

1

University of Warsaw

2

Faculty of Mathematics, Informatics and Mechanics

3

Katarzyna Kowalska

Student no. 371053

4

Approximation and Parameterized Algorithms for Segment Set Cover

5

6

Master's thesis

7

in COMPUTER SCIENCE

8

Supervisor:

dr Michał Pilipczuk

Institute of Informatics

9

June 2022

10 **Supervisor's statement**

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

13 Date

Supervisor's signature

14 **Author's statement**

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

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

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

21 Date

Author's signature

22

Abstract

23 The work presents a study of different geometric set cover problems. It mostly focuses on
24 segment set cover and its connection to the polygon set cover.

25

Keywords

26 set cover, geometric set cover, FPT, $W[1]$ -completeness, APX-completeness, PCP theorem,
27 NP-completeness

28

Thesis domain (Socrates-Erasmus subject area codes)

29 11.3 Informatyka

30

31

Subject classification

32 D. Software

33 D.127. Blabalgorithms

34 D.127.6. Numerical blabalalysis

35

Tytuł pracy w języku polskim

36 Algorytmy aproksymacyjne i parametryzowane dla problemu pokrywania punktów
37 odcinkami na płaszczyźnie

Contents

39	1. Introduction	5
40	2. Preliminaries	9
41	2.1. Geometric set cover	9
42	2.2. Parameterization	9
43	2.3. Approximation	9
44	2.4. δ -extension	10
45	2.5. Weighted setting	10
46	3. APX-hardness of geometric set cover problem	13
47	3.1. MAX-(3,3)-SAT and statement of reduction	13
48	3.2. Reduction	15
49	3.2.1. VARIABLE-gadget	15
50	3.2.2. OR-gadget	16
51	3.2.3. CLAUSE-gadget	18
52	3.2.4. Summary	20
53	3.2.5. Summary of construction	21
54	3.3. Construction lemmas and proof of Lemma 3.1	22
55	4. Fixed-parameter tractable algorithm for geometric set cover problem	25
56	4.1. Fixed-parameter tractable algorithm for unweighted segments	25
57	4.1.1. Axis-parallel segments	25
58	4.1.2. Segments in arbitrary directions	26
59	4.2. Fixed-parameter tractable algorithm for weighted segments with δ -extension	28
60	5. W[1]-hardness for axis-parallel weighted segments	33

Chapter 1

Introduction

Some problems in Computer Science are known to be NP-complete, meaning that assuming $P \neq NP$ 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 parametrization.

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 the literature to be inapproximable with factor $(1 - o(1)) \ln n$ assuming $P \neq NP$ (which is a stronger result than APX-hardness) proved in [Dinur and Steurer, 2014], and W[2]-hard with the natural parametrization. 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.

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] proves that Geometric Set Cover with unweighted axis-parallel rectangles is APX-hard; they also show similar hardness for Geometric Set Cover with many other standard geometric objects.

Parametrization 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, 2015] 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 parametrization of weighted problems. There does not seem to be a consensus of what parametrization 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 Subset

Sum or Knapsack are presented in [Etscheid et al., 2017]. Another work [Kim et al., 2021] considers weighted parametrization of Weighted Directed Feedback Set and Weighted *st*-Cut.

δ -extension In this paper, we focus on Geometric Set Cover with segments 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 an interesting result for the Maximum Weight Independent Set of Rectangles problem. In this problem one needs to find a set of non-overlapping weighted rectangles with maximum sum of weight possible. 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 on all sides. This problem is easier, because we compare this result 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 author in [Adamaszek et al., 2015] presents a PTAS for Maximum Weight Independent Set of Rectangles with δ -shrinking, which is later improved to EPTAS in [Pilipczuk et al., 2016] alongside presenting a new FPT result for this problem with natural parametrization. Later the similar δ -shrinking model was used in [Wiese, 2018] to present a PTAS for Maximum Weight Independent Set of Polygons with δ -shrinking.

Definition 1.2. 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 object $L^{+\delta} = \{(1 + \delta) \cdot (x - x_s, y - y_s) + (x_s, y_s) : (x, y) \in L\}$. That is, $L^{+\delta}$ is the image of L under homothety centred at S with scale $(1 + \delta)$.

Analogous to δ -shrinking, δ -extension provides a framework of 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 \mathcal{R} covers all the points in the universe \mathcal{C} after δ -extension, ie. by set $\mathcal{R}^{+\delta}$. The same concept could be used for Geometric Set Hitting problems.

For more elaborate 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 EPTAS [Har-Peled and Lee, 2009]. δ -extension model presented there is well-defined only for fat polygons. It extends the object by all the points that have distance to the closest point in the object P no larger than $\delta \cdot \text{rad}(P)$, where $\text{rad}(P)$ is a 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 this problem. Polygons extended by δ -extension defined in 1.2 covers a superset of set of points that object extended by δ -extension defined in [Har-Peled and Lee, 2009]. Since our definition is more permissive for any polygon, then EPTAS from [Har-Peled and Lee, 2009] also works for polygons extended by our δ -extension.

Our contribution

In this paper we make the following contributions.

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

Theorem 1.1. (*Axis-parallel segment set cover with $\frac{1}{2}$ -extension is APX-hard*). *Unweighted geometric set cover with axis-parallel segments in the 2D plane (even with $\frac{1}{2}$ -extension) is APX-hard. That is, assuming $P \neq NP$, there does not exist a PTAS for this problem.*

136 This expands the previous result of [Chan and Grant, 2014] that Geometric Set Cover with
 137 unweighted axis-parallel rectangles being APX-hard. This also proves that the assumption in
 138 [Har-Peled and Lee, 2009] for EPTAS about polygons being fat is necessary, because cover
 139 with arbitrary polygons with δ -extension is APX-hard.

140 We also provide two FPT algorithms for parameterized Geometric Set Cover with un-
 141 weighted segments (Theorem 1.2) and weighted segments relaxed with δ -extension (Theo-
 142 rem 1.3).

143 **Theorem 1.2. (FPT for segment cover).** *There exists an algorithm that given a fam-
 144 ily \mathcal{P} of segments (in any direction), a set of points \mathcal{C} and a parameter k , runs in time
 145 $k^{\mathcal{O}(k)}(|\mathcal{C}| \cdot |\mathcal{P}|)^2$, and outputs a solution $\mathcal{R} \subseteq \mathcal{P}$ such that $|\mathcal{R}| \leq k$ and \mathcal{R} covers all points
 146 in \mathcal{C} , or determines that such a set \mathcal{R} does not exist.*

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

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

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

157 On the other hand, we prove that Geometric Set Cover with weighted axis-parallel seg-
 158 ments is W[1]-hard (Theorem 1.4) and assuming ETH there does not exist algorithm for this
 159 problem that runs in time $f(k)(|\mathcal{C}| + |\mathcal{P}|)^{o(\sqrt{k})}$. See Figure 1.1 for a summary of parameterized
 160 results for the weighted setting.

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

167 Please see Section 2.1 for exact definitions of axis-parallel and right-diagonal segments.

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

171 Permissive FPT is a relaxed FPT problem, where we need to find solution of *any* size in
 172 FPT-time, but we compare it to the optimum solution of size at most k . Idea for permissive
 173 FPT in local search was presented in [Marx and Schlotter, 2011], [Gaspers et al., 2012].

174 Theorem 1.4 can be improved to show that a permissive FPT algorithm does not exist.
 175 This is formulated precisely in 5.2.

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

Figure 1.1: Our results for Geometric Set Cover problem with weighted segments parameterized by the size of solution. Results marked with * trivially follow from the results presented in this thesis.

176 **Future work.** There are two aforementioned problems that relate to Theorem 1.4 and were
177 not solved in this thesis. We have not presented W[1]-hardness proof of Geometric Set Cover
178 problem with axis-parallel weighted segments, but this construction maybe might be improved
179 to utilize segments in 2 directions instead of 3 directions. The other question is what is the
180 tight bound for this problem. The simple algorithm solving this problem is running in time
181 $\mathcal{O}(f(k)(|\mathcal{C}| + |\mathcal{P}|)^k)$.

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 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 $f : \mathcal{P} \rightarrow \mathbb{R}^+$ and we would like to find a solution \mathcal{R} that minimizes $\sum_{R \in \mathcal{R}} f(R)$.

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

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

2.2. Parameterization

In the parameterized setting of the 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. An **FPT** algorithm is...

Definition 2.4. Exponential Time Hypothesis is...

Theorem 2.1. *Problem is $W[1]$ -hard if assuming ETH there is no...*

2.3. Approximation

Let us recall some definitions related to optimization problems.

Definition 2.5. A **polynomial-time approximation scheme (PTAS)** for a minimization problem Π is a family of algorithms \mathcal{A}_ϵ for every $\epsilon > 0$ such that \mathcal{A}_ϵ 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 $(1 + \epsilon)\text{opt}(I)$, where $\text{opt}(I)$ is the weight of an optimal solution to I .

Definition 2.6. 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 Defintion 1.2.

The geometric set cover problem 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 2.7. 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. $|\mathcal{R}| \leq |\text{opt}(I)|$.

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

Definition 2.8. 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 strongly 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. $(1 + \epsilon)|\mathcal{R}| \leq |\text{opt}(I)|$.

2.5. Weighted setting

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

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

TODO: Restate below for FPT

250 **Definition 2.9.** The **weighted geometric set cover problem with δ -extension** is the
251 problem where for an input instance $I = (\mathcal{P}, \mathcal{C}, f)$ of weighted geometric set cover, the task is
252 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
253 larger than the optimal solution to the problem without extension, i.e. $\sum_{R \in \mathcal{R}} f(R) \leq |\text{opt}(I)|$.

Chapter 3

APX-hardness of geometric set cover problem

In this section we analyze whether there exists a PTAS for geometric set cover for rectangles. We show that we can restrict this problem to a very simple setting: segments parallel to axes and allow $(1/2)$ -extension, and the problem is still APX-hard. Note that segments are just degenerated rectangles with one side being very narrow.

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

Theorem 1.1. (*Axis-parallel segment set cover with $\frac{1}{2}$ -extension is APX-hard*). Unweighted geometric set cover with axis-parallel segments in the 2D plane (even with $\frac{1}{2}$ -extension) is APX-hard. That is, assuming $P \neq NP$, there does not exist a PTAS for this problem.

Theorem 1.1 implies the following.

Corollary 3.1. (*rectangle set cover is APX-hard*). Unweighted geometric set cover with axis-parallel rectangles (even with $1/2$ -extension) is APX-hard.

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 and statement of reduction

Definition 3.1. MAX-3SAT is the following maximization problem. We are given a 3-CNF formula, and need to find an 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 (α -satisfiable MAX-3SAT formula). 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 $(7/8 + \epsilon)$ -satisfiable (3,3)-SAT formulas.

Given an instance I of MAX-(3,3)-SAT, we construct an instance J of axis-parallel segment set cover problem such that for a sufficiently small $\epsilon > 0$, a polynomial time $(1 + \epsilon)$ -approximation algorithm for J would be able to distinguish whether an instance I of MAX-(3,3)-SAT is fully satisfiable or is at most $(7/8 + \epsilon)$ -satisfiable. However, according to Theorem 3.1 the latter problem is NP-hard. This would imply $P = NP$, contradicting the assumption.

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 $opt(S)$, we can construct an instance I of geometric set cover with axis-parallel segments in 2D such that:*

(1) *For every solution X of instance I , there exists a solution to S that satisfies at least $15n - |X|$ clauses.*

(2) *For every solution to instance S that satisfies w clauses, there exists a solution to I of size $15n - w$.*

(3) *Every solution with $1/2$ -extension of I is also a solution to the original instance I .*

Therefore, the optimum size of a solution to I is $opt(I) = 15n - opt(S)$.

TODO: Do the summary which dot corresponds to which lemma to have better structure
We prove Lemma 3.1 in subsequent sections, but meanwhile let us prove Theorem 1.1 using Lemma 3.1 and Theorem 3.1.

Proof of Theorem 1.1. Consider any $0 < \epsilon < 1/(15 \cdot 8)$.

Let us assume that there exists a polynomial-time $(1 + \epsilon)$ -approximation algorithm for unweighted geometric set cover with axis-parallel segments in 2D with $(1/2)$ -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 geometric set cover I using Lemma 3.1. We now use the $(1 + \epsilon)$ -approximation algorithm for geometric set cover on I . Denote the size of the solution returned by this algorithm as $approx(I)$. We prove that if in S one can satisfy at most $(\frac{7}{8} + \epsilon)n$ clauses, then $approx(I) \geq 15n - (\frac{7}{8} + \epsilon)n$ and if S is satisfiable, then $approx(I) < 15n - (\frac{7}{8} + \epsilon)n$.

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

$$opt(S) = n.$$

From Lemma 3.1 we have:

$$opt(I) = 14n.$$

Therefore,

$$\begin{aligned} approx(I) &\leq (1 + \epsilon)opt(I) = 14n(1 + \epsilon) = 14n + 14\epsilon \cdot n = \\ &= 14n + (15\epsilon - \epsilon)n < 14n + \left(\frac{1}{8} - \epsilon\right)n = 15n - \left(\frac{7}{8} + \epsilon\right)n. \end{aligned}$$

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

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

From Lemma 3.1 we have:

$$opt(I) \geq 15n - \left(\frac{7}{8} + \epsilon\right)n$$

313 Since a solution to I with $\frac{1}{2}$ -extension is also a solution without any extension, by Lemma
 314 3.1 (3), we have:

$$approx(I) \geq opt(I) = 15n - \left(\frac{7}{8} + \epsilon\right)n$$

315 Therefore, by using the assumed $(1 + \epsilon)$ -approximation algorithm, it is possible to distin-
 316 guish the case when S is satisfiable: from the case when it is at most $(\frac{7}{8} + \epsilon)n$ satisfiable,
 317 it suffices to compare $approx(I)$ with $15n - (\frac{7}{8} + \epsilon)n$. Hence, the assumed approximation
 318 algorithm cannot exist, unless $P = NP$. \square

319 3.2. Reduction

320 We proceed to the proof of Lemma 3.1. That is, we show a reduction from the MAX-(3,3)-
 321 SAT problem to geometric set cover with segments parallel to axis. Moreover, the obtained
 322 instance of geometric set cover will be robust to 1/2-extension (have the same optimal solution
 323 after 1/2-extension).

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

327 3.2.1. VARIABLE-gadget

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

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

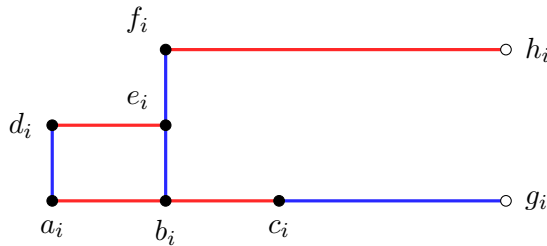


Figure 3.1: **VARIABLE-gadget**. We denote the set of points marked with black circles as pointsVariable_i , and they need to be covered (are part of the set \mathcal{C}). Note that some of the points are not marked as black dots and exists only to name segments for further reference. We denote the set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

332

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

Let us define:

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

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

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

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

334 **Segments.** Let us define:

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

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

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

335 We also name two of these segment for future reference: $\text{xTrueSegment}_i := (c_i, g_i)$,
336 $\text{xFalseSegment}_i := (f_i, h_i)$.

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

339 *Proof.* We can use either set $\text{chooseVariable}_i^{\text{true}}$ or $\text{chooseVariable}_i^{\text{false}}$. □

340 **Lemma 3.3.** *For any $1 \leq i \leq n$, points in pointsVariable_i can not be covered with fewer than*
341 *3 segments from $\text{segmentsVariable}_i$.*

342 *Proof.* No segment of $\text{segmentsVariable}_i$ covers more than one point from $\{d_i, f_i, c_i\}$, therefore
343 pointsVariable_i can not be covered with fewer than 3 segments. □

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

346 *Proof.* No segment from $\text{segmentsVariable}_i$ covers more than one point from $\{a_i, e_i\}$, therefore
347 $\text{pointsVariable}_i - \{c_i, f_i, g_i, h_i\}$ can not be covered with fewer than 2 segments. □

348 3.2.2. OR-gadget

349 OR-gadget connects input and output segments (see Figure 3.2) in a way that is supposed to
350 simulate a binary *or* function.

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

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

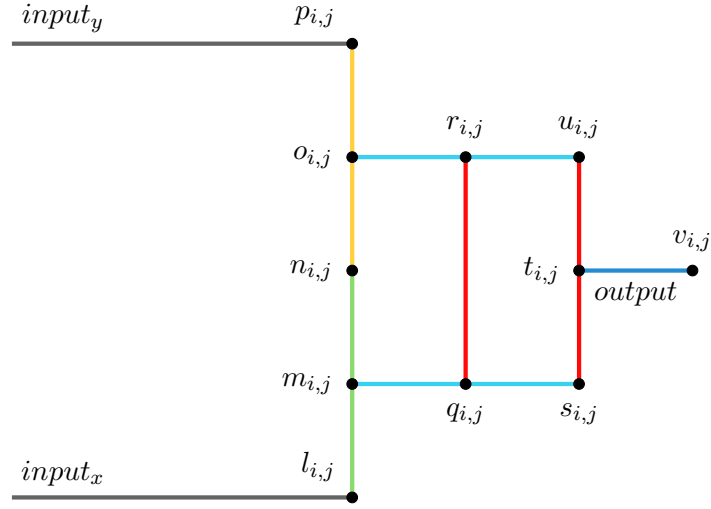


Figure 3.2: **OR-gadget**. Segments from $\text{chooseOr}_{i,j}^{false}$ are **red**, segments from $\text{chooseOr}_{i,j}^{true}$ are blue (both **light blue** and **dark blue**), segments from $\text{orMoveVariable}_{i,j}$ are **green** and **yellow**. **Dark blue** segment is the *output* segment.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

359 **Points.**

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

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

361 For integers i, j , define $\{l_{i,j}, m_{i,j} \dots v_{i,j}\}$ as $\{l_0, m_0 \dots v_0\}$ shifted by $vec_{i,j}$, i.e. $l_{i,j} =$
 362 $l_0 + vec_{i,j}$ etc.

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

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

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

365 **Segments.** We define set of segments in several parts:

$$\begin{aligned}
 \text{chooseOr}_{i,j}^{false} &:= \{(q_{i,j}, r_{i,j}), (s_{i,j}, u_{i,j})\}, \\
 \text{chooseOr}_{i,j}^{true} &:= \{(m_{i,j}, s_{i,j}), (o_{i,j}, u_{i,j}), (t_{i,j}, v_{i,j})\},
 \end{aligned}$$

$$\text{orMoveVariable}_{i,j} := \{(l_{i,j}, n_{i,j}), (n_{i,j}, p_{i,j})\}.$$

366 Finally all segments in OR-gadget are defined as:

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

367 **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 $\text{pointsOr}_{i,j} - \{x\} \cup$
 368 $\{v_{i,j}\}$ can be covered with 4 segments from $\text{segmentsOr}_{i,j}$.

369 *Proof.* We can do that using one segment from $\text{orMoveVariable}_{i,j}$, the one that does not cover
 370 x , and all segments from $\text{chooseOr}_{i,j}^{\text{true}}$. \square

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

373 *Proof.* We can do that using segments from $\text{orMoveVariable}_{i,j} \cup \text{chooseOr}_{i,j}^{\text{false}}$. \square

374 3.2.3. CLAUSE-gadget

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

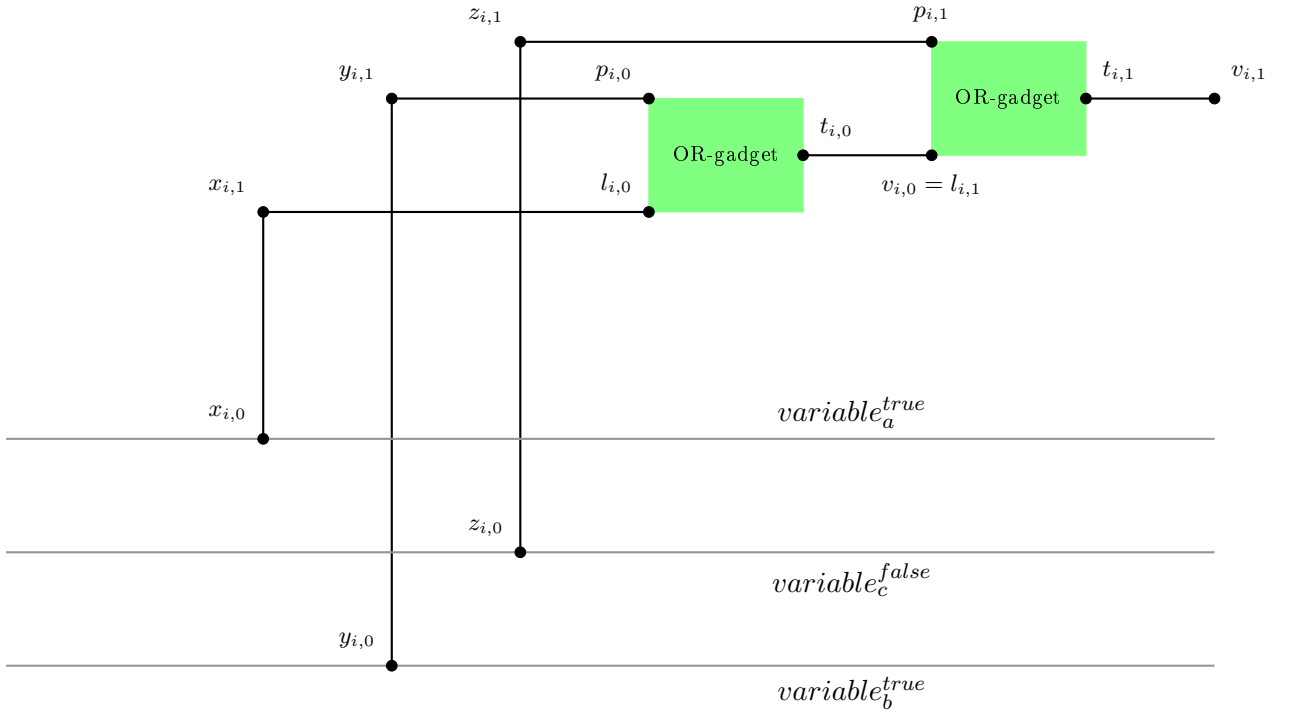


Figure 3.3: **CLAUSE-gadget for a clause $a \vee b \vee \neg c$.** Every green rectangle is an OR-gadget. y -coordinates of $x_{i,0}, y_{i,0}$ and $z_{i,0}$ depend on the variables in the i -th clause. Grey segments corresponds to the values of variables satisfying the i -th clause.

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

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

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

We are now ready to define set of points:

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

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

Note that these two points are equal: $v_{i,0} = l_{i,1}$. This translates to the fact, that output of the one OR-gadget is an input to the other OR-gadget to create *or* of 3 segments.

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

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

The CLAUSE-gadgets consist of two OR-gadgets. Ideally, we would place the i -th CLAUSE-gadget close to the $\text{xTrueSegment}_{j_1}$ or $\text{xFalseSegment}_{j_1}$ 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 $\text{xTrueSegment}_{j_2}$ or $\text{xFalseSegment}_{j_2}$ segments corresponding to the other literals.

Instead, we use simple auxiliary gadgets to *transfer* whether the segment is in a solution, i.e. segments $(x_{i,0}, x_{i,1}), (y_{i,0}, y_{i,1}), (z_{i,0}, z_{i,1})$ in this gadget. Each 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 $1 \leq i \leq n$ and $a \in \{x_{i,0}, y_{i,0}, z_{i,0}\}$, there is a set $\text{solClause}_i^{\text{true},a} \subseteq \text{segmentsClause}_i$ with $|\text{solClause}_i^{\text{true},a}| = 11$ that covers all points in $\text{pointsClause}_i - \{a\}$.*

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 $\text{chooseOr}_{i,0}^{\text{true}} \cup \text{chooseOr}_{i,1}^{\text{true}}$ 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 $a = z_{i,0}$: Using Lemma 3.6 and Lemma 3.5 with $x = p_{i,1}$, we obtain 8 segments in $\text{chooseOr}_{i,0}^{\text{false}} \cup \text{chooseOr}_{i,1}^{\text{true}}$ 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 $1 \leq i \leq n$ there is a set $\text{solClause}_i^{\text{false}} \subseteq \text{segmentsClause}_i$ with $|\text{solClause}_i^{\text{false}}| = 12$ that covers all points in pointsClause_i .*

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

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

- 422 (1) *points in pointsClause_i can not be covered using any subset of segments from segmentsClause_i*
 423 *of size smaller than 12;*
- 424 (2) *points in $\text{pointsClause}_i - \{x_{i,0}, y_{i,0}, z_{i,0}\}$ can not be covered using any subset of segments*
 425 *from segmentsClause_i of size smaller than 11.*

Proof of (1). No segment in segmentsClause_i covers more than 1 point from

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

426 Therefore we need to use at least 12 segments. \square

427 *Proof of (2).* We can define disjoint sets X, Y, Z such that $X \cup Y \cup Z \subseteq \text{pointsClause}_i -$
 428 $\{x_{i,0}, y_{i,0}, z_{i,0}\}$ such that there are no segments in segmentsClause_i covering points from dif-
 429 ferent sets. And we prove a lower bound for each of these sets. First, let:

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

430 No two points in X can be covered with one segment of segmentsClause_i , so it must be
 431 covered with 3 different segments. Next we define other sets:

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

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

432 For both Y and Z we can check all of the subsets of 3 segments of segmentsClause_i to
 433 conclude that none of them cover the considered, so both Y and Z have to be covered with
 434 disjoint sets of 4 segments each.

435 Therefore, $\text{pointsClause}_i - \{x_{i,0}, y_{i,0}, z_{i,0}\}$ must be covered with at least $3 + 4 + 4 = 11$
 436 segments from segmentsClause_i . \square

437 3.2.4. Summary

438 Add some smart lemmas that sets will be exclusive to each other.

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

441 *Proof.* We can just check every segment. Most of the segments s are collinear only with points
 442 that lie on s , so trivially $s^{+\frac{1}{2}}$ cannot cover more points than s does.

443 Within VARIABLE-gadget for any $1 \leq i \leq n$ after $\frac{1}{2}$ -extension: (c_i, g_i) does not cover b_i .

444 Within OR-gadget some of the segments are collinear and share one point; specifically, for
 445 any $1 \leq i \leq n$ and $j \in \{0, 1\}$, after $\frac{1}{2}$ -extension:

- 446 • $(l_{i,j}, n_{i,j})$ does not cover $o_{i,j}$,
- 447 • $(n_{i,j}, p_{i,j})$ does not cover $m_{i,j}$,
- 448 • $(t_{i,j}, v_{i,j})$ does not cover $n_{i,j}$.

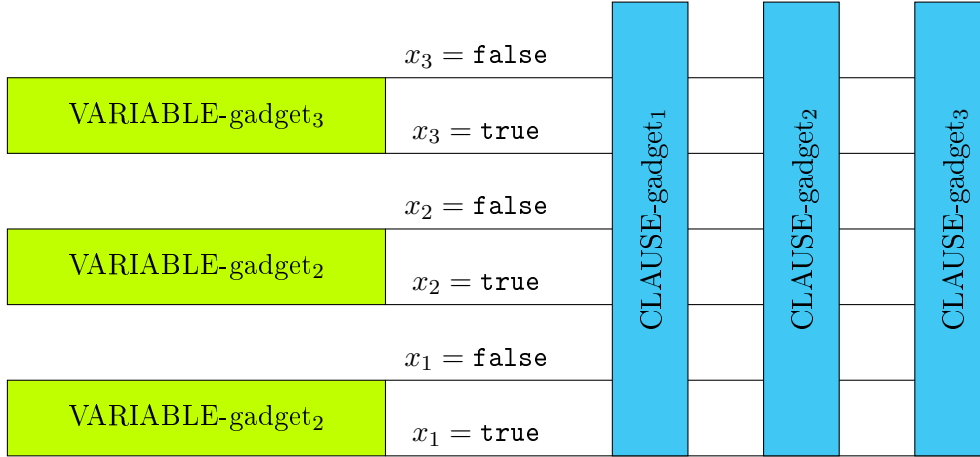


Figure 3.4: **Schema of the whole construction.**

General layout of VARIABLE-gadgets and CLAUSE-gadgets and how they interact with each other.

449 Within CLAUSE-gadget, for any $1 \leq i \leq n$ after $\frac{1}{2}$ -extension:

- 450 • $(o_{i,0}, u_{i,0})$ does not cover $m_{i,1}$,
- 451 • $(m_{i,1}, s_{i,1})$ does not cover $u_{i,0}$,
- 452 • $(y_{i,1}, p_{i,0})$ does not cover $n_{i,1}$.

453 For two consecutive VARIABLE-gadgets, for any $1 \leq i < n$ after $\frac{1}{2}$ -extension: (b_i, f_i) does
 454 not cover b_{i+1} (nor f_{i-1} for $i > 1$). Similarly (a_i, d_i) does not cover a_{i+1} (nor d_{i-1} for $i > 1$),
 455 because this segment is shorter than the previous one and a_i and b_i share y-coordinate.

456 For two consecutive CLAUSE-gadgets, segments from one do not cover anything from the
 457 other, as the gadgets have width 9 and every leftmost x-coordinate is divisible by 20. Hence
 458 two different gadgets do not interact with each other after $\frac{1}{2}$ -extension.

459 Next we need to check whether VARIABLE-gadget's segments do not cover any points
 460 $x_{i,0}, y_{i,0}$ or $z_{i,0}$ from CLAUSE-gadget. For any $1 \leq i \leq n$ and $1 \leq j \leq n$, all points $x_{j,0}, y_{j,0}$
 461 and $z_{j,0}$ have x-coordinate strictly positive. Segment (a_i, c_i) have length $2L$ and c_i has x-
 462 coordinate equal to $-L$, so after $\frac{1}{2}$ -extension this segment does not cover any points with a
 463 positive x-coordinate.

464 □

465 3.2.5. Summary of construction

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

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

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

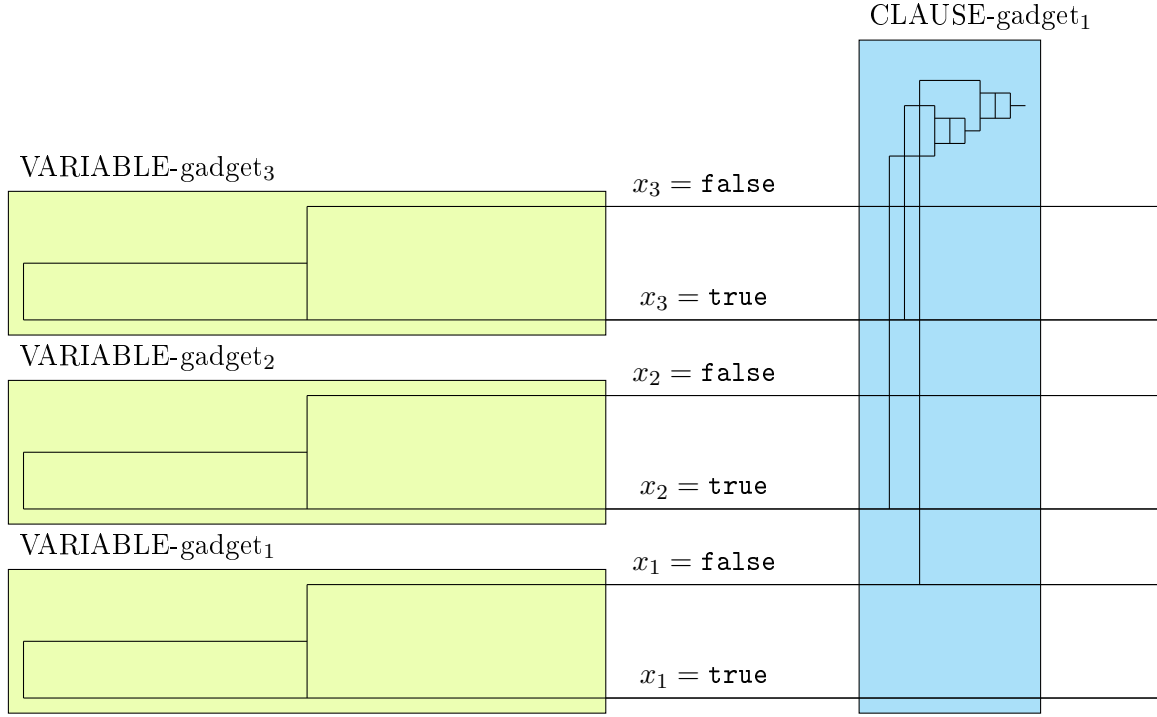


Figure 3.5: **Schema of the whole construction.**

General layout of VARIABLE-gadgets and CLAUSE-gadgets and how they interact with each other.

3.3. Construction lemmas and proof of Lemma 3.1

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 geometric set cover as described in Section 3.2 for the instance S of MAX-(3,3)-SAT.

Lemma 3.11. *Instance $(\mathcal{C}, \mathcal{P})$ of geometric set cover admits a solution of size $15n - k$.*

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

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 **true** in it as x_i and point corresponding to x_i in $\mathbf{pointsClause}_i$ as a . Points in $\mathbf{pointsClause}_i$ are covered with set $\mathbf{solClause}_i^{\mathbf{true}, a}$ described in Lemma 3.7. For every clause that is not satisfied, say c_j , points in $\mathbf{pointsClause}_j$ are covered with set $\mathbf{solClause}_j^{\mathbf{false}}$ 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:

$$\begin{aligned}
R_i &:= \begin{cases} \text{chooseVariable}_i^{\text{true}} & \text{if } \phi(x_i) = \text{true} \\ \text{chooseVariable}_i^{\text{false}} & \text{if } \phi(x_i) = \text{false} \end{cases} \\
C_i &:= \begin{cases} \text{solClause}_i^{\text{true},a} & \text{if } c_i \text{ satisfied by literal corresponding to point } a \\ \text{solClause}_i^{\text{false}} & \text{if } c_i \text{ not satisfied} \end{cases} \\
\mathcal{R} &:= \bigcup_{i=1}^n \{R_i \cup C_i : 1 \leq i \leq n\}.
\end{aligned}$$

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

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

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

486 **Lemma 3.12.** *Suppose we have a solution \mathcal{R} of the instance $(\mathcal{C}, \mathcal{P})$ of geometric set cover.*
487 *Then there exists a solution \mathcal{R}' , such that $|\mathcal{R}'| \leq |\mathcal{R}|$, and \mathcal{R}' contains at most one of the*
488 *segments xTrueSegment_i and xFalseSegment_i from each VARIABLE-gadget.*

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

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

498 Now, there exists at most one CLAUSE-gadget which needs adjustment to make \mathcal{R}' valid;
499 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
500 might be not covered, say $y_{j,0}$. We amend the solution by adding $(y_{j,0}, y_{j,1})$ to \mathcal{R}' .

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

503 **Lemma 3.13.** *Suppose we have a solution \mathcal{R} of the instance $(\mathcal{C}, \mathcal{P})$ of geometric set cover*
504 *that is of size w . Then there exists a solution to S that satisfies at least $15n - w$ clauses.*

505 *Proof.* Let the clauses in S be $c_1, c_2 \dots c_n$ and the variables be $x_1, x_2 \dots x_n$. Given a solution
506 \mathcal{R} of the instance $(\mathcal{C}, \mathcal{P})$ of geometric set cover, we use Lemma 3.12 to modify \mathcal{R} such that
507 for any i it contains at most one of xTrueSegment_i and xFalseSegment_i ; this may decrease the
508 cost of \mathcal{R} , but that does not matter in the subsequent construction. To simplify notation, in
509 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:

$$\phi : \{x_1, x_2 \dots x_n\} \rightarrow \{\text{true}, \text{false}\}$$

510 that satisfies at least $15n - w$ clauses in S .

511 **Definition of ϕ .** Recall that due to Lemma 3.12, \mathcal{R} contains at most one of xTrueSegment_i
 512 and xFalseSegment_i .

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

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

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

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

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

519 Let us denote a as the amount of such clauses c_i for which none of the points $x_{i,0}, y_{i,0}, z_{i,0}$
 520 in pointsClause_i were covered by segments from $\mathcal{R} \cap \text{segmentsVariable}_j$ for any $1 \leq j \leq n$.

521 Consider a clause c_i for which at least one of the points $x_{i,0}, y_{i,0}, z_{i,0}$ in pointsClause_i were
 522 covered by segments from $\mathcal{R} \cap \text{segmentsVariable}_j$ for some $1 \leq j \leq n$, then denote this
 523 point as t and say it corresponds to literal q and variable x_j . Point t can be only covered in
 524 $\text{segmentsVariable}_j$ by a corresponding segment xTrueSegment_j or xFalseSegment_j (depending
 525 on whether the literal q is negated or not). From the definition of ϕ and the fact that one of
 526 this segment is in \mathcal{R} , we know that $\phi(j)$ has the value that evaluates w to be true. Therefore,
 527 clause c_i is satisfied.

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

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

$$w \geq 3n + 11(n - a) + 12a = 3n + 11n + a = 14n + a,$$

hence

$$15n - w \leq 15n - 14n - a = n - a.$$

531 Therefore ϕ satisfies at least $15n - w$ clauses of S . □

532 We are ready to conclude the proof of Lemma 3.1.

Proof of Lemma 3.1. By Lemma 3.11, we know that there exists a solution to $(\mathcal{C}, \mathcal{P})$ of size
 $15n - k$, so:

$$\text{opt}((\mathcal{C}, \mathcal{P})) \leq 15n - k.$$

Since the optimum solution to S satisfies k clauses, then according to Lemma 3.13:

$$\text{opt}((\mathcal{C}, \mathcal{P})) \geq 15n - k.$$

533 Therefore, the solution given by Lemma 3.11 of size $15n - k$ is an optimum solution to the
 534 instance $(\mathcal{C}, \mathcal{P})$. □

Chapter 4

Fixed-parameter tractable algorithm for geometric set cover problem

In this chapter we show fixed-parameter tractable algorithms for the geometric set cover problem in two different settings. Section 4.1 shows a fixed-parameter tractable algorithm for geometric set cover with unweighted segments. The remainder of the chapter presents a fixed-parameter tractable algorithm for geometric set cover with weighted segments 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 segments

In this section we consider fixed-parameter tractable algorithms for unweighted geometric set cover with segments. The setting where segments are required to be axis-parallel (or limited to a constant number of directions) has an FPT algorithm already present in literature in the Parametrized Algorithms book [Cygan et al., 2015]. We present an FPT algorithm for geometric set cover with unweighted segments, where segments are in arbitrary directions.

4.1.1. Axis-parallel segments

Theorem 4.1. (*FPT for segment 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.*

We present here a simple algorithm from [Cygan et al., 2015] for completeness.

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 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 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 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, \dots, x_t in the order in which they appear on L . Our proposed set is defined as:

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

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 us define a set \mathcal{R}_L , which is defined as 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 $\{x_1, x_2, \dots, x_k\}$, therefore at least one of these points must be covered by segments from \mathcal{R}_L . We take the leftmost point from $\{x_1, x_2, \dots, x_k\}$ 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 segments with δ -extension

In this section we consider the geometric set cover problem for weighted segments 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 geometric set cover problem for weighted segments (without extension) is W[1]-hard, which means there does not exist any FPT algorithm parameterized by solution size for it, assuming $\text{FPT} \neq \text{W}[1]$.

Theorem 1.3. (*FPT for weighted segment 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 $\delta > 0$, such that it 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} .

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 .

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

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

683 **Base case for $k = 1$.** It is sufficient that A consists of the extreme points of C .

684 If they are covered with one segment, it must be a segment that includes the extreme
685 points from C , so it covers the whole set C .

686 There are at most 2 extreme points in C and $2 < 2 + \frac{2}{\delta}$.

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

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

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

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

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

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

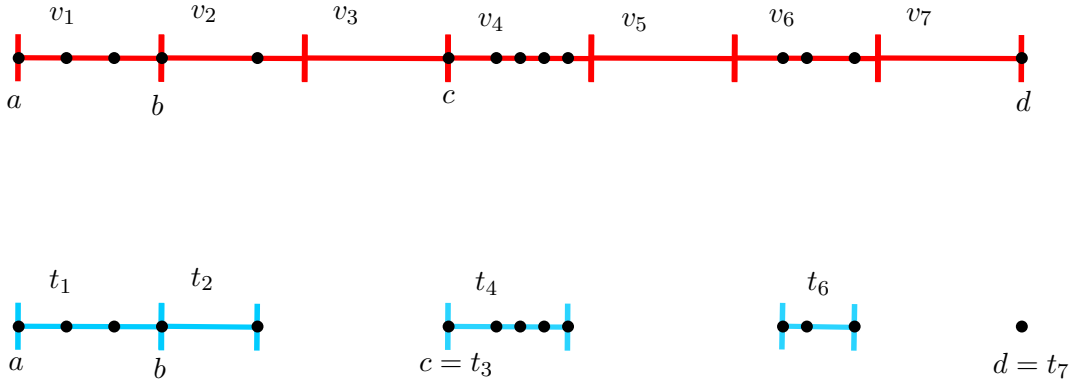


Figure 4.1: **Example of segments v_i and t_i .**

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 .

697 We use the inductive hypothesis to choose (k, δ) -dense sets A_i for sets C_i . Note that if
698 $|C_i| \leq 1$, then $A_i = C_i$ and it is still a (k, δ) -dense set for C_i .

699 Then we define $A = \bigcup_{i=1}^M A_i$. Thus A includes the extreme points of C , because they are
700 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:

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

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

703 For every segment t_i , if there exists a segment x in \mathcal{R} that is disjoint with t_i , then we have
704 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 ,
705 $(\mathcal{R} - \{x\})^{+\delta}$ covers C_i . So $\mathcal{R}^{+\delta}$ covers C_i as well.

706 If there exists a segment t_i for which a segment x as defined above does not exist, then
707 all $k + 1$ segments that cover A_i intersect t_i . An example of such segments is depicted in
708 Figure 4.2. Let us consider any such t_i . By the inductive hypothesis, the endpoints of s
709 are in A_1 and A_M respectively, so \mathcal{R} must cover them. For each endpoint of s , there exists
710 a segment that contains this endpoint and intersects t_i . Let us call these two segments y
711 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
712 $\max(|y|, |z|) \geq |s|(1 - \frac{\delta}{\delta + 2})/2 = \frac{|s|}{\delta + 2}$.

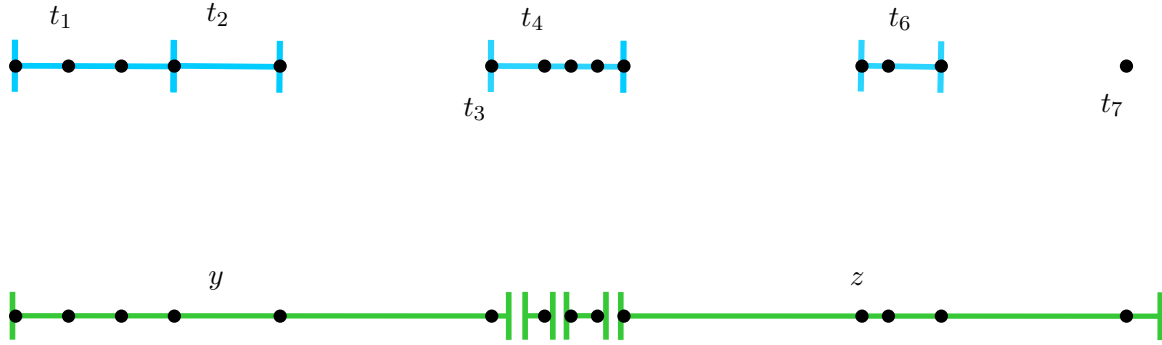


Figure 4.2: **Example of all $k + 1$ segments intersecting one segment t_i .**

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:

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

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

715 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:

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

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 $|\mathcal{R}| \leq k$. □

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 geometric set cover instance. □

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 $|\mathcal{R}_1| \leq k$, weight of \mathcal{R}_1 is not greater than 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 points in \mathcal{C}' , $\mathcal{R}_2^{+\delta}$ covers points in original instance \mathcal{C} .

If we found such sets $(\mathcal{C}', \mathcal{P}')$, using *Property 1* we know that optimum solution of size at most k to $(\mathcal{C}', \mathcal{P}')$ has no greater weight than 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 $\mathcal{O}((k^2 + k(2 + \frac{2}{\delta})^k)^2)$.

Proof of Property 2. First, 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 separated 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 $|\mathcal{R}_1| \leq k$ and the weight of \mathcal{R}_1 is not greater than the weight of \mathcal{R} .

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 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 for axis-parallel weighted segments

In this chapter we consider the geometric set cover problem with axis-parallel or right-diagonal weighted 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 below construction can be improved to only utilize the axis-parallel segments.

Theorem 1.4. *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 $w : \mathcal{P} \rightarrow \mathbb{R}^+$ 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.*

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 $\text{Pow}(A)$, as the set of all subsets of A , i.e. $\text{Pow}(A) = \{B : B \subseteq A\}$.

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

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.

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 .*

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

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**).

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 geometric 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 geometric 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 the geometric set cover). Therefore, geometric set cover with setting described in Theorem 1.4 is at least as hard as the grid tiling problem.

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 $f : \{1, \dots, k\} \times \{1, \dots, k\} \rightarrow \text{Pow}(\{1, \dots, n\} \times \{1, \dots, n\})$.

Let us also define constants:

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

which are going to be used when defining the weight of the constructed instance of geometric set cover with weighted segments.

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 geometric set cover with weighted segments 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 $W_{\text{hv}} + k^2\delta$;*

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

845 First, let us prove Theorem 1.4 using Lemma 5.1.

846 *Proof of Theorem 1.4.* Let us take any instance (n, l, f) of the grid tiling problem. We prove
 847 the theorem by contradiction, therefore we assume that geometric set cover with weighted
 848 segments parameterized by solution size k admits a $g(k) \cdot n^{o(\sqrt{k})}$ -time algorithm for some
 849 computable function g .

850 Using Lemma 5.1 let us construct an instance I for (n, l, f) . Let us assume that the
 851 optimum solution of size at most k to the instance I has weight u . Using (2) we know that if
 852 $u \leq W_{\text{hv}} + k^2\delta$, then the answer to (n, l, f) is YES. If $u > W_{\text{hv}} + k^2\delta$, then using (1) we know
 853 that the answer to (n, l, f) must be NO.

854 Therefore if we could find the solution in time $g(k) \cdot n^{o(\sqrt{k})}$, then we could solve the grid
 855 tiling problem in time $g(l) \cdot n^{o(l)}$ by constructing an instance of the set cover with weighted
 856 segments, solving it for parameter $k = 3l^2 + 2l$ in time $n^{o(\sqrt{3l^2+2l})}$ and then answering based
 857 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
 858 algorithm contradicts Theorem 5.1. Hence such an algorithm can not exist. \square

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

861 In the proof of Lemma 5.6 we do not use the assumption that the solution is bounded
 862 by the size, which the problem is parametrized by, $3k^2 + 2k$. If we had a permissive FPT
 863 algorithm that finds a solution of any size that still has weight no more than $W_{\text{hv}} + k^2\delta$, then
 864 we still would have a contradiction with grid tiling being W[1]-hard in proof of Theorem 1.4.
 865 Thus this reduction proves that the problem is not only W[1]-hard, but assuming ETH there
 866 also does not exist permissive FPT algorithm for this problem. Formally we state this in the
 867 Theorem 5.2.

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

873 Assuming ETH, there is no algorithm for this problem with running time $f(k) \cdot (|\mathcal{C}| +$
 874 $|\mathcal{P}|)^{o(\sqrt{k})}$ for any computable function f . Moreover, this holds even if all segments in \mathcal{P} are
 875 axis-parallel or right-diagonal.

876 **Construction.** We construct an instance $(\mathcal{C}, \mathcal{P}, w, 3k^2 + 2k)$ of geometric set cover as follows.

877 First, let us choose any bijection $\text{order} : \{1, \dots, n^2\} \rightarrow \{1, \dots, n\} \times \{1, \dots, n\}$.

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

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

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

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**.

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

$$\text{hor}_{i,j,t_1,t_2} := (h_{i,j,t_1}^R, h_{i+1,j,t_2}^L),$$

$$\text{ver}_{i,j,t_1,t_2} := (v_{i,j,t_1}^U, v_{i,j+1,t_2}^D),$$

$$\text{horBeg}_{i,t} := (h_{1,i,1}^L, h_{1,i,t}^L),$$

$$\text{horEnd}_{i,t} := (h_{k,i,t}^R, h_{k,i,n^2}^R),$$

$$\text{verBeg}_{i,t} := (v_{i,1,1}^D, v_{i,1,t}^D),$$

$$\text{verEnd}_{i,t} := (v_{i,k,t}^U, v_{i,k,n^2}^U).$$

Next, we define sets of vertical and horizontal segments:

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

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

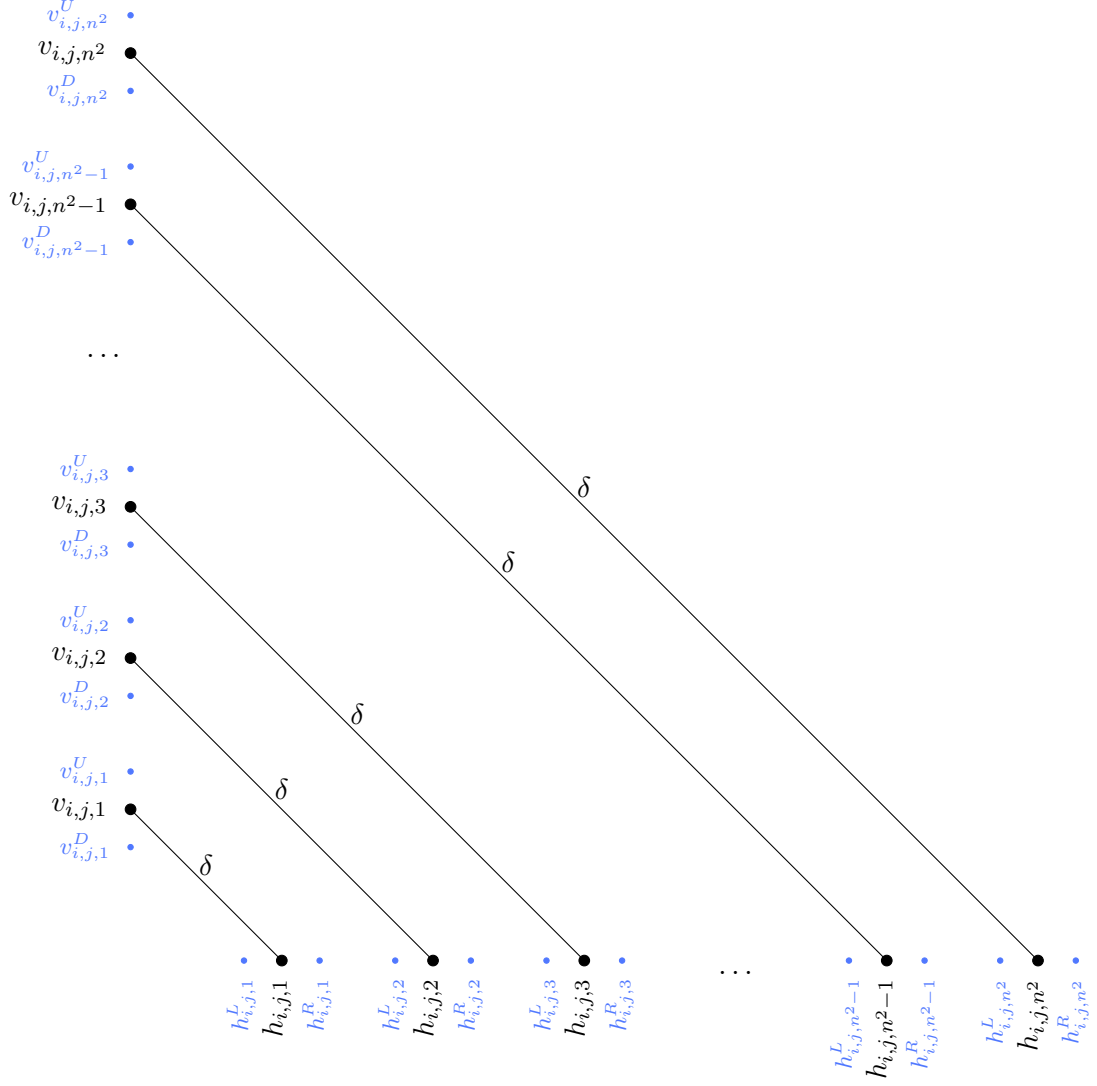


Figure 5.2: **Vertices and segments in DIAG.**

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).

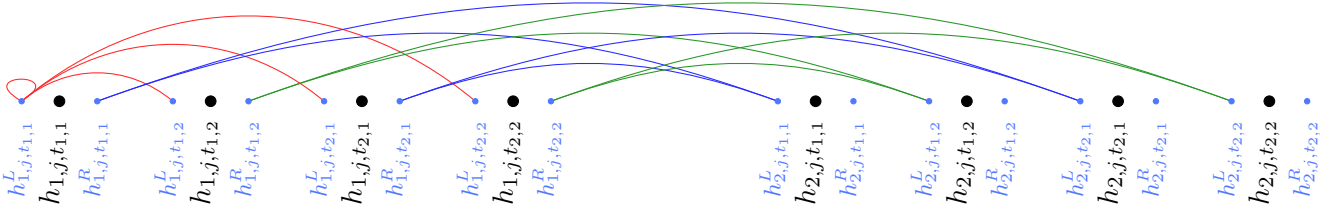


Figure 5.3: **Vertices and segments in HOR.**

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 $\text{order}^{-1}(i, j)$. Segments are represented as arcs between endpoints. You can see $\text{horBeg}_{j,t}$ segments in red. $\text{horBeg}_{j,1}$ is degenerated to a single point at $h_{1,1,t_{1,1}}^L$. Segments $\text{hor}_{i,j,t_{x_1,y},t_{x_2,y}}$ 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.

885 An example is depicted in Figure 5.3.

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

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

886 An example of such segments is depicted in Figure 5.2.

887 Every segment in **DIAG** connects points $(i(n^2+1)+t, j \cdot (n^2+1))$ and $(i \cdot (n^2+1), j(n^2+1) + t)$
 888 for some $1 \leq i, j \leq k, 1 \leq t \leq n^2$. The line on which it lies can be described by linear equation
 889 $x + y = t + (i + j)(n^2 + 1)$, thus these segments are in fact right-diagonal.

890 The constructed segment set is defined as:

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

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

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

893 Now, we prove that the constructed instance of geometric set cover with weighted segments
 894 indeed gives a correct and sound reduction of the grid tiling problem. Lemma 5.2 proves that
 895 if a solution to the instance of the grid tiling instance exists, then there exists a solution with
 896 suitably bounded size and weight of the constructed instance of geometric set cover. Then
 897 Lemma 5.6 proves that if there is a solution to the geometric set cover instance with bounded
 898 weight, then there exists a solution to the original grid tiling instance.

899 **Lemma 5.2.** *If there exists a solution to the grid tiling instance $(f_{i,j})$, then there exists*
 900 *a solution to the instance $(\mathcal{C}, \mathcal{P}, w, 3k^2 + 2k)$ of geometric set cover with weight $W_{\text{hv}} + k^2\delta$.*

901 *Proof.* Suppose there exists a solution x, y of the instance $(f_{i,j})$ of the grid tiling problem.

902 We define the proposed solution $\mathcal{R} \subseteq \mathcal{P}$ of the instance of geometric set cover in three

903 parts: $D \subseteq \text{DIAG}$, $A \subseteq \text{HOR}$ and $B \subseteq \text{VER}$:

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

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

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

905 Fix any $1 \leq j \leq k$ and define $t_i := \text{order}^{-1}(x(i), y(j))$. The two leftmost segments in A
906 for this j are $\text{horBeg}_{j,t_1} = (h_{1,j,1}^L, h_{1,j,t_1}^L)$ and $\text{hor}_{1,j,t_1,t_2} = (h_{1,j,t_1}^R, h_{2,j,t_2}^L)$. Therefore, points
907 $h_{1,j,x}^L, h_{1,j,x}^L$ and $h_{1,j,x}^R$ for all $1 \leq x \leq n^2$ are covered by horBeg_{j,t_1} and hor_{1,j,t_1,t_2} , excluding
908 point h_{1,j,t_1} .

909 Analogously for $2 \leq i \leq k-1$, the two consecutive segments $\text{hor}_{i-1,j,t_{i-1},t_i}$ and $\text{hor}_{i,j,t_i,t_{i+1}}$
910 cover points $h_{i,j,x}^L, h_{i,j,x}^L$ and $h_{i,j,x}^R$ for all $1 \leq x \leq n^2$, excluding point h_{i,j,t_i} .

911 Finally $\text{hor}_{k-1,j,t_{k-1},t_k}$ and horEnd_{j,t_k} cover all points $h_{k,j,x}^L, h_{k,j,x}^L$ and $h_{k,j,x}^R$ for $1 \leq x \leq n^2$,
912 excluding point h_{k,j,t_k} .

913 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:

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

914 Then, we need to compute the total weight of the solution \mathcal{R} . First, we compute the sum
915 of weights of segments in A . Fix $1 \leq j \leq k$ and consider segments collinear with the j -th
916 horizontal line. All points $h_{i,j,t}, h_{i,j,t}^L$ and $h_{i,j,t}^R$ for every $1 \leq i \leq k$ and $1 \leq t \leq n^2$ are covered
917 by A excluding points $h_{i,j, \text{order}^{-1}(x(i), y(j))}$. Every such point leaves a gap of length 2ϵ between
918 $h_{i,j, \text{order}^{-1}(x(i), y(j))}^L$ and $h_{i,j, \text{order}^{-1}(x(i), y(j))}^R$. Therefore, the total weight of segments in A that
919 lie on the line in question equals the length of the segment $(h_{i,1,1}^L, h_{i,k,n^2}^R)$ minus $2\epsilon k$, which is
920 $k(n^2 + 1) - 2(1 - \epsilon) - 2k\epsilon$. We need to multiply that by k , as we consider all possible values
921 of j .

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

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

924 Now we present a few additional properties of the constructed instance of the geometric
925 set cover that help us to prove Lemma 5.6.

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

- 927 • *the left and right guards of points in H (points in $\{p^L : p \in H\} \cup \{p^R : p \in H\}$) have*
928 *to be covered with segments from HOR ,*
- 929 • *the lower and upper guards of points in V (points in $\{p^D : p \in V\} \cup \{p^U : p \in V\}$) have*
930 *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 $1 \leq i \leq k$,
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 $1 \leq i, j \leq n$ 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 $1 \leq t \leq n^2$ must be covered with
segments from **HOR** or **VER**.*

Proof. We prove the claim for horizontal segments, as the proof for vertical segments is ana-
logous.

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

Point h_{i,j,t_1}^R has to be covered with a segment from **HOR** by Claim 5.1. Every segment in
HOR covering h_{i,j,t_1}^R , but not h_{i,j,t_1} must start at h_{i,j,t_1}^R 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 $1 \leq i \leq k$.

We provide a lower bound for the sum of lengths of vertical segments from $\mathcal{R} \cap \text{VER}$. 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 $v_{i,1,1}^D$ and v_{i,k,n^2}^U is:

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

(2) For every $1 \leq j \leq k$ there exists at most one $1 \leq t \leq n^2$ such that $v_{i,j,t}$ is not covered
by segments from **VER** (Claim 5.2). Its guards (see Definition 5.3) $v_{i,j,t}^U$ and $v_{i,j,t}^D$ 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 $v_{i,1,1}^D$ and v_{i,k,n^2}^U .

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

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

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 $W_{\text{hv}} + k^2\delta$. Then for every $1 \leq i, j \leq k$ there exists $1 \leq t \leq n^2$ 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) $\text{order}(t) \in f(i, j)$, that is, $\text{order}(t)$ is an allowed tile for (i, j) ;
- (4) for every $1 \leq s \leq n^2, s \neq t, v_{i,j,s}$ is covered in **VER**;

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

966 *Proof.* At most one of the points $\{h_{i,j,t_x} : 1 \leq t_x \leq n^2\}$ and one of the points $\{v_{i,j,t_y} : 1 \leq$
 967 $t_y \leq n^2\}$ is covered with **DIAG** (Claim 5.2).

968 Moreover, exactly one such point h_{i,j,t_x} and one such point v_{i,j,t_y} is covered with **DIAG**,
 969 because if none of them were covered, then the solution would have to have weight at least
 970 $W_{\text{hv}} + 2\epsilon$ (see the proof of Lemma 5.3), which is more than $W_{\text{hv}} + k^2\delta$.

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

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

978 We refer to the function mapping $1 \leq x \leq k$ to t_x from Lemma 5.4 as **diagonal** : $\{1 \dots k\} \times$
 979 $\{1 \dots k\} \rightarrow \{1 \dots n^2\}$.

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

982 1. for any $1 \leq i < k, 1 \leq j \leq k$, $\text{match}_h(\text{diagonal}(i, j), \text{diagonal}(i + 1, j))$ is **true**;

983 2. for any $1 \leq i \leq k, 1 \leq j < k$, $\text{match}_v(\text{diagonal}(i, j), \text{diagonal}(i, j + 1))$ is **true**.

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

985 Let us take any $1 \leq i < k, 1 \leq j \leq k$ and name $t_1 = \text{diagonal}(i, j)$ and $t_2 = \text{diagonal}(i +$
 986 $1, j)$. We also assume that $\text{match}_h(t_1, t_2)$ is **false**, which is equivalent to the fact that segment
 987 $(h_{i,j,t_1}^R, h_{i+1,j,t_2}^L)$ is not in set **HOR**.

988 Therefore h_{i,j,t_1} and h_{i+1,j,t_2} are not covered by segments from **HOR** (Lemma 5.4), while
 989 h_{i,j,t_1}^R and h_{i+1,j,t_2}^L have to be covered by segments from **HOR** (Claim 5.1).

990 Every segment from **HOR** either:

991 • starts at point h_{x,y,z_1}^R and ends at point h_{x+1,y,z_2}^L for some $1 \leq x < k, 1 \leq y \leq k$ and
 992 $1 \leq z_1, z_2 \leq n^2$; or

993 • is **horBeg** _{y,z} and starts at $h_{1,y,1}^L$ and ends at $h_{1,y,z}^L$ for some $1 \leq y \leq k$ and $1 \leq z \leq n^2$;
 994 or

995 • is **horEnd** _{y,z} and starts at $h_{k,y,z}^R$ and ends at h_{k,y,n^2}^R for some $1 \leq y \leq k$ and $1 \leq z \leq n^2$.

996 All of the points between h_{i,j,t_1}^R and h_{i+1,j,t_2}^L are covered by segments in **HOR** and there is no
 997 segment $(h_{i,j,t_1}^R, h_{i+1,j,t_2}^L)$ in **HOR**. Hence, there are at least two different segments covering
 998 them. If both of these segments are neither **horBeg** _{y,z} nor **horEnd** _{y,z} , then one of them must
 999 begin at h_{i,j,t_1}^R and end at h_{i+1,j,z_2}^L and there must be other one that begins at h_{i,j,z_1}^R and ends
 1000 at h_{i+1,j,t_2}^L for some $1 \leq z_1, z_2 \leq n^2$.

1001 Thus, the space between h_{i,j,z_1}^R and $h_{i,j+1,z_2}^L$ would be covered twice and is longer than ϵ .
 1002 The case when one of them is **horBeg** _{y,z} or **horEnd** _{y,z} is analogous. Note that they cannot be
 1003 both **horBeg** _{y,z} or **horEnd** _{y,z} .

1004 By the proof of Lemma 5.3, the lower bound for weight of such a solution is $W_{\text{hv}} + \epsilon$ which
 1005 is more than $W_{\text{hv}} + k^2\delta$.

1006 Therefore h_{i,j,t_1}^R and h_{i+1,j,t_2}^L must be covered by one segment from **HOR**, namely $(h_{i,j,t_1}^R, h_{i+1,j,t_2}^L)$.

1007 Hence $(h_{i,j,t_1}^R, h_{i+1,j,t_2}^L)$ is a segment in **HOR** and $\text{match}_h(t_1, t_2)$ is **true**. \square

1008 **Lemma 5.6.** *If there exists a solution to instance $(\mathcal{C}, \mathcal{P}, w, 3k^2 + 2k)$ with weight at most*
 1009 *$W_{\text{hv}} + k^2\delta$, then there exists a solution to the grid tiling instance $(f_{i,j})$.*

1010 *Proof.* Take **diagonal** function from Lemma 5.4.

1011 To define the x function for every $1 \leq i \leq k$ set $x(i) := x_i$ where $(x_i, a) = \text{order}(v_{i,1})$.
 1012 Similarly, to define the y function, for every $1 \leq i \leq k$ set $y(i) := y_i$ where $(b, y_i) = \text{order}(h_{1,i})$

1013 To prove that this is a correct solution to grid tiling, we need to prove that for every
 1014 $1 \leq i, j \leq k$, $(x(i), y(j))$ is in the allowed tiles set $f(i, j)$.

1015 Let us take any $1 \leq i, j \leq k$. By Lemma 5.5 and simple induction, we know that
 1016 $\text{match}_h(\text{diagonal}(1, j), \text{diagonal}(i, j))$ and $\text{match}_v(\text{diagonal}(i, 1), \text{diagonal}(i, j))$ are **true**. There-
 1017 fore $\text{order}(\text{diagonal}(i, j)) = (x(i), y(j))$. By Lemma 5.4 we know that $\text{order}(\text{diagonal}(i, j))$ is in
 1018 $f(i, j)$. 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 approximation 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, 2009] Har-Peled, S. and Lee, M. (2009). Weighted geometric set cover problems revisited. *Journal of Computational Geometry*, 3.
- [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.
- [Marx, 2005] Marx, D. (2005). Efficient approximation schemes for geometric problems? In Brodal, G. S. and Leonardi, S., editors, *Algorithms – ESA 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.

- 1053 [Marx and Pilipczuk, 2015] Marx, D. and Pilipczuk, M. (2015). Optimal parameterized algo-
1054 rithms for planar facility location problems using voronoi diagrams. *CoRR*, abs/1504.05476.
- 1055 [Marx and Schlotter, 2011] Marx, D. and Schlotter, I. (2011). Stable assignment with couples:
1056 Parameterized complexity and local search. *Discret. Optim.*, 8(1):25–40.
- 1057 [Mustafa et al., 2014] Mustafa, N. H., Raman, R., and Ray, S. (2014). Settling the apx-
1058 hardness status for geometric set cover. In *55th IEEE Annual Symposium on Foundations
1059 of Computer Science, FOCS 2014, Philadelphia, PA, USA, October 18-21, 2014*, pages
1060 541–550. IEEE Computer Society.
- 1061 [Mustafa and Ray, 2010] Mustafa, N. H. and Ray, S. (2010). Improved results on geometric
1062 hitting set problems. *Discret. Comput. Geom.*, 44(4):883–895.
- 1063 [Pilipczuk et al., 2016] Pilipczuk, M., van Leeuwen, E. J., and Wiese, A. (2016). Approxima-
1064 tion and parameterized algorithms for geometric independent set with shrinking. *CoRR*,
1065 abs/1611.06501.
- 1066 [Pilipczuk et al., 2020] Pilipczuk, M., van Leeuwen, E. J., and Wiese, A. (2020). Quasi-
1067 polynomial time approximation schemes for packing and covering problems in planar
1068 graphs. *Algorithmica*, 82(6):1703–1739.
- 1069 [Shachnai and Zehavi, 2017] Shachnai, H. and Zehavi, M. (2017). A multivariate framework
1070 for weighted FPT algorithms. *J. Comput. Syst. Sci.*, 89:157–189.
- 1071 [Wiese, 2018] Wiese, A. (2018). Independent set of convex polygons: From n^ϵ to $1 + \epsilon$ via
1072 shrinking. *Algorithmica*, 80(3):918–934.