University of Warsaw
Faculty of Mathematics, Informatics and Mechanics

Katarzyna Kowalska
Student no. 371053

Approximation and Parameterized Algorithms for Segment Set Cover

Master's thesis
in COMPUTER SCIENCE

Supervisor:
dr hab Michał Pilipczuk
Institute of Informatics

June 2022

Supervisor's statement

Hereby I confirm that the presented thesis was prepared under my supervision and that it fulfils the requirements for the degree of Master of Computer Science.

Date Supervisor's signature

Author's statement

Hereby I declare that the presented thesis was prepared by me and none of its contents was obtained by means that are against the law.

The thesis has never before been a subject of any procedure of obtaining an academic degree.

Moreover, I declare that the present version of the thesis is identical to the attached electronic version.

Date Author's signature

Abstract

In this thesis we study approximation and parameterized algorithms for a variant of the Set Cover problem, where the universe of elements to cover are points in the plane, and sets to cover objects with are segments. We call this problem Segment Set Cover. We also consider the problem relaxed with δ-extension, where we need to cover the points by segments, which are extended by a tiny fraction, but we compare the solution size to the optimum solution without extension. We prove that Segment Set Cover is APX-hard even if we restrict segments to be axis-parallel and allow $\frac{1}{2}$-extension. We provide FPT algorithms for unweighted Segment Set Cover parameterized by the size of the solution k and for Weighted Segment Set Cover with δ-extension. Finally, we prove that Weighted Segment Set Cover is W[1]-hard and there does not exist an algorithm running in time $f(k) \cdot n^{o(\sqrt{k})}$ solving this problem even if we restrict the segments to 3 directions.

Keywords

geometric set cover, weighted set cover, FPT, W[1]-hard, APX-hard

Thesis domain (Socrates-Erasmus subject area codes)

11.3 Informatyka

Subject classification

Theory of computation → Design and analysis of algorithms → Parameterized complexity and exact algorithms
Theory of computation → Design and analysis of algorithms → Approximation algorithms analysis → Packing and covering problems

Tytuł pracy w języku polskim

Algorytmy aproksymacyjne i parametryzowane dla problemu pokrywania punktów odcinkami na płaszczyźnie

Contents

Chapter 1

Introduction

1.1. Background

Some problems in Computer Science are known to be NP-complete, meaning that assuming P≠NP there is no polynomial-time algorithm that can solve these problems. Even so, they still can be amenable to different approaches, such as approximation or parameterization.

Definition 1.1. In the **Set Cover** problem we are given a set of elements (universe) $\mathcal{C}$ and a family of sets $\mathcal{P}$ that are subsets of the universe $\mathcal{C}$ and sum up to the whole $\mathcal{C}$. Our task is to find a set $\mathcal{R} \subseteq \mathcal{P}$ such that $\bigcup \mathcal{R} = \mathcal{C}$ and the size of $\mathcal{R}$ is minimum possible.

Set Cover is a classical example of an NP-complete problem, which has been proven in [Dinur and Steurer, 2014] to be inapproximable with factor $(1 - o(1)) \ln n$ assuming $\mathrm{P} \neq \mathrm{NP}$ (which is a stronger result than APX-hardness), and W[2]-complete with the natural parameterization, see Theorem 13.21 in [Cygan et al., 2015]. However, restricting the problem to various specialized settings can lead to more tractable special cases. In this thesis we take a closer look at the Geometric Set Cover problem in the plane, where elements to cover are points in the plane and sets to cover them with are geometric objects.

Definition 1.2. Segment Set Cover is Geometric Set Cover where objects that we cover the points with are segments in the plane.

Approximation Over the years there has been a lot of work related to approximation algorithms for Geometric Set Cover. Notably, Geometric Set Cover with unweighted unit disks admits a PTAS (see Corollary 1.1 in [Mustafa and Ray, 2010]). When we consider the same problem with weighted unit disks (or unit squares), the problem admits a QPTAS [Mustafa et al., 2014], see also [Pilipczuk et al., 2020]. On the other hand, [Chan and Grant, 2014] proved that Geometric Set Cover with unweighted axis-parallel fat rectangles is APX-hard; they also show similar hardness for Geometric Set Cover with many other standard geometric objects.

Parameterization We consider Geometric Set Cover parameterized by the size of solution. Geometric Set Cover with unit squares was first proven to be W[1]-hard in [Marx, 2005] (Theorem 5). A later follow-up work [Marx and Pilipczuk, 2022] shows that there is an algorithm running in time $n^{\mathcal{O}(\sqrt{k})}$ that solves Geometric Set Cover with unit squares or disks and that there is no algorithm running in time $f(k) \cdot n^{o(\sqrt{k})}$ for any computable f under the Exponential-Time Hypothesis, so this is a tight bound for this problem.

We also consider parameterization of weighted problems. There does not seem to be a consensus of what parameterization in the weighted setting is exactly; there was an attempt to introduce a quite complicated general framework of weighted parameterized setting in [Shachnai and Zehavi, 2017]. Kernels for several well-known weighted problems such as WEIGHTED SUBSET SUM or WEIGHTED KNAPSACK are presented in [Etscheid et al., 2017]. Another work [Kim et al., 2021] considers weighted parameterization of WEIGHTED DIRECTED FEEDBACK SET and WEIGHTED st-CUT.

δ-extension In this paper, we focus on SEGMENT SET COVER with δ-extension. δ-extension is a problem relaxation method based on the δ-shrinking model which was introduced in [Adamaszek et al., 2015] to provide interesting results for the MAXIMUM WEIGHT INDEPENDENT SET OF RECTANGLES problem. In this problem one is given a family of weighted rectangles and needs to find a set of non-overlapping rectangles with the largest possible total weight. In the δ-shrinking relaxed problem the returned set of rectangles must be non-overlapping after all the rectangles are shrunk by a tiny fraction δ towards the centre of symmetry. This problem is easier, because we compare the weight of the obtained solution to the optimum result before the shrinking. It might even lead to finding a set with result better than the optimum for the original problem. The authors in [Adamaszek et al., 2015] present a PTAS for MAXIMUM WEIGHT INDEPENDENT SET OF RECTANGLES with δ-shrinking, which was later improved to an EPTAS in [Pilipczuk et al., 2017], alongside with presenting a new FPT algorithm for this problem with the natural parameterization. A similar δ-shrinking model was used in [Wiese, 2018] to present a PTAS for MAXIMUM WEIGHT INDEPENDENT SET OF POLYGONS with δ-shrinking.

Definition 1.3. For any $\delta > 0$ and a centre-symmetric convex object L with centre of symmetry $S = (x_s, y_s)$, the **δ-extension** of L is the open set of points:

$$L^{+\delta} = \{(1+\epsilon) \cdot (x - x_s, y - y_s) + (x_s, y_s) : (x, y) \in L, 0 \leq \epsilon < \delta\}.$$

That is, $L^{+\delta}$ is the image of L under homothety centred at S with scale $(1 + \delta)$ but with the extreme points excluded. In particular, δ-extension turns a segment into a segment without endpoints and a rectangle into an interior of a rectangle.

Analogous to δ-shrinking, δ-extension provides a framework for relaxing GEOMETRIC SET COVER problems, where we allow the returned set of objects $\mathcal{R}$ to *almost* cover the points in the universe by requiring that they are covered by $\mathcal{R}$ after δ-extension, i.e. by the set $\mathcal{R}^{+\delta}$. The same concept could be used for GEOMETRIC HITTING SET problems.

For a longer discussion of this concept see Section 2.4.

Similar model is used to prove that GEOMETRIC SET COVER with fat polygons relaxed with δ-extension admits an EPTAS [Har-Peled and Lee, 2012]. The δ-extension model presented there is well-defined only for fat polygons. An object P is extended by all the points that are at distance to the closest point in the object P no larger than $\delta \cdot \mathsf{rad}(P)$, where $\mathsf{rad}(P)$ is the largest radius of a circle inscribed into P. Since segments do not have any circle inscribed into them, the definition presented there cannot be utilized for the setting of segments considered here. Polygon extended by δ-extension defined in Definition 1.3 covers a superset of points that the polygon extended by δ-extension defined in [Har-Peled and Lee, 2012] covers. Since our definition is more permissive for any polygon, the EPTAS from [Har-Peled and Lee, 2012] also works for polygons extended according to our definition of δ-extension.

1.2. Our contribution

In this thesis we make the following contributions.

We show that Segment Set Cover is APX-hard, even if segments are axis-parallel and we relax the problem with $\frac{1}{2}$-extension, (Theorem 1.1).

Theorem 1.1. *(**Segment Set Cover** is APX-hard). Segment Set Cover is APX-hard even when relaxed with $\frac{1}{2}$-extension and segments are axis-parallel. That is, assuming $P \neq NP$, there does not exist a PTAS for this problem.*

Theorem 1.1 implies the following. Note that segments are just degenerated rectangles.

Corollary 1.1. *(**Geometric Set Cover** with rectangles is APX-hard). Geometric Set Cover with axis-parallel rectangles is APX-hard even when relaxed with $\frac{1}{2}$-extension.*

This expands the previous result of [Chan and Grant, 2014] that Geometric Set Cover with axis-parallel fat rectangles is APX-hard, we improved the result that rectangles no longer have to be fat (Corollary 1.1) and it holds when the problem is relaxed with $\frac{1}{2}$-extension. It also proves that the assumption in [Har-Peled and Lee, 2012] about polygons being fat is necessary, because covering with arbitrary polygons with $\frac{1}{2}$-extension is APX-hard.

We also provide two FPT algorithms for parameterized Segment Set Cover (Theorem 1.2) and Weighted Segment Set Cover relaxed with δ-extension (Theorem 1.3).

Theorem 1.2. *(FPT for **Segment Set Cover**). There exists an algorithm that given a family $\mathcal{P}$ of segments (in any direction), a set of points $\mathcal{C}$ and a parameter k, runs in time $k^{\mathcal{O}(k)}(|\mathcal{C}| \cdot |\mathcal{P}|)^2$, and outputs a solution $\mathcal{R} \subseteq \mathcal{P}$ such that $|\mathcal{R}| \leq k$ and $\mathcal{R}$ covers all points in $\mathcal{C}$, or determines that such a set $\mathcal{R}$ does not exist.*

Theorem 1.3. *(FPT for **Weighted Segment Set Cover** with δ-extension). There exists an algorithm that given a family $\mathcal{P}$ of n weighted segments (in any direction), a set of m points $\mathcal{C}$, and parameters k and $\delta > 0$, runs in time $f(k, \delta) \cdot (nm)^c$ for some computable function f and a constant c and outputs a set $\mathcal{R}$ such that:*

- *$\mathcal{R} \subseteq \mathcal{P}$,*
- *$|\mathcal{R}| \leq k$,*
- *$\mathcal{R}^{+\delta}$ covers all points in $\mathcal{C}$,*
- *the weight of $\mathcal{R}$ is not greater than the weight of an optimum solution of size at most k for this problem without δ-extension,*

or determines that there is no set $\mathcal{R}$ with $|\mathcal{R}| \leq k$ such that $\mathcal{R}$ covers all points in $\mathcal{C}$.

On the other hand, we prove that Weighted Segment Set Cover is W[1]-hard even when segments are limited to 3 directions (Theorem 1.4) and assuming ETH there does not exist an algorithm for this problem that runs in time $f(k)(|\mathcal{C}| + |\mathcal{P}|)^{o(\sqrt{k})}$. See Figure 1.1 for a summary of parameterized results for Segment Set Cover and Weighted Segment Set Cover.

Theorem 1.4. *(**Weighted Segment Set Cover** is W[1]-hard). Consider the problem of covering a set $\mathcal{C}$ of points by selecting at most k segments from a set of segments $\mathcal{P}$ with non-negative weights $w : \mathcal{P} \to \mathbb{R}^+$ so that the weight of the cover is minimal. Then this problem is W[1]-hard when parameterized by k and assuming ETH, there is no algorithm for this problem with running time $f(k) \cdot (|\mathcal{C}| + |\mathcal{P}|)^{o(\sqrt{k})}$ for any computable function f. Moreover, this holds even if all segments in $\mathcal{P}$ are axis-parallel or right-diagonal.*

See Section 2.1 for exact definitions of axis-parallel and right-diagonal segments.

This result is particularly interesting, because the problem without weights is FPT, while the weighted variant is W[1]-hard. Moreover, δ-extension allowed us to provide an FPT algorithm for the problem which is W[1]-hard otherwise.

Note that the result of Theorem 1.4 is not tight: there exists a simple algorithm running in time $f(k)(|\mathcal{C}| + |\mathcal{P}|)^k$. So the question whether there exists an algorithm for this problem running in time $f(k) \cdot (|\mathcal{C}| + |\mathcal{P}|)^{o(k)}$ is still open.

Permissive FPT is a relaxed FPT problem, where we need to find a solution of *any* size in FPT-time, but we compare it to the optimum solution of size at most k. Idea for permissive FPT in local search was presented in [Marx and Schlotter, 2011], [Gaspers et al., 2012]. Theorem 1.4 can be improved to show that a permissive FPT algorithm does not exist. This is formulated precisely in Theorem 5.2.

	exact weighted	δ-extension weighted	exact unweighted
axis-parallel	?	FPT*	FPT*
3 directions	W[1]-hard	FPT*	FPT*
any direction	W[1]-hard*	FPT	FPT

Figure 1.1: Our results for Weighted Segment Set Cover and Segment Set Cover parameterized by the size of a solution. Results marked with * are not explicitly given in this thesis, but they trivially follow from stronger results shown in the other cells of the table.

Future work. There are two aforementioned problems that relate to Theorem 1.4 and were not solved in this thesis. We have given a W[1]-hardness proof for Weighted Segment Set Cover where segments are limited to 3 directions, but the segments in the construction may be also right-diagonal. However, it may be possible to improve this construction to use segments in 2 directions instead of 3 directions. The other question is what is the tight bound for this problem. The simple algorithm solving this problem is running in time $f(k)(|\mathcal{C}| + |\mathcal{P}|)^{\mathcal{O}(k)}$, while our lower bound refutes running time $f(k)(|\mathcal{C}| + |\mathcal{P}|)^{o(\sqrt{k})}$.

Another problem to consider is whether Geometric Hitting Set relaxed with δ-extension can yield some better results.

Acknowledgments. Words cannot express my gratitude to my supervisor, dr hab Michał Pilipczuk, for his patience and eager help with research and editing. I am also grateful to Krzysztof Maziarz for countless hours of proofreading this thesis.

Chapter 2

Preliminaries

In this chapter we present some basic definitions that will be used later.

2.1. Geometric Set Cover

Whenever speaking about Geometric Set Cover, we consider it in the 2-dimensional plane.

In the Geometric Set Cover problem we are are given $\mathcal{P}$ — a set of objects, which are connected subsets of the plane and $\mathcal{C}$ — a set of points in the plane. The task is to choose $\mathcal{R} \subseteq \mathcal{P}$ such that every point in $\mathcal{C}$ is inside some object from $\mathcal{R}$ and $|\mathcal{R}|$ is minimized. We will mostly consider the case where $\mathcal{P}$ consists of segments in the plane.

In the weighted setting, there is some given weight function $f : \mathcal{P} \to \mathbb{R}^+$ and we would like to find a solution $\mathcal{R}$ that minimizes $\sum_{R \in \mathcal{R}} f(R)$.

Definition 2.1. A segment is **axis-parallel** if it lies on a line that is either horizontal $y = c$ or vertical $x = c$.

Definition 2.2. A line is **right-diagonal** if it is described by the linear function $x + y = d$ for some $d \in \mathbb{R}$. A segment is **right-diagonal** if its direction is a right-diagonal line.

2.2. Parameterization

In the parameterized setting of Geometric Set Cover for a given k, our task is to either find a solution $\mathcal{R}$ such that $|\mathcal{R}| \leq k$ or decide that there is no such solution.

Definition 2.3. A **fixed-parameter (FPT)** algorithm for a problem with parameter k and instance size n is an algorithm running in time $f(k) \cdot n^c$ for some constant c and some computable function f.

Definition 2.4. Boolean formula is in **conjunctive normal form (CNF)** if it is a conjunction of one or more formulas, which are disjunction of literals. **k-CNF** formula is a CNF formula, where every disjunction consists of at most k literals.

Definition 2.5. **k-SAT** problem is a Boolean satisfiability problem of k-CNF formulas. Given k-CNF formula, one must answer if there exists any variable assignment that satisfies the formula.

Definition 2.6. For $k \geq 3$, let us define S_k as the set of constants σ such that there exists an algorithm solving k-SAT running in time $2^{\sigma n} \cdot n^{\mathcal{O}(1)}$. Let s_k be the infimum of the set S_k.

Exponential Time Hypothesis (ETH) asserts that $s_3 > 0$. This conjecture implies that there does not exist an algorithm solving 3-SAT running in time $2^{o(n)}$.

The definition of a W[1]-hard problem and W hierarchy can be found in Chapter 13.3 of [Cygan et al., 2015]. When proving that a problem is W[1]-hard, we are going to use Theorem 5.1 (W[1]-hardness of Grid Tiling), which was proved in [Marx, 2007].

2.3. Approximation

Let us recall some definitions related to optimization problems.

Definition 2.7. A **polynomial-time approximation scheme (PTAS)** for a minimization problem Π is a family of algorithms $\mathcal{A}_\epsilon$ for every $\epsilon > 0$ such that $\mathcal{A}_\epsilon$ takes an instance I of Π and in polynomial time finds a solution that is within a factor of $(1+\epsilon)$ of being optimal. This means that the reported solution has weight at most $(1+\epsilon)\mathsf{opt}(I)$, where $\mathsf{opt}(I)$ is the weight of an optimal solution to I.

Definition 2.8. A problem Π is **APX-hard** if assuming P $\neq$ NP, there exists $\epsilon > 0$ such that there is no polynomial-time $(1+\epsilon)$-approximation algorithm for Π.

2.4. δ-extension

Another idea presented here, which can be utilized only when considering the problems with geometric objects, is δ-extension. We define it specifically for the Geometric Set Cover problem with convex centre-symmetric objects.

Intuitively, we consider a problem with slightly larger objects, which makes the instance more permissive. However, we aim to find a solution that is not larger than the optimum solution to the original problem, so this is substantially easier than just solving the problem for the larger objects. It may even be the case that we are able to find a solution of size smaller than the optimum solution to the original problem.

Formal definition of δ-extended objects is present in Definition 1.3.

The Geometric Set Cover with δ-extension is a version of Geometric Set Cover with the following modifications.

- We need to cover all the points in $\mathcal{C}$ by selecting objects from $\{P^{+\delta} : P \in \mathcal{P}\}$ (which always include no fewer points than the objects before δ-extension).
- We look for a solution that is not larger than the optimum solution to the original problem. Note that it does not need to be an optimal solution in the modified problem.

Formally, we have the following.

Definition 2.9. The **Geometric Set Cover problem with δ-extension** is the problem where for an input instance $I = (\mathcal{P}, \mathcal{C})$ of Geometric Set Cover, the task is to output a solution $\mathcal{R} \subseteq \mathcal{P}$ such that the δ-extended set $\{R^{+\delta} : R \in \mathcal{R}\}$ covers $\mathcal{C}$ and is not larger than the optimal solution to the problem without extension, i.e. $|\mathcal{R}| \leq |opt(I)|$.

At last, we formulate a definition of the polynomial-time approximation scheme (PTAS) for a problem with δ-extension.

Definition 2.10. A **PTAS for Geometric Set Cover with δ-extension** is a family of algorithms $\{\mathcal{A}_{\delta,\epsilon}\}_{\delta,\epsilon>0}$ that each takes as an input instance $I = (\mathcal{P}, \mathcal{C})$ of Geometric Set Cover where objects are centre-symmetric and convex, and in polynomial-time outputs a solution $\mathcal{R} \subseteq \mathcal{P}$ such that the δ-extended set $\{R^{+\delta} : R \in \mathcal{R}\}$ covers $\mathcal{C}$ and is within a $(1+\epsilon)$ factor of the optimal solution to this problem without extension, i.e. $(1+\epsilon)|\mathcal{R}| \leq |opt(I)|$.

2.5. Weighted Geometric Set Cover

In this thesis we also consider a Weighted Geometric Set Cover problem, which is a combination of the weighted and parameterized setting described in Section 2.1. We already argued in the introduction that there is no consensus of how it is defined, but when we discuss the weighted parameterized setting we will consider the following definition. There is a given weight function $f : \mathcal{P} \to \mathbb{R}^+$ and we would like to find a solution $\mathcal{R}$ such that $|\mathcal{R}| \leq k$ and $\sum_{R\in\mathcal{R}} f(R)$ is minimum possible among such sets $\mathcal{R}$.

Definition 2.11. The **Weighted Geometric Set Cover problem with δ-extension** is the problem where for an input instance $I = (\mathcal{P}, \mathcal{C}, f)$ of Weighted Geometric Set Cover, the task is to output a solution $\mathcal{R} \subseteq \mathcal{P}$ such that the δ-extended set $\{R^{+\delta} : R \in \mathcal{R}\}$ covers $\mathcal{C}$ and it has weight not larger than the optimal solution to the problem without extension, i.e. $\sum_{R\in\mathcal{R}} f(R) \leq |\mathsf{opt}(I)|$.

We also consider weighted parameterized setting with δ-extension, which we formally define below.

Definition 2.12. The **Weighted Geometric Set Cover problem with δ-extension parameterized by the size of a solution** is a problem where for an input instance $I = (\mathcal{P}, \mathcal{C}, f, k)$ of Weighted Geometric Set Cover parameterized by the size of a solution k, the task is to output a solution $\mathcal{R} \subseteq \mathcal{P}$ such that the δ-extended set $\{R^{+\delta} : R \in \mathcal{R}\}$ covers $\mathcal{C}$, uses no more than k sets, i.e. $|\mathcal{R}| \leq k$, and it has weight not larger than the optimal solution to the problem without extension, i.e. $\sum_{R\in\mathcal{R}} f(R) \leq |\mathsf{opt}(I)|$.

Chapter 3

APX-hardness of Segment Set Cover

In this section we analyze whether there exists a PTAS for Geometric Set Cover for rectangles. We show that Segment Set Cover is APX-hard even if we can restrict this problem to a very simple setting: segments parallel to axes and allow $\frac{1}{2}$-extension.

Our result can be summarized in the following theorem and this section aims to prove it.

Theorem 1.1. ***(Segment Set Cover is APX-hard).*** Segment Set Cover *is APX-hard even when relaxed with $\frac{1}{2}$-extension and segments are axis-parallel. That is, assuming $P \neq NP$, there does not exist a PTAS for this problem.*

We prove Theorem 1.1 by taking a problem that is APX-hard and showing a reduction. For this problem we choose MAX-(3,3)-SAT which we define below.

3.1. MAX-(3,3)-SAT

See Definition 2.4 for the definition of a k-CNF formula.

Definition 3.1. **MAX-3SAT** is the following maximization problem. We are given a 3-CNF formula, and we need to find a Boolean assignment of variables that satisfies the most clauses.

Definition 3.2. **MAX-(3,3)-SAT** is a variant of MAX-3SAT with an additional restriction that every variable appears in exactly 3 clauses and every clause contains exactly 3 literals of 3 different variables. Note that thus, the number of clauses is equal to the number of variables.

In our proof of Theorem 1.1 we use hardness of approximation of MAX-(3,3)-SAT proved in [Håstad, 2001] and described in Theorem 3.1 below.

Definition 3.3. MAX-3SAT formula with m clauses is **at most α-satisfiable**, if every assignment of variables satisfies no more than αm clauses.

Theorem 3.1. ***([Håstad, 2001]).*** *For any $\epsilon > 0$, it is NP-hard to distinguish satisfiable (3,3)-SAT formulas from at most $(\frac{7}{8} + \epsilon)$-satisfiable (3,3)-SAT formulas.*

3.2. Statement of reduction

The following lemma encapsulates the properties of the reduction described in this section, and it allows us to prove Theorem 1.1.

Lemma 3.1. *Given an instance S of MAX-(3,3)-SAT with n variables and optimum value* $\mathsf{opt}(S)$*, we can construct an instance $(\mathcal{C}, \mathcal{P})$ of* Segment Set Cover *with axis-parallel segments in 2D such that:*

(1) For every solution to instance S that satisfies k clauses, there exists a solution to $(\mathcal{C}, \mathcal{P})$ of size $15n - k$.

(2) For every solution $\mathcal{R}$ to instance $(\mathcal{C}, \mathcal{P})$, there exists a solution to S that satisfies at least $15n - |\mathcal{R}|$ clauses.

(3) For every $\mathcal{R} \subseteq \mathcal{P}$, if $\mathcal{R}^{+\frac{1}{2}}$ is a solution to $(\mathcal{C}, \mathcal{P})$, then $\mathcal{R}$ is also a solution to $(\mathcal{C}, \mathcal{P})$.

Therefore, the optimum size of a solution to $(\mathcal{C}, \mathcal{P})$ is $\mathsf{opt}((\mathcal{C}, \mathcal{P})) = 15n - \mathsf{opt}(S)$.

We prove Lemma 3.1 in subsequent sections. Section 3.3 describes the proposed instance $(\mathcal{C}, \mathcal{P})$. Property (1) is proved by Lemma 3.11, (2) by Lemma 3.13, and finally (3) trivially follows from Lemma 3.10. Firstly let us prove Theorem 1.1 using Lemma 3.1 and Theorem 3.1.

Proof of Theorem 1.1. Consider any $0 < \epsilon < \frac{1}{15 \cdot 8}$.

Let us assume that there exists a polynomial-time $(1 + \epsilon)$-approximation algorithm for unweighted Segment Set Cover with axis-parallel segments in 2D with $\frac{1}{2}$-extension. We construct an algorithm that solves the problem stated in Theorem 3.1, thereby proving that P = NP.

Take an instance S of MAX-(3,3)-SAT to be distinguished and construct an instance of Segment Set Cover $(\mathcal{C}, \mathcal{P})$ using Lemma 3.1. We now use the $(1 + \epsilon)$-approximation algorithm for Segment Set Cover relaxed with $\frac{1}{2}$-extension on $(\mathcal{C}, \mathcal{P})$. Denote the size of the solution returned by this algorithm as $\mathsf{approx}^*((\mathcal{C}, \mathcal{P}))$. We prove that if in S one can satisfy at most $(\frac{7}{8} + \epsilon)n$ clauses, then $\mathsf{approx}^*((\mathcal{C}, \mathcal{P})) \geq 15n - (\frac{7}{8} + \epsilon)n$, and if S is satisfiable, then $\mathsf{approx}^*((\mathcal{C}, \mathcal{P})) < 15n - (\frac{7}{8} + \epsilon)n$.

Assume S satisfiable. From the definition of S being satisfiable, we have:

$$\mathsf{opt}(S) = n.$$

From Lemma 3.1 we have:

$$\mathsf{opt}((\mathcal{C}, \mathcal{P})) = 14n.$$

Therefore,

$$\mathsf{approx}^*((\mathcal{C}, \mathcal{P})) \leq (1 + \epsilon)\mathsf{opt}((\mathcal{C}, \mathcal{P})) = 14n(1 + \epsilon) = 14n + 14\epsilon \cdot n =$$
$$= 14n + (15\epsilon - \epsilon)n < 14n + \left(\frac{1}{8} - \epsilon\right) n = 15n - \left(\frac{7}{8} + \epsilon\right) n.$$

Assume S is at most $\left(\frac{7}{8} + \epsilon\right)$ satisfiable. From the definition of S being at most $\left(\frac{7}{8} + \epsilon\right) n$ satisfiable, we have:

$$\mathsf{opt}(S) \leq \left(\frac{7}{8} + \epsilon\right) n$$

From Lemma 3.1 we have:

$$\mathsf{opt}((\mathcal{C}, \mathcal{P})) \geq 15n - \left(\frac{7}{8} + \epsilon\right) n$$

Since a solution to $(\mathcal{C}, \mathcal{P})$ with $\frac{1}{2}$-extension is also a solution without any extension, by Lemma 3.1 (3), we have:

$$\mathsf{approx}^*((\mathcal{C}, \mathcal{P})) \geq \mathsf{opt}((\mathcal{C}, \mathcal{P})) = 15n - \left(\frac{7}{8} + \epsilon\right) n$$

Therefore, by using the assumed $(1+\epsilon)$-approximation algorithm, it is possible to distinguish the case when S is satisfiable from the case when it is at most $(\frac{7}{8} + \epsilon)n$ satisfiable: it suffices to compare $\mathsf{approx}^*((\mathcal{C}, \mathcal{P}))$ with $15n - (\frac{7}{8} + \epsilon)n$. Hence, the assumed approximation algorithm cannot exist, unless P = NP. $\square$

3.3. Construction of the Segment Set Cover instance

We proceed to the proof of Lemma 3.1. That is, we show a reduction from the MAX-(3,3)-SAT problem to Segment Set Cover with segments parallel to axes. Moreover, the obtained instance of Segment Set Cover will be robust to $\frac{1}{2}$-extension (have the same optimal solution after $\frac{1}{2}$-extension).

The construction will be composed of 2 types of gadgets: **VARIABLE-gadgets** and **CLAUSE-gadgets**. CLAUSE-gadgets will be constructed using two **OR-gadgets** connected together.

3.3.1. VARIABLE-gadget

VARIABLE-gadget is responsible for choosing the value of a variable in a CNF formula. It allows two minimum solutions of size 3 each. These two choices correspond to the two Boolean values of the variable corresponding to this gadget.

Points. Define points a, b, c, d, e, f, g, h as follows, where $L = 22n$:

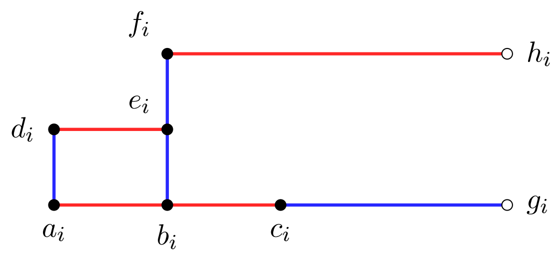

Figure 3.1: **VARIABLE-gadget.** We denote the set of points marked with black circles as $\mathsf{pointsVariable}_i$, and they need to be covered (are part of the set $\mathcal{C}$). Note that some of the points are not marked as black dots and exists only to name segments for further reference. We denote the set of red segments as $\mathsf{chooseVariable}_i^{\mathtt{false}}$ and the set of blue segments as $\mathsf{chooseVariable}_i^{\mathtt{true}}$.

$$\begin{array}{llll} a := (-3L, 0) & b := (-2L, 0) & c := (-L, 0) & d := (-3L, 1) \\ e := (-2L, 1) & f := (-2L, 2) & g := (L, 0) & h := (L, 2) \end{array}$$

Let us define:

$$\mathsf{pointsVariable} := \{a, b, c, d, e, f\}$$

and, for any $1 \leq i \leq n$,

$$\mathsf{pointsVariable}_i := \mathsf{pointsVariable} + (0, 4i).$$

We denote $a_i := a + (0, 4i)$ etc.

Segments. Let us define:

$$\mathsf{chooseVariable}_i^{\texttt{true}} := \{(a_i, d_i), (b_i, f_i), (c_i, g_i)\},$$

$$\mathsf{chooseVariable}_i^{\texttt{false}} := \{(a_i, c_i), (d_i, e_i), (f_i, h_i)\},$$

$$\mathsf{segmentsVariable}_i := \mathsf{chooseVariable}_i^{\texttt{true}} \cup \mathsf{chooseVariable}_i^{\texttt{false}}.$$

We also name two of these segments for future reference: $\mathsf{xTrueSegment}_i := (c_i, g_i)$, $\mathsf{xFalseSegment}_i := (f_i, h_i)$.

Lemma 3.2. *For any $1 \leq i \leq n$, points in $\mathsf{pointsVariable}_i$ can be covered using 3 segments from $\mathsf{segmentsVariable}_i$.*

Proof. We can use either set $\mathsf{chooseVariable}_i^{\texttt{true}}$ or $\mathsf{chooseVariable}_i^{\texttt{false}}$. ☐

Lemma 3.3. *For any $1 \leq i \leq n$, points in $\mathsf{pointsVariable}_i$ can not be covered with fewer than 3 segments from $\mathsf{segmentsVariable}_i$.*

Proof. No segment of $\mathsf{segmentsVariable}_i$ covers more than one point from $\{d_i, f_i, c_i\}$, therefore $\mathsf{pointsVariable}_i$ can not be covered with fewer than 3 segments. ☐

Lemma 3.4. *For every set $A \subseteq \mathsf{segmentsVariable}_i$ such that A covers $\mathsf{pointsVariable}_i$ and $\mathsf{xTrueSegment}_i, \mathsf{xFalseSegment}_i \in A$, it holds that $|A| \geq 4$.*

Proof. No segment from $\mathsf{segmentsVariable}_i$ covers more than one point from $\{a_i, e_i\}$, therefore $\mathsf{pointsVariable}_i - \{c_i, f_i\}$ can not be covered with fewer than 2 segments. ☐

3.3.2. OR-gadget

An OR-gadget connects input and output segments (see Figure 3.2) in a way that is supposed to simulate the binary disjunction.

Input segments are the only segments that cover points outside of the gadget, as their left ends lie outside of it. Point $v_{i,j}$ is the only one that can be covered by segments that do not belong to the gadget.

The OR-gadget has the property that every set of segments that covers all the points in the gadget uses at least 3 segments from it. Moreover, the output segment belongs to the solution of size 3 only if at least one of the input segments belongs to the solution. Therefore, optimum solutions restricted to the OR-gadget behave like a binary disjunction for the input segments.

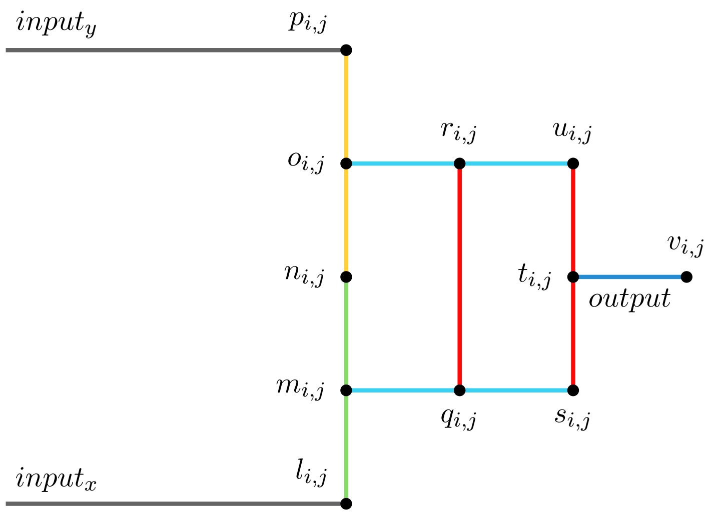

Figure 3.2: **OR-gadget.** Segments from $\mathsf{chooseOr}_{i,j}^{\texttt{false}}$ are red, segments from $\mathsf{chooseOr}_{i,j}^{\texttt{true}}$ are blue (both light blue and dark blue), segments from $\mathsf{orMoveVariable}_{i,j}$ are green and yellow. Dark blue segment is the *output* segment. Grey segments $input_x$ and $input_y$ are input segments that are not part of $\mathsf{segmentsOr}_{i,j}$.

Points. We define

$$\begin{array}{llll} l_0 := (0,0) & m_0 := (0,1) & n_0 := (0,2) & o_0 := (0,3) \\ p_0 := (0,4) & q_0 := (1,1) & r_0 := (1,3) & s_0 := (2,1) \\ t_0 := (2,2) & u_0 := (2,3) & v_0 := (3,2) & \end{array}$$

$$vec_{i,j} := (20i + 3 + 3j, 4(n+1) + 2j)$$

For integers i, j, define $\{l_{i,j}, m_{i,j}, \ldots, v_{i,j}\}$ as $\{l_0, m_0, \ldots, v_0\}$ shifted by $vec_{i,j}$, i.e. $l_{i,j} = l_0 + vec_{i,j}$ etc.

Note that $v_{i,0} = l_{i,1}$ (see Figure 3.3). Next, let

$$\mathsf{pointsOr}_{i,j} := \{l_{i,j}, m_{i,j}, n_{i,j}, o_{i,j}, p_{i,j}, q_{i,j}, r_{i,j}, s_{i,j}, t_{i,j}, u_{i,j}\}$$

Note that $\mathsf{pointsOr}_{i,j}$ does not include the point $v_{i,j}$.

Segments. We define the set of segments in several parts:

$$\begin{aligned} \mathsf{chooseOr}_{i,j}^{\texttt{false}} &:= \{(q_{i,j}, r_{i,j}), (s_{i,j}, u_{i,j})\}, \\ \mathsf{chooseOr}_{i,j}^{\texttt{true}} &:= \{(m_{i,j}, s_{i,j}), (o_{i,j}, u_{i,j}), (t_{i,j}, v_{i,j})\}, \\ \mathsf{orMoveVariable}_{i,j} &:= \{(l_{i,j}, n_{i,j}), (n_{i,j}, p_{i,j})\}. \end{aligned}$$

Finally all segments of an OR-gadget are defined as:

$$\mathsf{segmentsOr}_{i,j} := \mathsf{chooseOr}_{i,j}^{\texttt{false}} \cup \mathsf{chooseOr}_{i,j}^{\texttt{true}} \cup \mathsf{orMoveVariable}_{i,j}$$

Lemma 3.5. *For any $1 \leq i \leq n, j \in \{0, 1\}$ and $x \in \{l_{i,j}, p_{i,j}\}$, points in $\mathsf{pointsOr}_{,i,j} - \{x\} \cup \{v_{i,j}\}$ can be covered with 4 segments from $\mathsf{segmentsOr}_{i,j}$.*

Proof. We can do this using one segment from $\mathsf{orMoveVariable}_{i,j}$, the one that does not cover x, and all segments from $\mathsf{chooseOr}^{\mathtt{true}}_{i,j}$. ☐

Lemma 3.6. *For any $1 \le i \le n, j \in \{0,1\}$, points in $\mathsf{pointsOr}_{i,j}$ can be covered with 4 segments from $\mathsf{segmentsOr}_{i,j}$.*

Proof. We can do this using segments from $\mathsf{orMoveVariable}_{i,j} \cup \mathsf{chooseOr}^{\mathtt{false}}_{i,j}$. ☐

3.3.3. CLAUSE-gadget

A CLAUSE-gadget is responsible for determining whether variable values assigned in variable gadgets satisfy the corresponding clause in the input formula ϕ. It has a minimum solution of size w if and only if the clause is satisfied, i.e. at least one of the respective variables is assigned the correct value. Otherwise, its minimum solution has size $w+1$. In this way, by analyzing the size of the minimum solution to the entire constructed instance, we will be able to tell how many clauses it is possible to satisfy in an optimum solution to ϕ.

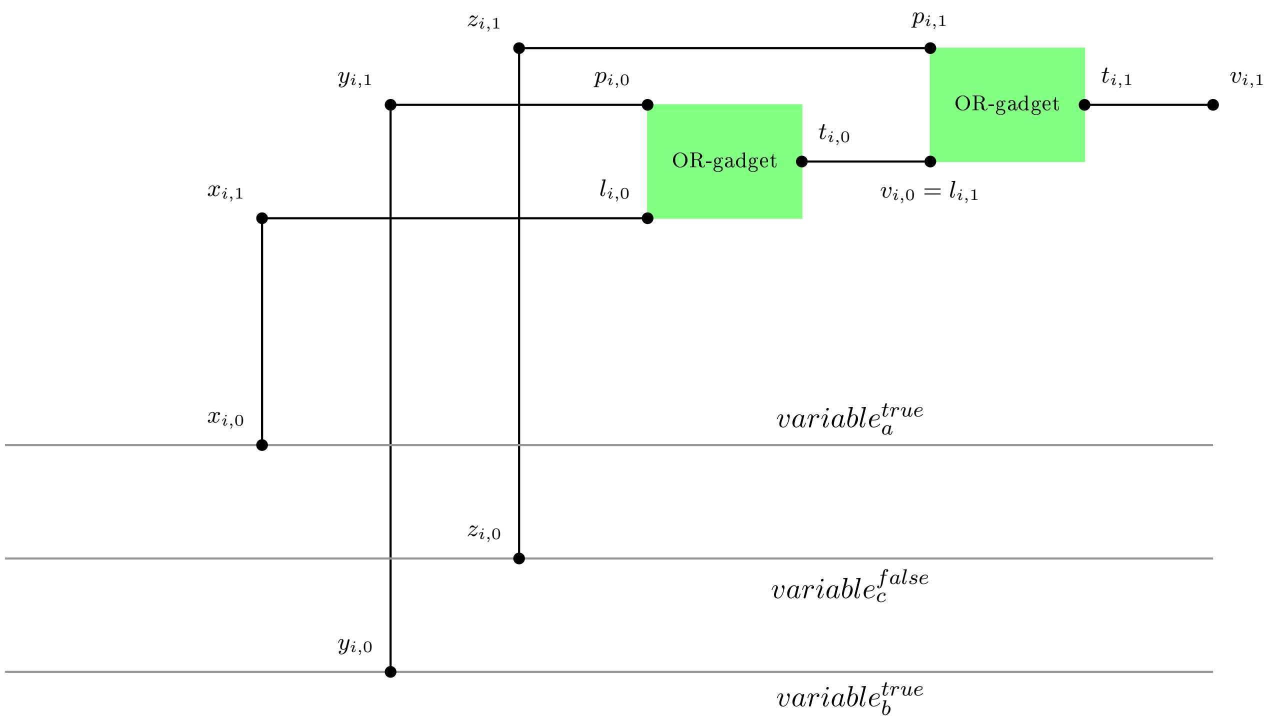

Figure 3.3: **CLAUSE-gadget for a clause $a \vee b \vee \neg c$.** Every green rectangle is an OR-gadget. y-coordinates of $x_{i,0}$, $y_{i,0}$ and $z_{i,0}$ depend on the variables in the i-th clause. Grey segments corresponds to the values of variables satisfying the i-th clause.

Points. First, we define auxiliary functions for literals. For a literal w, let $\mathsf{idx}(w)$ be the index of the variable in w, and $\mathsf{neg}(w)$ be the Boolean value (0 or 1) whether the variable is negated in w or not.

$$\begin{aligned}
\mathsf{idx}(w) &:= i \text{ when } w = x_i \\
\mathsf{neg}(w) &:= \begin{cases} 0 & \text{if } w = x_i \\ 1 & \text{if } w = \neg x_i \end{cases}
\end{aligned}$$

Let us assume that clause $C_i = a \vee b \vee c$ for any literals a, b, c. Then, we define points in the gadget as:

$$\begin{aligned} x_{i,0} &:= (20i, 4\cdot \mathsf{idx}(a) + 2\cdot \mathsf{neg}(c)), & x_{i,1} &:= (20i, 4(n+1)),\\ y_{i,0} &:= (20i+1, 4\cdot \mathsf{idx}(b) + 2\cdot \mathsf{neg}(b)), & y_{i,1} &:= (20i+1, 4(n+1)+4),\\ z_{i,0} &:= (20i+2, 4\cdot \mathsf{idx}(c) + 2\cdot \mathsf{neg}(c)), & z_{i,1} &:= (20i+2, 4(n+1)+6). \end{aligned}$$

We are now ready to define the set of points in a CLAUSE-gadget:

$$\mathsf{moveVariablePoints}_i := \{x_{i,j} : j \in \{0,1\}\} \cup \{y_{i,j} : j \in \{0,1\}\} \cup \{z_{i,j} : j \in \{0,1\}\},$$

$$\mathsf{pointsClause}_i := \mathsf{moveVariablePoints}_i \cup \mathsf{pointsOr}_{i,0} \cup \mathsf{pointsOr}_{i,1} \cup \{v_{i,1}\}.$$

Note that these two points are equal: $v_{i,0} = l_{i,1}$. This translates to the fact that the output of the first OR-gadget is an input to the second OR-gadget. This creates an *or* of 3 Boolean values.

Segments. We also define segments for the CLAUSE-gadget as below:

$$\begin{aligned} \mathsf{moveVariableSegments}_i &:= \{(x_{i,0}, x_{i,1}), (y_{i,0}, y_{i,1}), (z_{i,0}, z_{i,1}), (x_{i,1}, l_{i,0}), (y_{i,1}, p_{i,0}), (z_{i,1}, p_{i,1})\}\\ \mathsf{segmentsClause}_i &:= \mathsf{moveVariableSegments}_i \cup \mathsf{segmentsOr}_{i,0} \cup \mathsf{segmentsOr}_{i,1}. \end{aligned}$$

The CLAUSE-gadgets consist of two OR-gadgets. Ideally, we would place the i-th CLAUSE-gadget close to the $\mathsf{xTrueSegment}_{j_1}$ or $\mathsf{xFalseSegment}_{j_1}$ segments corresponding to the literals that occur in the i-th clause. It would be inconvenient to position them there, because between these segments there may be additional $\mathsf{xTrueSegment}_{j_2}$ or $\mathsf{xFalseSegment}_{j_2}$ segments corresponding to the other literals.

Instead, we use simple auxiliary gadgets to *transfer* whether the segment is in a solution, i.e. segments $(x_{i,0}, x_{i,1}), (y_{i,0}, y_{i,1}), (z_{i,0}, z_{i,1})$. Each transfer gadget consists of two segments $(x_{i,0}, x_{i,1}), (x_{i,1}, a)$. These are the only segments that can cover $x_{i,1}$. We place $x_{i,0}$ on a segment that we want to transfer (i.e. segment responsible for choosing the variable value satisfying the corresponding literal). If in some solution $x_{i,0}$ is already covered by this segment, then we can cover $x_{i,1}$ by $(x_{i,1}, a)$, thus also covering a. If $x_{i,0}$ is not covered by this segment, then the only way to cover $x_{i,0}$ is to use segment $(x_{i,0}, x_{i,1})$. Intuitively, in any optimal solution the two segments *transfer* the state of whether $x_{i,0}$ is covered onto whether a is covered. Therefore, the number of segments in the optimal solution is increased by one, and we get a point a that was effectively placed on some segment s, but it can be placed anywhere in the plane instead, consequently simplifying the construction.

Lemma 3.7. *For any* $1 \le i \le n$ *and* $a \in \{x_{i,0}, y_{i,0}, z_{i,0}\}$*, there is a set* $\mathsf{solClause}_i^{\mathit{true},a} \subseteq \mathsf{segmentsClause}_i$ *with* $|\mathsf{solClause}_i^{\mathit{true},a}| = 11$ *that covers all points in* $\mathsf{pointsClause}_i - \{a\}$.

Proof. For $a = x_{i,0}$ (analogous proof for $y_{i,0}$): First we use Lemma 3.5 twice with excluded $x = l_{i,0}$ and $x = l_{i,1} = v_{i,0}$, resulting with 8 segments in $\mathsf{chooseOr}_{i,0}^{\mathtt{true}} \cup \mathsf{chooseOr}_{i,1}^{\mathtt{true}}$ which cover all required points apart from $x_{i,1}, y_{i,0}, y_{i,1}, z_{i,0}, z_{i,1}, l_{i,0}$. We cover those using additional 3 segments: $\{(x_{i,1}, l_{i,0}), (y_{i,0}, y_{i,1}), (z_{i,0}, z_{i,1})\}$.

For $a = z_{0,i}$: Using Lemma 3.6 and Lemma 3.5 with $x = p_{i,1}$, we obtain 8 segments in $\mathsf{chooseOr}_{i,0}^{\mathtt{false}} \cup \mathsf{chooseOr}_{i,1}^{\mathtt{true}}$ which cover all required points apart from $x_{i,0}, x_{i,1}, y_{i,0}, y_{i,1}, z_{i,1}, p_{i,1}$. We cover those using additional 3 segments: $\{(x_{i,0}, x_{i,1}), (y_{i,0}, y_{i,1}), (z_{i,1}, p_{i,1})\}$. $\square$

Lemma 3.8. *For any* $1 \le i \le n$ *there is a set* $\mathsf{solClause}_i^{\mathit{false}} \subseteq \mathsf{segmentsClause}_i$ *with* $|\mathsf{solClause}_i^{\mathit{false}}| = 12$ *that covers all points in* $\mathsf{pointsClause}_i$.

Proof. Using Lemma 3.6 twice we can cover $\mathsf{pointsOr}_{i,0}$ and $\mathsf{pointsOr}_{i,1}$ with 8 segments. To cover the remaining points we additionally use: $\{(x_{i,0}, x_{i,1}), (y_{i,0}, y_{i,1}), (z_{i,0}, z_{i,1}), (t_{i,1}, v_{i,1})\}$. □

Lemma 3.9. *For any $1 \leq i \leq n$:*

(1) points in $\mathsf{pointsClause}_i$ can not be covered using any subset of segments from $\mathsf{segmentsClause}_i$ of size smaller than 12;

(2) points in $\mathsf{pointsClause}_i - \{x_{i,0}, y_{i,0}, z_{i,0}\}$ can not be covered using any subset of segments from $\mathsf{segmentsClause}_i$ of size smaller than 11.

Proof of (1). No segment in $\mathsf{segmentsClause}_i$ covers more than 1 point from

$$\{x_{i,0}, y_{i,0}, z_{i,0}, l_{i,0}, p_{i,0}, q_{i,0}, u_{i,0}, v_{i,0} = l_{i,1}, p_{i,1}, q_{i,1}, u_{i,1}, v_{i,1}\}.$$

Therefore we need to use at least 12 segments. □

Proof of (2). We can define disjoint sets X, Y, Z such that

$$X \cup Y \cup Z \subseteq \mathsf{pointsClause}_i - \{x_{i,0}, y_{i,0}, z_{i,0}\}$$

and there are no segments in $\mathsf{segmentsClause}_i$ covering points from different sets. And we prove a lower bound for each of these sets. First, let:

$$X := \{x_{i,1}, y_{i,1}, z_{i,1}\}.$$

No two points in X can be covered with one segment of $\mathsf{segmentsClause}_i$, so it must be covered with 3 different segments. Next, we define the other sets:

$$Y := \mathsf{pointsOr}_{i,0} - \{l_{i,0}, p_{i,0}\},$$

$$Z := \mathsf{pointsOr}_{i,1} - \{l_{i,1}, p_{i,1}\}.$$

For both Y and Z we can check all of the subsets of 3 segments of $\mathsf{segmentsClause}_i$ to conclude that none of them cover the considered points, so both Y and Z have to be covered with disjoint sets of 4 segments each.

Therefore, $\mathsf{pointsClause}_i - \{x_{i,0}, y_{i,0}, z_{i,0}\}$ must be covered with at least $3 + 4 + 4 = 11$ segments from $\mathsf{segmentsClause}_i$. □

3.3.4. Summary

Finally we define the set of points and segments for the constructed instance:

$$\mathcal{C} := \bigcup_{1 \leq i \leq n} \mathsf{pointsVariable}_i \cup \mathsf{pointsClause}_i,$$

$$\mathcal{P} := \bigcup_{1 \leq i \leq n} \mathsf{segmentsVariable}_i \cup \mathsf{segmentsClause}_i.$$

Lemma 3.10. ***(Robustness to $\frac{1}{2}$-extension).*** *For every segment $s \in \mathcal{P}$, s and $s^{+\frac{1}{2}}$ cover the same points from $\mathcal{C}$.*

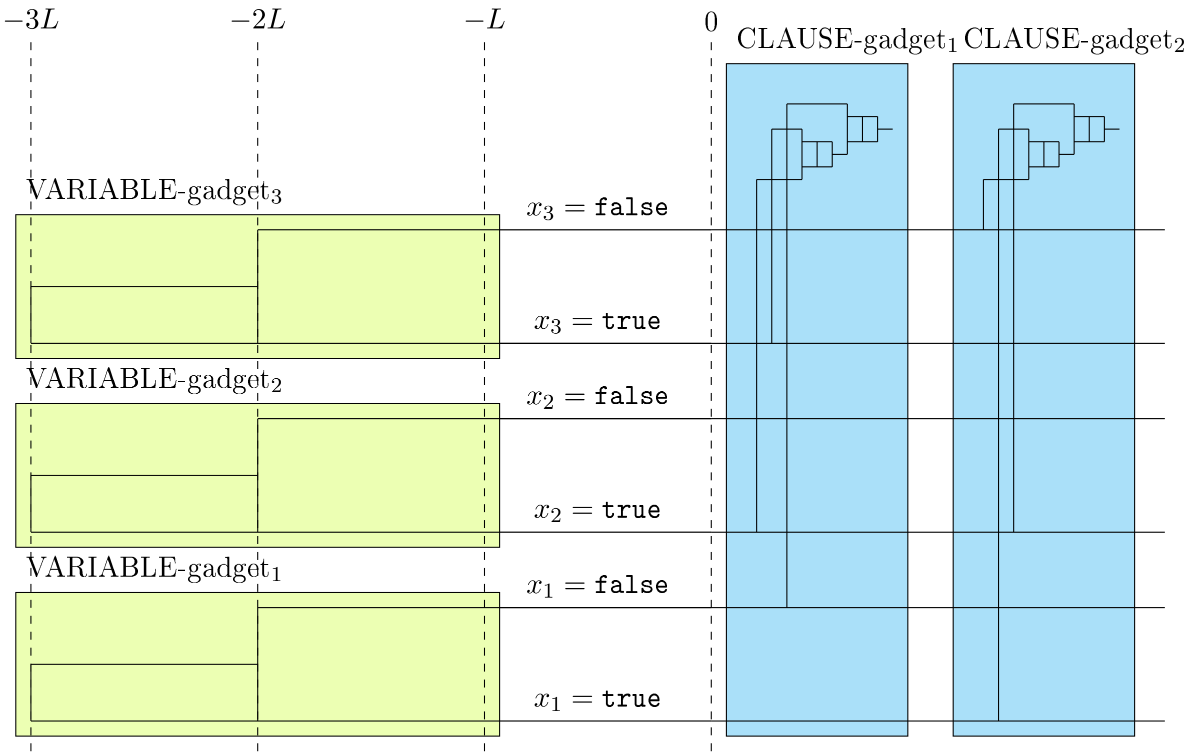

Figure 3.4: **Scheme of the whole construction.**
General layout of VARIABLE-gadgets and CLAUSE-gadgets and how they interact with each other. In green and blue we depict bounding boxes given by Claims 3.1 and 3.4, respectively.

In order to prove this lemma we will define a bounding rectangle R for every gadget, with the following property: R fits both segments and points from the gadget and $R^{+\frac{1}{2}}$ (R after $\frac{1}{2}$-extension) does not cover any points outside of R. Checking that the property from the above lemma holds for points and segments within the same gadget can be easily done using the figures above as references. We omit the proofs, and only express the necessary assertions in claims below.

Note that the claims stated below also encapsulate the interaction between the gadgets, which are also mentioned in the helper lemmas above, and prove that gadgets are independent otherwise.

First, let us define points to cover inside of rectangle R as:

$$\mathsf{points}(R) := \text{ points from } \mathcal{C} \text{ that lie in rectangle } R.$$

Claim 3.1. *For any $1 \le i \le n$, $\mathsf{pointsVariable}_i$ fit in the rectangle defined as:*

$$R_2 := [-3L, -L] \times [4i, 4i+2].$$

(1) The only points in R_2 are $\mathsf{pointsVariable}_i$: $\mathsf{points}(R_2) = \mathsf{pointsVariable}_i$.

(2) R_2 covers the same points from $\mathcal{C}$ before and after $\frac{1}{2}$-extension, i.e.

$$\mathsf{points}(R_2) = \mathsf{points}(R_2^{+\frac{1}{2}}).$$

(3) All segments of $\mathsf{segmentsVariable}_i - \{\mathsf{xTrueSegment}_i, \mathsf{xFalseSegment}_i\}$ fit fully inside of R_2.

Claim 3.2. *For any $1 \le i \le n$, $\mathsf{segmentsVariable}_i$ fit in the rectangle defined by points a_i and h_i from VARIABLE-gadget:*

$$R_1 := [-3L, L] \times [4i, 4i+2].$$

(1) The only points in R_1 are $\mathsf{pointsVariable}_i$ and $x_{j,0}, y_{j,0}$ or $z_{j,0}$ points from CLAUSE-gadgets:

$$\mathsf{pointsVariable}_i \subseteq \mathsf{points}(R_1) \subseteq \mathsf{pointsVariable}_i \cup \{x_{j,0}, y_{j,0}, z_{j,0} : 1 \le j \le n\}.$$

(2) R_1 covers the same points from $\mathcal{C}$ before and after $\frac{1}{2}$-extension, i.e. $\mathsf{points}(R_1) = \mathsf{points}(R_1^{+\frac{1}{2}})$.

(3) All segments of $\mathsf{segmentsVariable}_i$ fit fully inside of R_1.

Claim 3.3. *For any $1 \le i \le n$ and $j \in \{0, 1\}$, points from OR-gadget $\mathsf{pointsOr}_{i,j}$ and segments $\mathsf{segmentsOr}_{i,j} - \{(t_{i,j}, v_{i,j})\}$ fit in the rectangle defined as:*

$$Q_j := [x, x+2] \times [y, y+4], \text{ where } x = 20i + 3j + 3, y = 4(n+1) + 2j.$$

(1) Q_j covers only $\mathsf{pointsOr}_{i,j}$, i.e. $\mathsf{points}(Q_j) = \mathsf{pointsOr}_{i,j}$.

(2) Q_j covers the same points from $\mathcal{C}$ before and after $\frac{1}{2}$-extension, i.e. $\mathsf{points}(Q_j) = \mathsf{points}(Q_j^{+\frac{1}{2}})$.

(3) All segments of $\mathsf{segmentsOr}_{i,j} - \{(t_{i,j}, v_{i,j})\}$ fit fully inside of Q_j.

Claim 3.4. *For any* $1 \le i \le n$, $\mathsf{segmentsClause}_i$ *and* $\mathsf{pointsClause}_i$ *fit in the rectangle:*

$$Q := [20i, 20i+9] \times [0, 4(n+1)+6].$$

(1) Q covers only $\mathsf{pointsClause}_i$, *i.e.* $\mathsf{points}(Q) = \mathsf{pointsClause}_i$.

(2) Q covers the same points from $\mathcal{C}$ before and after $\frac{1}{2}$-extension, i.e. $\mathsf{points}(Q) = \mathsf{points}(Q^{+\frac{1}{2}})$.

(3) All segments of $\mathsf{segmentsClause}_i$ *fit fully inside of Q.*

With claims asserted, we can give a proof of Lemma 3.10.

Proof of Lemma 3.10. First, we check one by one for every segment within every VARIABLE-gadget and OR-gadget that if it covers some point after $\frac{1}{2}$-extension, then it covered that point before extension. In other words, every segment does not cover any new point from the same gadget after $\frac{1}{2}$-extension.

Next, we consider interactions of segments and points from different gadgets.

VARIABLE-gadget Let us fix $1 \le i \le n$ and consider segments from the i-th VARIABLE-gadget. We use Claim 3.2 and name the resulting rectangle R_1. $\mathsf{segmentsVariable}_i$ do not cover any point outside of R_1 after $\frac{1}{2}$-extension. However, some points from $\mathsf{pointsClause}_j$ for some j can lie within R_1, hence we use Claim 3.1 and name the resulting rectangle R_2. R_2 covers only points from $\mathsf{pointsVariable}_i$ (even after $\frac{1}{2}$-extension), then all points from CLAUSE-gadgets inside of R_1 lie on either $\mathsf{xTrueSegment}_i$ or $\mathsf{xFalseSegment}_i$, and it is enough to check that these segments cover exactly the same points from CLAUSE-gadgets before and after $\frac{1}{2}$-extension. They both cover all points from any CLAUSE-gadget that are collinear with these segments, so they cover exactly the same set of points after extension.

CLAUSE-gadget Let us fix $1 \le i \le n$ and consider segments from the i-th CLAUSE-gadget. We use Claim 3.3 for $j \in \{0,1\}$ to get rectangles Q_0 and Q_1 respectively. We need to check whether segments $\mathsf{moveVariableSegments}_i \cup \{(t_{i,j}, v_{i,j}) : j \in \{0,1\}\}$ cover any new points from $\mathsf{pointsClause}_i$ after $\frac{1}{2}$-extension, because their interaction is not considered by Claim 3.3 for Q_0 and Q_1.

Then we use Claim 3.4 to conclude that no segment from $\mathsf{segmentsClause}_i$ after $\frac{1}{2}$-extension covers any point from a different CLAUSE-gadget or any VARIABLE-gadget. ☐

3.4. Proof that the reduction is correct

In order to prove Lemma 3.1 we introduce several auxiliary lemmas proving properties of the construction described in the previous section.

Consider an instance S of MAX-(3,3)-SAT of size n with optimum solution satisfying k clauses. Let us construct an instance $(\mathcal{C}, \mathcal{P})$ of Segment Set Cover as described in Section 3.3 for the instance S of MAX-(3,3)-SAT.

Lemma 3.11. *The instance $(\mathcal{C}, \mathcal{P})$ of* Segment Set Cover *admits a solution of size* $15n-k$.

Proof. Let the clauses in S be $c_1, c_2, \ldots, c_n$ and the variables be $x_1, x_2, \ldots, x_n$. Let the variable assignment in the optimum solution to S be $\phi : \{x_1, x_2, \ldots, x_n\} \to \{\texttt{true}, \texttt{false}\}$.

We cover every VARIABLE-gadget with solution described in Lemma 3.2, where in the i-th gadget we choose the set of segments corresponding to the value of $\phi(x_i)$.

For every clause that is satisfied, say c_i, let us name the variable that is $\texttt{true}$ in it as x_i and the point corresponding to x_i in $\mathsf{pointsClause}_i$ as a. Points in $\mathsf{pointsClause}_i$ are covered with set $\mathsf{solClause}_i^{\texttt{true},a}$ described in Lemma 3.7. For every clause that is not satisfied, say c_j, points in $\mathsf{pointsClause}_j$ are covered with set $\mathsf{solClause}_i^{\texttt{false}}$ described in Lemma 3.8.

Formally, we define sets responsible for choosing variable assignment and satisfying clauses, R_i and C_i respectively, as following:

$$R_i := \begin{cases} \mathsf{chooseVariable}_i^{\texttt{true}} & \text{if } \phi(x_i) = \texttt{true} \\ \mathsf{chooseVariable}_i^{\texttt{false}} & \text{if } \phi(x_i) = \texttt{false} \end{cases}$$

$$C_i := \begin{cases} \mathsf{solClause}_i^{\texttt{true},a} & \text{if } c_i \text{ satisfied by the literal corresponding to point } a \\ \mathsf{solClause}_i^{\texttt{false}} & \text{if } c_i \text{ not satisfied} \end{cases}$$

$$\mathcal{R} := \bigcup_{i=1}^{n} \{R_i \cup C_i : 1 \leq i \leq n\}.$$

This set covers all the points from $\mathcal{C}$, because the sets R_i, C_i individually cover their corresponding gadgets, as proved in the respective lemmas.

All of these sets are disjoint, so the size of the obtained solution is:

$$|\mathcal{R}| = \sum_{i=1}^{n} R_i + \sum_{i=1}^{n} C_i = 3n + 11k + 12(n-k) = 15n - k. \quad \square$$

Lemma 3.12. *Suppose we have a solution $\mathcal{R}$ of the instance $(\mathcal{C}, \mathcal{P})$ of* SEGMENT SET COVER. *Then there exists a solution $\mathcal{R}'$ such that $|\mathcal{R}'| \leq |\mathcal{R}|$ and $\mathcal{R}'$ contains at most one of the segments* $\mathsf{xTrueSegment}_i$ *and* $\mathsf{xFalseSegment}_i$ *from each VARIABLE-gadget.*

Proof. Assume that we have $\{\mathsf{xTrueSegment}_i, \mathsf{xFalseSegment}_i\} \subseteq \mathcal{R}$ for some i. We will show how to modify $\mathcal{R}$ into $\mathcal{R}'$, such that the number of such i decreases, while $\mathcal{R}'$ is still a valid solution to $(\mathcal{C}, \mathcal{P})$, and $|\mathcal{R}'| \leq |\mathcal{R}|$. Then, by repeating this procedure, we can eventually construct a solution satisfying the property from the Lemma.

To construct $\mathcal{R}'$, we first remove from $\mathcal{R}$ all segments belonging to $\mathsf{segmentsVariable}_i$. Recall that the i-th VARIABLE-gadget corresponds to variable x_i in S. As every variable in S is used in exactly 3 clauses, then one literal x_i or $\neg x_i$ must appear in at least 2 clauses. If that literal is x_i, then we add to the constructed solution all segments from $\mathsf{chooseVariable}_i^{\texttt{true}}$, otherwise we add all segments from $\mathsf{chooseVariable}_i^{\texttt{false}}$.

Now, there exists at most one CLAUSE-gadget which needs adjustment to make $\mathcal{R}'$ valid; assuming it is the j-th clause, then one of the points $x_{j,0}, y_{j,0}$ or $z_{j,0}$ for this CLAUSE-gadget might be not covered, say $y_{j,0}$. We amend the solution by adding $(y_{j,0}, y_{j,1})$ to $\mathcal{R}'$.

By Lemma 3.4 we know that $\mathcal{R}$ used at least 4 segments from $\mathsf{segmentsVariable}_i$. Therefore, we removed at least 4 segments and added at most 4 segments, so $|\mathcal{R}'| \leq |\mathcal{R}|$. $\square$

Lemma 3.13. *Suppose we have a solution $\mathcal{R}$ of the instance $(\mathcal{C}, \mathcal{P})$ of* SEGMENT SET COVER. *Then there exists a solution to S that satisfies at least $15n - |\mathcal{R}|$ clauses.*

Proof. Let the clauses in S be $c_1, c_2, \ldots, c_n$ and the variables be $x_1, x_2, \ldots, x_n$. Given a solution $\mathcal{R}$ of the instance $(\mathcal{C}, \mathcal{P})$ of SEGMENT SET COVER, we use Lemma 3.12 to modify $\mathcal{R}$ so that for any i, $\mathcal{R}$ contains at most one of $\mathsf{xTrueSegment}_i$ and $\mathsf{xFalseSegment}_i$; this may decrease the size of $\mathcal{R}$, but that does not matter in the subsequent construction. To simplify notation, in the remainder of this proof we use $\mathcal{R}$ to refer to the modified solution.

Given $\mathcal{R}$, we construct a solution to S by defining an assignment of variables:

$$\phi : \{x_1, x_2, \ldots, x_n\} \to \{\texttt{true}, \texttt{false}\}$$

that satisfies at least $15n - |\mathcal{R}|$ clauses in S.

Definition of ϕ. Recall that due to Lemma 3.12, $\mathcal{R}$ contains at most one of $\mathsf{xTrueSegment}_i$ and $\mathsf{xFalseSegment}_i$.

We define the value $\phi(x_i)$ for the variable x_i as follows:

$$\phi(x_i) := \begin{cases} \texttt{true} & \text{if } \mathsf{xTrueSegment}_i \in \mathcal{R}, \\ \texttt{false} & \text{otherwise} \end{cases}$$

Moreover, from Lemma 3.3 we get $|\mathsf{segmentsVariable}_i \cap \mathcal{R}| \geq 3$ for every i.

Clauses satisfied with the chosen variable assignment. For a clause c_i, $\mathcal{R}$ needs to use at least 11 segments to cover $\mathsf{pointsClause}_i - \{x_{i,0}, y_{i,0}, z_{i,0}\}$ in the i-th CLAUSE-gadget (Lemma 3.9).

Moreover, if none of the points $\{x_{i,0}, y_{i,0}, z_{i,0}\}$ are covered by the segments from $\mathcal{R} \cap \mathsf{segmentsVariable}_i$, then $\mathcal{R}$ needs to cover $\mathsf{pointsClause}_i$ with at least 12 segments by Lemma 3.9.

Let a be the number of clauses c_i for which none of the points $x_{i,0}, y_{i,0}, z_{i,0}$ in $\mathsf{pointsClause}_i$ are covered by segments from $\mathcal{R} \cap \mathsf{segmentsVariable}_j$ for any $1 \leq j \leq n$.

Consider a clause c_i for which at least one of the points $x_{i,0}, y_{i,0}, z_{i,0}$ in $\mathsf{pointsClause}_i$ is covered by segments from $\mathcal{R} \cap \mathsf{segmentsVariable}_j$ for some $1 \leq j \leq n$. Denote this point as t and say it corresponds to literal q and variable x_j. Point t can be only covered in $\mathsf{segmentsVariable}_j$ by a corresponding segment $\mathsf{xTrueSegment}_j$ or $\mathsf{xFalseSegment}_j$ (depending on whether the literal q is negated or not). From the definition of ϕ and the fact that one of these segments is in $\mathcal{R}$, we know that $\phi(j)$ has the value that evaluates q to be $\texttt{true}$. Therefore, clause c_i is satisfied.

Consequently, ϕ satisfies all but at most a clauses in S.

To conclude, given a solution $\mathcal{R}$ to $(\mathcal{C}, \mathcal{P})$ we constructed a variable assignment ϕ that satisfies at least $n - a$ clauses of S. Finally, note that

$$|\mathcal{R}| \geq 3n + 11(n - a) + 12a = 3n + 11n + a = 14n + a,$$

hence

$$15n - |\mathcal{R}| \leq 15n - 14n - a = n - a.$$

Therefore, ϕ satisfies at least $15n - |\mathcal{R}|$ clauses of S. $\square$

Now Lemma 3.1 follows immediately from Lemmas 3.11, 3.13 and 3.10.

Chapter 4

Fixed-parameter tractable algorithm for SEGMENT SET COVER

In this chapter we show fixed-parameter tractable algorithms for the SEGMENT SET COVER problem in two different settings. Section 4.1 shows a fixed-parameter tractable algorithm for unweighted SEGMENT SET COVER. The remainder of the chapter presents a fixed-parameter tractable algorithm for WEIGHTED SEGMENT SET COVER with δ-extension. We show an algorithm for the setting with δ-extension, because the original problem with weights is W[1]-hard, as we show in Chapter 5.

We start with a shared definition for this problem. We define *extreme points* for a set of collinear points.

Definition 4.1. For a set of collinear points C in the plane, **extreme points** of C are the endpoints of the smallest segment that covers all points from set C.

If C consists of one point or is empty, then there are 1 or 0 extreme points respectively.

4.1. Fixed-parameter tractable algorithm for unweighted SEGMENT SET COVER

In this section we consider fixed-parameter tractable algorithms for SEGMENT SET COVER. The setting where segments are required to be axis-parallel (or limited to a constant number of directions) has a trivial FPT algorithm. We present an FPT algorithm for SEGMENT SET COVER, where segments are in arbitrary directions.

4.1.1. Axis-parallel segments

Theorem 4.1. ***(FPT for* SEGMENT SET COVER *with axis-parallel segments).*** *There exists an algorithm that given a family $\mathcal{P}$ of axis-parallel segments, a set of points $\mathcal{C}$ and a parameter k, runs in time $\mathcal{O}(2^k)$, and outputs a solution $\mathcal{R} \subseteq \mathcal{P}$ such that $|\mathcal{R}| \leq k$ and $\mathcal{R}$ covers all points in $\mathcal{C}$, or determines that such a set $\mathcal{R}$ does not exist.*

Proof. We show an $\mathcal{O}(2^k)$-time branching algorithm. In each step, the algorithm selects a point a which is not yet covered, branches to choose one of the two directions, and greedily chooses a segment a in that direction to cover. This proceeds until either all points are covered or k segments are chosen.

Let us take the point $a = (x_a, y_a)$ which is the smallest among points that are not yet covered in the lexicographic ordering of points in $\mathbb{R}^2$. We need to cover a with some of the remaining segments.

Branch over the choice of one of the coordinates (x or y); without loss of generality, let us assume we chose x. Among the segments lying on line $x = x_a$, we greedily add to the solution the one that covers the most points. As a was the smallest in the lexicographical order, all points on the line $x = x_a$ have the y-coordinate larger than y_a. Therefore, if we denote the greedily chosen segment as s, then any other segment on the line $x = x_a$ that covers a can only cover a subset of points covered by s. Thus, greedily choosing s is optimal.

In each step of the algorithm we add one segment to the solution, thus the recursion can be stopped at depth k. If no branch finds a solution, then this means that a solution of size at most k does not exist. $\square$

Note that the same algorithm can be used for segments in d directions, where we branch over d choices of directions, and it runs in complexity $\mathcal{O}(d^k)$.

4.1.2. Segments in arbitrary directions

In this section we consider the setting where segments are not constrained to a constant number of directions. We present a fixed-parameter tractable algorithm, parameterized by the size of the solution.

Theorem 1.2. ***(FPT for*** **Segment Set Cover*****).*** *There exists an algorithm that given a family $\mathcal{P}$ of segments (in any direction), a set of points $\mathcal{C}$ and a parameter k, runs in time $k^{\mathcal{O}(k)}(|\mathcal{C}| \cdot |\mathcal{P}|)^2$, and outputs a solution $\mathcal{R} \subseteq \mathcal{P}$ such that $|\mathcal{R}| \leq k$ and $\mathcal{R}$ covers all points in $\mathcal{C}$, or determines that such a set $\mathcal{R}$ does not exist.*

We will need the following lemmas proving properties of any instance of the problem.

Lemma 4.1. *Given an instance $(\mathcal{P}, \mathcal{C})$ of the* Segment Set Cover *problem, without loss of generality we can assume that no segment covers a superset of what another segment covers. That is, for any distinct $A, B \in \mathcal{P}$, we have $A \cap \mathcal{C} \not\subseteq B \cap \mathcal{C}$ and $A \cap \mathcal{C} \not\supseteq B \cap \mathcal{C}$.*

Proof. Assume towards a contradiction that there is an instance $(\mathcal{P}, \mathcal{C})$, and two distinct subsets of $\mathcal{P}$, A, B, such that $A \cap \mathcal{C} \subseteq B \cap \mathcal{C}$.

We construct a set $\mathcal{P}' := \mathcal{P} - \{A\}$. We prove that for any solution $\mathcal{R}$ of $(\mathcal{P}, \mathcal{C})$, we can construct a solution $\mathcal{R}' \subseteq \mathcal{P}'$, such that $|\mathcal{R}'| \leq |\mathcal{R}|$. Let us take any solution $\mathcal{R}$ of $(\mathcal{P}, \mathcal{C})$. If $A \in \mathcal{R}$, then $\mathcal{R}' := \mathcal{R} \cup \{B\} - \{A\}$, otherwise $\mathcal{R}' := \mathcal{R}$. Let us consider the case when $A \in \mathcal{R}$, because the other case is trivial. Since $A \cap \mathcal{C} \subseteq B \cap \mathcal{C}$, then $\mathcal{R} \cup \{B\} - \{A\}$ covers any point from $\mathcal{C}$ that was covered by $\mathcal{R}$. Also, $|\mathcal{R} \cup \{B\} - \{A\}| \leq |\mathcal{R}|$. $\square$

Lemma 4.2. *Given an instance $(\mathcal{P}, \mathcal{C})$ of the* Segment Set Cover *problem transformed by Lemma 4.1, if there exists a line L with at least $k+1$ points on it, then there exists a subset $A \subseteq \mathcal{P}$, of size at most k, such that every solution $\mathcal{R}$ with $|\mathcal{R}| \leq k$ satisfies $|A \cap \mathcal{R}| \geq 1$. Moreover, such a subset can be found in polynomial time.*

Proof. Let us enumerate the points from $\mathcal{C}$ that lie on L as $x_1, x_2, \ldots, x_t$ in the order in which they appear on L. Our proposed set is defined as:

$$A := \{\text{segment collinear with } L \text{ that covers } x_i \text{ and does not cover } x_{i-1} : i \in \{1, \ldots, k\}\},$$

where for $i = 1$ we just take a segment that covers x_1. If such a segment does not exist for any point x as above, then x does not give rise to any segment in A.

We prove the lemma by contradiction. Let us assume that there exists a solution $\mathcal{R}$ of size at most k such that $\mathcal{R} \cap A = \emptyset$.

Let $\mathcal{R}_L$ be the set of segments from $\mathcal{R}$ that are collinear with L.

Every segment that is not collinear with L can cover at most one of the points that lie on this line. Hence, if $\mathcal{R}_L$ was empty, then $\mathcal{R}$ would cover at most k points on line L, but L had at least $k+1$ different points from $\mathcal{C}$ on it.

Therefore, we know that $\mathcal{R}_L$ is not empty and $|\mathcal{R} - \mathcal{R}_L| \leq k-1$. Segments from $\mathcal{R} - \mathcal{R}_L$ can cover at most $k-1$ points among $\{x_1, x_2, \ldots, x_k\}$, therefore at least one of these points must be covered by segments from $\mathcal{R}_L$. We take the leftmost point from $\{x_1, x_2, \ldots, x_k\}$ that is covered in $\mathcal{R}_L$ and name it a. After the transformation from Lemma 4.1, in $\mathcal{R}$ there is only one segment that starts in a and is collinear with L, therefore this segment must be in both $\mathcal{R}$ and A. This contradiction concludes the proof that $|A \cap \mathcal{R}| \geq 1$ for any solution $\mathcal{R}$ of size at most k. $\square$

We are now ready to prove Theorem 1.2.

Proof of Theorem 1.2. We will prove this theorem by presenting a branching algorithm that works in desired complexity. It first branches over the choice of segments to cover the lines with *many* points and then solves a small instance (where every line has at most k points) by checking all possible solutions.

Algorithm. We present a recursive algorithm. Given an instance of the problem:

(1) Use Lemma 4.1 to remove some redundant segments from our instance.

(2) If there exists a line with at least $k+1$ points from $\mathcal{C}$, we branch over the choice of adding to the solution one of the at most k possible segments provided by Lemma 4.2; name this segment s and name the set of points from $\mathcal{C}$ that lie on s as S. By recursion, we find a solution $\mathcal{R}$ for the instance $(\mathcal{C} - S, \mathcal{P} - \{s\})$, and parameter $k-1$. We return $\mathcal{R} \cup \{s\}$. Note that if Lemma 4.2 returned $\emptyset$, then we respond `NO`.

(3) If every line has at most k points on it and $|\mathcal{C}| > k^2$, then answer `NO`.

(4) If $|\mathcal{C}| \leq k^2$, solve the problem by brute force: check all subsets of $\mathcal{P}$ of size at most k.

Correctness. Lemma 4.2 proves that at least one segment that we branch over in (1) must be present in every solution $\mathcal{R}$ with $|\mathcal{R}| \leq k$. Therefore, the recursive call can find a solution, provided there exists one.

In (2) the answer is no, because every line covers no more than k points from $\mathcal{C}$, which implies the same about every segment from $\mathcal{P}$. Under this assumption we can cover only k^2 points with a solution of size k, which is less than $|\mathcal{C}|$.

Checking all possible solutions in (3) is trivially correct.

Complexity. In the leaves of the recursion we have $|\mathcal{C}| \leq k^2$, so $|\mathcal{P}| \leq k^4$, because every segment can be uniquely identified by the two extreme points it covers (by Lemma 4.1). Therefore, there are $\binom{k^4}{k}$ possible solutions to check, each can be checked in time $\mathcal{O}(k|\mathcal{C}|)$. Thus, (3) takes time $k^{\mathcal{O}(k)}$.

In this branching algorithm our parameter k is decreased with every recursive call, so we have at most k levels of recursion with branching over k possibilities. Candidates to branch over can be found on each level in time $\mathcal{O}((|\mathcal{C}| \cdot |\mathcal{P}|)^{\mathcal{O}(1)})$.

Reduction from Lemma 4.1 can be implemented in time $\mathcal{O}((|\mathcal{C}| \cdot |\mathcal{P}|)^{\mathcal{O}(1)})$.

It follows that the overall complexity is $\mathcal{O}(((|\mathcal{C}| \cdot |\mathcal{P}|)^{\mathcal{O}(1)}) \cdot k^{\mathcal{O}(k)})$ $\square$

4.2. Fixed-parameter tractable algorithm for Weighted Segment Set Cover with δ-extension

In this section we consider the Weighted Segment Set Cover problem relaxed with δ-extension. We show that this problem admits an FPT algorithm when parameterized by the size of the solution and δ. In the next chapter we show that the assumption about the problem being relaxed with δ-extension is necessary: we prove that Weighted Segment Set Cover problem (without extension) is W[1]-hard, which means there does not exist any FPT algorithm parameterized by solution size for it, assuming FPT $\neq$ W[1].

Theorem 1.3. ***(FPT for* Weighted Segment Set Cover *with δ-extension).*** *There exists an algorithm that given a family $\mathcal{P}$ of n weighted segments (in any direction), a set of m points $\mathcal{C}$, and parameters k and $\delta > 0$, runs in time $f(k, \delta) \cdot (nm)^c$ for some computable function f and a constant c and outputs a set $\mathcal{R}$ such that:*

- $\mathcal{R} \subseteq \mathcal{P}$,
- $|\mathcal{R}| \leq k$,
- $\mathcal{R}^{+\delta}$ *covers all points in* $\mathcal{C}$,
- *the weight of $\mathcal{R}$ is not greater than the weight of an optimum solution of size at most k for this problem without δ-extension,*

or determines that there is no set $\mathcal{R}$ with $|\mathcal{R}| \leq k$ such that $\mathcal{R}$ covers all points in $\mathcal{C}$.

4.2.1. Dense subsets

To solve this problem we will introduce a lemma about choosing a *dense* subset of points. A dense subset of points for a set of collinear points C and parameters k and δ is a subset of C such that if we cover it with at most k segments, these segments after δ-extension will cover all of the points from C. We will prove that such set of size bounded by some function $f(k, \delta)$ always exists (Lemma 4.3). Later, Lemma 4.3 will allow us to find a kernel for our original problem.

Definition 4.2. For a set of collinear points C, a subset $A \subseteq C$ is **(k, δ)-dense** if for any set of segments R that covers A and such that $|R| \leq k$, it holds that $R^{+\delta}$ covers C.

Lemma 4.3. *For any set of collinear points C, $\delta > 0$ and $k \geq 1$, there exists a (k, δ)-dense set $A \subseteq C$ of size at most $(2 + \frac{2}{\delta})^k$. Moreover, there exists an algorithm that computes the (k, δ)-dense set in time $\mathcal{O}(|C| \cdot (2 + \frac{2}{\delta})^k)$.*

Proof. We prove this for a fixed δ by induction on k.

Inductive hypothesis. For any set of collinear points C, there exists a set A such that:

- A is subset of C,
- A is (ℓ, δ)-dense for every $1 \leq \ell \leq k$,
- $|A| \leq (2 + \frac{2}{\delta})^k$,
- the extreme points of C are in A.

Base case for $k = 1$. It is sufficient that A consists of the extreme points of C. If they are covered with one segment, it must be a segment that includes the extreme points from C, so it covers the whole set C. There are at most 2 extreme points in C and $2 < 2 + \frac{2}{\delta}$.

Inductive step. Assuming inductive hypothesis for any set of collinear points C and for parameter k, we will prove it for $k + 1$.

Let s be the minimal segment that includes all points from C. That is, the extreme points of C are endpoints of s.

We define $M = \lceil 1 + \frac{2}{\delta} \rceil$ subsegments of s by splitting s into M closed segments of equal length. We name these segments v_i, note that $|v_i| = \frac{|s|}{M}$ for each $1 \leq i \leq M$.

Let C_i be the subset of C consisting of points lying on v_i.

Let t_i be the segment with endpoints being the extreme points of C_i. It might be a degenerate segment if C_i consists of one point, or t_i might be empty if C_i is empty.

Figure 4.1 presents an example of such segments v_i and t_i.

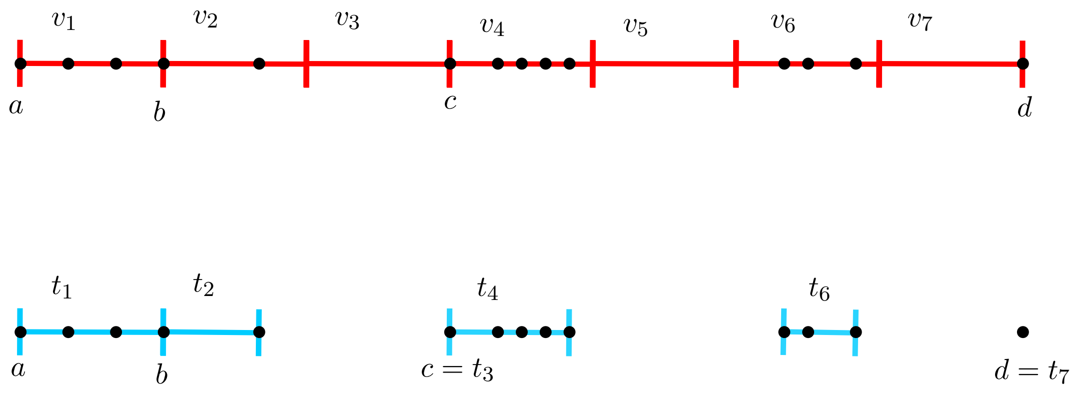

Figure 4.1: **Example of segments v_i and t_i.**
Example for $M = 7$ and some set of points (marked with black circles). The top panel shows segments v_i and the bottom panel shows segments t_i on the same set of points. a and b are the extreme points and therefore segment s ends at a and b. Red segments depict the split into M segments of equal length v_i. Blue segments depict the segments t_i. t_5 is an empty segment, because there are no points that lie on segment v_5. Segments t_3 and t_7 are degenerated to one point – c and d, respectively. Segments t_1 and t_2 share one point b.

We use the inductive hypothesis to choose (k, δ)-dense sets A_i for sets C_i. Note that if $|C_i| \leq 1$, then $A_i = C_i$ and it is still a (k, δ)-dense set for C_i.

Then we define $A = \bigcup_{i=1}^{M} A_i$. Thus A includes the extreme points of C, because they are included in the sets A_1 and A_M.

The size of each A_i is at most $(2 + \frac{2}{\delta})^k$ from the inductive hypothesis, therefore size of A is at most:

$$M\left(2 + \frac{2}{\delta}\right)^k = \left\lceil 1 + \frac{2}{\delta} \right\rceil \cdot \left(2 + \frac{2}{\delta}\right)^k \leq \left(2 + \frac{2}{\delta}\right)^{k+1}.$$

Proof that A is $(k+1, \delta)$-dense for C. Let us take any cover of A with $k+1$ segments and call it $\mathcal{R}$.

For every segment t_i, if there exists a segment x in $\mathcal{R}$ that is disjoint with t_i, then we have a cover of A_i with at most k segments using $\mathcal{R} - \{x\}$. Since A_i is (k, δ)-dense for t_i and C_i, $(\mathcal{R} - \{x\})^{+\delta}$ covers C_i. So $\mathcal{R}^{+\delta}$ covers C_i as well.

If there exists a segment t_i for which a segment x as defined above does not exist, then all $k+1$ segments that cover A_i intersect t_i. An example of such segments is depicted in Figure 4.2. Let us consider any such t_i. By the inductive hypothesis, the endpoints of s are in A_1 and A_M respectively, so $\mathcal{R}$ must cover them. For each endpoint of s, there exists a segment that contains this endpoint and intersects t_i. Let us call these two segments y and z. It follows that: $|y| + |z| + |t_i| \geq |s|$. Since $|t_i| \leq |v_i| = \frac{|s|}{M} \leq \frac{|s|}{1+\frac{2}{\delta}} = \frac{|s|\delta}{\delta+2}$, we have $\max(|y|, |z|) \geq |s|(1 - \frac{\delta}{\delta+2})/2 = \frac{|s|}{\delta+2}$.

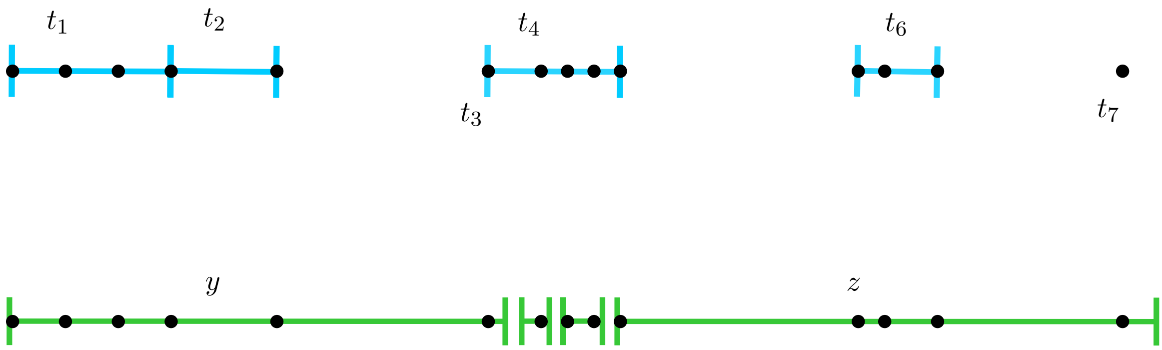

Figure 4.2: **Example of all $k+1$ segments intersecting one segment t_i.** Both panels show the same set $\mathcal{C}$ (black circles), the same as in Figure 4.1. The top panel shows blue segments t_i for $M = 7$. The bottom panel shows green segments – solution $\mathcal{R}$ of size 4. All segments from $\mathcal{R}$ intersect t_4. Segments z and y are named in the figure.

After δ-extension, the longer of these segments will expand at both ends by at least:

$$\max(|y|, |z|)\delta \geq \frac{|s|\delta}{\delta+2} = \frac{|s|}{1+\frac{2}{\delta}} \geq \frac{|s|}{M} = |v_i| \geq |t_i|.$$

Therefore, the longer of segments y and z will cover the whole segment t_i after δ-extension. We conclude that $\mathcal{R}^{+\delta}$ covers C_i.

Since $C = \bigcup_{i=1}^{M} C_i$, it follows that $\mathcal{R}^{+\delta}$ covers C.

Algorithm. We can simulate the inductive proof presented above by a recursive algorithm with the following complexity:

$$O\left(|C| + \frac{1}{\delta}\right) + O\left(|C| \cdot \left(2 + \frac{2}{\delta}\right)^k\right).$$

□

4.2.2. Algorithm

Let us now formulate some claims about the properties for the problem parameterized by the solution size. These properties provide bounds for different objects in the problem instance, which help us to find a small kernel for the problem or conclude that the optimum solution to this instance must be, in terms of size, above some threshold.

Definition 4.3. A line in the plane is **long** if there are at least $k+1$ points from $\mathcal{C}$ on it.

Claim 4.1. *If there are more than k different long lines, then $\mathcal{C}$ can not be covered with k segments.*

Proof. We prove the claim by contradiction. Let us assume that we have at least $k+1$ different long lines in our instance of the problem and there is a solution $\mathcal{R}$ of size at most k covering points $\mathcal{C}$.

Choose any long line L. Every segment from $\mathcal{R}$ which is not collinear with L, covers at most one point that lies on L. L is long, so there are at least $k+1$ points from $\mathcal{C}$ that lie on L. This implies that there must be a segment in $\mathcal{R}$ that is collinear with L.

Since we have at least $k+1$ different long lines, there are at least $k+1$ segments in $\mathcal{R}$ collinear with different lines. This contradicts with the assumption that $|\mathcal{R}| \leq k$. $\square$

Claim 4.2. *If there are more than k^2 points from $\mathcal{C}$ that do not lie on any long line, then $\mathcal{C}$ can not be covered with k segments.*

Proof. We prove the claim by contradiction. Let us assume that we have at least k^2+1 points from $\mathcal{C}$ that do not lie on any long line, call this set A, and a solution $\mathcal{R}$ of size at most k covering all points in $\mathcal{C}$.

Every segment s from $\mathcal{R}$ covers at most k points from A. This is because if s covered at least $k+1$ points from A, then the line in the direction of s would be a long line and that contradicts the definition of A.

If every segment from $\mathcal{R}$ covers at most k points from A and $|\mathcal{R}| \leq k$, then at most k^2 points from A are covered by $\mathcal{R}$ and that contradicts the fact that $\mathcal{R}$ is a solution to the given Weighted Segment Set Cover instance. $\square$

We are now ready to give a proof of Theorem 1.3.

Proof of Theorem 1.3. Our goal is to either answer `NO` or to find a kernel $(\mathcal{C}', \mathcal{P}')$ of size bounded by $f(k)$ for some function f, such that:

- *(Property 1)* for every solution $\mathcal{R}$ to $(\mathcal{C}, \mathcal{P})$ of size at most k, there exists a set $\mathcal{R}_1 \subseteq \mathcal{P}'$ such that $|\mathcal{R}_1| \leq k$, the weight of $\mathcal{R}_1$ is not greater than the weight of $\mathcal{R}$, and $\mathcal{R}_1$ covers $\mathcal{C}'$;

- *(Property 2)* for every set $\mathcal{R}_2 \subseteq \mathcal{P}'$ such that $|\mathcal{R}_2| \leq k$ and $\mathcal{R}_2$ covers all points in $\mathcal{C}'$, $\mathcal{R}_2^{+\delta}$ covers all points in the original set $\mathcal{C}$.

If we found such sets $(\mathcal{C}', \mathcal{P}')$, using *Property 1* we know that an optimum solution of size at most k to $(\mathcal{C}', \mathcal{P}')$ has no greater weight than an optimum solution of size at most k to $(\mathcal{C}, \mathcal{P})$. Using *Property 2* we know that any solution to $(\mathcal{C}', \mathcal{P}')$ after δ-extension covers $\mathcal{C}$.

Therefore, finding such sets and solving the instance $(\mathcal{C}', \mathcal{P}')$ by iterating over all of the subsets of $\mathcal{P}'$ of size at most k in desired complexity is sufficient to prove Theorem 1.3.

Definition of $\mathcal{C}'$ and $\mathcal{P}'$. Let us name the number of different long lines as l. Applying Claims 4.1 and 4.2, if we have more than k different long lines or more than k^2 points from $\mathcal{C}$ that do not lie on any long line, then we answer `NO`, because these lemmas prove that there is no solution of size at most k to this instance.

Otherwise, we can split $\mathcal{C}$ into at most $k+1$ sets:

- D: points that do not lie on any long line, $|D| \leq k^2$;

- C_i for $1 \leq i \leq l$: points that lie on the i-th long line, $|C_i| > k$.

Note that sets C_i do not need to be disjoint.

Then, for every set C_i we can use Lemma 4.3 to obtain a (k, δ)-dense set A_i for C_i with $|A_i| \leq (2 + \frac{2}{\delta})^k$.

We define $\mathcal{C}' := D \cup (\bigcup A_i)$. $\mathcal{C}'$ has size at most $k^2 + k(2 + \frac{2}{\delta})^k$. We define $\mathcal{P}'$ as follows: for every pair of points $\mathcal{C}'$, we choose one segment from $\mathcal{P}$ that has the lowest weight among segments that cover these points or decide that there is no segment that covers them. There are at most $|\mathcal{C}'|^2$ different segments in $\mathcal{P}'$, therefore both $\mathcal{P}'$ and $\mathcal{C}'$ have size bounded by $\mathcal{O}\left((k^2 + k(2 + \frac{2}{\delta}))^2\right)$.

Proof of Property 2. Firstly, we prove that for every set $\mathcal{R}_2 \subseteq \mathcal{P}'$ such that $|\mathcal{R}_2| \leq k$ and $\mathcal{R}_2$ covers points in $\mathcal{C}'$, $\mathcal{R}_2^{+\delta}$ covers points in the original instance $\mathcal{C}$.

Let us take such a set $\mathcal{R}_2$.

$\mathcal{C}$ is partitioned into several parts – sets D and C_i. Points from D are covered by $\mathcal{R}_2$, because D is part of $\mathcal{C}'$. Each point from any A_i is covered, because A_i is a part of $\mathcal{C}'$; A_i is a (k, δ)-dense set for C_i, therefore $\mathcal{R}_2^{+\delta}$ covers all points in C_i. Therefore, $\mathcal{R}_2^{+\delta}$ covers all points in $\mathcal{C}$.

Proof of Property 1. Secondly, we prove that for every solution $\mathcal{R}$ to $(\mathcal{C}, \mathcal{P})$ of size at most k, there exists a set $\mathcal{R}_1 \subseteq \mathcal{P}'$ such that $|\mathcal{R}_1| \leq k$, the weight of $\mathcal{R}_1$ is not greater than the weight of $\mathcal{R}$ and $\mathcal{R}_1$ covers $\mathcal{C}'$.

For every segment in $\mathcal{R}$, say s, let us look at the points from $\mathcal{C}'$ that lie on s and call this set of points F. F is of course a set of collinear points. We can cover F with any segment that covers extreme points of F, because all other points lie on the segment between these points. Therefore, we can replace s with a segment s' that has lowest weight among the points that cover the extreme points of F. Such a segment belongs to $\mathcal{P}'$, because this is how it was defined. Segment s' has weight no greater than the weight of s, because s also covers F.

Therefore, we produced the set $\mathcal{R}_1$ that has size not greater than the size of $\mathcal{R}$ (because some segments s can map to the same segment s'), weight not greater than $\mathcal{R}$, and it covers $\mathcal{C}'$.

Complexity We find a solution of $(\mathcal{C}', \mathcal{P}')$ by iterating over all the possible subsets of $\mathcal{P}'$. Finding sets $\mathcal{P}'$ and $\mathcal{C}'$ and then solving problem for kernel has overall complexity $(|\mathcal{P}| + |\mathcal{C}|)^{\mathcal{O}(1)}\mathcal{O}((2 + \frac{2}{\delta})^k) + \mathcal{O}((k^2 + k(2 + \frac{2}{\delta})^k)^k)$. $\square$

Chapter 5

W[1]-hardness of Weighted Segment Set Cover

In this chapter we consider the Weighted Segment Set Cover problem with axis-parallel or right-diagonal segments. In Theorem 1.4 below, we prove that this problem is W[1]-hard when parameterized by the size of the solution. We believe that the construction can be improved to only utilize the axis-parallel segments.

Theorem 1.4. ***(Weighted Segment Set Cover is W[1]-hard).*** *Consider the problem of covering a set $\mathcal{C}$ of points by selecting at most k segments from a set of segments $\mathcal{P}$ with non-negative weights $w : \mathcal{P} \to \mathbb{R}^+$ so that the weight of the cover is minimal. Then this problem is W[1]-hard when parameterized by k and assuming ETH, there is no algorithm for this problem with running time $f(k) \cdot (|\mathcal{C}| + |\mathcal{P}|)^{o(\sqrt{k})}$ for any computable function f. Moreover, this holds even if all segments in $\mathcal{P}$ are axis-parallel or right-diagonal.*

5.1. Grid Tiling

In order to prove Theorem 1.4 we will show a reduction from a W[1]-hard problem: Grid Tiling. This problem was introduced in [Marx, 2007] (the author called it matrix tiling instead). It was originally described as an approximation problem, but W[1]-hardness follows directly from the theorems stated there. For a more contemporary description of this problem and a proof of W[1]-hardness, see Chapter 14 of [Cygan et al., 2015].

Definition 5.1. We define the **powerset** of a set A, denoted as $\mathsf{Pow}(A)$, as the set of all subsets of A, i.e. $\mathsf{Pow}(A) = \{B : B \subseteq A\}$.

Definition 5.2. In the **Grid Tiling** problem we are given integers n and k, and a function $f : \{1,\ldots,k\} \times \{1,\ldots,k\} \to \mathsf{Pow}(\{1,\ldots,n\} \times \{1,\ldots,n\})$ specifying the set of allowed tiles for each cell of a $k \times k$ grid. The task is to decide whether there exist functions $x, y : \{1,\ldots,k\} \to \{1,\ldots,n\}$ that assign colors from $\{1,\ldots,n\}$ to respectively columns and rows of the grid, so that $(x(i), y(j)) \in f(i,j)$ for all $i, j \in \{1,\ldots,k\}$.

In short, in the Grid Tiling problem one needs to assign numbers to rows and columns in such a way that for every pair of a row and a column, the pair of colors assigned to the row and column belongs to the allowed set of tiles for this pair. The next theorem describes the complexity of this problem, which is W[1]-hard when parameterized by the size of the grid.

	$x(1) = 3$	$x(2) = 1$	$x(3) = 3$	$x(4) = 7$
$y(4) = 1$	$(2,1)$; $(2,2)$; $\mathbf{(3{,}1)}$; $(3,9)$	$\mathbf{(1{,}1)}$; $(3,1)$	$\mathbf{(3{,}1)}$; $(7,2)$	$(2,1)$; $\mathbf{(7{,}1)}$
$y(3) = 1$	$(2,1)$; $\mathbf{(3{,}1)}$; $(4,2)$; $(8,2)$	$\mathbf{(1{,}1)}$; $(1,3)$	$\mathbf{(3{,}1)}$; $(4,3)$	$\mathbf{(2{,}2)}$; $\mathbf{(7{,}1)}$
$y(2) = 6$	$(2,6)$; $\mathbf{(3{,}6)}$	$(1,2)$; $\mathbf{(1{,}6)}$; $(2,6)$	$(2,6)$; $\mathbf{(3{,}6)}$	$(2,6)$; $\mathbf{(7{,}6)}$
$y(1) = 4$	$(2,4)$; $(2,6)$; $\mathbf{(3{,}4)}$; $(3,9)$	$\mathbf{(1{,}4)}$; $(1,9)$	$\mathbf{(3{,}4)}$; $(3,9)$	$(2,9)$; $\mathbf{(7{,}4)}$

Figure 5.1: **Example of a Grid Tiling instance and its solution.**
In the first row and column of the table you can see the solution: functions x and y. The tiles used in this solution are marked in **bold**. If we instead chose the tiles marked in blue (whenever there is one, taking the tile marked in **bold** otherwise), then that corresponds to setting $x(1) = 2$, and would also form a correct solution. On the other hand, if we instead chose the tiles marked in red (as before), then this corresponds to setting $y(1) = 9$ and $x(4) = 2$ and that would **not** form a correct solution. Even though the first row is correct, the cell with coordinates (3,4) requires tile (2,1), not (2,2) (marked in **bold red**).

Theorem 5.1. ***([Marx, 2007]).*** *Grid Tiling is W[1]-hard when parameterized by k and assuming ETH, there is no $f(k) \cdot n^{o(k)}$-time algorithm solving the Grid Tiling problem for any computable function f.*

The remainder of this section is devoted to proving Theorem 1.4 by a reduction from a Grid Tiling problem instance with parameter k (number of rows in the grid) to a Weighted Segment Set Cover instance with parameter k^2 (size of solution). This reduction is described in Lemma 5.1. This proves the W[1]-hardness of the Weighted Segment Set Cover problem, because if we could solve it with an FPT algorithm, then we could also solve the Grid Tiling problem (which we reduced to Weighted Segment Set Cover). Therefore, Weighted Segment Set Cover with setting described in Theorem 1.4 is at least as hard as the Grid Tiling problem.

5.2. Statement of reduction

Let us denote an instance of Grid Tiling problem as (n, k, f) consisting of:

- the number of colors n,
- the size of the grid k,
- the function specifying the allowed tiles $f : \{1, \ldots, k\} \times \{1, \ldots, k\} \to \mathsf{Pow}(\{1, \ldots, n\} \times \{1, \ldots, n\})$.

Let us also define constants:

$$\begin{aligned} \epsilon &:= \frac{1}{2k^2} \\ \delta &:= \frac{1}{4k^4} \\ W_{\mathsf{hv}} &:= 2k^2(n^2+1) - 4k^2\epsilon - 4k(1-\epsilon) \end{aligned}$$

which are going to be used when defining the weight of the constructed instance of Weighted Segment Set Cover.

Lemma 5.1. *Given an instance (n, k, f) of the* Grid Tiling *problem, we can construct an instance $(\mathcal{C}, \mathcal{P}, w, 3k^2 + 2k)$ of* Weighted Segment Set Cover *such that:*

(1) if the answer to (n, k, f) is `YES`*, then there exists a solution to $(\mathcal{C}, \mathcal{P}, w, 3k^2 + 2k)$ of weight at most $W_{\mathsf{hv}} + k^2\delta$;*

(2) if there exists a solution to $(\mathcal{C}, \mathcal{P}, w, 3k^2 + 2k)$ of weight at most $W_{\mathsf{hv}} + k^2\delta$, then the answer to (n, k, f) is `YES`*.*

First, let us prove Theorem 1.4 using Lemma 5.1.

Proof of Theorem 1.4. Let us take any instance (n, l, f) of the Grid Tiling problem. We prove the theorem by contradiction, therefore we assume that Weighted Segment Set Cover parameterized by solution size $k = 3l^2 + 2l$ admits a $g(k) \cdot n^{o(\sqrt{k})}$-time algorithm for some computable function g.

Using Lemma 5.1 let us construct an instance I for (n, l, f). Let us assume that the optimum solution of size at most k to the instance I has weight u. Using (2) we know that if $u \leq W_{\mathsf{hv}} + k^2\delta$, then the answer to (n, l, f) is `YES`. If $u > W_{\mathsf{hv}} + k^2\delta$, then using (1) we know that the answer to (n, l, f) must be `NO`.

Therefore if we could find the solution in time $g(k) \cdot n^{o(\sqrt{k})}$, then we could solve the Grid Tiling problem in time $g(l) \cdot n^{o(l)}$ by constructing an instance of Weighted Segment Set Cover, solving it for parameter k in time $n^{o(\sqrt{3l^2+2l})}$ and then answering based on the weight of the optimum solution. As $\mathcal{O}(n^{o(l)}) \subseteq \mathcal{O}(n^{o(\sqrt{3l^2+2l})})$, the existence of this algorithm contradicts Theorem 5.1. Hence such an algorithm can not exist. $\square$

We prove Lemma 5.1 in subsequent sections. First, we define a constructed instance I, later property (1) is proved by Lemma 5.2 and property (2) is proved by Lemma 5.6.

In the proof of Lemma 5.1 (see proof of Lemma 5.6) we do not use the assumption that the solution is bounded by the size, which the problem is parameterized by, $3k^2 + 2k$. If we had a permissive FPT algorithm that finds a solution of any size that still has weight no more than $W_{\mathsf{hv}} + k^2\delta$, then we still would have a contradiction with Grid Tiling being W[1]-hard in proof of Theorem 1.4. Thus, this reduction proves that the problem is not only W[1]-hard, but assuming ETH there also does not exist permissive FPT algorithm for this problem. Formally we state this in Theorem 5.2 below.

Theorem 5.2. ***(Permissive FPT does not exist).*** *Consider the problem of covering a set $\mathcal{C}$ of points using segments from a set $\mathcal{P}$ with non-negative weights $w : \mathcal{P} \to \mathbb{R}^+$ so that the weight of the cover is minimal. Let $\mathcal{R}^k$ be the optimum solution to this problem of size at most k. The task is to find a solution $\mathcal{R}$ of any size such that weight of $\mathcal{R}$ is not greater than the weight of $\mathcal{R}^k$.*

Assuming ETH, there is no algorithm for this problem with running time $f(k) \cdot (|\mathcal{C}| + |\mathcal{P}|)^{o(\sqrt{k})}$ for any computable function f. Moreover, this holds even if all segments in $\mathcal{P}$ are axis-parallel or right-diagonal.

5.3. Construction of the Segment Set Cover instance

We construct an instance $(\mathcal{C}, \mathcal{P}, w, 3k^2 + 2k)$ of Segment Set Cover as follows.

First, let us choose any bijection $\mathsf{order} : \{1, \ldots, n^2\} \to \{1, \ldots, n\} \times \{1, \ldots, n\}$.

Define $\mathsf{match}_v(i, j)$ and $\mathsf{match}_h(i, j)$ as Boolean functions denoting whether two points share x or y coordinate:

$$\mathsf{match}_v(i, j) \text{ is } \mathtt{true} \iff \mathsf{order}(i) \text{ and } \mathsf{order}(j) \text{ have the same x coordinate,}$$

$$\mathsf{match}_h(i, j) \text{ is } \mathtt{true} \iff \mathsf{order}(i) \text{ and } \mathsf{order}(j) \text{ have the same y coordinate.}$$

5.3.1. Points

For $1 \leq i, j \leq k$ and $1 \leq t \leq n^2$ define points:

$$h_{i,j,t} := (i \cdot (n^2 + 1) + t, j \cdot (n^2 + 1)),$$

$$v_{i,j,t} := (i \cdot (n^2 + 1), j \cdot (n^2 + 1) + t).$$

Let us define sets H and V as:

$$H := \{h_{i,j,t} : 1 \leq i, j \leq k, 1 \leq t \leq n^2\},$$

$$V := \{v_{i,j,t} : 1 \leq i, j \leq k, 1 \leq t \leq n^2\}.$$

Let us recall that $\epsilon = \frac{1}{2k^2}$. For a point $p = (x, y)$ we define points:

$$p^L := (x - \epsilon, y),$$

$$p^R := (x + \epsilon, y),$$

$$p^U := (x, y + \epsilon),$$

$$p^D := (x, y - \epsilon).$$

Then we define the point set as follows:

$$\mathcal{C} := H \cup \{p^L : p \in H\} \cup \{p^R : p \in H\} \cup V \cup \{p^U : p \in V\} \cup \{p^D : p \in V\}.$$

Definition 5.3. For every point $p \in H$, we name point p^L its **left guard** and point p^R its **right guard**.

Similarly for every points $p \in V$, we name point p^D its **lower guard** and point p^U its **upper guard**.

5.3.2. Segments

For $1 \leq i, j \leq k$ and $1 \leq t, t_1, t_2 \leq n^2$ define segments:

$$\begin{aligned}
\mathsf{hor}_{i,j,t_1,t_2} &:= (h^R_{i,j,t_1}, h^L_{i+1,j,t_2}), \\
\mathsf{ver}_{i,j,t_1,t_2} &:= (v^U_{i,j,t_1}, v^D_{i,j+1,t_2}), \\
\mathsf{horBeg}_{i,t} &:= (h^L_{1,i,1}, h^L_{1,i,t}), \\
\mathsf{horEnd}_{i,t} &:= (h^R_{k,i,t}, h^R_{k,i,n^2}), \\
\mathsf{verBeg}_{i,t} &:= (v^D_{i,1,1}, v^D_{i,1,t}), \\
\mathsf{verEnd}_{i,t} &:= (v^U_{i,k,t}, v^U_{i,k,n^2}).
\end{aligned}$$

Next, we define sets of vertical and horizontal segments:

$$\begin{aligned}\mathsf{HOR} &:= \{\mathsf{hor}_{i,j,t_1,t_2} : 1 \le i < k, 1 \le j \le k, 1 \le t_1, t_2 \le n^2, \mathsf{match}_h(t_1,t_2) \text{ holds}\}\\ &\cup \;\; \{\mathsf{horBeg}_{i,t} : 1 \le i \le k, 1 \le t \le n^2\}\\ &\cup \;\; \{\mathsf{horEnd}_{i,t} : 1 \le i \le k, 1 \le t \le n^2\},\end{aligned}$$

$$\begin{aligned}\mathsf{VER} &:= \{\mathsf{ver}_{i,j,t_1,t_2} : 1 \le i \le k, 1 \le j < k, 1 \le t_1, t_2 \le n^2, \mathsf{match}_v(t_1,t_2) \text{ holds}\}\\ &\cup \;\; \{\mathsf{verBeg}_{i,t} : 1 \le i \le k, 1 \le t \le n^2\}\\ &\cup \;\; \{\mathsf{verEnd}_{i,t} : 1 \le i \le k, 1 \le t \le n^2\}.\end{aligned}$$

An example is depicted in Figure 5.3.

Finally, we also define a set of right-diagonal segments:

$$\mathsf{DIAG} := \{(h_{i,j,t}, v_{i,j,t}) : 1 \le i,j \le k, 1 \le t \le n^2, \mathsf{order}(t) \in f(i,j)\}.$$

An example of such segments is depicted in Figure 5.2.

Every segment in DIAG connects points $(i(n^2+1)+t, j(n^2+1))$ and $(i(n^2+1), j(n^2+1)+t)$ for some $1 \le i,j \le k, 1 \le t \le n^2$. The line on which it lies can be described by linear equation $x + y = t + (i+j)(n^2+1)$, thus these segments are in fact right-diagonal.

The constructed segment set is defined as:

$$\mathcal{P} := \mathsf{HOR} \cup \mathsf{VER} \cup \mathsf{DIAG}.$$

The weight of each segment in $\mathsf{HOR} \cup \mathsf{VER}$ is equal to its length, while every segment in DIAG has weight δ.

$$w(s) = \begin{cases} length(s) & \text{if } s \in \mathsf{HOR} \cup \mathsf{VER} \\ \delta & \text{if } s \in \mathsf{DIAG} \end{cases}$$

5.4. Proof that the reduction is correct

Now, we prove that the constructed instance of Weighted Segment Set Cover indeed gives a correct and sound reduction of the Grid Tiling problem. Lemma 5.2 proves that if a solution to the instance of the Grid Tiling instance exists, then there exists a solution with suitably bounded size and weight of the constructed instance of Weighted Segment Set Cover. Then Lemma 5.6 proves that if there is a solution to the Weighted Segment Set Cover instance with bounded weight, then there exists a solution to the original Grid Tiling instance.

Lemma 5.2. *If there exists a solution to the* Grid Tiling *instance (n,k,f), then there exists a solution to the instance $(\mathcal{C}, \mathcal{P}, w, 3k^2+2k)$ of* Weighted Segment Set Cover *with weight $W_{\mathsf{hv}} + k^2\delta$.*

Proof. Suppose there exists a solution x, y of the instance (n,k,f) of the Grid Tiling problem.

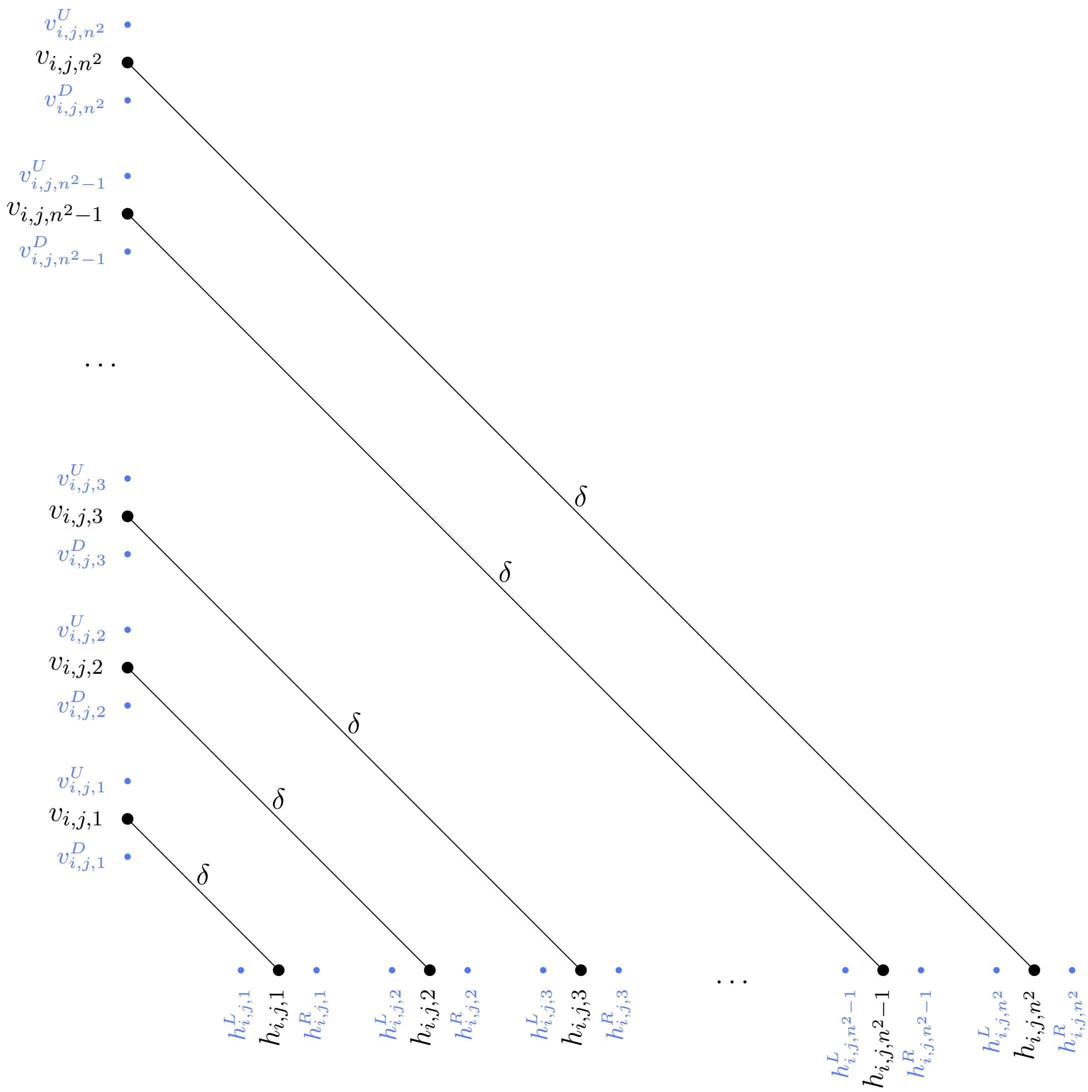

Figure 5.2: **Vertices and segments in DIAG.**
This is an example of constructed points any $1 \leq i, j \leq k$. Points from H and V are marked in black, their guards are marked in blue. You can also see segments from DIAG with their weights (equal to δ).

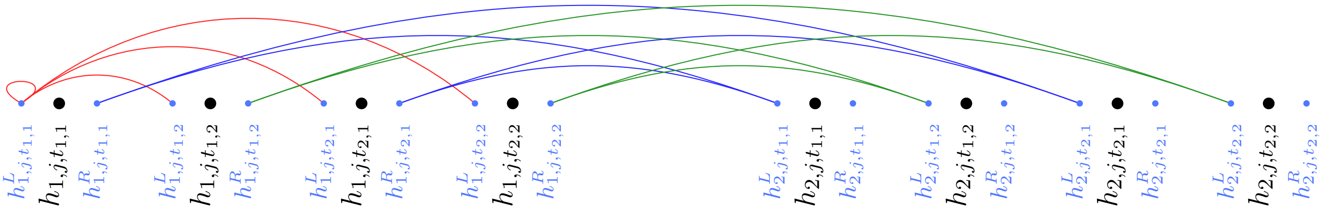

Figure 5.3: **Vertices and segments in HOR.**
This is an example for $n = 2$ and any $1 \leq j \leq k$. Points from H are marked in black, their guards are marked in light blue. $t_{i,j}$ is a notation that we use for $\mathsf{order}^{-1}(i, j)$. Segments are represented as arcs between endpoints. You can see $\mathsf{horBeg}_{j,t}$ segments in red. $\mathsf{horBeg}_{j,1}$ is degenerated to a single point at $h^L_{1,1,t_{1,1}}$. Segments $\mathsf{hor}_{i,j,t_{x_1,y},t_{x_2,y}}$ are marked in blue and green. Blue segments connect $t_{x_1,y}$ and $t_{x_2,y}$ such that they share y-coordinate equal to 1, for green segments it is equal to 2.

We define the proposed solution $\mathcal{R} \subseteq \mathcal{P}$ of the instance of Weighted Segment Set Cover in three parts: $D \subseteq \mathsf{DIAG}$, $A \subseteq \mathsf{HOR}$ and $B \subseteq \mathsf{VER}$:

$$
\begin{aligned}
D &:= \{(v_{i,j,t}, h_{i,j,t}) : 1 \leq i, j \leq k, t = \mathsf{order}^{-1}(x(i), y(j))\},\\
A &:= \{\mathsf{horBeg}_{i,\mathsf{order}^{-1}(x(1),y(i))} : 1 \leq i \leq k\}\\
&\cup \{\mathsf{horEnd}_{i,\mathsf{order}^{-1}(x(k),y(i))} : 1 \leq i \leq k\}\\
&\cup \{\mathsf{hor}_{i,j,\mathsf{order}^{-1}(x(i),y(j)),\mathsf{order}^{-1}(x(i+1),y(j))} : 1 \leq i < k, 1 \leq j \leq k\},\\
B &:= \{\mathsf{verBeg}_{i,\mathsf{order}^{-1}(x(i),y(1))} : 1 \leq i \leq k\}\\
&\cup \{\mathsf{verEnd}_{i,\mathsf{order}^{-1}(x(i),y(k))} : 1 \leq i \leq k\}\\
&\cup \{\mathsf{ver}_{i,j,\mathsf{order}^{-1}(x(i),y(j)),\mathsf{order}^{-1}(x(i),y(j+1))} : 1 \leq i \leq k, 1 \leq j < k\},
\end{aligned}
$$

$$\mathcal{R} := D \cup A \cup B.$$

Since $\mathcal{C} = H \cup V$, we show that $\mathcal{R}$ covers the whole set H; the proof for V is analogous.

Fix any $1 \leq j \leq k$ and define $t_i := \mathsf{order}^{-1}(x(i), y(j))$. The two leftmost segments in A for this j are $\mathsf{horBeg}_{j,t_1} = (h^L_{1,j,1}, h^L_{1,j,t_1})$ and $\mathsf{hor}_{1,j,t_1,t_2} = (h^R_{1,j,t_1}, h^L_{2,j,t_2})$. Therefore, points $h_{1,j,x}, h^L_{1,j,x}$ and $h^R_{1,j,x}$ for all $1 \leq x \leq n^2$ ale covered by horBeg_{j,t_1} and $\mathsf{hor}_{1,j,t_1,t_2}$, excluding point h_{1,j,t_1}.

Analogously for $2 \leq i \leq k-1$, the two consecutive segments $\mathsf{hor}_{i-1,j,t_{i-1},t_i}$ and $\mathsf{hor}_{i,j,t_i,t_{i+1}}$ cover points $h_{i,j,x}, h^L_{i,j,x}$ and $h^R_{i,j,x}$ for all $1 \leq x \leq n^2$, excluding point h_{i,j,t_i}.

Finally $\mathsf{hor}_{k-1,j,t_{k-1},t_k}$ and horEnd_{j,t_k} cover all points $h_{k,j,x}, h^L_{k,j,x}$ and $h^R_{k,j,x}$ for $1 \leq x \leq n^2$, excluding point h_{k,j,t_k}.

D covers all points h_{i,j,t_i} and v_{i,j,t_i}. As j was chosen arbitrarily, all points in H are covered.

The size of this proposed solution is:

$$|\mathcal{R}| = |D| + |A| + |B| = k^2 + (k+1)k + (k+1)k = 3k^2 + 2k.$$

Then, we need to compute the total weight of the solution $\mathcal{R}$. First, we compute the sum of weights of segments in A. Fix $1 \leq j \leq k$ and consider segments collinear with the j-th horizontal line. All points $h_{i,j,t}$, $h^L_{i,j,t}$ and $h^R_{i,j,t}$ for every $1 \leq i \leq k$ and $1 \leq t \leq n^2$ are covered by A excluding points $h_{i,j,\mathsf{order}^{-1}(x(i),y(j))}$. Every such point leaves a gap of length 2ϵ between $h^L_{i,j,\mathsf{order}^{-1}(x(i),y(j))}$ and $h^R_{i,j,\mathsf{order}^{-1}(x(i),y(j))}$. Therefore, the total weight of segments in A that lie on the line in question equals the length of the segment $(h^L_{i,1,1}, h^R_{i,k,n^2})$ minus $2\epsilon k$, which is $k(n^2+1) - 2(1-\epsilon) - 2k\epsilon$. We need to multiply that by k, as we consider all possible values of j.

Computation for vertical segments is analogous and yields the same result. Every segment in D has weight δ, therefore the sum of all weights is equal to:

$$2k(k(n^2+1) - 2(1-\epsilon) - 2k\epsilon) + k^2\delta = W_{\mathsf{hv}} + k^2\delta. \quad \square$$

Now we present a few additional properties of the constructed instance of the Weighted Segment Set Cover that help us to prove Lemma 5.6.

Claim 5.1. *In any solution to the instance $(\mathcal{C}, \mathcal{P}, w, 3k^2+2k)$:*

- *the left and right guards of points in H (points in $\{p^L : p \in H\} \cup \{p^R : p \in H\}$) have to be covered with segments from* HOR*,*
- *the lower and upper guards of points in V (points in $\{p^D : p \in V\} \cup \{p^U : p \in V\}$) have to be covered with segments from* VER*.*

Proof. We prove the claim for the points from H as the proof for points from V is analogous.

Every segment in VER is vertical and has x-coordinate equal to $i(n^2+1)$ for some $1 \le i \le k$, so they all have different x-coordinate than any left or right guard of points in H.

For every point x which is a left or right guard of a point in H, there are kn^2 segments from DIAG that intersect with the horizontal line that goes through x. All of these segments intersect with this line in points from set H, therefore none of them covers any of the guards.

Therefore none of the segments from VER or DIAG covers any of the guards of the points in H. $\square$

Claim 5.2. *For any $1 \le i, j \le n$ and any solution to the instance $(\mathcal{C}, \mathcal{P}, w, 3k^2+2k)$, all but at most one point $h_{i,j,t}$ and at most one point $v_{i,j,t}$ for $1 \le t \le n^2$ must be covered with segments from* HOR *or* VER*.*

Proof. We prove the claim for horizontal segments, as the proof for vertical segments is analogous.

We prove this by contradiction. Assume that we have two points $h_{i,j,t_1}, h_{i,j,t_2}, 1 \le t_1 < t_2 \le n^2$, such that they are not covered with segments from HOR.

Point h^R_{i,j,t_1} has to be covered with a segment from HOR by Claim 5.1. Every segment in HOR covering h^R_{i,j,t_1} but not h_{i,j,t_1} must start at h^R_{i,j,t_1} and all such segments cover also h_{i,j,t_2}. This contradicts the assumption, which concludes the proof. $\square$

Lemma 5.3. *For every solution $\mathcal{R}$ to the instance $(\mathcal{C}, \mathcal{P}, w, 3k^2+2k)$, the sum of weights of segments chosen from sets* HOR *and* VER *is at least W_{hv}.*

Proof. Let us fix $1 \le i \le k$.

We provide a lower bound for the sum of lengths of vertical segments from $\mathcal{R} \cap \mathsf{VER}$. This bound is the same for each i and is the same for horizontal lines, thus we need to multiply such a bound by $2k$.

(1) The total length between $v^D_{i,1,1}$ and v^U_{i,k,n^2} is:

$$(k(n^2+1) + n^2 + \epsilon) - ((n^2+1) + 1 - \epsilon) = k(n^2+1) - 2(1-\epsilon).$$

(2) For every $1 \le j \le k$ there exists at most one $1 \le t \le n^2$ such that $v_{i,j,t}$ is not covered by segments from VER (Claim 5.2). Its guards (see Definition 5.3) $v^U_{i,j,t}$ and $v^D_{i,j,t}$ have to be covered in VER (Claim 5.1). Therefore, at most k spaces of length 2ϵ can be left not covered by segments from VER between $v^D_{i,1,1}$ and v^U_{i,k,n^2}.

The sum of these lower bounds for vertical and horizontal lines is:

$$2k(k(n^2+1) - 2k\epsilon - 2(1-\epsilon)) = 2k^2(n^2+1) - 4k^2\epsilon - 4k(1-\epsilon) = W_{\mathsf{hv}}. \quad \square$$

Lemma 5.4. *Let $\mathcal{R}$ be a solution to a constructed instance $(\mathcal{C}, \mathcal{P}, w, 3k^2+2k)$ with weight at most $W_{\mathsf{hv}} + k^2\delta$. Then for every $1 \le i, j \le k$ there exists $1 \le t \le n^2$ such that:*

(1) $v_{i,j,t}, h_{i,j,t}$ are not covered by segments from VER or HOR;

(2) segment $(v_{i,j,t}, h_{i,j,t})$ is in solution $\mathcal{R}$;

(3) $\mathsf{order}(t) \in f(i,j)$, that is, $\mathsf{order}(t)$ is an allowed tile for (i,j);

(4) for every $1 \le s \le n^2$, $s \ne t$, $v_{i,j,s}$ is covered in VER;

(5) for every $1 \le s \le n^2$, $s \ne t$, $h_{i,j,s}$ is covered in HOR.

Proof. At most one of the points $\{h_{i,j,t_x} : 1 \le t_x \le n^2\}$ and one of the points $\{v_{i,j,t_y} : 1 \le t_y \le n^2\}$ is covered with DIAG (Claim 5.2).

Moreover, exactly one such point h_{i,j,t_x} and one such point v_{i,j,t_y} is covered with DIAG, because if none of them were covered, then the solution would have to have weight at least $W_{\mathsf{hv}} + 2\epsilon$ (see the proof of Lemma 5.3), which is more than $W_{\mathsf{hv}} + k^2\delta$.

We observe that points h_{i,j,t_x} and v_{i,j,t_y} have to be covered with the same segment from DIAG. Indeed we need to use at least k^2 of them to use exactly one DIAG segment for every pair of $1 \le i, j \le k$, if we used 2 segments from DIAG for one pair (i,j), then we would have used total weight at least $W_{\mathsf{hv}} + k^2\delta + \delta$ (Lemma 5.3), which is more than $W_{\mathsf{hv}} + k^2\delta$. Since points h_{i,j,t_x} and v_{i,j,t_y} are covered by a single segment from DIAG, we have $t_x = t_y$.

Therefore $t_x = t_y$ and $\mathsf{order}(t_x)$ is an allowed tile for (i,j) because the corresponding segment is in DIAG. $\square$

We refer to the function mapping from a pair (i,j), where $1 \le i, j \le k$, to a number t_x from Lemma 5.4 as $\mathsf{diagonal} : \{1, \dots, k\} \times \{1, \dots, k\} \to \{1, \dots, n^2\}$.

Lemma 5.5. *Let $\mathcal{R}$ be any solution of a constructed instance $(\mathcal{C}, \mathcal{P}, w, 3k^2+2k)$ with weight at most $W_{\mathsf{hv}} + k^2\delta$. Then:*

1. *for any $1 \le i < k, 1 \le j \le k$, $\mathsf{match}_h(\mathsf{diagonal}(i,j), \mathsf{diagonal}(i+1,j))$ is `true`;*

2. *for any $1 \le i \le k, 1 \le j < k$, $\mathsf{match}_v(\mathsf{diagonal}(i,j), \mathsf{diagonal}(i,j+1))$ is `true`.*

Proof. We prove (1) by contradiction, the proof of (2) is analogous.

Let us take any $1 \le i < k, 1 \le j \le k$ and name $t_1 = \mathsf{diagonal}(i,j)$ and $t_2 = \mathsf{diagonal}(i+1,j)$. We also assume that $\mathsf{match}_h(t_1,t_2)$ is `false`, which is equivalent to the fact that segment $(h^R_{i,j,t_1}, h^L_{i+1,j,t_2})$ is not in set HOR.

Therefore h_{i,j,t_1} and h_{i+1,j,t_2} are not covered by segments from HOR (Lemma 5.4), while h^R_{i,j,t_1} and h^L_{i+1,j,t_2} have to be covered by segments from HOR (Claim 5.1).

Every segment from HOR either:

- starts at point h^R_{x,y,z_1} and ends at point h^L_{x+1,y,z_2} for some $1 \le x < k$,$1 \le y \le k$ and $1 \le z_1, z_2 \le n^2$; or
- is $\mathsf{horBeg}_{y,z}$ and starts at $h^L_{1,y,1}$ and ends at $h^L_{1,y,z}$ for some $1 \le y \le k$ and $1 \le z \le n^2$; or
- is $\mathsf{horEnd}_{y,z}$ and starts at $h^R_{k,y,z}$ and ends at h^R_{k,y,n^2} for some $1 \le y \le k$ and $1 \le z \le n^2$.

All of the points between h^R_{i,j,t_1} and h^L_{i+1,j,t_2} are covered by segments in HOR and there is no segment $(h^R_{i,j,t_1}, h^L_{i+1,j,t_2})$ in HOR. Hence, there are at least two different segments covering them. If both of these segments are neither $\mathsf{horBeg}_{y,z}$ nor $\mathsf{horEnd}_{y,z}$, then one of them must begin at h^R_{i,j,t_1} and end at h^L_{i+1,j,z_2} and there must be other one that begins at h^R_{i,j,z_1} and ends at h^L_{i+1,j,t_2} for some $1 \le z_1, z_2 \le n^2$.

Thus, the space between h^R_{i,j,z_1} and $h^L_{i,j+1,z_2}$ would be covered twice and is longer than ϵ. The case when one of them is $\mathsf{horBeg}_{y,z}$ or $\mathsf{horEnd}_{y,z}$ is analogous. Note that they cannot be both $\mathsf{horBeg}_{y,z}$ or $\mathsf{horEnd}_{y,z}$.

By the proof of Lemma 5.3, the lower bound for weight of such a solution is $W_{\mathsf{hv}} + \epsilon$ which is more than $W_{\mathsf{hv}} + k^2\delta$.

Therefore h^R_{i,j,t_1} and h^L_{i+1,j,t_2} must be covered by one segment from HOR, namely $(h^R_{i,j,t_1}, h^L_{i+1,j,t_2})$. Hence $(h^R_{i,j,t_1}, h^L_{i+1,j,t_2})$ is a segment in HOR and $\mathsf{match}_h(t_1, t_2)$ is `true`. $\square$

Lemma 5.6. *If there exists a solution to instance $(\mathcal{C}, \mathcal{P}, w, 3k^2 + 2k)$ with weight at most $W_{\mathsf{hv}} + k^2\delta$, then there exists a solution to the* Grid Tiling *instance (n, k, f).*

Proof. Take $\mathsf{diagonal}$ function from Lemma 5.4.

To define the x function for every $1 \le i \le k$ set $x(i) := x_i$ where $(x_i, a) = \mathsf{order}(v_{i,1})$. Similarly, to define the y function, for every $1 \le i \le k$ set $y(i) := y_i$ where $(b, y_i) = \mathsf{order}(h_{1,i})$

To prove that this is a correct solution to Grid Tiling, we need to prove that for every $1 \le i, j \le k$, $(x(i), y(j))$ is in the allowed tiles set $f(i, j)$.

Let us take any $1 \le i, j \le k$. By Lemma 5.5 and simple induction, we know that $\mathsf{match}_h(\mathsf{diagonal}(1, j), \mathsf{diagonal}(i, j))$ and $\mathsf{match}_v(\mathsf{diagonal}(i, 1), \mathsf{diagonal}(i, j))$ are `true`. Therefore $\mathsf{order}(\mathsf{diagonal}(i, j)) = (x(i), y(j))$. By Lemma 5.4 we know that $\mathsf{order}(\mathsf{diagonal}(i, j))$ is in $f(i, j)$. Therefore $(x(i), y(j))$ is in $f(i, j)$. $\square$

Bibliography

[Adamaszek et al., 2015] Adamaszek, A., Chalermsook, P., and Wiese, A. (2015). How to tame rectangles: Solving independent set and coloring of rectangles via shrinking. In Garg, N., Jansen, K., Rao, A., and Rolim, J. D. P., editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2015, August 24-26, 2015, Princeton, NJ, USA*, volume 40 of *LIPIcs*, pages 43–60. Schloss Dagstuhl - Leibniz-Zentrum für Informatik.

[Chan and Grant, 2014] Chan, T. M. and Grant, E. (2014). Exact algorithms and apx-hardness results for geometric packing and covering problems. *Comput. Geom.*, 47(2):112–124.

[Cygan et al., 2015] Cygan, M., Fomin, F. V., Kowalik, L., Lokshtanov, D., Marx, D., Pilipczuk, M., Pilipczuk, M., and Saurabh, S. (2015). *Parameterized Algorithms*. Springer.

[Dinur and Steurer, 2014] Dinur, I. and Steurer, D. (2014). Analytical approach to parallel repetition. In Shmoys, D. B., editor, *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 624–633. ACM.

[Etscheid et al., 2017] Etscheid, M., Kratsch, S., Mnich, M., and Röglin, H. (2017). Polynomial kernels for weighted problems. *J. Comput. Syst. Sci.*, 84:1–10.

[Gaspers et al., 2012] Gaspers, S., Kim, E. J., Ordyniak, S., Saurabh, S., and Szeider, S. (2012). Don't be strict in local search! In Hoffmann, J. and Selman, B., editors, *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence, July 22-26, 2012, Toronto, Ontario, Canada*. AAAI Press.

[Har-Peled and Lee, 2012] Har-Peled, S. and Lee, M. (2012). Weighted geometric set cover problems revisited. *J. Comput. Geom.*, 3(1):65–85.

[Håstad, 2001] Håstad, J. (2001). Some optimal inapproximability results. *J. ACM*, 48(4):798–859.

[Kim et al., 2021] Kim, E. J., Kratsch, S., Pilipczuk, M., and Wahlström, M. (2021). Directed flow-augmentation. *CoRR*, abs/2111.03450. To appear at STOC 2022.

[Marx, 2005] Marx, D. (2005). Efficient approximation schemes for geometric problems? In Brodal, G. S. and Leonardi, S., editors, *13th European Symposium on Algorithms 2005*, pages 448–459, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Marx, 2007] Marx, D. (2007). 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.

[Marx and Pilipczuk, 2022] Marx, D. and Pilipczuk, M. (2022). Optimal parameterized algorithms for planar facility location problems using voronoi diagrams. *ACM Trans. Algorithms*, 18(2):13:1–13:64.

[Marx and Schlotter, 2011] Marx, D. and Schlotter, I. (2011). Stable assignment with couples: Parameterized complexity and local search. *Discret. Optim.*, 8(1):25–40.

[Mustafa et al., 2014] Mustafa, N. H., Raman, R., and Ray, S. (2014). Settling the apx-hardness status for geometric set cover. In *55th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2014, Philadelphia, PA, USA, October 18-21, 2014*, pages 541–550. IEEE Computer Society.

[Mustafa and Ray, 2010] Mustafa, N. H. and Ray, S. (2010). Improved results on geometric hitting set problems. *Discret. Comput. Geom.*, 44(4):883–895.

[Pilipczuk et al., 2017] Pilipczuk, M., van Leeuwen, E. J., and Wiese, A. (2017). Approximation and parameterized algorithms for geometric independent set with shrinking. In Larsen, K. G., Bodlaender, H. L., and Raskin, J., editors, *42nd International Symposium on Mathematical Foundations of Computer Science, MFCS 2017, August 21-25, 2017 - Aalborg, Denmark*, volume 83 of *LIPIcs*, pages 42:1–42:13. Schloss Dagstuhl - Leibniz-Zentrum für Informatik.

[Pilipczuk et al., 2020] Pilipczuk, M., van Leeuwen, E. J., and Wiese, A. (2020). Quasi-polynomial time approximation schemes for packing and covering problems in planar graphs. *Algorithmica*, 82(6):1703–1739.

[Shachnai and Zehavi, 2017] Shachnai, H. and Zehavi, M. (2017). A multivariate framework for weighted FPT algorithms. *J. Comput. Syst. Sci.*, 89:157–189.

[Wiese, 2018] Wiese, A. (2018). Independent set of convex polygons: From n^ϵ to $1+\epsilon$ via shrinking. *Algorithmica*, 80(3):918–934.