geq 2$, we construct a graph with relations and portals (G, U), where $U = \{u_1, \ldots, u_k\}$ are the vertices in the scope of the relation. We add a single provider from Lemma 4 (for arbitrary, but fixed r, s) to the graph. Note that this provider has constant size, and let v be the portal vertex of the provider. For each $i \in [1, k]$, we add a $\texttt{HW}^{(2)}_{=1}$-relation with scope v, u_i to the graph. Fig. 4.1 depicts the realization.

It is easy to see that this gadget realizes $\texttt{EQ}^{(k)}$. In particular, if all vertices of U are selected, v can be unselected, and we can easily obtain a solution by extending the selection to the provider. Similarly, we can select v if all vertices of U are unselected. However, it is not possible that some vertices of U are selected and others are not, as the $\texttt{HW}^{(2)}_{=1}$-relations could then not be fulfilled anymore.

For the size of G, it is easy to see that it is $O(k)$ because G contains k portal vertices and k relations with constant size, and because the size of the provider is constant. For the pathwidth, we notice that, when we replace each relation with a clique, we essentially end up with a star graph, which has constant pathwidth, plus the provider. By putting all vertices of the provider in a bag with its portal vertex, we can see that the pathwidth of G is constant. □

Now, we are finally ready to prove the main lemma of this section.

Lemma 10 ([FMI+23c, Corollary 8.8])**.** *Let σ, ρ be non-empty sets and $\rho \neq \{0\}$. For all constants d there is a polynomial-time reduction from (σ, ρ)-$\textsc{GenDomSet}^{\textsc{Rel}}$ on instances of arity at most d and pathwidth pw to (σ, ρ)-$\textsc{GenDomSet}^{\textsc{Rel}}$ on instances of arity at most $2^d + 1$ and pathwidth $\mathsf{pw} + O(1)$, where each relation is a hamming-weight 1 relation.*

Proof. The proof is attributed to Focke et al. [FMI+23c].

Let G be a graph with relations, such that each relation is of arity at most d, and assume that we are provided with a path decomposition of width pw. We first utilize Lemma 11 to replace all relations with equality and hamming-weight 1 relations. Then, we use

Lemma 12 to get rid of the equality relations, leaving us with an equivalent instance that only contains hamming-weight 1 relations.

As replacing arbitrary relations leads to an arity bound of $2^d + 1$, and removing equality relations does not change this, the output instance has arity at most $2^d + 1$. For the pathwidth, we can proceed by taking the original path composition of G, and enhancing it a bit. Concretely, we know that a path decomposition of G is a path composition of $\hat{G}$, where $\hat{G}$ is G in which the vertices in the scope of each relation form a clique. Hence, for each relation, there exists a bag in the decomposition that contains the vertices in its scope. Now, let G' be the output graph of the reduction. We can obtain a path decomposition of $\hat{G}'$ by taking the decomposition of $\hat{G}$, and, for each relation of G that is realized in G', we copy the bag of the decomposition that contains the vertices in the relation scope twice (and connect the copies), and add the vertices of the realization to the center copy of the overall three copies. This way, we obtain a valid decomposition. Notice that we increase the pathwidth at most by the number of vertices in the gadgets that realize a relation. We see that the replacement of the arbitrary relations leads to a graph with $O(2^d \cdot d)$ vertices, and by replacing the equality relations, this number is increased by at most $O(2^d)$, as that is an upper bound on the number of equality relations that need replacement, and each replacement increases the number of vertices by a constant amount (the number of vertices of a provider). Hence, the output path decomposition has pathwidth in $\mathsf{pw} + O(1)$, as d is constant.

Finally, it is easy to see that the procedure can be done in polynomial-time, as replacing a relation can actually be done in constant time, given that d is a constant. □

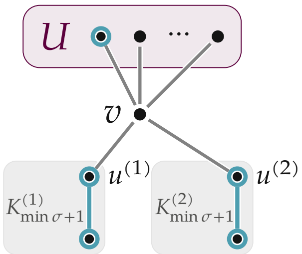
4.2 Realizing $\mathtt{HW}_{=1}$

Now, in our final step, we must get rid of the hamming-weight 1 relations in the graph by replacing them with suitable gadgets. One challenge in directly realizing such gadgets in our settings is, that for any vertex, having 1 selected neighbor is the same as having $\mathrm{m} + 1$ selected neighbors. Thus, we will first show that we can create a gadget that ensures that the number of selected vertices in the relation scope is congruent 1 modulo m. We denote the relation with this property of arity d as the $\mathtt{HW}^{(d)}_{\in \rho - \min \rho + 1}$-relation. In a second step, we can then build the $\mathtt{HW}_{=1}$ relation for arbitrary arities d by utilizing these $\mathtt{HW}_{\in \rho - \min \rho + 1}$-relations as building blocks.

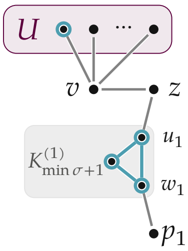
Lemma 13 ([GSW24, Lemma 6.11])**.** *Let σ and ρ be two difficult periodic sets with the same period* m*. Then, the relation $\mathtt{HW}_{\in \rho - \min \rho + 1}$ can be realized.*

Proof. The proof is also provided in [GSW24].

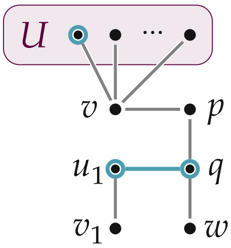
We consider different cases, depending on σ and ρ, and show that we can realize the relation in each case by proving four different claims later on in this proof. Note that σ, ρ being difficult implies that $\mathrm{m} \geq 3$ and $\min \rho \geq 1$.



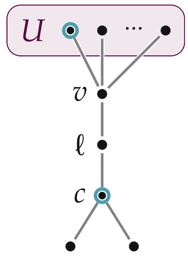
(a) The case $\min\rho \geq 2$ illustrated for $\min\sigma = 1, \min\rho = 3$.



(b) The case $\min\rho = 1, \min\sigma \geq 2$ illustrated for $\min\sigma = 2, \min\rho = 1$ and $r = 1$.



(c) The case $\min\rho = \min\sigma = 1$ illustrated for $r = 1, s = 1$.



(d) The case $\min\rho = 1, \min\sigma = 0$.

Figure 4.2: The gadget constructions from Lemma 13. In each sketched construction we mark vertices corresponding to a feasible solution within the gadget. From the relation scope U, we select an arbitrary vertex. This figure and the captions are also displayed in [GSW24, Figure 4].

- If $\min\rho \geq 2$, then we use Claim 1.
- If $\min\rho = 1$ and $\min\sigma \geq 2$, then we use Claim 2.
- If $\min\rho = 1$, $\min\sigma = 1$, then we use Claim 3.
- If $\min\rho = 1$, $\min\sigma = 0$, then we use Claim 4.

Consult Fig. 4.2 for a visualization of the gadgets used in the different cases of the proof.

In the following, we describe the gadgets in the strongest-possible way, that is, we describe the minimal requirements for the gadgets to work.

We start with the first case where $\min\rho \geq 2$. Clearly, m ≥ 3 implies that $\min\sigma + 1 \notin \sigma$.

Claim 1. *Let σ and ρ be arbitrary non-empty sets such that $\min\rho \geq 2$ and $\min\sigma + 1 \notin \sigma$. Then, $\texttt{HW}_{\in\rho-\min\rho+1}$ is realizable.*

Proof. For all $i \in [1, \min\rho - 1]$, we create a clique $K^{(i)}_{\min\sigma+1}$ on $\min\sigma + 1$ vertices. We create a vertex v that is made adjacent to all vertices in the scope of the relation. Finally, we select a vertex $u^{(i)}$ from the clique $K^{(i)}_{\min\sigma+1}$, and make v adjacent to $u^{(i)}$ for all $i \in [1, \min\rho - 1]$.

The correctness relies on the fact that all vertices of each clique must be selected in a solution. To see this, fix some $i \in [1, \min\rho - 1]$ and consider the clique $K^{(i)}_{\min\sigma+1}$.

If $\min\sigma + 1 = 1$, then the clique consists of a single vertex $u^{(i)}$ that must be selected since $u^{(i)}$ only has a single neighbor, and if the vertex would be unselected, it would need at least two neighbors because of $\min\rho \geq 2$.

If $\min\sigma + 1 = 2$, then the clique consists of two vertices connected by an edge. Let w be the vertex of the clique that is not $u^{(i)}$. Since w has only a single neighbor, vertex w must be selected. But, since $\min\sigma = 1$, this means that also $u^{(i)}$, the only neighbor of w, must be selected.

If $\min\sigma + 1 \geq 3$, assume that some vertex $v^{(i)}$ of the clique that is not $u^{(i)}$ is not selected in a solution. Then, $v^{(i)}$ must have two selected neighbors in the clique. At least one of them, denote it by $w^{(i)}$, must be different from $u^{(i)}$, Hence, $w^{(i)}$ is selected and requires $\min\sigma$ selected neighbors, which implies that all vertices of the clique must be selected, contradicting that $v^{(i)}$ is not selected.

By the previous argument vertex v is adjacent to a vertex $u^{(i)}$ such that $u^{(i)}$ has exactly $\min\sigma$ selected neighbors in the clique. Since $\min\sigma + 1 \notin \sigma$, vertex v cannot be selected. Furthermore, v is adjacent to $\min\rho - 1$ vertices that are selected, and we know that $\min\rho - 1 \notin \rho$. Thus, at least one vertex in the scope of the relation must be selected, so that v can have enough neighbors. Moreover, exactly r vertices from the scope must be selected where $r + \min\rho - 1 \in \rho$. □

For all the remaining cases we assume that $\min\rho = 1$. We proceed with the case where $\min\sigma \geq 2$.

Claim 2. *Let σ and ρ be arbitrary non-empty sets such that $\min\rho = 1$, there exists an $r \in \rho$ with $r + 1 \notin \rho$, $\min\sigma \geq 2$, and $\min\sigma + 1 \notin \sigma$. Then, $\mathtt{HW}_{\in\rho-\min\rho+1}$ is realizable.*

Observe that in our case choosing $r = \min\rho$ is possible.

Proof. The gadget contains the $r + 2$ vertices $v, z, p_1, \ldots, p_r$ and r cliques $K^{(i)}_{\min\sigma+1}$ on $\min\sigma + 1$ vertices each where $i \in [1, r]$, and u_i and w_i denote two distinct vertices of the clique $K^{(i)}_{\min\sigma+1}$. We make v adjacent to all vertices of the scope and to z. Vertex z is adjacent to all u_i and vertex w_i is adjacent to p_i for all i.

In any solution, all vertices of each clique $K^{(i)}_{\min\sigma+1}$ must be selected due to vertex p_i: Vertex p_i has only a single neighbor and hence, cannot be selected as $\min\sigma \geq 2$. Moreover, since $\min\rho = 1$, the unique neighbor w_i of p_i must also be selected. The selected vertex w_i then needs $\min\sigma$ selected neighbors, and so the whole clique must be selected. Then, vertex u_i has $\min\sigma$ selected neighbors, which implies that its only other neighbor z

cannot be selected. Since the unselected vertex z already has r selected neighbors u_i, vertex v cannot be selected as $r+1 \notin \rho$. However, as v is unselected and has no selected neighbors, it requires to have t selected neighbors for some $t \in \rho$ which have to stem from the scope of the relation. □

Next we change the requirement for $\min \sigma$ by assuming $\min \sigma = 1$.

Claim 3. *Let σ and ρ be arbitrary non-empty sets such that $\min \rho = 1$, there exists an $r \in \rho$ with $r+1 \notin \rho$, $\min \sigma = 1$, $2 \notin \sigma$, and there is an $s \in \sigma$ with $s+1, s+2 \notin \sigma$. Then $\texttt{HW}_{\in \rho - \min \rho + 1}$ is realizable.*

Observe that this covers our case by setting $r = 1$ and $s = 1$ as $\mathrm{m} \geq 3$.

Proof. Assume we can construct a gadget F where a distinguished vertex p is forced to be not selected, and the only possible solution also provides exactly one selected neighbor for this vertex p. Then we can realize the relation as follows. Create r copies of F which we denote by $F_1, \ldots, F_r$ where we identify all vertices $p_1, \ldots, p_r$ with a new vertex p We additionally add a vertex v and make v adjacent to all vertices in the scope of the relation and p.

By the properties of the gadget F, the vertex p is not selected and has one neighbor in each copy F_i. Hence, vertex v cannot be selected as $r+1 \notin \rho$. Therefore, the number of vertices that are selected from the scope must be t for some $t \in \rho$.

It remains to construct the gadget we assumed to exist above. For this we introduce $2s+3$ vertices p, q, w and $u_1, v_1, \ldots, u_s, v_s$ where p, q, w are connected to a path and q, u_i, v_i are also connected to form a path for all i.

Since $\min \rho = \min \sigma = 1$, all vertices u_i must be selected because of the requirement of each v_i. Moreover, because the only neighbor of w is q, vertex q must be selected as well. Hence, as $2 \notin \sigma$ and q is selected, each vertex u_i forces v_i to not be selected. With this selection vertex q has s selected neighbors. Since $s+1, s+2 \notin \sigma$, it is not possible to select any of the vertices w and p which have one selected neighbor each.

This concludes the construction of the auxiliary gadget F with p as the distinguished vertex. □

For the remaining case we now assume that $\min \sigma = 0$.

Claim 4. *Let σ and ρ be arbitrary non-empty sets such that $\min \rho = 1$, and $2, 3 \notin \rho$, $\min \sigma = 0$, and $1 \notin \sigma$. Then $\texttt{HW}_{\in \rho - \min \rho + 1}$ is realizable.*

Observe that our case is covered since $\mathrm{m} \geq 3$, $\min \rho = 1$ and $\min \sigma = 0$ implies all conditions required by Claim 4.

Proof. We start by creating a star graph S_3 with 3 leaves. Let c be the center of the star. Add a vertex v to the graph, and make v adjacent to one leaf of S_3, and to all vertices in the scope of the relation.

We first argue that in any solution, vertex c must be selected, and furthermore, all of its neighbors cannot be selected. Towards a contradiction, assume that c is not selected. Then, there are two leaves of c that must be selected to become happy. [1] However, for c, two out of three of its neighbors are now selected, and $2 \notin \rho, 3 \notin \rho$. Thus, c must be selected in any solution. The leaves of c that have no other neighbors cannot be selected now, as they have selected neighbor c and $1 \notin \sigma$.

Consider the leaf ℓ of S_3 that is a neighbor of v. This leaf has two neighbors, one of which is selected. As c is selected and $1 \notin \sigma$, vertex ℓ cannot be selected. Furthermore, v cannot be selected either, because selecting v would give ℓ two selected neighbors and $2 \notin \rho$. Thus, v is not selected and requires at least one more selected vertex from the scope of the relation.

Finally, we can make all vertices of the gadget happy if $r \in \rho$ vertices of the relation scope are selected. In that case, c is the only gadget vertex that must be selected. The leaves of c are happy, because they have a single selected neighbor and are unselected. Furthermore, vertex v is happy, because it has r selected neighbors in the scope of the relation. ☐

This finishes the proof of Lemma 13 by combining Claims 1 to 4. ☐

Notice that $\texttt{HW}_{\in\rho-\min\rho+1}$ relations are the same as $\texttt{HW}_{=1}$ relations if the arity of the relation is small enough. This observation is formalized in the next corollary.

Corollary 1 ([GSW24, Corollary 6.4])**.** *Let σ and ρ be two difficult periodic sets with the same period* m*. Then, the relation $\texttt{HW}^{(k)}_{=1}$ can be realized for $k \in [1, 3]$.*

Proof. A similar proof is given in [GSW24].

Because σ, ρ are difficult, we know that $\mathrm{m} \geq 3$. Per Lemma 13, we can realize the $\texttt{HW}_{\in\rho-\min\rho+1}$ relation for these sets. Since $\mathrm{m} \geq 3$, and $1 \in \rho - \min\rho + 1$, it can be easily observed that $2, 3, \notin \rho - \min\rho + 1$. Thus, the $\texttt{HW}_{\in\rho-\min\rho+1}$ relation is the same as the $\texttt{HW}_{=1}$ relation for the considered arities. ☐

Now that we can realize $\texttt{HW}_{=1}$ for arities one, two, and three, we can already proceed to the realization of $\texttt{HW}_{=1}$ relations of arbitrary arity.

Lemma 14 (follows from [GSW24, Lemma 6.6])**.** *Let σ and ρ denote periodic sets with period* m *such that $\texttt{HW}^{(1)}_{=1}$, $\texttt{HW}^{(2)}_{=1}$, and $\texttt{HW}^{(3)}_{=1}$ can be realized, and $0 \notin \rho$. Then, for all $k \geq 1$, the relation $\texttt{HW}^{(k)}_{=1}$ can be realized by a gadget of size $O(k)$.*

[1] We call a selected vertex happy if the number of selected neighbors it has is in σ, and call an unselected vertex happy if the number of selected neighbors it has is in ρ.

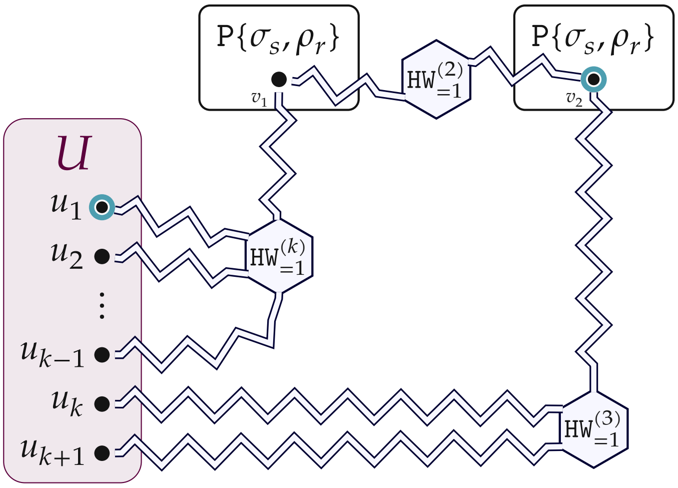


Figure 4.3: The gadget from Lemma 14. The figure is also presented in [GSW24, Figure 3].

Proof. The proof is also presented in [GSW24].

We use the same ideas as in the proof of [MSS21, Lemma 4.4], which uses a simple approach of obtaining a higher degree relation by combining $\mathtt{HW}_{=1}^{(2)}$ and $\mathtt{HW}_{=1}^{(3)}$ relations in a path-like manner.

We proceed by strong induction. The base cases, $1 \le k \le 3$ hold by assumption.

For the induction step ($k \ge 4$), we assume that we can realize the relation for all arities from 1 to k, and show that we can realize the relation for arity $k+1$. We first describe the gadget construction; then we argue about its properties. Also consult Fig. 4.3 for a visualization of the construction.

Denote by $u_1, \dots, u_{k+1}$ the vertices of the relation scope. First, we set $s = \min \sigma \in \sigma$ and $r = \min \rho \in \rho$, and we add to the graph two independent copies of the gadget from Lemma 4 for s and r; call the portal vertices of said gadgets v_1 and v_2, respectively. Next, we add the relation $\mathtt{HW}_{=1}^{(k)}$ with scope $u_1, \dots, u_{k-1}, v_1$ to the graph. Then, we add the relation $\mathtt{HW}_{=1}^{(2)}$ with scope v_1, v_2 to the graph. After that, we add the relation $\mathtt{HW}_{=1}^{(3)}$ with scope v_2, u_k, u_{k+1} to the graph. Finally, we replace all relations with the respective realization gadgets, which exist by assumption. For the rest of this proof, call the resulting graph G.

Claim 5. *The graph G realizes the $\mathtt{HW}_{=1}^{(k+1)}$ relation.*

Proof. First, assume that no vertex of the relation scope is selected. In this case, both v_1, and v_2 must be selected, which is not possible due to the $\mathtt{HW}_{=1}^{(2)}$ relation.

Next, it is not possible that two vertices of $u_1, \dots, u_{k-1}$ or two vertices of u_k, u_{k+1} are selected, due to the $\mathtt{HW}_{=1}^{(k)}$ and $\mathtt{HW}_{=1}^{(3)}$ relations. If one vertex of $u_1, \dots, u_{k-1}$ and one vertex of u_k and u_{k+1} are selected, then both v_1, and v_2 cannot be selected, which is once again impossible in any solution.

Finally, if one vertex of $u_1, \ldots, u_{k+1}$ is selected, then exactly one of v_1 and v_2 must be selected, and all relations are fulfilled. Moreover, the vertices v_1 and v_2 can always receive a feasible number of neighbors, regardless of their selection status. □

To conclude the proof, we analyze the size of the gadget. The gadgets for arities 1, 2 and 3, and the gadgets from Lemma 4 have constant size each. Hence, the size of the gadget grows only by a constant amount as we go from one arity to the next, proving that the size of the gadget for arity k is linear in k. □

Having the ability to realize $\texttt{HW}_{=1}$-relations for arbitrary arities, we can now prove the main result of the section.

Theorem 4 ([GSW24, Lemma 6.8])**.** *Let σ and ρ be two difficult periodic sets with the same period* m*. For all constants d, there is a polynomial-time reduction from (σ, ρ)-GenDomSet$^{\text{Rel}}$ on instances with arity d given with a path decomposition of width* pw *to (σ, ρ)-GenDomSet on instances given with a path decomposition of width* $\mathsf{pw} + O(2^d)$.

Proof. We restate the proof of Lemma 6.8 of [GSW24] almost exactly as stated there.

Let σ and ρ denote sets as in the statement of the lemma. Further, let I_1 denote an instance of (σ, ρ)-GenDomSet$^{\text{Rel}}$, let pw denote the pathwidth of the graph corresponding to I_1, and let d denote the arity of the graph corresponding to I_1.

First, we apply Lemma 10 to obtain an equivalent instance I_2 of (σ, ρ)-GenDomSet$^{\text{Rel}}$ with pathwidth $\mathsf{pw} + O(1)$ and arity $2^d + 1$ in which all relations are hamming-weight 1 relations. By Corollary 1 and Lemma 14, we can replace all remaining relations of the graph with their realizations. To do this, observe that any remaining relation is a $\texttt{HW}_{=1}$ relation. To replace such a relation with a graph, we add the graph that realizes this relation, and unify its portal vertices with the vertices of the relation. Write I_3 to denote the resulting instance of (σ, ρ)-GenDomSet$^{\text{Rel}}$.

Claim 6. *The instances I_2 and I_3 are equivalent.*

Proof. First, assume that I_2 is a yes-instance. Then, selecting the same vertices that are selected in I_2, and extending this solution to the newly added graphs results in a solution for instance I_3. Because all relations of I_2 are fulfilled, such an extension is indeed possible by Definition 17.

Now, assume that I_3 is a yes-instance. Since no solution can select neighbors of vertices of I_3 that were not yet present in I_2, restricting the solution of I_3 to graph I_2 ensures that all vertices of I_2 receive a feasible number of neighbors. Moreover, the gadgets that were added to replace the relations ensure that exactly one vertex of each relation of I_2 must be selected in any solution, hence, this solution also fulfills all relations. □

Claim 7. *The instance I_3 has a pathwidth of* $\mathsf{pw} + O(2^d)$.

Proof. Recall from Definition 13 that the pathwidth of I_2 is defined as the pathwidth of the graph obtained by forming a clique out of all vertices in the relation scope for each relation. Let $\hat{I}_2$ denote the graph that is obtained from I_2 by applying the aforementioned transformation. Consider a path decomposition of $\hat{I}_2$. For any relation, there exists a bag of the decomposition in which all vertices of the relation are present. We can duplicate this bag and reconnect the bags in the natural way. Then, we simply add all vertices of the gadget that realizes the relation to the duplicated bag. It is easy to see that one can obtain a path decomposition of I_3 by performing this operation for each relation such that we never add the vertices of two realization gadgets to the same bag. The width of this decomposition is the width of the decomposition of I_2 plus the size of the largest gadget that was added to the graph. Using Lemma 14, we observe that this results in a decomposition of width $\mathsf{pw} + O(2^d)$, as desired. □

Combining Claims 6 and 7, we obtain the claimed result. □

4.3 Finalizing the Proof

In Chapter 3, we have presented a lower bound for (σ, ρ)-GenDomSet$^{\text{Rel}}$ under the SETH in Theorem 3. Using the reduction from Theorem 4, we can now prove that also (σ, ρ)-GenDomSet does not admit a faster algorithm unless the SETH is false. This is the main theorem of the thesis.

Main Theorem 1 ([GSW24, Main Theorem 2])**.** *Write $\sigma, \rho \subseteq \mathbb{N}$ for difficult periodic sets that both have the same period $\mathrm{m} \geq 2$. Unless SETH fails, for all $\varepsilon > 0$, there is no algorithm that can decide in time $(\mathrm{m} - \varepsilon)^{\mathsf{pw}} \cdot |G|^{O(1)}$ whether the input graph G has a (σ, ρ)-set, when a path decomposition of width pw is given with the input.*

Proof. The proof is also given in [GSW24].

Assume we are given a faster algorithm for (σ, ρ)-GenDomSet for some $\varepsilon > 0$. Let d be the constant from Theorem 3 such that there is no algorithm solving (σ, ρ)-GenDomSet$^{\text{Rel}}$ in time $(\mathrm{m} - \varepsilon)^{\mathsf{pw}} \cdot |G|^{O(1)}$ when the input instance G is given with a path decomposition of with pw.

Consider an instance G of (σ, ρ)-GenDomSet$^{\text{Rel}}$ with arity d along with a path decomposition of width $\mathsf{pw}(G)$. We use Theorem 4 to transform this instance into an instance G' of (σ, ρ)-GenDomSet with a path decomposition of width $\mathsf{pw}(G') = \mathsf{pw}(G) + O(2^d)$.

We apply the fast algorithm for (σ, ρ)-GenDomSet to the instance G' which correctly outputs the answer for the original instance G of (σ, ρ)-GenDomSet$^{\text{Rel}}$. The running time of this entire procedure is

$$|G|^{O(1)} + (\mathrm{m} - \varepsilon)^{\mathsf{pw}(G')} \cdot |G'|^{O(1)} = (\mathrm{m} - \varepsilon)^{\mathsf{pw}(G)+O(2^d)} \cdot |G|^{O(1)} = (\mathrm{m} - \varepsilon)^{\mathsf{pw}(G)} \cdot |G|^{O(1)}$$

since d is a constant only depending on ε. This contradicts SETH and concludes the proof. □

CHAPTER 5

Turn The Lights Off!

In this chapter, we deal with the non-trivial cases of (σ, ρ)-GenDomSet for periodic σ, ρ with the same period, that we did not yet cover. As mentioned in Chapter 2, the decision problem is solvable in polynomial time if m = 2. Recall that we refer to the problem where $\sigma = \{0, 2, 4, \dots\}$, $\rho = \{1, 3, 5, \dots\}$ as Refl-AllOff, and the problem where $\sigma = \rho = \{1, 3, 5, \dots\}$ as AllOff. As these problems correspond to natural board games, we are interested in the optimization variants of them, which are NP-hard [Sut88, HKT00b, CGK01].

In this chapter, we use the constructions by Sutner [Sut88], as presented in [FY13], as a basis, and extend/modify them to yield tight complexity lower bounds under SETH. Concretely, we prove Main Theorem 2. In Section 5.1, we deal with the minimization problem of Refl-AllOff, and in Section 5.2, we cover the minimization version of AllOff. The content of Sections 5.1 and 5.2 is also presented in [GSW24, Section 7].

Main Theorem 2 ([GSW24, Main Theorem 3])**.** *Unless SETH fails, for all $\varepsilon > 0$, there is no algorithm for each of the problems Refl-AllOff and AllOff deciding in time $(2-\varepsilon)^{\mathsf{pw}} \cdot |G|^{O(1)}$ whether there exists a solution of size at most k for a graph G that is given with a path decomposition of width pw.*

Finally, we also deal with the maximization problems in Section 5.3 which culminates in Main Theorem 3. Since the maximization problem is difficult even if $0 \in \rho$, we show the lower bound for the maximization problem for all four different combinations of σ, ρ with period 2. The content of Section 5.3 is presented in this thesis for the first time.

Main Theorem 3. *Let σ, ρ be periodic sets with period 2. Unless SETH fails, for all $\varepsilon > 0$, there is no algorithm for (σ, ρ)-GenDomSet deciding in time $(2-\varepsilon)^{\mathsf{pw}} \cdot |G|^{O(1)}$ whether there exists a solution of size at least k for a graph G that is given with a path decomposition of width pw.*

5.1 Lower Bound for Refl-AllOff

In the following we prove the lower bound for Refl-AllOff by a reduction from k-SAT to an equivalent instance of the minimization version of Refl-AllOff with small pathwidth.

Theorem 5 ([GSW24, Theorem 7.1])**.** *Unless the SETH fails, for all $\varepsilon > 0$, there is no algorithm for Refl-AllOff that can decide in time $(2-\varepsilon)^{\mathsf{pw}} \cdot |G|^{O(1)}$ whether there exists a solution of size at most α for a graph G that is given with a path decomposition of width* pw*.*

Proof. The proof is also presented in [GSW24].

We prove the lower bound by a reduction from k-SAT. Fix some $\varepsilon > 0$ for this and let k be the smallest integer such that k-SAT does not have a $(2-\varepsilon)^n \cdot (n+m)^{O(1)}$ algorithm where n is the number of variables and m the number of clauses.

Consider an arbitrary k-SAT formula φ with n variables $x_1, \dots, x_n$ and m clauses $C^1, \dots, C^m$ as input.[1] In the following we construct a graph G_φ as an instance of Refl-AllOff. The graph is built based on variable gadgets, clause gadgets, and a single negation gadget.

We first construct the gadgets and then describe how they are connected. For every variable x_i where $i \in [1, n]$, the *variable gadget* V_i consists of the two vertices v_i and $\bar{v}_i$ that are connected by an edge.

For every clause $C^j = \lambda_1^j \vee \dots \vee \lambda_k^j$ where $j \in [1, m]$, the *clause gadget* D^j contains the following vertices and edges. There are k *literal vertices* $t_1^j, \dots, t_k^j$ where each vertex corresponds to one literal of the clause. Moreover, the gadget D^j contains so-called *subset vertices* s_L^j for all $L \subsetneq [1, k]$, that is, for every proper subset of the literals of the clause, there exists a vertex labeled with this subset (and the gadget index). For each subset $L \subsetneq [1, k]$, the subset vertex s_L^j is connected to the literal vertex t_ℓ^j if and only if $\ell \in L$. Moreover, all subset vertices together form a clique on $2^k - 1$ vertices.

The negation gadget consists of three vertices q_0, q_1, and q_2 that are connected to a path on three vertices with q_1 in the middle.

As a last step it remains to connect the vertices of the different gadgets. Intuitively, each literal vertex of the clause gadget is connected to the corresponding variable vertex of the variable gadgets. Consider literal λ_ℓ^j, that is, the ℓth literal in the jth clause. If this literal is positive, i.e., if $\lambda_\ell^j = x_i$ for some variable x_i, then the vertex t_ℓ^j is adjacent to vertex v_i. If the literal is negative, i.e., if $\lambda_\ell^j = \neg x_i$ for some variable x_i, then the vertex t_ℓ^j is also adjacent to vertex v_i but additionally also to vertex q_1.

[1] We assume that every clause contains exactly k literals. This restriction is not of technical nature, as the constructions works for the general case, but rather to keep notation simple and clean.

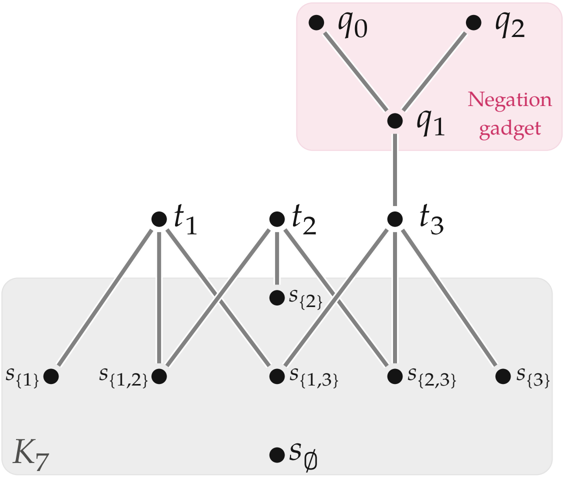


Figure 5.1: A depiction of the clause gadget for the clause $x_1 \vee x_2 \vee \neg x_3$ as well as the negation gadget. Some indices are omitted for simplicity. This figure is also presented as [GSW24, Figure 5].

This concludes the description of G_φ. See Fig. 5.1 for an illustration of the clause gadget and the negation gadget. To prove the correctness of this reduction we set $m + n + 1$ as the upper bound for the number of selected vertices.

Claim 8. *If φ is a yes-instance of k-SAT, then G_φ has a solution for* Refl-AllOff *of size at most $n + m + 1$.*

Proof. Let π be a satisfying assignment for the k-SAT formula φ. We select the following vertices:

- In the variable gadget of variable x_i, we select v_i if $\pi(x_i) = 1$ and $\bar{v}_i$ otherwise.
- In the clause gadget of clause C^j, let $L \subsetneq [1, k]$ be the set of literal indices of this clause that are *not* satisfied. (This is well-defined as π is a satisfying which means that not all literals are unsatisfied.) We select vertex s_L^j.
- We select vertex q_1 from the negation gadget.

Let S denote the set of all selected vertices.

Since we select exactly one vertex from each gadget, the size of S is precisely $n+m+1$. It remains to prove that S is indeed a solution, that is, S is a (σ, ρ)-set where $\sigma = \{0, 2, 4, \dots\}$ and $\rho = \{1, 3, 5, \dots\}$.

First, consider the vertices of the negation gadget. Since only vertex q_1 is selected, the vertices q_0 and q_2 have exactly one selected neighbors. As none of the literal vertices are selected, vertex q_1 is adjacent to zero selected vertices.

Consider any vertex of the variable gadgets and observe that none of them have a selected neighbor outside the gadget. By our choice of S, exactly one of the two vertices is selected and has no selected neighbors, while the other vertex is not selected and has exactly one selected neighbor, the one in the variable gadget.

It remains to check the vertices of the clause gadget. For the subset vertices we notice that in every clause gadget exactly one of them is selected. Recall that all subset vertices of one clause gadget are connected to each other, and they are not connected to any other vertices outside the gadget but only to the unselected literal vertices of the gadget. This implies that exactly one of the subset vertices is selected and has no selected neighbors, whereas the other subset vertices are not selected and have exactly one selected neighbor.

Next we check the literal vertices. Consider a positive literal, say $\lambda_\ell^j = x_i$. By our selection t_ℓ^j is unselected and has vertex v_i as selected neighbor if $\pi(x_i) = 1$. In this case no subset vertex s_L^j with $\ell \in L$ is selected by definition of S. Hence, the vertex has exactly one selected neighbor. If $\pi(x_i) = 0$, then the literal does not satisfy the clause and hence, by the definition of S, a vertex s_L^j is selected where $\ell \in L$.

As a last step we check the literal vertices corresponding to negated variables, say $t_\ell^j = \neg x_i$. Once more, this vertex is not selected but always adjacent to the selected vertex q_1 from the negation gadget. If $\pi(x_i) = 1$, then also the neighboring vertex v_i is selected. However, this is not a problem since in this case the literal does not satisfy the clause and hence, a subset vertex s_L^j with $\ell \in L$ must be selected. Thus, the literal vertex is adjacent to three selected vertices. If $\pi(x_i) = 0$, then the literal satisfies the clause and no subset vertex s_L^j with $\ell \in L$ is selected; the literal vertex has exactly one selected neighbor.

We conclude that every selected vertex of the graph has no selected neighbors, whereas every unselected vertex has either exactly one or exactly three selected neighbors. □

As a next step we show the reverse direction of the correctness.

Claim 9. *If G_φ has a solution for* Refl-AllOff *of size at most $n + m + 1$, then φ is a yes-instance of k-SAT.*

Proof. Consider a solution S of the Refl-AllOff instance of size at most $n + m + 1$. Recall that S is a (σ, ρ)-set for G_φ where $\sigma = \{0, 2, 4, \dots\}$ and $\rho = \{1, 3, 5, \dots\}$.

In the negation gadget at least one vertex must be selected as q_0 and q_2 must be selected themselves or require one selected neighbor which can only be q_1. Similarly, in each variable gadget V_i, the set S contains at least one vertex; vertex $\bar{v}_i$ is either selected itself or it requires a selected neighbor (which must be v_i).

In each clause gadget D^j, we must also select at least one vertex which must be a subset vertex. Indeed, the subset vertex $s_\emptyset^j$ must either be selected or have a selected neighbor. As all neighbors of this subset vertex are also subset vertices, the claim follows.

Hence, we see that any solution S of size at most $n + m + 1$ must select *exactly one* vertex of each variable gadget, as well as exactly one subset vertex of each clause gadget, and vertex q_1 of the negation gadget.

Based on these observations we define an assignment π for the formula φ by setting $\pi(x_i) = 1$ if and only if $v_i \in S$ and $\pi(x_i) = 0$ otherwise.

In the following we prove that π satisfies φ. For this consider an arbitrary clause C^j. By the above discussion, we know that there is some selected subset vertex s^j_L of the clause gadget D^j. Since $L \subsetneq [1, k]$, there is, by the construction of G_φ, a literal vertex t^j_ℓ where $\ell \in [1, k] \setminus L$. Since this vertex t^j_ℓ is not selected, it must have an odd number of neighbors in S. We first consider the case that the corresponding literal λ^j_ℓ is positive, that is, $\lambda^j_\ell = x_i$ for some variable x_i. In this case the only selected neighbor of t^j_ℓ is v_i. From $v_i \in S$ the definition of π gives $\pi(x_i) = 1$ which implies that the clause C^j is satisfied by literal λ^j_ℓ. Now consider the case when $\lambda^j_\ell = \neg x_i$ for some variable x_i. In this case vertex t^j_ℓ is adjacent to the selected vertex q_1 of the negation gadget, by the construction of G_φ and the above observations. As the literal vertex is still unselected and only adjacent to unselected subset vertices and one additional variable vertex, the variable vertex v_i cannot be selected. Hence, the definition of the assignment π gives $\pi(x_i) = 0$ which directly implies that the clause C^j is satisfied by the literal λ^j_ℓ. □

Before we combine all parts of the proof to obtain the lower bound, we first provide a bound on the pathwidth of the constructed graph.

Claim 10. *G_φ has pathwidth at most $2^k + k + n$.*

Proof. If we delete vertex q_1 and, for all $i \in [1, n]$, the variable vertex v_i, then the graph decomposes into small components of size at most $2^k - 1 + k$. This allows us to get a path decomposition of small width.

Concretely, we create a node for every variable gadget, clause gadget, and the negation gadget. We add the vertices of the corresponding gadgets to the bag of the node. We connect these nodes in an arbitrary way to form a path. As a last step we extend all bags by adding the vertices in $\{v_i \mid i \in [1, n]\} \cup \{q_1\}$. It is easy to see that this is a valid path decomposition.

The bags corresponding to the variable gadgets have size $1 + (n + 1)$ and the bags of the clause gadgets have size $2^k - 1 + k + (n + 1)$. Finally, the bag of the negation gadget has size $2 + (n + 1)$. Hence, the pathwidth of the graph is bounded by $2^k + k + n$. □

Recall that we fixed some $\varepsilon > 0$ and set k to the smallest integer such that k-SAT does not have an algorithm with running time $(2 - \varepsilon)^n \cdot n^{O(1)}$ where n is the number of variables. For a given k-SAT formula φ with n variables and m clauses we constructed an Refl-AllOff instance G_φ that has a solution of size $n + m + 1$ if and only if φ is satisfiable.

Towards a contradiction, assume that the minimization variant of Refl-AllOff can be solved in time $(2-\varepsilon)^{\mathsf{pw}} \cdot N^{O(1)}$ on instances of size N. Applying this algorithm to the constructed instance G_φ and asking for a solution of size at most $n+m+1$, yields, by using Claims 8 to 10, an algorithm for k-SAT with running time

$$(2-\varepsilon)^{\mathsf{pw}} \cdot (n+m)^{O(1)} \leq (2-\varepsilon)^{2^k+k+n} \cdot (n+m)^{O(1)} = (2-\varepsilon)^n \cdot (n+m)^{O(1)}$$

as the constant k only depends on the fixed value ε. This directly contradicts SETH and finishes the proof. □

5.2 Lower Bound for AllOff

We now proceed to AllOff, that is, (σ,ρ)-GenDomSet with $\sigma = \rho = \{1,3,\dots\}$.

Theorem 6 ([GSW24, Theorem 7.5])**.** *Unless the SETH is false, for all $\varepsilon > 0$, there is no algorithm for AllOff that can decide in time $(2-\varepsilon)^{\mathsf{pw}} \cdot |G|^{O(1)}$ whether there exists a solution of size at most α for a graph G that is given with a path decomposition of width* pw.

Proof. The proof is also presented in [GSW24].

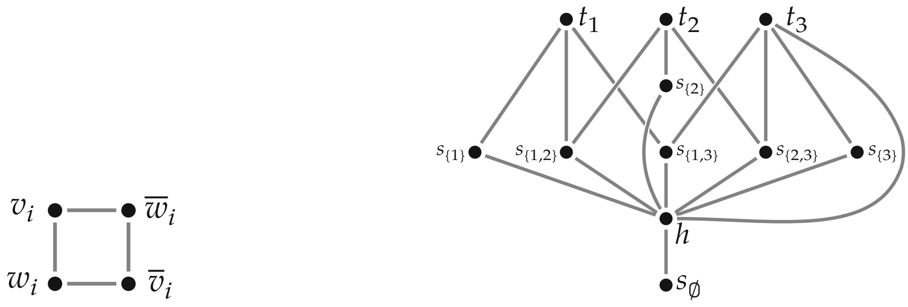
Similar to Theorem 5, we prove the bound by a reduction from k-SAT. For this fix some $\varepsilon > 0$ and let k be the smallest integer such that k-SAT has no $(2-\varepsilon)^n \cdot (n+m)^{O(1)}$ algorithm under SETH where n is the number of variables and m the number of clauses.

Let φ be an arbitrary k-SAT instance with n variables $x_1,\dots,x_n$ and m clauses $C^1,\dots,C^m$ as input.[2] We construct a graph G_φ that consists of variable and clause gadgets. See Fig. 5.2 for an illustration of the construction.

For every variable x_i where $i \in [1,n]$, the graph G_φ contains a variable gadget V_i. Gadget V_i consists of a cycle of length four on the four vertices v_i, w_i, $\bar{v}_i$, and $\bar{w}_i$ (in this order).

For every clause $C^j = \lambda_1^j \vee \dots \vee \lambda_k^j$ where $j \in [1,m]$, the graph G_φ contains a clause gadget D^j. This gadget D^j contains k literal vertices $t_1^j,\dots,t_k^j$, that is, one distinguished vertex for every literal of the clause. Additionally, D^j contains, for every subset $L \subsetneq [1,k]$, a so called *subset*-vertex s_L^j, that is, for every proper subset of the literals of the clause, there exists a vertex labeled with the subset (and the gadget index). These subset vertices are used to indicate which literals of the clause are *not* satisfied by the encoded assignment. As a last vertex there is an additional vertex h^j (for happy) in D^j. There are two different groups of edges in the gadget; first, each subset vertex s_L^j is connected to each literal vertex t_ℓ^j if and only if $\ell \in L$. Second, the vertex h^j is connected to all subset vertices of the gadget D^j.

[2] As for the proof of Theorem 5 we assume purely for the ease of the presentation that each clause contains exactly k literals.



(a) A depiction of the variable gadget.

(b) A depiction of the clause gadget for the clause $x_1 \vee x_2 \vee \neg x_3$.

Figure 5.2: A depiction of a literal gadget and a clause gadget from the proof of the lower bound for AllOff. Some indices are omitted for simplicity. This figure is also presented as [GSW24, Figure 6].

As a last step of the construction we describe the edges encoding the appearance of variables in the clauses. Intuitively each literal vertex of the clause gadget is connected to the corresponding variable vertex of the variable gadgets. Formally, if the ℓth literal of the jth clause is positive, i.e., if $\lambda_\ell^j = x_i$ for some variable x_i, then vertex t_ℓ^j is adjacent to vertex v_i. If the ℓth literal is negative, i.e., if $\lambda_\ell^j = \neg x_i$ for some variable x_i, then vertex t_ℓ^j is again adjacent to vertex v_i but also to vertex h^j.

This concludes the description of G_φ. We prove in the following that the AllOff instance G_φ has a solution of size at most $2m + 2n$ if and only if φ is satisfiable.

We first show that if the k-SAT formula φ is satisfiable, then the constructed AllOff instance is a yes-instance.

Claim 11. *If φ is a yes-instance of k-SAT, then G_φ has a solution for AllOff of size at most $2m + 2n$.*

Proof. Consider a satisfying assignment π for φ. We select the following vertices:

- For all $i \in [1, n]$, if $\pi(x_i) = 1$, then we select the two vertices v_i and w_i. If $\pi(x_i) = 0$, then we select the other two vertices of the variable gadget, which are $\bar{v}_i$ and $\bar{w}_i$.
- For all $j \in [1, m]$, we select in the clause gadget the vertex h^j. Moreover, let $L \subsetneq [1, k]$ be the set of all positions of the literals of the clause that are *not* satisfied by π. Since the clause is satisfied by the assignment, the set L cannot contain all literals of the clause. Thus, there is a subset vertex s_L^j corresponding to this set L. We select this vertex s_L^j.

Let S denote the set of all selected vertices. Clearly, this set contains exactly $2m + 2n$ vertices as we select exactly two vertices from every gadget. It remains to show that S is indeed a solution, that is, every vertex of G_φ has an odd number of neighbors in S.

Consider any vertex of the variable gadgets and observe that none of them have a selected neighbor outside the gadget. With this it is easy to see that each vertex of a variable gadget has exactly one selected neighbor in S.

Each subset vertex of a clause gadget D^j has exactly one selected neighbor, namely vertex h^j. Furthermore, vertex h^j has exactly one selected neighbor, namely the selected subset vertex of the clause gadget. Now consider an arbitrary literal vertex t^j_ℓ. First assume that the corresponding literal is positive, that is, $\lambda^j_\ell = x_i$ for some variable x_i. If this literal is satisfied, the vertex v_i in the variable gadget is selected, otherwise if the literal is not satisfied, then vertex t^j_ℓ is a neighbor of the selected subset vertex.

Now assume that the corresponding literal is negative, that is, $\lambda^j_\ell = \neg x_i$ for some variable x_i. If the variable x_i is not satisfied by π, then the only selected neighbor of t^j_ℓ is h^j. If the variable x_i is satisfied by π, then the literal λ^j_ℓ is not satisfied which implies that the literal vertex is adjacent to the selected subset vertex of this clause gadget. Moreover, vertex t^j_ℓ is adjacent to v_i and thus, to three selected vertices which is a valid number.

We conclude the proof by observing that every vertex has either one or three selected neighbors and thus, we constructed a valid solution. □

As a next step we prove the reverse direction of the correctness.

Claim 12. *If G_φ has a solution for* AllOff *of size $2m + 2n$, then φ is a yes-instance of k-SAT.*

Proof. Consider a solution S to the AllOff instance of size at most $2m + 2n$. Recall that every vertex of G_φ is adjacent to an odd number of vertices in S.

We first start with some observations about the solution S. In each variable gadget V_i, vertex $\bar{v}_i$ must have a selected neighbor in S (which is either w_i or $\bar{w}_i$), and this selected neighbor itself requires a selected neighbor in S (which is either v_i or $\bar{v}_i$). Hence, at least two vertices must be selected from each variable gadget.

In each clause gadget D^j, the vertex $s^j_\emptyset$ must have a selected neighbor which forces its only neighbor, which is vertex h^j, to be selected. The vertex h^j also requires a selected neighbor. If this neighbor was a literal vertex, say vertex t^j_ℓ, then, the subset vertex $s^j_{\{\ell\}}$ would have exactly two selected neighbors, namely h^j and t^j_ℓ. As 2 is neither in σ nor in ρ, we see that the selected neighbor is not a literal vertex, and must thus be a subset vertex. From the given bound on the solution size, we conclude that in every variable gadget and every clause gadget exactly two vertices are selected. Moreover, from each clause gadget D^j exactly one subset vertex, and vertex h^j are selected.

We define the assignment π for the variables of φ such that $\pi(x_i) = 1$ if and only if $v_i \in S$ and $\pi(x_i) = 0$ otherwise.

It remains to show that π satisfies φ. To prove this, consider an arbitrary clause C^j. Let s^j_L be the selected subset vertex from the clause gadget D^j. As $L \neq [1,k]$ by the construction of G_φ, there is some literal vertex t^j_ℓ such that $\ell \notin L$. If the corresponding literal is positive, i.e., if $\lambda^j_\ell = x_i$, then t^j_ℓ is not adjacent to h^j. As t^j_ℓ must have one selected neighbor in S and since the vertex is not adjacent to any selected subset vertex, the only remaining neighbor of t^j_ℓ, i.e., vertex v_i, must be selected which implies that π was defined such that $\pi(x_i) = 1$.

If the literal is negative, i.e., if $\lambda^j_\ell = \neg x_i$, then the vertex t^j_ℓ is adjacent to h^j by the construction of G_φ. Since h^j is selected, vertex t^j_ℓ cannot have further selected neighbors as all other adjacent subset vertices are unselected. Hence, the vertex v_i is also unselected which implies that, by the definition of π, the variable x_i is not satisfied but the literal λ^j_ℓ is satisfied which makes the clause C^j true. □

As a last step we prove a bound on the pathwidth of the constructed graph.

Claim 13. *G_φ has pathwidth at most $n + k + 1$.*

Proof. Intuitively the idea is as follows. If we delete all the variable vertices v_i for all $i \in [1,n]$ in the variable gadgets, the graph decomposes into small components. We use this to construct a path decomposition in the following by providing a node search strategy (see e.g. [CFK+15, Section 7.5]).

We start by placing one searcher on each vertex v_i for every $i \in [1,n]$. Each of the variable gadget can be cleaned by using 3 additional searchers which we just place on all vertices.

For the clause gadgets we use a more complex approach to clean all vertices. Fix a clause gadget D^j for this. We first place k new searchers on the k literal vertices of the gadget and one more searcher on the vertex h^j. The remaining subset vertices can then be cleaned by using one additional searcher which we put one subset vertex after the other. Repeating this procedure for all clause gadgets cleans the entire graph without recontamination.

This approach uses at most $n + k + 2$ searchers simultaneously. Thus, the claimed bound on the pathwidth follows immediately. □

Recall that we fixed some $\varepsilon > 0$ and chose k as the smallest integer such that k-SAT has no $(2-\varepsilon)^n \cdot (n+m)^{O(1)}$ algorithm under SETH where n is the number of variables and m the number of clauses. For a given k-SAT instance φ with n variables and m clauses we constructed an AllOff instance G_φ that has a solution of size $2m + 2n$ if and only if φ is satisfiable together with a path decomposition.

For the sake of a contradiction, now assume that the minimization version of AllOff can be solved in time $(2-\varepsilon)^{\mathsf{pw}} \cdot N^{O(1)}$ on graphs of size N. If we apply this algorithm to the constructed instance and ask for a solution of size at most $2m + 2n$, we also solve, by Claims 11 to 13 the k-SAT instance in time

$$(2-\varepsilon)^{\mathsf{pw}} \cdot N^{O(1)} \leq (2-\varepsilon)^{n+k+1} \cdot (n+m)^{O(1)} = (2-\varepsilon)^{n} \cdot (n+m)^{O(1)}$$

as k depends only on the fixed value ε and thus, only contributes a constant factor to the running time. This then directly contradicts SETH and finishes the proof. □

Now, our first main result of this section follows immediately.

Main Theorem 2 ([GSW24, Main Theorem 3])**.** *Unless SETH fails, for all $\varepsilon > 0$, there is no algorithm for each of the problems* Refl-AllOff *and* AllOff *deciding in time $(2-\varepsilon)^{\mathsf{pw}} \cdot |G|^{O(1)}$ whether there exists a solution of size at most k for a graph G that is given with a path decomposition of width* pw*.*

Proof. Is a direct consequence of Theorems 5 and 6. □

5.3 Maximization Variants

In the previous sections, we showed that, under the SETH, the minimization versions of AllOff and Refl-AllOff cannot be solved in time $(2-\varepsilon)^{\mathsf{pw}} \cdot |G|^{O(1)}$. This leaves the question of the complexity of the second optimization problem, the maximization variant, still open. We now show that the same lower bound holds for these problem variants. Moreover, the maximization problem is also interesting for the cases where $0 \in \rho$, which were trivial for the minimization problem. Hence, we show the lower bound for all four maximization problems, where σ, ρ are sets with period 2.

Similar to the approach in [HKT00b], we will first show the lower bound for the problem where $\sigma = \rho = \{0, 2, 4, \dots\}$, and later extend this result to the other problems via simple reductions.

5.3.1 The Gadgets

The proof for the case where $\sigma = \rho = \{0, 2, 4, \dots\}$ relies on two somewhat specific gadgets, the *unselection gadget* and *satisfiability gadget*. These gadgets will be utilized many times within the construction, and since their properties are less-obvious than those of the gadgets for the minimization problems, we will prove their essential properties separately.

Unselection Gadget

The first gadget we need is the so-called *unselection gadget*, which is a gadget that has an additional parameter k as part of its definition. This parameter k allows us to tweak the number of selected neighbors the portal vertex receives in one of the solutions.

Definition 18 (Unselection gadget)**.** *A graph with portals $G' = (G, \{u\})$ is an unselection gadget with advantage k and order c if and only if*

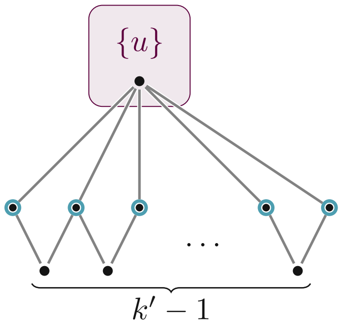


Figure 5.3: The unselection gadget for $\sigma = \rho = \{0, 2, 4, \dots\}$ and odd order. The highlighted solution is the largest partial solution that does not select the portal vertex.

- *there is a partial solution S of G' that does not select u in which u has exactly c selected neighbors,*
- *for all partial solutions $S' \neq S$ of G' we have $|S| - |S'| \geq k$.*

We now show that these gadgets can be created for a fixed c and suitable k values.

Lemma 15. *Let $\sigma = \rho = \{0, 2, 4, \dots\}$. There exists a*

- *unselection gadget with odd order, advantage at least k, size in $O(k)$, pathwidth in $O(1)$, and there exists a*
- *unselection gadget with even order, advantage at least k, size in $O(k)$, pathwidth in $O(1)$,*

for all integers $k \geq 1$.

Proof. We first proof the existence of the gadget with odd order. Set k' to the smallest odd integer that is larger or equal to $\max(k, 3)$.

Let G' be the graph created as follows. Start with a graph consisting of a center vertex u, and k' pendants $\ell_1, \dots, \ell_{k'}$ attached to u. That is, the graph thus far is a star graph with k' leaves. Moreover, add $k' - 1$ vertices $v_1, \dots, v_{k'-1}$, and connect vertex v_i to ℓ_i and ℓ_{i+1} for all $i \in [1, k' - 1]$. The gadget is then the graph with portals $G = (G', \{u\})$. We depict it in Fig. 5.3.

The size bound is clear. Regarding the pathwidth, note that the graph consists of a path on $2k' - 1$ vertices, and center vertex u. Hence, the pathwidth is constant, as we can simply take a path decomposition of the path of length $2k' - 1$ and add u to all bags.

Consider the set $S = \{\ell_i, | \; i \in [1, k']\}$. Then, each vertex ℓ_i has zero selected neighbors, which is allowed. For all $i \in [1, k' - 1]$ vertex v_i has two selected neighbors, which is also fine. Hence, S is a partial solution of size k'.

Now, consider an arbitrary partial solution S'. We first argue that it cannot be the case that $u \in S'$. For this purpose, assume that $u \in S'$. We know that vertex ℓ_1 requires either 0 or 2 selected neighbors, and since its neighbor u is selected, it requires 2 selected neighbors, which means that vertex v_1 is necessarily selected too. Then, vertex ℓ_2 has selected v_1 and u as neighbors, and hence its only other neighbor, v_2, cannot be selected. By continuing this chain of reasoning, we see that v_i, for $i \in [1, k'-1]$, is in S' if and only if i is odd. However, vertex $\ell_{k'}$ requires that vertex $v_{k'-1}$ is selected, but $k'-1$ is even, and thus $v_{k'-1}$ is not selected, a contradiction. Thus, we have $u \notin S'$. Due to this, we immediately see that vertex v_1 cannot be in S', as otherwise ℓ_1 would only have a single selected neighbor, which is not allowed. Then, v_2 cannot be in S' either because of ℓ_2. We see that vertex v_i cannot be in S' for any $i \in [1, k'-1]$.

Now, assume that for some i the vertex ℓ_i is not in S'. Then, v_i forces ℓ_{i+1} (if that vertex exists) to not be in S' either, and similarly v_{i-1} ensures that also ℓ_{i-1} (if it exists) is not in S'. Hence, no vertex of the graph can be selected, and the solution is the empty set, which is sufficiently small. Overall, we see that the only partial solutions that exist are the empty set of size 0, and the solution of size $k' \geq k$, hence the advantage claim is proven.

Next, we construct the gadget with even order. The idea is, that we can simply use the same gadget we used for the odd case twice.

Let $(G_1, \{u_1\})$ and $(G_2, \{u_2\})$ be two distinct copies of the gadget with advantage at least k and odd order we constructed in the previous paragraphs. Create graph G' by taking the disjoint union of G_1 and G_2 and identifying u_1 and u_2 into the new vertex u. Then, $(G', \{u\})$ is the gadget we were looking for.

It is clear that the size is in $O(k)$, and that the pathwidth is constant. Moreover, the number of neighbors of the center vertex is clearly even, as it is two times an odd number. Thus, we have a partial solution of even size, which is just the union of the two large partial solutions for $(G_1, \{u_1\})$ and $(G_2, \{u_2\})$. Moreover, we know that $(G_1, \{u_1\})$ and $(G_2, \{u_2\})$ only have the empty set, and the large partial solution as possible selections. Hence, it is clear that we have an advantage of at least k, as already each of the two parts of the final gadget admits this advantage. □

Satisfiability Gadget

The *satisfiability gadget* is defined next. This gadget will behave differently from the previous one, in particular, the solution that selects the portal vertex will actually be the larger solution. Hence, in a large solution for the overall graph, the portal vertex of these gadgets will usually have to be selected. Within the construction, these gadgets are employed to ensure that each clause is satisfied, hence the name.

Definition 19 (Satisfiability gadget)**.** *A graph with portals $G = (G', \{u\})$ is a satisfiability gadget with advantage k if*

- *there is a* (σ, ρ)*-set* S_1 *of* G' *that selects* u*, and a* (σ, ρ)*-set* S_2 *of* G' *that does not select* u*,*
- $|S_1| \geq k$ *and* $|S_2| = 0$*,*
- S_1 *and* S_2 *are the only partial solutions of* G*.*

Again, we show the existence of such a gadget for the considered sets σ and ρ.

Lemma 16. *If* $\sigma = \rho = \{0, 2, 4, \dots\}$*, there exists a satisfiability gadget with advantage at least* k *and pathwidth at most* 1*, and size in* $O(k)$ *for all integers* $k \geq 1$*.*

Proof. Let G' be the path $v_1, \dots, v_{2k-1}$. Then, $(G', \{v_1\})$ is the desired gadget.

It is clear that the pathwidth is at most 1, and that the size is in $O(k)$.

Moreover, the set $S = \{v_1, v_3, \dots, v_{2k-1}\}$ is a solution, every selected vertex has no selected neighbor, every unselected vertex two selected neighbors. The set S has cardinality k exactly.

Now, consider an arbitrary partial solution S'. Since v_{2k-1} requires an even number of selected neighbors, vertex v_{2k-2} cannot be selected. Then, if $v_{2k-1} \in S'$, we know that v_{2k-3} must be selected to ensure that v_{2k-2} has an even number of selected neighbors. Continuing the argumentation, we see that S' must be equal to S exactly. In the other case, $v_{2k-1} \notin S'$. Then, also $v_{2k-3} \notin S'$ as v_{2k-2} would otherwise have an odd number of selected neighbors. We see that in this case, one obtains $S' = \emptyset$, and that there are no other partial solutions. The advantage claim is thus true. □

5.3.2 The Reduction

We can now prove the lower bound for $\sigma = \rho = \{0, 2, 4, \dots\}$, that is, we prove Lemma 17.

Lemma 17. *Unless SETH fails, there is no algorithm that can decide whether a graph* G*, provided together with a path decomposition of width* pw*, has a* $(\{0, 2, 4, \dots\}, \{0, 2, 4, \dots\})$*-set of size at least* α *in time* $(2 - \varepsilon)^{\mathsf{pw}} \cdot |G|^{O(1)}$ *for any* $\varepsilon > 0$*.*

Proof. We follow the same proof template as for the other proofs in this chapter. Fix some $\varepsilon > 0$, and let k be the smallest integer such that k-SAT cannot be solved in time $(2 - \varepsilon)^n \cdot (n + m)^{O(1)}$ for instances with n variables and m clauses.

We will output a graph G of sufficiently small pathwidth, such that this graph has a large (σ, ρ)-set if and only if the input k-SAT instance is satisfiable. Let the variables of the input instance be $x_1, \dots, x_n$ and the clauses $C_1, \dots, C_m$. [3]

By Lemma 15, we can create unselection gadgets for the considered sets with constant pathwidth, advantage at least c and even/odd order for any integer $c \geq 1$. By Lemma 16,

[3]We again assume that each clause contains exactly k variables for the sake of a simpler exposition.

we can also build satisfiability gadgets with constant pathwidth and advantage c for any positive integer c.

For all these gadgets, it also holds that their size is linear in the lower bound on their advantage.

The definitions of the unselection gadgets and satisfiability gadgets allow us to play with the difference of the solution size of the solutions that select or don't select the portal vertex. For the unselection gadgets, these values will be chosen such that we can guarantee that the portal vertex is never selected. For the satisfiability gadgets, these values are chosen such that the portal vertex is only rarely unselected.

Concretely, we will use the smallest satisfiability gadgets with advantage at least $n+1$, and define selSz_S to be the size of the single solution that selects the portal vertex. Then, let $a_\mathsf{u} = n + m \cdot \mathsf{selSz}_S + 1$, and observe that, a_u is polynomial in $n \cdot m$. We will utilize the smallest unselection gadgets with advantage at least a_u of both even and odd order. Let $\mathsf{unselSz}_\mathsf{o}$ be the size of the largest solution of the unselection gadget with odd order, and $\mathsf{unselSz}_\mathsf{e}$ be the size of the largest solution of the unselection gadget with even order.

For each $i \in [1, n]$, we create a variable gadget V_i for variable x_i, that simply consists of a singleton variable vertex v_i.

For each clause C_j (for $j \in [1, m]$) consisting of the literals $\lambda_1, \ldots, \lambda_k$, we create a clause gadget. For each $i \in [1, k]$, the gadget contains an unselection gadget T_i^j:

- If λ_i is a positive literal, then T_i^j is the unselection gadget with even order.
- If λ_i is a negative literal, then T_i^j is the unselection gadget with odd order.

Denote the portal vertex of T_i^j as the *literal vertex* t_i^j.

Moreover, for each $L \subsetneq [1, k]$, we create a copy of the satisfiability gadget called S_L^j. Denote the portal vertex of S_L^j as the *subset vertex* s_L^j. Connect vertex t_i^j to vertex s_L^j if and only if $i \in L$.

Next, add a copy of the unselection gadget with even order to the graph. Denote the portal vertex of this copy as f^j. Make f^j adjacent to all subset vertices of the clause gadget of C_j. Finally, for all $i \in [1, k]$, let x_ℓ be the variable appearing in literal λ_i. Connect vertex t_i^j to vertex v_ℓ of the variable gadgets.

This concludes the description of the output graph G.

Let $\mathsf{unselSz}_\mathsf{o}$ be the number of unselection gadgets of odd order, and $\mathsf{unselSz}_\mathsf{e}$ be the number of unselection gadgets of even order. Then, the input instance is a yes-instance if and only if G has a (σ, ρ)-set of size at least $m \cdot (2^k - 2) \cdot \mathsf{selSz}_S + c_\mathsf{o} \cdot \mathsf{unselSz}_\mathsf{o} + c_\mathsf{e} \cdot \mathsf{unselSz}_\mathsf{e}$.

Claim 14. *If the input k-SAT instance is satisfiable, then the graph G has a (σ, ρ)-set of size at least $m \cdot (2^k - 2) \cdot \mathsf{selSz}_S + c_\mathsf{o} \cdot \mathsf{unselSz}_\mathsf{o} + c_\mathsf{e} \cdot \mathsf{unselSz}_\mathsf{e}$.*

Proof. Let π be a variable assignment that satisfies the input k-SAT instance.

From the variable gadgets, we select v_i if and only if $\pi(x_i) = 1$.

From all unselection gadgets, we select the vertices according to the largest solution that does not select the portal vertex.

Only the selection within the satisfiability gadgets is unclear. For this purpose, consider the clause gadget of clause C_j. Because π is a satisfying assignment, not all literals of C_j are unsatisfied by π. Let the literals of the clause be $\lambda_1, \ldots, \lambda_k$, and L be the indexes of the set of literals that are not satisfied by π. From the satisfiability gadget that contains vertex s^j_L, we select no vertex. From the other satisfiability gadgets in the same clause gadget, we choose vertices according to the solution that does select the portal vertex.

Let us now examine the size of the solution we obtain this way. From the unselection gadgets, we always select vertices according to the solution that does not select the portal vertex. This means that we select $c_{\mathsf{o}} \cdot \mathsf{unselSz}_{\mathsf{o}} + c_{\mathsf{e}} \cdot \mathsf{unselSz}_{\mathsf{e}}$ vertices in all unselection gadgets combined. Finally, we select 0 vertices from m satisfiability gadgets (one per clause), and selSz_S vertices from the remaining $m \cdot (2^k - 2)$ satisfiability gadgets. Hence, we select at least $m \cdot (2^k - 2) \cdot \mathsf{selSz}_S + c_{\mathsf{o}} \cdot \mathsf{unselSz}_{\mathsf{o}} + c_{\mathsf{e}} \cdot \mathsf{unselSz}_{\mathsf{e}}$ vertices overall, which meets the requirement.

All that remains is showing that the selection, call it S, is a (σ, ρ)-set of G. First, consider the variable gadgets. There, we know that the single vertex of such a gadget has no selected neighbors, and it is thus happy regardless of whether it is selected itself.

Now, let us take a look at the clause gadget of clause C_j for some $j \in [1, m]$. The first vertex we examine is vertex f^j. This vertex is adjacent to $2^k - 2$, an even number, of selected subset vertices, as exactly one of the $2^k - 1$ subset vertices of the clause gadget is not selected. The vertex receives an even number of selected neighbors from its unselection gadget, hence, it has an even number of selected neighbors overall, as required.

Each subset vertex is happy because it has no selected neighbor outside its own satisfiability gadget, and within the gadget the selection represents a solution.

Let us now examine the literal vertices. If literal vertex t^j_i corresponds to a literal that is satisfied by π, then, all neighbors of t^j_i that are subset vertices are selected. Hence, t^j_i has $2^{k-1} - 1$ selected neighbors that are subset vertices, which is an odd count.

If the literal of t^j_i is positive, then t^j_i receives an even number of selected neighbors from its unselection gadget. This means that from the clause gadget, t^j_i overall receives an odd number of selected neighbors. As the literal is satisfied by π, the only neighbor of t^j_i outside its own gadget, a variable vertex, is selected, and t^j_i has an even number of selected neighbors in total.

If the literal of t^j_i is negative, then t^j_i receives an odd number of selected neighbors from its unselection gadget, and an odd number of selected subset vertices. The overall even number of selected neighbors is allowed, as the literal is negative and satisfied by π, the variable vertex it has as a neighbor is not selected.

A similar chain of arguments shows that the vertex also receives a number of neighbors that is in ρ if it corresponds to a literal that is not satisfied by π. In particular, in that case, the literal vertex has an even number of selected subset vertices as neighbors. This difference compared to the case that the literal is satisfied is then offset by the fact that its adjacent variable vertex is unselected if the literal is positive, and selected if it is negative.

All other vertices are only adjacent to vertices within their own gadget, and happy because we select vertices according to a solution within each gadget. □

Next, we proof the second direction of correctness.

Claim 15. *If G has a (σ,ρ)-set of size at least $m\cdot(2^k-2)\cdot \mathsf{selSz}_S + c_{\mathsf{o}}\cdot \mathsf{unselSz}_{\mathsf{o}} + c_{\mathsf{e}}\cdot \mathsf{unselSz}_{\mathsf{e}}$, then the input k-SAT instance is satisfiable.*

Proof. Let S be a (σ,ρ)-set of G of size at least $m\cdot(2^k-2)\cdot \mathsf{selSz}_S + c_{\mathsf{o}}\cdot \mathsf{unselSz}_{\mathsf{o}} + c_{\mathsf{e}}\cdot \mathsf{unselSz}_{\mathsf{e}}$. We first need to argue that no literal vertex is selected, and all except for one subset vertex of each clause is selected in S.

Consider any variable gadget, unselection gadget, or satisfiability gadget. Per the definitions of these gadgets, there are very specific properties of partial solutions of them. In particular, these properties guarantee that specific solutions do not exist, even when the portal vertices are connected to other vertices in the rest of the graph. From each variable gadget, one can clearly select at most one vertex. Moreover, S can select at most $\mathsf{unselSz}_{\mathsf{o}}$ vertices of each unselection gadget with odd order, at most $\mathsf{unselSz}_{\mathsf{e}}$ vertices of each unselection gadget with even order. Finally, S can select at most selSz_S vertices of each satisfiability gadget. The output graph G contains no vertices outside one of these gadgets. We see that the maximum solution size is already quite close to the lower bound we require. In particular, we have just established an upper bound on the solution size of exactly $n + m\cdot(2^k-1)\cdot \mathsf{selSz}_S + c_{\mathsf{e}}\cdot \mathsf{unselSz}_{\mathsf{e}} + c_{\mathsf{o}}\cdot \mathsf{unselSz}_{\mathsf{o}}$.

The different between the lower bound of $|S|$ and the upper bound is only $n + m\cdot \mathsf{selSz}_S$. However, the advantage of any unselection gadget is larger than that, and hence, the solution within such a gadget must be the largest solution that does not select the portal vertex. Thus, the unselection gadgets fulfill their job of ensuring that their portal vertices cannot be selected, and the job of providing an odd/even number of selected neighbors to the portal vertex.

Next, consider an arbitrary clause C_j. Within the clause gadget of C_j, vertex f^j is not selected, moreover, the number of selected neighbors of f^j within the unselection gadget is even. Hence, the number of selected subset vertices within the clause gadget must be even, and in particular, not all subset vertices can be selected, as there is an odd number of them in the gadget.

This means that in each of the m clause gadgets, at most $2^k - 2$ of the subset vertices can be selected. Each subset vertex is part of a satisfiability gadget, and has no selected neighbors outside the gadget. There is only a single solution for the satisfiability gadget that selects the portal vertex, and only a single solution (of size 0) that does not select it. Hence, within a clause gadget, we can select at most $(2^k - 2) \cdot \mathsf{selSz}_S$ vertices of the satisfiability gadgets.

We know that the solution has size at least $m \cdot (2^k - 2) \cdot \mathsf{selSz}_S + c_\mathsf{o} \cdot \mathsf{unselSz}_\mathsf{o} + c_\mathsf{e} \cdot \mathsf{unselSz}_\mathsf{e}$. Moreover, at most n vertices of the variable gadget, and at most $\mathsf{unselSz}_\mathsf{o}$, respectively $\mathsf{unselSz}_\mathsf{e}$, vertices from unselection gadgets with odd and even order can be in S. This means that we must select at least $m \cdot (2^k - 2) \cdot \mathsf{selSz}_S - n$ vertices from the satisfiability gadgets. Recalling that each satisfiability gadget has advantage at least $n + 1$, we see that S must contain all subset vertices except for exactly one of each clause gadget.

Consider the variable gadget of variable x_i. We define a variable assignment π that sets variable x_i to true if and only if $v_i \in S$.

We now argue that π is a satisfying assignment. Consider a clause C_j, and let the unselected subset vertex of the respective clause gadget be s^j_L. Then, as $L \neq [1, k]$, there is some $i \in [1, k]$ such that $i \notin L$.

Hence, all subset vertices adjacent to t^j_i are selected, and we have previously also established that t^j_i, which is the portal vertex of an unselection gadget, is not selected. Moreover, this means that vertex t^j_i has exactly $2^{k-1} - 1$ selected subset vertices as neighbors, which is an odd number. In the coming arguments, let λ_i be the ith literal of C_j, and let x_ℓ be the variable appearing in λ_i.

If λ_i is a positive literal, then t^j_i has an even number of selected neighbors from its unselection gadget. Since t^j_i requires an even number of neighbors overall, its only other neighbor, v_ℓ, is selected, and then $\pi(x_\ell) = 1$, satisfying the clause.

If λ_i is a negative literal, then t^j_i has an odd number of selected neighbors from its unselection gadget. Since t^j_i requires an even number of neighbors overall, its only other neighbor, v_ℓ, cannot be selected, and then $\pi(x_\ell) = 0$, satisfying the clause.

We see that all clauses are satisfied by π, concluding the proof. □

Claim 16. *The output graph G has pathwidth $n + O(1)$ and size polynomial in $n + m$.*

Proof. We prove this by sketching how a path decomposition of such a width can be obtained and by arguing about the size of the gadgets. When the n variable vertices of the graph are deleted, the graph is decomposed into connected components, such that each connected component can only contain vertices of a single clause gadget.

Each clause gadget consists of $2^k - 1$ satisfiability gadgets, and $k + 1$ unselection gadgets. Each unselection gadget and satisfiability gadget has constant pathwidth, and size polynomial in $n + m$. As k is a constant also the size of the clause gadget is polynomial in

$n + m$, and it follows that the size of the output instance is polynomial in $n + m$ overall. Regarding the pathwidth of a clause gadget, observe that we can arbitrarily connect the path decompositions of the smaller gadgets within it, and add the k literal vertices and vertex f^j to each bag. This way, we obtain a path decomposition of a clause gadget that still has constant pathwidth.

Finally, we can obtain a path decomposition of G of width $n + O(1)$ by connecting the decompositions of each individual gadget in an arbitrary path, and adding the n variable vertices to each bag. □

To finish the proof, we proceed as usual. Assume that one can decide whether a graph G provided with a path decomposition of width pw has a (σ, ρ)-set of size at least α in time $(2-\varepsilon)^{\mathsf{pw}} \cdot |G|^{O(1)}$ for some $\varepsilon > 0$. Then, we can take an input instance of k-SAT, and compute the reduction described in this section in polynomial time. We can also compute a path decomposition of width $n+O(1)$ in polynomial time, let pw be the width of that decomposition. When applying the hypothetical fast algorithm on this output instance, we can decide k-SAT in time

$$(2-\varepsilon)^{\mathsf{pw}} \cdot |G|^{O(1)} = (2-\varepsilon)^{n+O(1)} \cdot (n+m)^{O(1)} = (2-\varepsilon)^{n} \cdot (n+m)^{O(1)},$$

which contradicts the SETH. □

5.3.3 The Other Variants

Now, we will quickly reduce this case where $\sigma = \rho = \{0, 2, 4, \dots\}$ to the cases where σ, ρ are different, but still periodic with period 2. Concretely, we will repeat reductions introduced in [HKT00b], and show that they are also pathwidth preserving.

Lemma 18 (See [HKT00b, Theorem 6])**.** *Let G be an arbitrary graph, and G' the graph obtained by adding a pendant to each vertex in G. Then G' has a $(\{1,3,5,\dots\}, \{0,2,4,\dots\})$-set of size $2k$ if and only if G has a $(\{0,2,4,\dots\}, \{0,2,4,\dots\})$-set of size k. Moreover, all $(\{1,3,5,\dots\}, \{0,2,4,\dots\})$-sets of G' have even cardinality.*

Proof. The proof is attributed to [HKT00b].

Let G and G' be like in the statement of the lemma. For any vertex $v \in V(G)$ denote its unique pendant that is present in G' but not in G as v'.

Assume that G has a $(\{0,2,4,\dots\}, \{0,2,4,\dots\})$-set X. Then, the set $S = X \cup \{v' \mid v \in X\}$ is a $(\{1,3,5,\dots\}, \{0,2,4,\dots\})$ set of G'. If $v \in V(G')$ was in $V(G)$ too, it had an even number of selected neighbors in X. If v was itself selected by X, the selected neighbor count is now odd, otherwise it remains even. If v is a new pendant, it has either zero selected neighbors if v is not in S, or one selected neighbor if v is in S, both numbers are fine.

Now, assume that G' has a $(\{1,3,5,\dots\}, \{0,2,4,\dots\})$-set S^*. Then, the set $S^* \cap V(G)$ is a $(\{0,2,4,\dots\}, \{0,2,4,\dots\})$-set of G. To see this, consider an arbitrary vertex v of G.

If v is in S^*, then the pendant of v in G' must have been selected too, as $1 \notin \{0, 2, 4, \dots\}$. As v had an odd number of selected neighbors in S^*, the number of neighbors from within G is even. If v is not in S^*, then its pendant in G' must have not been selected because $0 \notin \{1, 3, 5, \dots\}$. Thus, v also has an even number of selected neighbors in S^*.

By the fact that pendant v' of $v \in V(G)$ is selected in a solution for G' if and only if v is also selected, we see that solutions of size k are mapped to solutions of size $2k$ and the other way around, and that the cardinality of any $(\{1, 3, 5, \dots\}, \{0, 2, 4, \dots\})$-set of G' is even. □

Lemma 19 (See [HKT00b, Theorem 6])**.** *Let G be a graph, and G' the graph obtained by adding three new vertices x_i, y_i, z_i for every $v_i \in V(G)$, and adding the edges x_iy_i, x_iz_i, x_iv_i. That is, we attach the center vertex of a new path on three vertices to each vertex of G. Then, G' has a $(\{0, 2, 4, \dots\}, \{1, 3, 5, \dots\})$-set of size $|V(G)| + 2k$ if and only if G has a $(\{0, 2, 4, \dots\}, \{0, 2, 4, \dots\})$-set of size k. Moreover, the cardinality of any $(\{0, 2, 4, \dots\}, \{1, 3, 5, \dots\})$-set of G' is $|V(G)| + 2c$ for some non-negative integer c.*

Proof. The proof is attributed to [HKT00b].

Let G, G' be like in the statement of the lemma, and assume that the input graph G has a $(\{0, 2, 4, \dots\}, \{0, 2, 4, \dots\})$-set S_1 of size k. Then, we define

$$S_1' = S_1 \cup \{x_i \mid v_i \in V(G), v_i \notin S_1\} \cup \{y_i, z_i \mid v_i \in S_1\}.$$

It is easy to see that all vertices of $V(G)$ are happy in G' under the selection S_1', if the vertex was selected, it has no new neighbor that is selected now, if it was unselected, it has a new selected neighbor, and its number of selected neighbors is thus now odd. The vertices of G' which were not part of G are also all happy, in any case the selected vertices have no selected neighbors, and the unselected vertices have one or three selected neighbors. The size of S_1' is exactly $|V(G)| + 2k$.

Now, assume that G' has a $(\{0, 2, 4, \dots\}, \{1, 3, 5, \dots\})$-set S_2' of size $|V(G)| + 2k$. Take a look at the newly attached paths of length three for some $v_i \in V(G)$. If vertex x_i is selected, then vertices y_i and z_i must be unselected since $1 \notin \{0, 2, 4, \dots\}$. Then, vertex v_i must also be unselected, as x_i would otherwise have exactly one selected neighbor. On the other hand, if x_i is not selected, then y_i and z_i must necessarily be selected. To avoid an even selected neighbor count for x_i, vertex v_i must also be selected. We see that, if x_i is selected, then v_i is unselected and the number of selected neighbors v_i has in the rest of the graph must be even. If x_i is unselected, v_i is selected, and the number of selected neighbors of v_i in the remaining graph must already be even. Thus, the set $S_2 = S_2' \cap V(G)$ is a $(\{0, 2, 4, \dots\}, \{0, 2, 4, \dots\})$-set of G. Regarding the size we notice that, as only two selections are possible in the attached gadgets, a solution of size $|V(G)| + 2k$ can only be reached if we select the solution that selects v_i in k gadgets, and hence $|S_2| = k$. Finally, any $(\{0, 2, 4, \dots\}, \{1, 3, 5, \dots\})$-set of G' must have a cardinality of the form $|V(G)| + 2c$ due to the possible selections in the attached gadgets. □

Lemma 20 (See [HKT00b, Theorem 6])**.** *Let G be a graph, and G' the graph obtained by adding fresh vertices x_i, y_i, z_i, w_i to the graph for each $v_i \in V(G)$, and connecting y_i, z_i, w_i to form a clique of size three, and adding edges between x_i and v_i and between x_i and y_i. Then, G' has a $(\{1,3,5,\dots\},\{1,3,5,\dots\})$-set of size $2 \cdot |V(G)| + 2k$ if and only if G has a $(\{0,2,4,\dots\},\{0,2,4,\dots\})$-set of size k. Moreover, any $(\{1,3,5,\dots\},\{1,3,5,\dots\})$-set of G' must have a cardinality of the form $2 \cdot |V(G)| + 2c$ for some non-negative integer c.*

Proof. The proof is attributed to [HKT00b].

Let G, G' be like in the statement of the lemma. Assume that the input graph G has a $(\{0,2,4,\dots\},\{0,2,4,\dots\})$-set S_1 of size k. Then, we can build

$$S_1' = S_1 \cup \{x_i, z_i, w_i \mid v_i \in S_1\} \cup \{x_i, y_i \mid v_i \in V(G), v_i \notin S_1\},$$

which is a $(\{1,3,5,\dots\},\{1,3,5,\dots\})$-set of G'. As x_i is selected for all $v_i \in V(G)$, the number of selected neighbors of v_i in G' is odd. Moreover, x_i, y_i, z_i, w_i have exactly one or three selected neighbors in any case. The size of S_1' is exactly as required.

Now assume that G' has a $(\{1,3,5,\dots\},\{1,3,5,\dots\})$-set S_2' of size $2 \cdot |V(G)| + 2k$. We inspect the graph induced by x_i, y_i, z_i, w_i, v_i for an arbitrary $v_i \in V(G)$. As vertex x_i has exactly two neighbors, we know that either v_i or y_i must be selected, but not both. If y_i is selected, then neither w_i nor z_i can be selected, as one of them would otherwise have exactly two selected neighbors, which is not allowed. However, vertex y_i still requires a selected neighbor, so x_i is selected. If y_i is not selected, then v_i must be selected. Then, both z_i and w_i must be in the solution too, as these vertices would otherwise not have a selected neighbor. Now, as y_i requires an odd number of selected neighbors, x_i must be selected as well. We see that there are only two possible selections, and the selection that selects v_i selects four vertices of the gadget, the selection that does not select v_i selects two vertices of the gadget. Moreover, v_i gets a selected neighbor from the attached gadget in any case. Thus, the set $S_2 = S_2' \cap V(G)$ is a $(\{0,2,4,\dots\},\{0,2,4,\dots\})$-set of G of size exactly k. Finally, also the claim about the cardinalities of $(\{1,3,5,\dots\},\{1,3,5,\dots\})$-sets of G' follows since only two selections are possible within each attached gadget. □

Having these reductions out of the way, we can prove the final main result of the thesis.

Main Theorem 3. *Let σ, ρ be periodic sets with period 2. Unless SETH fails, for all $\varepsilon > 0$, there is no algorithm for (σ,ρ)-GenDomSet deciding in time $(2-\varepsilon)^{\mathsf{pw}} \cdot |G|^{O(1)}$ whether there exists a solution of size at least k for a graph G that is given with a path decomposition of width pw.*

Proof. By Lemma 17 the statement holds when $\sigma = \rho = \{0,2,4,\dots\}$. For the other cases, the reductions given in Lemmas 18 to 20 provide reductions that extend the claim to these problems.

Concretely, they all prove claims about graphs G' that are created from an input graph G by attaching constant sized gadgets to each vertex of G. Given a path decomposition of

G and a vertex $v \in V(G)$, we can take an arbitrary bag that contains v, copy it, and add all vertices of the gadget attached to v to the bag. If we do this cleverly, we will obtain a path decomposition of G' and only increase the pathwidth by a constant amount, as the vertices of at most one gadget are added to a single bag. So, it is clear that we can create a path decomposition of width $\mathsf{pw} + O(1)$ for G' in polynomial time when having a path decomposition of width pw of G as input.

We now consider the remaining three cases individually. If we have $\sigma = \{1, 3, 5, \dots\}$, and $\rho = \{0, 2, 4, \dots\}$, then we use Lemma 18 that shows G' has a $(\{1, 3, 5, \dots\}, \{0, 2, 4, \dots\})$-set of size $2k$ if and only if G has a $(\{0, 2, 4, \dots\}, \{0, 2, 4, \dots\})$-set of size k. If we have $\sigma = \{0, 2, 4, \dots\}, \rho = \{1, 3, 5, \dots\}$, then Lemma 19 shows that the graph G' has a $(\{0, 2, 4, \dots\}, \{1, 3, 5, \dots\})$-set of size $|V(G)| + 2k$ if and only if the graph G has a $(\{0, 2, 4, \dots\}, \{0, 2, 4, \dots\})$-set of size k. Similarly, Lemma 20 works for the case that $\sigma = \rho = \{1, 3, 5, \dots\}$.

The argumentation works the same way for all three of these cases, so it will be illustrated for the case that $\sigma = \{0, 2, 4, \dots\}, \rho = \{1, 3, 5, \dots\}$. If G has a $(\{0, 2, 4, \dots\}, \{0, 2, 4, \dots\})$-set of size at least k, then G' has a $(\{0, 2, 4, \dots\}, \{1, 3, 5, \dots\})$-set of size $|V(G)| + 2k$. On the other hand, if G' has a $(\{0, 2, 4, \dots\}, \{1, 3, 5, \dots\})$-set S of size at least $|V(G)| + 2k$, then, by Lemma 19, $|S| = |V(G)| + 2c$ for some non-negative integer $c \geq k$. Hence, G has a $(\{0, 2, 4, \dots\}, \{0, 2, 4, \dots\})$-set of size $c \geq k$.

An algorithm for the maximization variant where $\sigma = \{0, 2, 4, \dots\}, \rho = \{1, 3, 5, \dots\}$ with running time $(2-\varepsilon)^{\mathsf{pw}} \cdot |G|^{O(1)}$ would immediately yield an algorithm with a running time of the same form for the case where $\sigma = \rho = \{0, 2, 4, \dots\}$, because the reduction above can be performed in polynomial time, and a path decomposition of G can be transformed into a path decomposition for G' in polynomial time while only increasing the pathwidth by a constant amount. By Lemma 17 this would contradict the SETH. □

CHAPTER 6

Conclusion

The thesis dealt with the (σ, ρ)-GenDomSet problem with periodic sets σ and ρ, that is, σ and ρ are residue classes modulo m, for some fixed $\mathrm{m} \geq 2$. The problem was investigated using the paradigm of parameterized complexity, concretely utilizing the common parameter treewidth. Literature related to the topics treewidth, parameterized complexity, (σ, ρ)-GenDomSet and the natural problem Lights Out was examined.

In the main contribution, the focus was on proving lower bounds based on the *Strong Exponential Time Hypothesis*. Concretely, it was shown that as long as σ, ρ are not residue classes modulo 2 and $0 \notin \rho$, it is not possible to decide whether a (σ, ρ)-set exists for a given input graph G provided with a path decomposition of width pw in time $(\mathrm{m} - \varepsilon)^{\mathsf{pw}} \cdot |G|^{O(1)}$ for any $\varepsilon > 0$, unless the SETH is false. Moreover, for the case where σ, ρ are residue classes modulo 2, it was shown that under the SETH the maximization problem cannot be solved in time $(2 - \varepsilon)^{\mathsf{pw}} \cdot |G|^{O(1)}$, and if $0 \notin \rho$, the same holds true for the minimization problems. Thus, assuming the SETH, the precise complexity for the problem is known in all scenarios for sets with the same period, except for the cases where the period is larger than 3 and $0 \in \rho$.

Together with the results by Focke et al. [FMI+23a], we have now obtained a reasonable picture of the exact complexity of the problem. Still, there are numerous further possibilities that should be examined. Firstly, although Focke et al. [FMI+23a] conclusively settle the question of the complexity for finite σ, ρ for the *decision* problem, the complexity of the *maximization* problem is not yet clear. In particular, their bounds do not apply to the case where $0 \in \rho$, however, one would expect the maximization problem to be hard in that case.

Similarly, our bounds only work for the case where $0 \in \rho$ when the period is 2. Hence, showing tight lower bounds for $0 \in \rho$ and a larger period is still an open problem. The case where $0 \in \rho$ generally presents a challenge, as the empty set is always a solution, and hence, interesting interactions between vertices can only be obtained by considering the solution size. Thus, studying the problem with $0 \in \rho$ could be of interest.

The considered scenario where σ and ρ are periodic sets with the same period is only a first step in providing algorithms for sets that are neither finite nor cofinite. In general, we only know that the problem is FPT when σ and ρ are *ultimately periodic* sets [Cha10], and that it is presumably not FPT in certain cases. It would be a good next step to precisely classify which properties σ and ρ must have for the problem to be FPT.

Moreover, bounds for the problem for different infinite sets than those we have considered are also of interest. For instance, one could consider the scenario where σ, ρ are periodic sets with a different period, or the scenario where one set is finite and the other periodic. For this situation, we do not even know what the concrete running time should be (although our work and the work by Focke et al. [FMI+23a] indicates a certain running time). Then, when an algorithm is obtained, showing that it is optimal under SETH would of course also be the ultimate goal. Another natural scenario would be considering sets σ, ρ that have the form $c + \mathrm{m} \cdot \mathbb{N}$, that is, σ, ρ are residue classes modulo m which are shifted by a potentially large constant. Especially in this scenario, the creation of a quick algorithm could be interesting, as it seems that one would perhaps even require new convolution techniques to obtain a good running time. Finally, considering sets that are unions of different residue classes, that is, unions of periodic sets with the same period, is another natural extension of our setting that one might want to take a look at.

As one can see by this discussion, we are far from settling the complexity of (σ, ρ)-GenDomSet parameterized by treewidth. Nonetheless, this work represents a first important step into the domain of sets that are neither cofinite nor finite, and it can serve as reference and basis for further work in the area.

Übersicht verwendeter Hilfsmittel

Es wurden keine generativen KI-Tools zur Erstellung der Arbeit verwendet.

Bibliography

[ABHS22] Amir Abboud, Karl Bringmann, Danny Hermelin, and Dvir Shabtay. SETH-based lower bounds for subset sum and bicriteria path. *ACM Transactions on Algorithms*, 18(1):6:1–6:22, 2022.

[ABR+10] Isolde Adler, Binh-Minh Bui-Xuan, Yuri Rabinovich, Gabriel Renault, Jan Arne Telle, and Martin Vatshelle. On the boolean-width of a graph: Structure and applications. In Dimitrios M. Thilikos, editor, *Graph Theoretic Concepts in Computer Science - 36th International Workshop, WG 2010, Zarós, Crete, Greece, June 28-30, 2010 Revised Papers*, volume 6410 of *Lecture Notes in Computer Science*, pages 159–170, 2010.

[ABW15] Amir Abboud, Arturs Backurs, and Virginia Vassilevska Williams. Tight hardness results for LCS and other sequence similarity measures. In Venkatesan Guruswami, editor, *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, pages 59–78. IEEE Computer Society, 2015.

[ACP87] Stefan Arnborg, Derek G. Corneil, and Andrzej Proskurowski. Complexity of finding embeddings in a k-Tree. *SIAM Journal on Algebraic Discrete Methods*, 8(2):277–284, 1987.

[AEFM89] Karl R. Abrahamson, John A. Ellis, Michael R. Fellows, and Manuel E. Mata. On the Complexity of Fixed Parameter Problems (Extended Abstract). In *30th Annual Symposium on Foundations of Computer Science, Research Triangle Park, North Carolina, USA, 30 October - 1 November 1989*, pages 210–215. IEEE Computer Society, 1989.

[AF98] Marlow Anderson and Todd Feil. Turning Lights Out with Linear Algebra. *Mathematics Magazine*, 71(4):300–303, October 1998.

[BBH21] Abraham Berman, Franziska Borer, and Norbert Hungerbühler. Lights Out on graphs. *Mathematische Semesterberichte*, 68(2):237–255, October 2021.

[BGS17] Huck Bennett, Alexander Golovnev, and Noah Stephens-Davidowitz. On the quantitative hardness of CVP. In Chris Umans, editor, *58th IEEE Annual*

Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017, pages 13–24. IEEE Computer Society, 2017.

[BIS17] Arturs Backurs, Piotr Indyk, and Ludwig Schmidt. On the fine-grained complexity of empirical risk minimization: Kernel methods and neural networks. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 4308–4318, 2017.

[BL16] Glencora Borradaile and Hung Le. Optimal dynamic program for r-Domination problems over tree decompositions. In Jiong Guo and Danny Hermelin, editors, *11th International Symposium on Parameterized and Exact Computation, IPEC 2016, August 24-26, 2016, Aarhus, Denmark*, volume 63 of *LIPIcs*, pages 8:1–8:23. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2016.

[BM93] Hans L. Bodlaender and Rolf H. Möhring. The pathwidth and treewidth of cographs. *SIAM J. Discret. Math.*, 6(2):181–188, 1993.

[Bod98] Hans L. Bodlaender. A partial k-arboretum of graphs with bounded treewidth. *Theoretical Computer Science*, 209(1-2):1–45, 1998.

[BvvV10] Hans L. Bodlaender, Erik Jan van Leeuwen, Johan M. M. van Rooij, and Martin Vatshelle. Faster algorithms on branch and clique decompositions. In Petr Hlinený and Antonín Kucera, editors, *Mathematical Foundations of Computer Science 2010, 35th International Symposium, MFCS 2010, Brno, Czech Republic, August 23-27, 2010. Proceedings*, volume 6281 of *Lecture Notes in Computer Science*, pages 174–185. Springer, 2010.

[CCF+05] Jianer Chen, Benny Chor, Mike Fellows, Xiuzhen Huang, David W. Juedes, Iyad A. Kanj, and Ge Xia. Tight lower bounds for certain parameterized NP-hard problems. *Inf. Comput.*, 201(2):216–231, 2005.

[CDL+16] Marek Cygan, Holger Dell, Daniel Lokshtanov, Dániel Marx, Jesper Nederlof, Yoshio Okamoto, Ramamohan Paturi, Saket Saurabh, and Magnus Wahlström. On problems as hard as CNF-SAT. *ACM Transactions on Algorithms*, 12(3):41:1–41:24, 2016.

[CFG+16] Marek Cygan, Fedor V. Fomin, Alexander Golovnev, Alexander S. Kulikov, Ivan Mihajlin, Jakub Pachocki, and Arkadiusz Socala. Tight bounds for graph homomorphism and subgraph isomorphism. In Robert Krauthgamer, editor, *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 1643–1649. SIAM, 2016.

[CFK+15] Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015.

[CGK01] Yair Caro, John L. Goldwasser, and William Klostermeyer. Odd and residue domination numbers of a graph. *Discuss. Math. Graph Theory*, 21(1):119–136, 2001.

[Cha10] Mathieu Chapelle. Parameterized complexity of generalized domination problems on bounded tree-width graphs. *CoRR*, abs/1004.2642v5, 2010.

[CHKX06] Jianer Chen, Xiuzhen Huang, Iyad A. Kanj, and Ge Xia. Strong computational lower bounds via parameterized complexity. *Journal of Computer and System Sciences*, 72(8):1346–1367, 2006.

[CIP09] Chris Calabro, Russell Impagliazzo, and Ramamohan Paturi. The complexity of satisfiability of small depth circuits. In Jianer Chen and Fedor V. Fomin, editors, *Parameterized and Exact Computation, 4th International Workshop, IWPEC 2009, Copenhagen, Denmark, September 10-11, 2009, Revised Selected Papers*, volume 5917 of *Lecture Notes in Computer Science*, pages 75–85. Springer, 2009.

[CJ03] Liming Cai and David W. Juedes. On the existence of subexponential parameterized algorithms. *Journal of Computer and System Sciences*, 67(4):789–807, 2003.

[CKX10] Jianer Chen, Iyad A. Kanj, and Ge Xia. Improved upper bounds for vertex cover. *Theoretical Computer Science*, 411(40-42):3736–3756, 2010.

[CM16] Radu Curticapean and Dániel Marx. Tight conditional lower bounds for counting perfect matchings on graphs of bounded treewidth, cliquewidth, and genus. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 1650–1669, 2016.

[CNP+22] Marek Cygan, Jesper Nederlof, Marcin Pilipczuk, Michal Pilipczuk, Johan M. M. van Rooij, and Jakub Onufry Wojtaszczyk. Solving connectivity problems parameterized by treewidth in single exponential time. *ACM Transactions on Algorithms*, 18(2):17:1–17:31, 2022.

[CPP16] Marek Cygan, Marcin Pilipczuk, and Michal Pilipczuk. Known algorithms for edge clique cover are probably optimal. *Siam Journal On Computing*, 45(1):67–83, 2016.

[DF92a] Rodney G. Downey and Michael R. Fellows. Fixed-parameter intractability. In *Proceedings of the Seventh Annual Structure in Complexity Theory Conference, Boston, Massachusetts, USA, June 22-25, 1992*, pages 36–49. IEEE Computer Society, 1992.

[DF92b] Rodney G. Downey and Michael R. Fellows. Fixed parameter tractability and completeness. *Congressus Numerantium*, 87:161–178, 1992.

[DF95a] Rodney G. Downey and Michael R. Fellows. Fixed-parameter tractability and completeness I: Basic results. *Siam Journal On Computing*, 24(4):873–921, 1995.

[DF95b] Rodney G. Downey and Michael R. Fellows. Fixed-parameter tractability and completeness II: On completeness for W[1]. *Theoretical Computer Science*, 141(1&2):109–131, 1995.

[DF13] Rodney G. Downey and Michael R. Fellows. *Fundamentals of Parameterized Complexity*. Texts in Computer Science. Springer, 2013.

[DHM+14] Holger Dell, Thore Husfeldt, Dániel Marx, Nina Taslaman, and Martin Wahlen. Exponential time complexity of the permanent and the tutte polynomial. *ACM Transactions on Algorithms*, 10(4):21:1–21:32, 2014.

[DW01] Yevgeniy Dodis and Peter Winkler. Universal configurations in light-flipping games. In S. Rao Kosaraju, editor, *Proceedings of the Twelfth Annual Symposium on Discrete Algorithms, January 7-9, 2001, Washington, DC, USA*, pages 926–927. ACM/SIAM, 2001.

[FFL+11] Michael R. Fellows, Fedor V. Fomin, Daniel Lokshtanov, Frances A. Rosamond, Saket Saurabh, Stefan Szeider, and Carsten Thomassen. On the complexity of some colorful problems parameterized by treewidth. *Information and Computation*, 209(2):143–153, 2011.

[FGK+09] Fedor V. Fomin, Petr A. Golovach, Jan Kratochvíl, Dieter Kratsch, and Mathieu Liedloff. Sort and search: Exact algorithms for generalized domination. *Inf. Process. Lett.*, 109(14):795–798, 2009.

[FGK+11] Fedor V. Fomin, Petr A. Golovach, Jan Kratochvíl, Dieter Kratsch, and Mathieu Liedloff. Branch and recharge: Exact algorithms for generalized domination. *Algorithmica*, 61(2):252–273, 2011.

[FGLS14] Fedor V. Fomin, Petr A. Golovach, Daniel Lokshtanov, and Saket Saurabh. Almost optimal lower bounds for problems parameterized by clique-width. *Siam Journal On Computing*, 43(5):1541–1563, 2014.

[FMI+23a] Jacob Focke, Dániel Marx, Fionn Mc Inerney, Daniel Neuen, Govind S. Sankar, Philipp Schepper, and Philip Wellnitz. Tight complexity bounds for counting generalized dominating sets in bounded-treewidth graphs. In Nikhil Bansal and Viswanath Nagarajan, editors, *Proceedings of the 2023 ACM-SIAM Symposium on Discrete Algorithms, SODA 2023, Florence, Italy, January 22-25, 2023*, pages 3664–3683. SIAM, 2023.

[FMI+23b] Jacob Focke, Dániel Marx, Fionn Mc Inerney, Daniel Neuen, Govind S. Sankar, Philipp Schepper, and Philip Wellnitz. Tight complexity bounds for counting generalized dominating sets in bounded-treewidth graphs part I: Algorithmic results. *CoRR*, abs/2211.04278, 2023.

[FMI+23c] Jacob Focke, Dániel Marx, Fionn Mc Inerney, Daniel Neuen, Govind S. Sankar, Philipp Schepper, and Philip Wellnitz. Tight complexity bounds for counting generalized dominating sets in bounded-treewidth graphs part II: Hardness results. *CoRR*, abs/2306.03640, 2023.

[FT08] Fedor V. Fomin and Dimitrios M. Thilikos. An annotated bibliography on guaranteed graph searching. *Theoretical Computer Science*, 399(3):236–245, 2008.

[FY13] Rudolf Fleischer and Jiajin Yu. A survey of the game "Lights Out!". In Andrej Brodnik, Alejandro López-Ortiz, Venkatesh Raman, and Alfredo Viola, editors, *Space-Efficient Data Structures, Streams, and Algorithms - Papers in Honor of J. Ian Munro on the Occasion of His 66th Birthday*, volume 8066 of *Lecture Notes in Computer Science*, pages 176–198. Springer, 2013.

[GGHN08] Jens Gramm, Jiong Guo, Falk Hüffner, and Rolf Niedermeier. Data reduction and exact algorithms for clique cover. *ACM Journal of Experimental Algorithmics*, 13, 2008.

[GH08] Elisabeth Gassner and Johannes Hatzl. A parity domination problem in graphs with bounded treewidth and distance-hereditary graphs. *Computing*, 82(2-3):171–187, 2008.

[GJT76] M. R. Garey, David S. Johnson, and Robert Endre Tarjan. The planar hamiltonian circuit problem is NP-complete. *Siam Journal On Computing*, 5(4):704–714, 1976.

[GK07] John L. Goldwasser and William Klostermeyer. Odd and Even Dominating Sets with Open Neighborhoods. *Ars Comb.*, 83, 2007.

[GKS12] Petr A. Golovach, Jan Kratochvíl, and Ondrej Suchý. Parameterized complexity of generalized domination problems. *Discret. Appl. Math.*, 160(6):780–792, 2012.

[GKT97] John Goldwasser, William Klostermeyer, and George Trapp. Characterizing switch-setting problems. *Linear and Multilinear Algebra*, 43(1-3):121–135, January 1997.

[GKTZ95] John Goldwasser, William Klostermeyer, George Trapp, and CQ Zhang. Setting switches on a grid. Technical Report Technical Report TR-95-20, Department of Statistics and Computer Science, West Virginia University,, 1995.

[GSW24] Jakob Greilhuber, Philipp Schepper, and Philip Wellnitz. Shining light on periodic dominating sets in bounded-treewidth graphs. *CoRR*, abs/2403.07524v1, 2024.

[HKT00a] Magnús M. Halldórsson, Jan Kratochvíl, and Jan Arne Telle. Independent sets with domination constraints. *Discret. Appl. Math.*, 99(1-3):39–54, 2000.

[HKT00b] Magnús M. Halldórsson, Jan Kratochvíl, and Jan Arne Telle. Mod-2 Independence and Domination in Graphs. *Int. J. Found. Comput. Sci.*, 11(3):355–363, 2000.

[HT98] Pinar Heggernes and Jan Arne Telle. Partitioning graphs into generalized dominating sets. *Nord. J. Comput.*, 5(2):128–142, 1998.

[IP01] Russell Impagliazzo and Ramamohan Paturi. On the complexity of k-SAT. *Journal of Computer and System Sciences*, 62(2):367–375, 2001.

[IPZ01] Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity? *Journal of Computer and System Sciences*, 63(4):512–530, 2001.

[JKST19] Lars Jaffke, O-joung Kwon, Torstein J. F. Strømme, and Jan Arne Telle. Mim-width III. Graph powers and generalized distance domination problems. *Theoretical Computer Science*, 796:216–236, 2019.

[JRS19] Ashwin Jacob, Venkatesh Raman, and Vibha Sahlot. Deconstructing parameterized hardness of fair vertex deletion problems. In Ding-Zhu Du, Zhenhua Duan, and Cong Tian, editors, *Computing and Combinatorics - 25th International Conference, COCOON 2019, Xi'an, China, July 29-31, 2019, Proceedings*, volume 11653 of *Lecture Notes in Computer Science*, pages 325–337. Springer, 2019.

[KL23] Tuukka Korhonen and Daniel Lokshtanov. An improved parameterized algorithm for treewidth. In Barna Saha and Rocco A. Servedio, editors, *Proceedings of the 55th Annual ACM Symposium on Theory of Computing, STOC 2023, Orlando, FL, USA, June 20-23, 2023*, pages 528–541. ACM, 2023.

[KP85] Lefteris M. Kirousis and Christos H. Papadimitriou. Interval graphs and searching. *Discret. Math.*, 55(2):181–184, 1985.

[KP86] Lefteris M. Kirousis and Christos H. Papadimitriou. Searching and pebbling. *Theoretical Computer Science*, 47(3):205–218, 1986.

[Lam20] Michael Lampis. Finer tight bounds for coloring on clique-width. *SIAM J. Discret. Math.*, 34(3):1538–1558, 2020.

[LMS11] Daniel Lokshtanov, Dániel Marx, and Saket Saurabh. Lower bounds based on the exponential time hypothesis. *Bull. EATCS*, 105:41–72, 2011.

[LMS18a] Daniel Lokshtanov, Dániel Marx, and Saket Saurabh. Known algorithms on graphs of bounded treewidth are probably optimal. *ACM Transactions on Algorithms*, 14(2):13:1–13:30, 2018.

[LMS18b] Daniel Lokshtanov, Dániel Marx, and Saket Saurabh. Slightly superexponential parameterized problems. *Siam Journal On Computing*, 47(3):675–702, 2018.

[LT02] Chin Lung Lu and Chuan Yi Tang. Weighted efficient domination problem on some perfect graphs. *Discrete Applied Mathematics*, 117(1-3):163–182, 2002.

[Mar07] Dániel Marx. On the optimality of planar and geometric approximation schemes. In *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2007), October 20-23, 2007, Providence, RI, USA, Proceedings*, pages 338–348. IEEE Computer Society, 2007.

[Mar10] Dániel Marx. Can you beat treewidth? *Theory of Computing. An Open Access Journal*, 6(1):85–112, 2010.

[MFMP20] Mohsen Alambardar Meybodi, Fedor V. Fomin, Amer E. Mouawad, and Fahad Panolan. On the parameterized complexity of [1, j]-domination problems. *Theoretical Computer Science*, 804:207–218, 2020.

[MM16] Dániel Marx and Valia Mitsou. Double-exponential and triple-exponential bounds for choosability problems parameterized by treewidth. In Ioannis Chatzigiannakis, Michael Mitzenmacher, Yuval Rabani, and Davide Sangiorgi, editors, *43rd International Colloquium on Automata, Languages, and Programming, ICALP 2016, July 11-15, 2016, Rome, Italy*, volume 55 of *LIPIcs*, pages 28:1–28:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2016.

[MSS21] Dániel Marx, Govind S. Sankar, and Philipp Schepper. Degrees and gaps: Tight complexity results of general factor problems parameterized by treewidth and cutwidth. In Nikhil Bansal, Emanuela Merelli, and James Worrell, editors, *48th International Colloquium on Automata, Languages, and Programming, ICALP 2021, July 12-16, 2021, Glasgow, Scotland (Virtual Conference)*, volume 198 of *LIPIcs*, pages 95:1–95:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.

[MSS22] Dániel Marx, Govind S. Sankar, and Philipp Schepper. Anti-factor is FPT parameterized by treewidth and list size (but counting is hard). In Holger Dell and Jesper Nederlof, editors, *17th International Symposium on Parameterized and Exact Computation, IPEC 2022, September 7-9, 2022,* *Potsdam, Germany*, volume 249 of *LIPIcs*, pages 22:1–22:23. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.

[PW10] Mihai Pătraşcu and Ryan Williams. On the possibility of faster SAT algorithms. In Moses Charikar, editor, *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2010, Austin, Texas, USA, January 17-19, 2010*, pages 1065–1075. SIAM, 2010.

[RS83] Neil Robertson and Paul D. Seymour. Graph minors. I. Excluding a forest. *J. Comb. Theory, Ser. B*, 35(1):39–61, 1983.

[RS86] Neil Robertson and Paul D. Seymour. Graph minors. II. algorithmic aspects of tree-width. *Journal of Algorithms*, 7(3):309–322, 1986.

[RW13] Liam Roditty and Virginia Vassilevska Williams. Fast approximation algorithms for the diameter and radius of sparse graphs. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 515–524. ACM, 2013.

[Sut88] Klaus Sutner. Additive Automata On Graphs. *Complex Syst.*, 2(6), 1988.

[Sut89] Klaus Sutner. Linear cellular automata and the garden-of-eden. *The Mathematical Intelligencer*, 11(2):49–53, March 1989.

[Tel94] Jan Arne Telle. Complexity of domination-type problems in graphs. *Nord. J. Comput.*, 1(1):157–171, 1994.

[TP93] Jan Arne Telle and Andrzej Proskurowski. Practical algorithms on partial k-Trees with an application to domination-like problems. In Frank K. H. A. Dehne, Jörg-Rüdiger Sack, Nicola Santoro, and Sue Whitesides, editors, *Algorithms and Data Structures, Third Workshop, WADS '93, Montréal, Canada, August 11-13, 1993, Proceedings*, volume 709 of *Lecture Notes in Computer Science*, pages 610–621. Springer, 1993.

[van20] Johan M. M. van Rooij. Fast algorithms for join operations on tree decompositions. In Fedor V. Fomin, Stefan Kratsch, and Erik Jan van Leeuwen, editors, *Treewidth, Kernels, and Algorithms - Essays Dedicated to Hans L. Bodlaender on the Occasion of His 60th Birthday*, volume 12160 of *Lecture Notes in Computer Science*, pages 262–297. Springer, 2020.

[van21] Johan M. M. van Rooij. A generic convolution algorithm for join operations on tree decompositions. In Rahul Santhanam and Daniil Musatov, editors, *Computer Science - Theory and Applications - 16th International Computer Science Symposium in Russia, CSR 2021, Sochi, Russia, June 28 - July 2, 2021, Proceedings*, volume 12730 of *Lecture Notes in Computer Science*, pages 435–459. Springer, 2021.

[vBR09] Johan M. M. van Rooij, Hans L. Bodlaender, and Peter Rossmanith. Dynamic programming on tree decompositions using generalised fast subset convolution. In Amos Fiat and Peter Sanders, editors, *Algorithms - ESA 2009, 17th Annual European Symposium, Copenhagen, Denmark, September 7-9, 2009. Proceedings*, volume 5757 of *Lecture Notes in Computer Science*, pages 566–577. Springer, 2009.