

1

University of Warsaw

2

Faculty of Mathematics, Informatics and Mechanics

3

Katarzyna Kowalska

Student no. 371053

4

5

# Approximation and Parametrized Algorithms for Segment Set Cover

6

Master's thesis

7

in COMPUTER SCIENCE

8

Supervisor:

**dr Michał Pilipczuk**

Instytut Informatyki

9

June 2020

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 blabalysis

35

## **Tytuł pracy w języku polskim**

36 Algorytmy parametryzowania i trudność aproksymacji problemu pokrywania zbiorów  
37 odcinkami na płaszczyźnie



# Contents

39	<b>1. Introduction</b>	5
40	<b>2. Definitions</b>	7
41	2.1. Geometric Set Cover	7
42	2.2. Approximation	7
43	2.3. $\delta$ -extensions	7
44	<b>3. Geometric Set Cover with segments</b>	9
45	3.1. FPT for segments	9
46	3.1.1. Axis-parallel segments	9
47	3.1.2. Segments in arbitrary directions	9
48	3.2. APX-completeness for segments parallel to axes	11
49	3.2.1. MAX-(3,3)-SAT and statement of reduction	11
50	3.2.2. Reduction	13
51	3.2.2.1. VARIABLE-gadget	13
52	3.2.2.2. OR-gadget	14
53	3.2.2.3. CLAUSE-gadget	16
54	3.2.2.4. Summary	18
55	3.2.2.5. Summary of construction	18
56	3.2.3. Construction lemmas and proof of Lemma 3	19
57	3.3. FPT for weighted segments with $\delta$ -extensions	21
58	3.4. W[1]-completeness for weighted segments in 3 directions	23
59	3.5. What is missing	25
60	<b>4. Geometric Set Cover with lines</b>	27
61	4.1. Lines parallel to one of the axis	27
62	4.2. FPT for arbitrary lines	27
63	4.3. APX-completeness for arbitrary lines	27
64	4.4. 2-approximation for arbitrary lines	28
65	4.5. Connection with general set cover	28
66	<b>5. Geometric Set Cover with polygons</b>	29
67	5.1. State of the art	29
68	<b>6. Conclusions</b>	31



# Chapter 1

## Introduction

The Set Cover problem is one of the most common NP-complete problems. [tutaj referencja]  
We are given a family of sets and have to choose the smallest subfamily of these sets that cover  
all their elements. This problem naturally extends to settings where we put different weights  
on the sets and look for the subfamily of the minimal weight. This problem is NP-complete  
even without weights and if we put restrictions on what the sets can be. One of such variants  
is Vertex Cover problem, where sets have size 2 (they are edges in a graph).

In this work we focus on another such variant where the sets correspond to some geometric  
shapes and only some points of the plane have to be covered. When these shapes are rectangles  
with edges parallel to the axis, the problem can be proven to be W[1]-complete (solution of  
size  $k$  cannot be found in  $n^o(k)$  time), APX-complete (for sufficiently small  $\epsilon > 0$ , the problem  
does not admit  $1 + \epsilon$ -approximation scheme) [referencje].

Some of these settings are very easy. Set cover with lines parallel to one of the axis can  
be solved in polynomial time.

There is a notion of  $\delta$ -expansions, which loosen the restrictions on geometric set cover. We  
allow the objects to cover the points after  $\delta$ -expansion and compare the result to the original  
setting. This way we can produce both FPT and EPTAS for the rectangle set cover with  
 $\delta$ -extensions [referencje].

**Our contribution.** In this work, we prove that unweighted geometric set cover with seg-  
ments is fixed parameter tractable (FPT).

Moreover, we show that geometric set cover with segments is APX-complete for unweighted  
axis-parallel segments, even with  $1/2$ -extensions. So the problem for very thin rectangles  
also can't admit PTAS. Therefore, in the efficient polynomial-time approximation scheme  
(EPTAS) for *fat polygons* by [Har-Peled and Lee, 2009], the assumption about polygons  
being fat is necessary.

Finally, we show that geometric set cover with weighted segments in 3 directions is  
W[1]-complete. However, geometric set cover with weighted segments is FPT if we allow  
 $\delta$ -extension.

This result is especially interesting, since it's counter-intuitive that the unweighted setting  
is FPT and the weighted setting is W[1]-complete. Most of such problems (like vertex cover  
or [wiecej przykladow]) are equally hard in both weighted and unweighted settings.





## Chapter 2

## Definitions

### 2.1. Geometric Set Cover

In the geometric set cover problem we are given  $\mathcal{P}$  – a set of objects, which are connected subsets of the plane,  $\mathcal{C}$  – 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 element from  $\mathcal{R}$  and  $|\mathcal{R}|$  is minimized.

In the parametrized setting for a given  $k$ , we only look for a solution  $\mathcal{R}$  such that  $|\mathcal{R}| \leq k$ .

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  $\sum_{R \in \mathcal{R}} f(R)$ .

### 2.2. Approximation

Let us recall some definitions related to optimization problems that are used in the following sections.

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

**Definition 2.** A problem  $\Pi$  is **APX-hard** if assuming  $P \neq NP$ , there exists  $\epsilon > 0$  such that there is no polynomial-time  $(1 + \epsilon)$ -approximation algorithm for  $\Pi$ .

### 2.3. $\delta$ -extensions

TODO PLACEHOLDER for introductory text

$\delta$ -extensions is one of the modifications to a problem, that makes geometric set cover problem easier, it has been already used in literature (place some refrence here).

**Definition 3** ( $\delta$ -extensions for center-symmetric objects). For any  $\delta > 0$  and a center-symmetric object  $L$  with centre of symmetry  $S = (x_s, y_s)$ , the  $\delta$ -**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 centered at  $S$  with scale  $(1 + \delta)$

The geometric set cover problem with  $\delta$ -extensions is a modified version of geometric set cover where:

- We need to cover all the points in  $\mathcal{C}$  with objects from  $\{P^{+\delta} : P \in \mathcal{P}\}$  (which always include no fewer points than the objects before  $\delta$ -extensions);

- We look for a solution that is no larger than the optimum solution for the original problem. Note that it does not need to be an optimal solution in the modified problem.

Formally, we have the following.

**Definition 4** (Geometric set cover problem with  $\delta$ -extensions). The geometric set cover problem with  $\delta$ -extensions is the problem where for an input instance  $I = (\mathcal{P}, \mathcal{C})$ , the task is to output a solution  $\mathcal{R} \subseteq \mathcal{P}$  such that the  $\delta$ -extended set  $\{R^{+\delta} : R \in \mathcal{R}\}$  covers  $\mathcal{C}$  and is no larger than the optimal solution for the problem without extensions, i.e.  $|\mathcal{R}| \leq |\text{opt}(I)|$ .

TODO: Some text

**Definition 5** (Geometric set cover PTAS with  $\delta$ -extensions). We define a PTAS for geometric set cover with  $\delta$ -extensions as a family of algorithms  $\{\mathcal{A}_{\delta, \epsilon}\}_{\delta, \epsilon > 0}$  that each takes as an input instance  $I = (\mathcal{P}, \mathcal{C})$ , and in polynomial-time outputs a solution  $\mathcal{R} \subseteq \mathcal{P}$  such that the  $\delta$ -extended set  $\{R^{+\delta} : R \in \mathcal{R}\}$  covers  $\mathcal{C}$  and is within a  $(1 + \epsilon)$  factor of the optimal solution for this problem without extensions, i.e.  $(1 + \epsilon)|\mathcal{R}| \leq |\text{opt}(I)|$ .

## Chapter 3

# Geometric Set Cover with segments

### 3.1. FPT for segments

In this section we consider the fixed-parameter tractable algorithms for unweighted geometric set cover with segments. Setting where segments are limited to be axis-parallel (or limited to constant number of directions) has an FPT algorithm already present in literature. We present an FPT algorithm for unweighted geometric set cover with segments, where segments are in arbitrary directions.

#### 3.1.1. Axis-parallel segments

You can find this in Platypus book. (TODO add referece)

We show an  $\mathcal{O}(2^k)$ -time branching algorithm. The algorithm covers one of the points with a segment in one of the directions greedily until it covers all points or decide the is no solution of size  $k$ .

Let us take the point  $a$  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 choice of one of the coordinates  $x$  or  $y$ . Among segments that share this coordinate with  $a$ , we greedily choose the segment that covers the most points. As  $a$  was the smallest in the lexicographical order, then all points that one of the coordinates with  $a$ , they have larger second coordinate. Therefore if we choose the segment that covers the most points  $s$ , it would be the segment that have the end with highest second coordinate. It also hold that every other segment covering  $a$  in this direction does not cover any point that is not covered by  $s$ . Therefore greedy choice of this segments is optimal.

If after  $k$  steps we do not cover all of the points, there is no solution of size at most  $k$ .

TODO: Maybe split it into theorem + algorithm + explanation like in section 3.1.2

TODO: Should it use Remark template?

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

#### 3.1.2. Segments in arbitrary directions

In this section we consider setting where segments are not constrained to only  $d$  directions. We present a fixed-parameter tractable algorithm, where parameter is the size of the solution.

**Theorem 1. (FPT for segment cover).** *There exists an algorithm that given a family  $\mathcal{P}$  of  $n$  segments (in any direction), a set of  $m$  points  $\mathcal{C}$  and a parameter  $k$ , runs in time*

178  $f(k) \cdot (nm)^c$  for some computable function  $f$  and constant  $c$ , and outputs a subfamily  $\mathcal{R} \subseteq \mathcal{P}$   
 179 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  
 180 exist.

181 We will need the following lemmas.

182 **Lemma 1.** *Given an instance  $(\mathcal{P}, \mathcal{C})$  of the segment cover problem, without a loss of generality*  
 183 *we can assume that no segment covers a superset of what another segment covers. That is,*  
 184 *for any distinct  $A, B \in \mathcal{P}$ , we have  $A \cap C \not\subseteq B \cap C$  and  $A \cap C \not\supseteq B \cap C$ .*

185 *Proof.* Trivial. □

186 **Lemma 2.** *Given an instance  $(\mathcal{P}, \mathcal{C})$  of the segment cover problem, if there exists a line  $L$*   
 187 *with at least  $k + 1$  points on it, then there exists a subset  $\mathcal{A} \subseteq \mathcal{P}$ ,  $|\mathcal{A}| \leq k$ , such that every*  
 188 *solution  $\mathcal{R}$  with  $|\mathcal{R}| \leq k$  satisfies  $|\mathcal{A} \cap \mathcal{R}| \geq 1$ . Moreover, such a subset can be found in*  
 189 *polynomial time.*

190 *Proof.* First we use Lemma 1.

191 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  
 192 they appear on  $L$ . Every segment that is not collinear with  $L$  can cover at most one of these  
 193 points. Therefore, in any solution of size not larger than  $k$ , among any  $k$  of these points at  
 194 least one must be covered with segment collinear with  $L$ .

195 Therefore, every solution needs to take one of the segments collinear with  $L$  that covers  
 196 any of the points  $x_1, x_2, \dots, x_k$ . After using reduction from Lemma 1, there are at most  $k$  such  
 197 segments that are distinct. □

198 We are ready to prove Theorem 1.

199 *Proof of Theorem 1.*

200 **Algorithm.** First we use Lemma 1.

201 Next, we present a recursive algorithm. Given an instance of the problem:

202 (1) If there exist a line with at least  $k + 1$  points from  $\mathcal{C}$ , we branch over adding to the  
 203 solution one of the at most  $k$  possible segments provided by Lemma 2; name this segment  
 204  $S$ . Then we find a solution  $\mathcal{R}$  for the problem for points  $\mathcal{C} - S$ , segments  $\mathcal{P} - \{S\}$ , and  
 205 parameter  $k - 1$ . We return  $\mathcal{R} \cup \{S\}$ .

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

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

208 **Correctness.** Lemma 2 proves that at least one segment that we branch over in (1)  
 209 must be present in every solution  $\mathcal{R}$  with  $|\mathcal{R}| \leq k$ . Therefore, the recursive call can find a  
 210 solution, provided there exists one.

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

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

215 **Complexity.** In the leaves of recursion we have  $|\mathcal{C}| \leq k^2$ , so  $|\mathcal{P}| \leq k^4$ , because every  
 216 segments can be uniquely identified by the two extreme points it covers (by Lemma 1).  
 217 Therefore, there are  $\binom{k^4}{k}$  possible solutions to check, each can be checked in time  $O(k|\mathcal{C}|)$ .  
 218 Therefore, (3) takes time  $k^{O(k)}$ .

219 In this branching algorithm our parameter  $k$  is decreased with every recursive call, so we  
 220 have at most  $k$  levels of recursion with branching over  $k$  possibilities. Candidates to branch  
 221 over can be found on each level in time  $O(nm \log(nm))$ .

222 Reduction from Lemma 1 can be implemented in time  $O(n^2m)$ .

223 It follows that the overall complexity is  $O(n^2m + nm \log(nm) \cdot f(k))$   $\square$

## 224 3.2. APX-completeness for segments parallel to axes

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

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

230 **Theorem 2. (*axis-parallel segment set cover with  $1/2$ -extension is APX-hard*).**  
 231 *Unweighted geometric set cover with axis-parallel segments in 2D (even with  $1/2$ -extension)*  
 232 *is APX-hard. That is, assuming  $P \neq NP$ , there does not exist a PTAS for this problem.*

233 Theorem 2 implies the following.

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

236 We prove Theorem 2 by taking a problem that is APX-hard and showing a reduction. For  
 237 this problem we choose MAX-(3,3)-SAT which we define below.

### 238 3.2.1. MAX-(3,3)-SAT and statement of reduction

239 **Definition 6.** MAX-3SAT is the following maximization problem. We are given a 3-CNF  
 240 formula, and need to find an assignment of variables that satisfies the most clauses.

241 **Definition 7.** MAX-(3,3)-SAT is a variant of MAX-3SAT with an additional restriction  
 242 that every variable appears in exactly 3 clauses. Note that thus, the number of clauses is  
 243 equal to the number of variables.

244 In our proof of Theorem 2 we use hardness of approximation of MAX-(3,3)-SAT proved  
 245 in [Håstad, 2001] and described in Theorem 3 below.

246 **Definition 8** ( $\alpha$ -satisfiable MAX-3SAT formula). MAX-3SAT formula of size  $n$  is at most  
 247  $\alpha$ -satisfiable, if every assignment of variables satisfies no more than  $\alpha n$  clauses.

248 **Theorem 3. [Håstad, 2001]**

249 *For any  $\epsilon > 0$ , it is NP-hard to distinguish satisfiable (3,3)-SAT formulas from at most*  
 250  *$(7/8 + \epsilon)$ -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) 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 2.

**Lemma 3.** *Given an instance  $S$  of MAX-(3,3)-SAT with  $n$  variables and optimum value  $\text{opt}(S)$ , 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 of  $S$  that satisfies at least  $15n - |X|$  clauses.*

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

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

*Therefore, the optimum size of a solution of  $I$  is  $\text{opt}(I) = 15n - \text{opt}(S)$ .*

We prove Lemma 3 in subsequent sections, but meanwhile let us prove Theorem 2 using Lemma 3 and Theorem 3.

TODO: This below can't use current template

*Proof of Theorem 2.* 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  $(1/2)$ -extensions. We construct an algorithm that solves the problem stated in Theorem 3, 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. 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  $\text{approx}(I)$ . We prove that if in  $S$  one can satisfy at most  $(\frac{7}{8} + \epsilon)n$  clauses, then  $\text{approx}(I) \geq 15n - (\frac{7}{8} + \epsilon)n$  and if  $S$  is satisfiable, then  $\text{approx}(I) < 15n - (\frac{7}{8} + \epsilon)n$ .

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

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

From Lemma 3 we have:

$$\text{opt}(I) = 14n.$$

Therefore,

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

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

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

From Lemma 3 we have:

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

281 Since a solution to  $I$  with  $\frac{1}{2}$ -extensions is also a solution without extensions, by Lemma 3  
282 (3.), we have:

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

283 Therefore, by using the assumed  $(1 + \epsilon)$ -approximation algorithm, it is possible to dis-  
284 tinguish the case when  $S$  is satisfiable from the case when it is at most  $(\frac{7}{8} + \epsilon)n$  satisfiable,  
285 it suffices to compute  $\text{approx}(I)$  with  $15n - (\frac{7}{8} + \epsilon)n$ . Hence, the assumed approximation  
286 algorithm cannot exist, unless  $P = NP$ .  $\square$

### 287 3.2.2. Reduction

288 We proceed to the proof of Lemma 3. That is, we show a reduction from MAX-(3,3)-SAT  
289 problem to geometric set cover with segments parallel to axis. Moreover, the obtained instance  
290 of geometric set cover will be robust to 1/2-extensions (have the same optimal solution after  
291 1/2-extension).

292 The construction will be composed of 2 types of gadgets: **VARIABLE-gadgets** and  
293 **CLAUSE-gadgets**. **CLAUSE-gadgets** would be constructed using two **OR-gadgets** con-  
294 nected together. Every gadget consists of a point set and a segment set.

#### 295 3.2.2.1. VARIABLE-gadget

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

299 **Points.** Define points  $a, b, c, d, e, f, g, h$  as follows, where  $L = 12n$ :

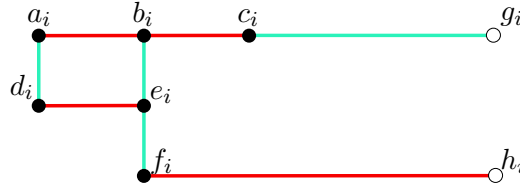


Figure 3.1: **VARIABLE-gadget**. We denote the set of points marked with black circles as  $C_{var}^i$ , and they need to be covered (are part of the set  $\mathcal{C}$ ). Note that some of the points are not marked as black dots and exists only to name segments for further reference. We denote the set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

300

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

Let us define:

$$C_{var} = \{a, b, c, d, e, f\}$$

and

$$C_{var}^i = C_{var} + (0, 4i)$$

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

302 **Segments.** Let us define:

$$X_{true}^i = \{(a_i, d_i), (b_i, f_i), (c_i, g_i)\}$$

$$X_{false}^i = \{(a_i, c_i), (d_i, e_i), (f_i, h_i)\}$$

$$P_{var}^i = X_{true}^i \cup X_{false}^i$$

303 **Lemma 4.** For any  $1 \leq i \leq n$ , points in  $C_{var}^i$  can be covered using 3 segments from  $P_{var}^i$ .

304 *Proof.* We can use either set  $X_{true}^i$  or  $X_{false}^i$ . □

305 **Lemma 5.** For any  $1 \leq i \leq n$ , points in  $C_{var}^i$  can not be covered with fewer than 3 segments  
306 from  $P_{var}^i$ .

307 *Proof.* No segment of  $P_{var}^i$  covers more than one point from  $\{d_i, f_i, c_i\}$ , therefore  $C_{var}^i$  can not  
308 be covered with fewer than 3 segments. □

309 **Lemma 6.** For every set  $A \subseteq P_{var}^i$  such that  $A$  covers  $C_{var}^i$  and  $(c_i, g_i), (f_i, h_i) \in A$ , it holds  
310 that  $|A| \geq 4$ .

311 *Proof.* No segment from  $P_{var}^i$  covers more than one point from  $\{a_i, e_i\}$ , therefore  $C_{var}^i$  -  
312  $\{c_i, f_i, g_i, h_i\}$  can not be covered with fewer than 2 segments. □

### 313 3.2.2.2. OR-gadget

314 OR-gadget has 3 important segments -  $x, y, result$ .  $x$  and  $y$  don't count to the weight of  
315 solution of OR-gadget (they are part of different gadgets). It has a minimal solution of weight  
316  $w$  and  $result$  can be chosen only if  $x$  or  $y$  are also chosen for the solution. If none of them  
317 are chosen, then solution choosing  $result$  segment has weight at least  $w + 1$ . Therefore the  
318 following formula holds for a solution  $R$  assuming that  $R$  uses only  $w$  from this OR-gadget:

$$(x \in R) \vee (y \in R) \iff result \in R$$

319 **Points.**

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

$$vec_{i,j} = (10i + 3 + 3j, 4n + 2j)$$

321 Define  $\{l_{i,j}, m_{i,j} \dots v_{i,j}\}$  as  $\{l_0, m_0 \dots v_0\}$  shifted by  $vec_{i,j}$

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

$$C\_or\_gadget_{i,j} = \{l_{i,j}, m_{i,j}, n_{i,j}, o_{i,j}, p_{i,j}, q_{i,j}, r_{i,j}, s_{i,j}, t_{i,j}, u_{i,j}\}$$





Figure 3.2: **OR-gadget.** We denote these point as  $or\_gadget_{i,j}$ . We denote set of red segments as  $or_{i,j}^{false}$ , set of blue segments as  $or_{i,j}^{true}$ , green and yellow segments as  $or\_move\_variable_{i,j}$ .

323 **Segments.** We define names subsets of segments, to refer to them in lemmas.

$$or_{i,j}^{false} = \{(q_{i,j}, r_{i,j}), (s_{i,j}, u_{i,j})\}$$

$$or_{i,j}^{true} = \{(m_{i,j}, s_{i,j}), (o_{i,j}, u_{i,j}), (t_{i,j}, v_{i,j})\}$$

$$or\_move\_variable_{i,j} = \{(l_{i,j}, n_{i,j}), (n_{i,j}, p_{i,j})\}$$

324 Segments in OR-gadget:

$$P\_or\_gadget_{i,j} = or_{i,j}^{false} \cup or_{i,j}^{true} \cup or\_move\_variable_{i,j}$$

325 **Lemma 7.** For any  $1 \leq i \leq n, j \in \{0, 1\}$  and  $x \in \{l_{i,j}, p_{i,j}\}$  we can cover points in  
 326  $C\_or\_gadget_{i,j} - \{x\} \cup \{v_{i,j}\}$  with 4 segments from  $P\_or\_gadget_{i,j}$ .

327 *Proof.* We can do that using one segment from  $or\_move\_variable_{i,j}$  (chosen depending on  
 328 the value of  $x$ ) and all segments from  $or_{i,j}^{true}$ .  $\square$

329 **Lemma 8.** For any  $1 \leq i \leq n, j \in \{0, 1\}$ , we can cover points in  $C\_or\_gadget_{i,j}$  with 4  
 330 segments from  $P\_or\_gadget_{i,j}$ .

331 *Proof.* We can do that using  $or\_move\_variable_{i,j}$  and  $or_{i,j}^{false}$ .  $\square$

### 3.2.2.3. CLAUSE-gadget

CLAUSE-gadget is responsible for calculating if choice of the variable values meets the clause in formula. It has minimal solution of weight  $w$  if at least one variable in the clause has a correct value. Otherwise it has minimal solution  $w + 1$ . This way by the minimal solution for the whole problem, we can tell how many clauses were satisfiable.

The CLAUSE-gadgets consist of two OR-gadgets. We don't want the CLAUSE-gadgets to be crammed somewhere between the very long variable segments. That's why we have a simple gadget to *pass* the value of the segment, ie. segments  $(x_{i,0}, x_{i,1}), (y_{i,0}, y_{i,1}), (z_{i,0}, z_{i,1})$ . Two segments and one of them is chosen if  $x$  was chosen in the solution and the other one if  $x$  wasn't.

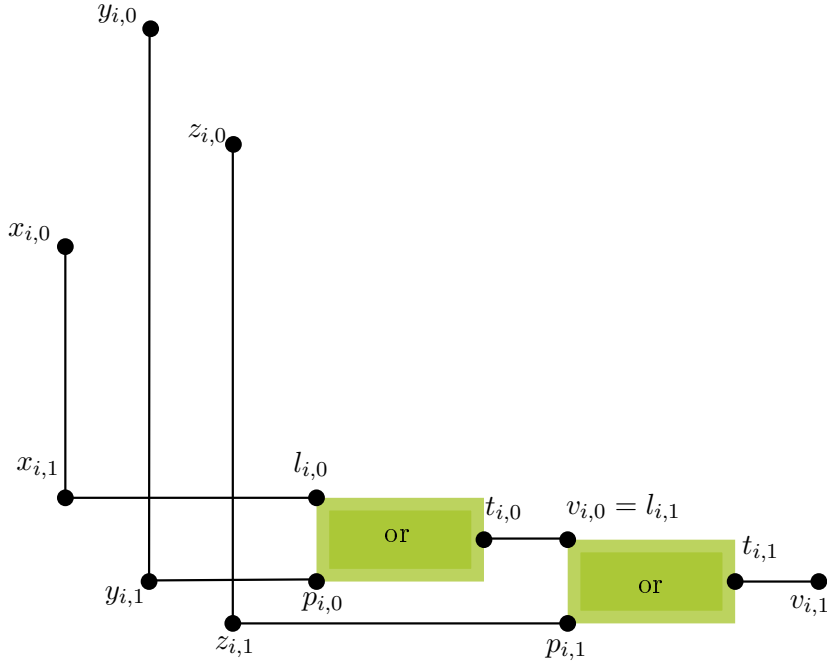


Figure 3.3: **CLAUSE-gadget**. We denote set of these points as  $C\_clause_i$ . Every green rectangle is an OR-gadget.  $y$ -coordinates of  $x_{i,0}$ ,  $y_{i,0}$  and  $z_{i,0}$  depend on the values of variables in the  $i$ -th clause.

**Points.** TODO: Rephrase it

Assuming clause  $C_i = x_i \vee y_i \vee z_i$ , function  $idx(w)$  is returning index of the variable  $w$ , function  $neg(w)$  is returning whether variable  $w$  is negated in a clause.

$$\begin{aligned} x_{i,0} &= (10i + 1, 4 \cdot idx(x_i) + 2 \cdot neg(x_i)) & x_{i,1} &= (10i + 1, 4n) \\ y_{i,0} &= (10i + 2, 4 \cdot idx(y_i) + 2 \cdot neg(y_i)) & y_{i,1} &= (10i + 2, 4n + 4) \\ z_{i,0} &= (10i + 3, 4 \cdot idx(z_i) + 2 \cdot neg(z_i)) & z_{i,1} &= (10i + 3, 4n + 6) \end{aligned}$$

$$move\_variable_i = \{x_{i,j} : j \in \{0, 1\}\} \cup \{y_{i,j} : j \in \{0, 1\}\} \cup \{z_{i,j} : j \in \{0, 1\}\}$$

$$C\_clause_i = move\_variable_i \cup C\_or\_gadget_{i,0} \cup C\_or\_gadget_{i,1} \cup \{v_{i,1}\}$$

## Segments.

$$P\_clause_i = \{(x_{i,0}, x_{i,1}), (y_{i,0}, y_{i,1}), (z_{i,0}, z_{i,1}), (x_{i,1}, l_{i,0}), (y_{i,1}, p_{i,0}), (z_{i,1}, p_{i,1}), \} \cup \\ \cup P\_or\_gadget_{i,0} \cup P\_or\_gadget_{i,1}$$

346 **Lemma 9.** For any  $1 \leq i \leq n$  and  $a \in \{x_{i,0}, y_{i,0}, z_{i,0}\}$ , points in  $C\_clause_i - \{a\}$  can be  
347 covered with a set of segments  $P\_true_i^a$ , a subset of  $P\_clause_i$  such that  $|P\_true_i^a| = 11$ .

348 *Proof.* For  $a = x_{i,0}$  (analogous proof for  $y_{i,0}$ ): First we use Lemma 7 twice with excluded  
349  $x = l_{i,0}$  and  $x = l_{i,1} = v_{i,0}$ , resulting with 8 segments  $or_{i,0}^{true} \cup or_{i,1}^{true}$  which cover all required  
350 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:  
351  $\{(x_{i,1}, l_{i,0}), (y_{i,0}, y_{i,1}), (z_{i,0}, z_{i,1})\}$

352 For  $a = z_{i,0}$ : Using Lemma 8 and Lemma 7 with  $x = p_{i,1}$ , resulting with 8 segments  
353  $or_{i,0}^{false} \cup or_{i,1}^{true}$  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  
354 those using additional 3 segments:  $\{(x_{i,0}, x_{i,1}), (y_{i,0}, y_{i,1}), (z_{i,1}, p_{i,1})\}$ .  $\square$

355 **Lemma 10.** For any  $1 \leq i \leq n$ , points in  $C\_clause_i$  can be covered with a set of segments  
356  $P\_false_i^a$ , a subset of  $P\_clause_i$  such that  $|P\_false_i^a| = 12$ .

357 *Proof.* Using Lemma 8 twice we can cover  $or\_gadget_{i,0}$  and  $or\_gadget_{i,1}$  with 8 segments.

358 To cover the remaining points we additionally use:  $\{(x_{i,0}, x_{i,1}), (y_{i,0}, y_{i,1}), (z_{i,0}, z_{i,1}), (t_{i,1}, v_{i,1})\}$   
359  $\square$

360 **Lemma 11.** For any  $1 \leq i \leq n$ :

361 (1) points in  $C\_clause_i - \{x_{i,0}, y_{i,0}, z_{i,0}\}$  can not be covered using any subset of segments  
362 from  $P\_clause_i$  of size less than 11;

363 (2) points in  $C\_clause_i$  can not be covered using any subset of segments from  $P\_clause_i$   
364 of size less than 12.

365 *Proof of no cover with fewer than 12 segments.* There is independent set of 12 points in  $C\_clause_i \supseteq$   
366  $\{x_{i,0}, y_{i,0}, z_{i,0}, l_{i,0}, p_{i,0}, q_{i,0}, u_{i,0}, v_{i,0} = l_{i,1}, p_{i,1}, q_{i,1}, u_{i,1}, v_{i,1}\}$ .  $\square$

367 *Proof of no cover with fewer than 11 segments.* We can choose disjoint sets  $X, Y, Z$  such that  
368  $X \cup Y \cup Z \subseteq C\_clause_i - \{x_{i,0}, y_{i,0}, z_{i,0}\}$  and there are no segments covering points from  
369 different sets. And we prove lower bounds for each of these sets.

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

370 Set  $X$  is an indendent set, so it must be covered with 3 segments.

$$Y = or\_gadget_{i,0} - \{l_{i,0}, p_{i,0}\}$$

$$Z = or\_gadget_{i,1} - \{l_{i,1}, p_{i,1}\}$$

371 For both  $Y$  and  $Z$  we can check all of the subsets of 3 segments with brutforce that none  
372 of them cover, so they have to be covered with 4 segments.

373 TODO: Funny fact, neither  $Y$  nor  $Z$  doesn't have independent set of size 4.

374 Therefore  $C\_clause_i$  must be covered with at least  $3 + 4 + 4 = 11$  segments.  $\square$

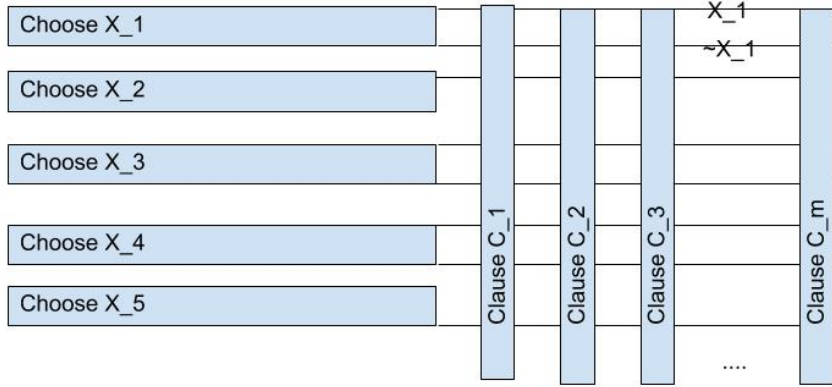


Figure 3.4: **General schema.**

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

TODO: Rename Choose X to VARIABLE-gadget and Clause C to CLAUSE-gadget.

#### 3.2.2.4. Summary

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

**Lemma 12. Robustness to 1/2-extensions.** *For every segment  $s \in \mathcal{P}$ ,  $s$  and  $s^{+1/2}$  cover the same points from  $\mathcal{C}$ .*

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

TODO: list problematic segments here

□

#### 3.2.2.5. Summary of construction

We define:

$$\mathcal{C} := \bigcup_{1 \leq i \leq n} C\_variable_i \cup C\_clause_i$$

$$\mathcal{P} := \bigcup_{1 \leq i \leq n} P\_variable_i \cup P\_clause_i$$

The subsequent sections define these sets.

384 We prove some properties of different gadgets. Every segment for a gadget will only cover  
 385 points in this gadget (won't interact with any different gadget), so we can prove lemmas *locally*.  
 386 TODO:  $y$  axis is increasing values downward on figures (not upwards like in normal).

### 387 3.2.3. Construction lemmas and proof of Lemma 3

388 **Lemma 13.** *Given an instance  $S$  of MAX-(3,3)-SAT of size  $n$  with optimum solution satis-*  
 389 *fying  $k$  clauses  $\text{opt}(S) = k$ . Instance of geometric set cover, constructed for  $S$  as described in*  
 390 *Section 3.2.2, can be solved with a solution of size  $15n - k$ .*

391 *Proof.* Let us name the assignments of the variables in the optimum solution of an instance  
 392  $S$  as  $y_1, y_2 \dots y_n$  and clauses as  $c_1, c_2 \dots c_n$ .

393 We cover every VARIABLE-gadget with solution described in Lemma 4, in the  $i$ -th gadget  
 394 choosing the set of segments corresponding to the value of  $y_i$ . CLAUSE-gadgets that are  
 395 satisfied, let us name the variable that is true in them  $a$ , are covered with set  $P\_true_i^a$   
 396 described in Lemma 9 and unsatisfied with set  $P\_false_i$  described in Lemma 10.

$$\begin{aligned} R_i &= \begin{cases} X_i^{true} & \text{if } y_i \\ X_i^{false} & \text{if } \neg y_i \end{cases} \\ C_i &= \begin{cases} P\_true_i^a & \text{if } c_i \text{ satisfied} \\ P\_false_i & \text{if } c_i \text{ not satisfied} \end{cases} \\ \mathcal{R} &= \bigcup_{i=1}^n \{R_i \cup C_i : 1 \leq i \leq n\} \end{aligned}$$

397 This set covers all points from  $\mathcal{C}$ , because the smaller sets individually cover their corre-  
 398 sponding gadgets (proved in respective lemmas).

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

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

400 □

401 **Lemma 14.** *Given an instance  $S$  of MAX-(3,3)-SAT of size  $n$  and a solution of an instance*  
 402 *of geometric set cover, as described in Section 3.2.2, that is of size  $w$ . There exists a solution*  
 403 *of an instance  $S$  that satisfied at least  $15n - w$  clauses.*

404 *Proof.*

405 Given a solution  $\mathcal{R}$  of the instance of geometric set cover, we construct a solution of the  
 406 instance  $S$  by constructing an assignment of variables that satisfies at least  $15n - w$  clauses in  
 407  $S$ .

408 **Variables** We need to use at least 3 segments to cover VARIABLE-gadget (Lemma 5).  
 409 If we have chosen both segments  $(c_i, g_i)$  and  $(f_i, h_i)$ , then we have used at least 4 segments  
 410 (Lemma 6).

$$\begin{cases} |C_{var}^i \cap \mathcal{R}| \geq 4 & \text{if } (c_i, g_i) \in \mathcal{R} \wedge (f_i, h_i) \in \mathcal{R} \\ |C_{var}^i \cap \mathcal{R}| \geq 3 & \text{otherwise} \end{cases}$$

If we chose at most one of the segments  $(c_i, g_i)$  and  $(f_i, h_i)$ , choose the corresponding variable value to the solution. If we chose both segments, choose the value that appears in most clauses. Every variable is in exactly 3 clauses, so one value appears in at least 2 of them. If we have chosen none of the segments, set value to false. Formally, we define the value of the  $x_i$  variable as follows:

$$\begin{cases} x_i = \text{majority}(X_i) & \text{if } (c_i, g_i) \in \mathcal{R} \wedge (f_i, h_i) \in \mathcal{R} \\ x_i = \text{true} & \text{if } (c_i, g_i) \in \mathcal{R} \\ x_i = \text{false} & \text{if } (f_i, h_i) \in \mathcal{R} \\ x_i = \text{false} & \text{otherwise} \end{cases} \quad (3.1)$$

TODO: Maybe remove section below, because we do this calculation at the end anyway To cover  $\bigcup_{1 \leq i \leq n} C_{var}^i$  we have used at least  $3n + a$  segments, where  $a$  is the number of  $i$  such that we have chosen both values  $(c_i, g_i)$  and  $(f_i, h_i)$ .

**Clauses** For a clause  $C_i = x \vee y \vee z$ , we need to use at least 11 segments to cover  $C_{\text{clause}_i} - \{x, y, z\}$  in CLAUSE-gadget (Lemma 11).

TODO: maybe put something with cases and names of sets as above

Moreover, if all of the points  $\{x_{i,0}, y_{i,0}, z_{i,0}\}$  are not covered by the segments from  $P_{var}^i$ , then we need to cover  $C_{\text{clause}_i}$  with at least 12 segments by Lemma 11.

TODO: Maybe remove section below, because we do this calculation at the end anyway We covered CLAUSE-gadget with at least 11 or at least 12 segments:  $|\bigcup_{i=1}^n P_{\text{clause}_i} \cap \mathcal{R}| \geq 11n + b$ , where  $b$  is the number of clauses where none of the segments covering the points  $x_{i,0}, y_{i,0}, z_{i,0}$  were chosen in  $P_{var}^j$ .

**Satisfied clauses with chosen variables assignment** Clauses for which none of the points  $x_{i,0}, y_{i,0}, z_{i,0}$  were covered by segments in  $P_{var}^j$ , are not satisfied in our variables assignment, but not all clauses that cover one of these points with segment in  $P_{var}^j$  are satisfied.

Let us look at such clause  $C_i$  and of points  $x_{i,0}, y_{i,0}, z_{i,0}$  that are covered in  $P_{var}^j$ . Consider the cases of choosing variable value in equation (3.1).

If only one of the segments  $(c_i, g_i)$  and  $(f_i, h_i)$  are chosen in  $P_{var}^j$ , then the value of  $x_j$  is the same as the one satisfying  $C_i$  and clause is satisfied.

If we chose neither  $(c_i, g_i)$  or  $(f_i, h_i)$ , then it is impossible that this point is covered in  $P_{var}^j$ .

If we chose both  $(c_i, g_i)$  and  $(f_i, h_i)$ , then there are 3 clauses for which this point is covered by  $P_{var}^j$ . We chose variable value in a way that only one clause using  $x_j$  is not satisfied by the value of  $x_j$ . Therefore there are at most  $a$  clauses that are covered with 11 segments from CLAUSE-gadget, but are not satisfied.

So in the solution to this MAX-(3,3)-SAT instance that we have shown, there are at most  $a + b$  unsatisfied clauses.

**Conclusions** We proved that given a solution of size  $w$  we have the variables assignment that satisfies at least  $n - (a + b)$  clauses of  $S$ . At last we prove that  $n - (a + b) \geq 15n - w$ .

$$w \geq 3(n - a) + 4a + 11(n - b) + 12b = 3n + a + 11n + b = 14n + a + b$$

$$15n - w \leq 15n - 14n - a - b = n - (a + b)$$

445

□

446 *Proof of Lemma 3.* Given an instance  $S$  of MAX-(3,3)-SAT of size  $n$  with optimum solution  
 447 satisfying  $k$  clauses. Let us construct an instance of geometric set cover for  $S$  as described in  
 448 Section 3.2.2 and name it  $I$ .

Given the Lemma 13, we know that there exists a solution of  $I$  of size  $15n - k$ , so:

$$\text{opt}(I) \leq 15n - k.$$

Since the optimum solution of  $S$  satisfies  $k$  clauses, then according to Lemma 14:

$$\text{opt}(I) \geq 15n - k.$$

449 Therefore solution from Lemma 13 of size  $15n - k$  is an optimum solution for instance  
 450  $I$ . □

### 451 3.3. FPT for weighted segments with $\delta$ -extensions

452 **Theorem 4** (FPT for weighted segment cover with  $\delta$ -extensions). *There exists an algorithm*  
 453  *$\mathcal{A}$  that given a family  $\mathcal{P}$  of  $n$  weighted segments (in any direction), a set of  $m$  points  $\mathcal{C}$ , and*  
 454 *parameters  $k$  and  $\delta$ , runs in time  $f(k, \delta) \cdot (nm)^c$  for some computable function  $f$  and a constant*  
 455  *$c$ , and outputs a set  $\mathcal{R} \subseteq \mathcal{P}$  such that  $|\mathcal{R}| \leq k$  and  $\mathcal{R}^{+\delta}$  covers all points in  $\mathcal{C}$  or determines*  
 456 *that such a set  $\mathcal{R}$  does not exist.*

457 To solve this problem we will introduce a lemma about choosing *good* subsets of points.

458 **Definition 9.** For a set of collinear points  $C$ , a subset  $A \subseteq C$  is  $(k, \delta)$ -**good** if for any set of  
 459 segments  $R$  that covers set  $A$  such that  $|R| \leq k$ , it holds that  $R^{+\delta}$  covers  $C$ .

460 **Lemma 15.** *There exists an algorithm that for any set of collinear points  $C$ ,  $\delta > 0$  and*  
 461  *$k \geq 1$ , outputs a  $(k, \delta)$ -good set of size at most  $f(k, \delta)$  for some computable function  $f$ . This*  
 462 *algorithm runs in time  $O(|C| \cdot f(k, \delta))$ .*

463 *Proof.* We prove this for a fixed  $\delta$  by induction over  $k$  for any set of collinear points  $C$ .

464 **Inductive hypothesis** For any set of collinear points  $C$ , there exists an algorithm that  
 465 runs in time  $O(|C|k(1 + \frac{1}{\delta}))$  and finds a set  $A$  such that:

- 466 •  $A$  is  $(l, \delta)$  – *good* for every  $1 \leq l \leq k$ ,
- 467 •  $A$  has size  $|A| < f(\delta, k)$  for some computable function  $f$ ,
- 468 • extreme points from  $C$  are in  $A$ .

469 **Base case for  $k = 1$**  It is sufficient that  $A$  consists of 2 points: extreme points from  $C$   
 470 or a single point if  $|C| = 1$ .

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

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

Let us name  $s$  the minimal segment that includes all points from  $C$ .

We define  $M = \lceil 1 + \frac{2}{\delta} \rceil$  subsegments of  $s$  in the following way. We split  $s$  into  $M$  parts  $v_i$  of equal length, that is  $|v_i| = \frac{|s|}{M}$  for any  $1 \leq i \leq M$ .

$C_i$  is a subset of  $C$  such that they lay on  $v_i$ .

$t_i$  is a segment connecting leftmost and rightmost point in  $C_i$  (it might be degenerated segment if  $|C_i| = 1$  or it might be empty if  $C_i$  is empty).

TODO: Add a picture with  $v_i$  and  $t_i$  here

We use inductive hypothesis to choose  $(k, \delta)$ -good sets  $A_i$  for sets  $C_i$ . If  $|C_i| \leq 1$ , then  $A_i = C_i$  and it's still a  $(k, \delta)$ -good set.

Then we define  $A = \bigcup_{i=1}^M A_i$ . It includes ends of  $s$ , because they are in sets  $A_1$  and  $A_M$ .

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

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

If there exists a segment  $t_i$  for which a segment  $x$  as defined above does not exist, then all  $k + 1$  segments that cover  $A_i$  intersect with  $t_i$ . (Note: There exists only one such segment  $t_i$ ). From the inductive hypothesis ends of  $s$  are in  $A_1$  and  $A_M$  respectively, so  $\mathcal{R}$  must cover them. Hence there must exist segments starting in the ends of  $s$  and ending somewhere in  $t_i$ . Let us name these two segments  $y$  and  $z$ . It follows that:  $|y| + |z| + |t_i| \geq |s|$ . Since  $|t_i| \leq |v_i| = \frac{|s|}{M} \leq \frac{|s|}{1 + \frac{2}{\delta}} = \frac{|s|\delta}{\delta + 2}$ , therefore  $\max(|y|, |z|) > |s|(1 - \frac{\delta}{\delta + 2})/2 = \frac{|s|}{\delta + 2}$ .

TODO: Add a picture with such segments here

After  $\delta$ -extension, the longer of these segments will lengthen both ways by at least:

$$\frac{|s|\delta}{\delta + 2} = \frac{|s|}{1 + \frac{2}{\delta}} > \frac{|s|}{M} = v_i > t_i.$$

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

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

**Complexity** We use the recursive algorithm for subsets  $C_i$ . Every point from  $C$  belongs to at most 2 sets  $C_i$ .

Apart from recursive algorithm we perform operations linear in size of  $|C| + M$  to calculate the sets  $C_i$ .

Therefore it has complexity:

$$O(|C| + M) + \sum_i^M O(|C_i|k(1 + \frac{1}{\delta})) = O(|C| + (1 + \frac{1}{\delta})) + O((\sum_i^M |C_i|)k(1 + \frac{1}{\delta})) \leq O(|C|k(1 + \frac{1}{\delta})).$$

□

*Proof of Theorem 4.* To construct an algorithm for this problem let us formulate some claims about the problem first.

**Definition 10.** Line is **long** if there are at least  $k + 1$  points from  $\mathcal{C}$  on it.



508 **Claim 1.** *If there are more than  $k$  long lines, then  $\mathcal{C}$  can not be covered with  $k$  segments.*

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

511 Applying the above claims, if we have more than  $k$  long lines or more than  $k^2$  points form  
 512  $\mathcal{C}$  that do not lie on any long line, then we answer that there is no solution of size at most  $k$ .

513 Otherwise, we can split  $\mathcal{C}$  into at most  $k + 1$  sets:  $D$ , at most  $k^2$  points that do not lie on  
 514 any long line and  $C_i$  – points that lay on  $i$ -th long line. Sets  $C_i$  do not need to be disjoint.

515 Then for every set  $C_i$ , we can use Lemma 15 to get  $(k, \delta)$ -good set  $A_i$  for  $C_i$ .

516 Then we have set  $D \cup \bigcup A_i$  of size at most  $f(k, \delta)$  for some computable function  $f$ , that  
 517 if we have a solution  $\mathcal{R}$  of size at most  $k$  that covers  $D \cup \bigcup A_i$ , then  $\mathcal{R}^{+\delta}$  covers  $\mathcal{C}$ . This is  
 518 because  $\mathcal{R}$  already covers points  $D$ , they cover  $C_i$ , because they cover  $(k, \delta)$ -good set  $A_i$  with  
 519 at most  $k$  segments, so  $\mathcal{R}^{+\delta}$  covers  $C_i$ .

520 After that we shrunk down size of  $\mathcal{C}$  to size of  $f(k, \delta)$  for some computable function  $f$ .  
 521 Then we would like to shrink down size of  $\mathcal{P}$ . For every collinear subset of  $D$ , we can choose  
 522 one segment from  $\mathcal{P}$  that covers these points and have the lowest weight or decide there is  
 523 no segment that cover them. There are at most  $|D|^2$  different segments, because we can  
 524 distinguish these collinear sets by their extreme points.

525 This has complexity  $O(|D|^2|\mathcal{P}|)$  and produce shrunk down set  $\mathcal{P}$  of size  $f(k, \delta)$  for some  
 526 computable function  $f$ .

527 Then we can iterate over all subsets of shrunk down set  $\mathcal{P}$  and choose the set with the  
 528 lowest sum of weights that cover  $D$ . This solution would have weight not larger than optimal  
 529 solution for the problem without extension, because we iterate over all possibilities of covering  
 530 the subset of  $\mathcal{C}$ .

531 □

### 532 3.4. W[1]-completeness for weighted segments in 3 directions

533 **Theorem 5.** *W[1]-completeness for weighted segments in 3 directions. Consider the*  
 534 *problem of covering a set  $\mathcal{C}$  of points by selecting  $k$  axis-parallel or right-diagonal weighted*  
 535 *segments with weights from a set  $\mathcal{P}$  with minimal weight. Assuming ETH, there is no algorithm*  
 536 *for this problem with running time  $f(k) \cdot (|\mathcal{C}| + |\mathcal{P}|)^{o(\sqrt{k})}$  for any computable function  $f$ .*

537 We will show reduction from grid tiling problem.

538 Let's have an instance of grid tiling problem – size of the grid  $k$ , number of elements  
 539 available  $n$  and  $k^2$  sets of available pairs in every tile  $S_{i,j} \subseteq \{1, n\} \times \{1, n\}$ .

540 **Construction.** We construct a set  $\mathcal{P}$  of segments and a set  $\mathcal{C}$  of points.

541 First let's choose any ordering of  $n^2$  elements  $\{1, n\} \times \{1, n\}$  and name this sequence  
 542  $a_1 \dots a_{n^2}$ .

$$match_v(i, j) \iff a_i = \{x_i, y_i\} \wedge a_j = \{x_j, y_j\} \wedge x_i = x_j$$

$$match_h(i, j) \iff a_i = \{x_i, y_i\} \wedge a_j = \{x_j, y_j\} \wedge y_i = y_j$$

**Points.** Define points:

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

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

Let's define sets  $H$  and  $V$  as:

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

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

543 Let's define  $\epsilon = 0.1$ . For a point  $\{x, y\} = p$  we define points  $p^L = \{x - \epsilon, y\}$ ,  $p^R = \{x + \epsilon, y\}$ ,  
 544  $p^U = \{x, y - \epsilon\}$ , and  $p^D = \{x, y + \epsilon\}$ .

Then we define:

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

545 **Segments.** Define horizontal segments.

$$hor_{i,j,t_1,t_2} = (h_{i,j,t_1}^R, h_{i,j+1,t_2}^L)$$

$$ver_{i,j,t_1,t_2} = (v_{i,j,t_1}^D, v_{i,j+1,t_2}^U)$$

$$horbeg_{i,t} = (h_{i,1,1}^L, h_{i,1,t}^L)$$

$$horend_{i,t} = (h_{i,n,t}^R, h_{i,n,n^2}^R)$$

$$verbeg_{i,t} = (v_{i,1,1}^U, v_{i,1,t}^U)$$

$$verend_{i,t} = (v_{i,n,t}^D, v_{i,n,n^2}^D)$$

$$\begin{aligned} HOR &= \{hor_{i,j,t_1,t_2} : 1 \leq i \leq k, 1 \leq j < k, 1 \leq t_1, t_2 \leq n^2, match_h(t_1, t_2)\} \\ &\cup \{horbeg_{i,t} : 1 \leq i \leq k, 1 \leq t \leq n^2\} \\ &\cup \{horend_{i,t} : 1 \leq i \leq k, 1 \leq t \leq n^2\} \end{aligned}$$

$$\begin{aligned} VER &= \{ver_{i,j,t_1,t_2} : 1 \leq i \leq k, 1 \leq j < k, 1 \leq t_1, t_2 \leq n^2, match_v(t_1, t_2)\} \\ &\cup \{verbeg_{i,t} : 1 \leq i \leq k, 1 \leq t \leq n^2\} \\ &\cup \{verend_{i,t} : 1 \leq i \leq k, 1 \leq t \leq n^2\} \end{aligned}$$

$$DIAG := \{(h_{i,j,t}, v_{j,i,t}) : 1 \leq i, j \leq k, 1 \leq t \leq n^2, a_t \in S_{i,j}\}$$

546 TODO: explain that these segments are in fact diagonal

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

547 **Lemma 16.** *If there exists solution for grid tiling, then there exists solution for our construc-*  
 548 *tion using  $2(k+1)k + k^2$  segments with weight exactly  $2k \cdot (k(n^2 + 1) - 2 - 2\epsilon(k-1))$ .*

**Claim 3.** *If there exists a solution to the grid tiling  $c_1 \dots c_k$  and  $r_1 \dots r_k$ , then there exists a solution covering all points*

$$\{h_{i,j,t} : 1 \leq i, j \leq k, t = (c_i, r_j)\} \cup \{v_{i,j,t} : 1 \leq i, j \leq k, t = (c_j, r_i)\}$$

549 *with segments in DIAG and the rest in VER or HOR and has weight  $2k \cdot (k(n^2 + 1) -$*   
 550  *$2 - 2\epsilon(k - 1))$ .*

551 **Proof.** TODO: jakiś prosty z definicji

552 **Lemma 17.** *If there exists solution for our construction using  $2(k + 1)k + k^2$  segments with*  
 553 *weight exactly  $2k \cdot (k(n^2 + 1) - 2 - 2\epsilon(k - 1))$ , then there exists a solution for grid tiling*

554 **Proof.** This follows from Lemma 18, because we just take which points are covered with  
 555 *DIAG.*

556 **Claim 4.** *Points  $p^L, p^R, p^U, p^D$  cannot be covered with DIAG.*

557 **Claim 5.** *Points in  $H \cup \{p^L : p \in H\} \cup \{p^R : p \in H\}$  cannot be covered with VER.*

558 *Points in  $V \cup \{p^U : p \in V\} \cup \{p^D : p \in V\}$  cannot be covered with HOR.*

559 **Claim 6.** *For given  $i, j$  if none of the points  $h_{i,j,t}$  ( $v_{i,j,t}$ ) for  $1 \leq t \leq n^2$  are covered with*  
 560 *DIAG, then some spaces between neighbouring points were covered twice.*

561 **Claim 7.** *For given  $i, j$  two points  $h_{i,j,t_1}, h_{i,j,t_2}$  ( $v_{i,j,t_1}, v_{i,j,t_2}$ ) for  $1 \leq t_1 < t_2 \leq n^2$  are covered*  
 562 *with DIAG, then one of them had to be also covered with a segment from HOR (VER).*

563 **Proof.** Point  $v_{i,j,t_2}^L$  had to be covered with VER from Claims 4 and 5. And every segment  
 564 in VER covering  $v_{i,j,t_2}^L$ , covers also  $v_{i,j,t_1}^L$ .

565 **Lemma 18.** *If there exists solution for our construction with weight at most (exactly)  $2k \cdot$*   
 566  *$(k(n^2 + 1) - 2 - 2\epsilon(k - 1))$ , then for every  $i, j$  there must be exactly one  $t$  such that  $h_{i,j,t}$  ( $v_{i,j,t}$ )*  
 567 *is covered with DIAG and moreover if  $h_{i,j,t_1}$  and  $h_{i,j+1,t_2}$  are uncovered, then  $\text{math}_h(t_1, t_2)$ .*  
 568 *Analogically for  $v$ .*

569 **Proof.** Only  $k^2$  points can be covered only in DIAG, the rest has to be covered with  
 570 *VER  $\cup$  HOR.* Therefore every result must be at least *ALL\_LINES* -  $2k^2\epsilon$ , because only  
 571  $2k^2$  spaces of length  $\epsilon$  can be uncovered in this axis.

572 Of course if  $h_{i,j,t_1}$  and  $h_{i,j+1,t_2}$  are uncovered, then there must exist a segment in HOR  
 573 between  $h_{i,j,t_1}^R$  and  $h_{i,j+1,t_2}^L$ , so  $\text{math}_h(t_1, t_2)$  must be true.

### 574 3.5. What is missing

575 We don't know FPT for axis-parallel segments without  $\delta$ -extensions.



## 576 Chapter 4

# 577 Geometric Set Cover with lines

### 578 4.1. Lines parallel to one of the axis

579 When  $\mathcal{R}$  consists only of lines parallel to one of the axis, the problem can be solved in  
580 polynomial time.

581 We create bipartial graph  $G$  with node for every line on the input split into sets:  $H$  –  
582 horizontal lines and  $V$  – vertical lines. If any two lines cover the same point from  $\mathcal{C}$ , then we  
583 add edge between them.

584 Of course there will be no edges between nodes inside  $H$ , because all of them are pararell  
585 and if they share one point, they are the same lines. Similar argument for  $V$ . So the graph is  
586 bipartial.

587 Now Geometric Set Cover can be solved with Vertex Cover on graph  $G$ . Since Vertex  
588 Cover (even in weighted setting) on bipartial graphs can be solved in polynomial time.

589 Short note for myself just to remember how to this in polynomial time:

590 Non-weighted setting - Konig theorem + max matching

591 Weighted setting - Min cut in graph of  $\neg A$  or  $\neg B$  (edges directed from  $V$  to  $H$ )

### 592 4.2. FPT for arbitrary lines

593 You can find this is Platypus book. We will show FPT kernel of size at most  $k^2$ .

594 (Maybe we need to reduce lines with one point/points with one line).

595 For every line if there is more than  $k$  points on it, you have to take it. At the end, if there  
596 is more than  $k^2$  points, return NO. Otherwise there is no more than  $k^4$  lines.

597 In weighted settings among the same lines with different weights you leave the cheapest  
598 one and use the same algorithm.

### 599 4.3. APX-completeness for arbitrary lines

600 We will show a reduction from Vertex Cover problem. Let's take an instance of the Vertex  
601 Cover problem for graph  $G$ . We will create a set of  $|V(G)|$  pairwise non-pararell lines, such  
602 that no three of them share a common point.

603 Then for every edge in  $(v, w) \in E(G)$  we put a point on crossing of lines for vertices  $v$   
604 and  $w$ . They are not pararell, so there exists exactly one such point and any other line don't  
605 cover this point (any three of them don't cross in the same point).

Solution of Geometric Set Cover for this instance would yield a sound solution of Vertex Cover for graph  $G$ . For every point (edge) we need to choose at least one of lines (vertices)  $v$  or  $w$  to cover this point.

Vertex Cover for arbitrary graph is APX-complete, so this problem is also APX-complete.

#### 4.4. 2-approximation for arbitrary lines

Vertex Cover has an easy 2-approximation algorithm, but here very many lines can cross through the same point, so we can do  $d$ -approximation, where  $d$  is the biggest number of lines crossing through the same point. So for set where any 3 lines don't cross in the same point it yields 2-approximation.

The problematic cases are where through all points cross at least  $k$  points and all lines have at least  $k$  points on them. It can be created by casting  $k$ -grid in  $k$ -D space on 2D space.

Greedy algorithm yields  $\log |\mathcal{R}|$ -approximation, but I have example for this for bipartial graph and reduction with taking all lines crossing through some point (if there are no more than  $k$ ) would solve this case. So maybe it works.

Unfortunately I haven't done this :(

I can link some papers telling it's hard to do.

#### 4.5. Connection with general set cover

Problem with finite set of lines with more dimensions is equivalent to problem in 2D, because we can project lines on the plane which is not perpendicular to any plane created by pairs of (point from  $\mathcal{C}$ , line from  $\mathcal{P}$ ).

Of course every two lines have at most one common point, so is every family of sets that have at most one point in common equivalent to some geometric set cover with lines?

No, because of Desargues's theorem. Have to write down exactly what configuration is banned.

## Chapter 5

# Geometric Set Cover with polygons

### 5.1. State of the art

Covering points with weighted discs admits PTAS [Li and Jin, 2015] and with fat polygons with  $\delta$ -extensions with unit weights admits EPTAS [Har-Peled and Lee, 2009].

Although with thin objects, even if we allow  $\delta$ -expansion, the Set Cover with rectangles is APX-complete (for  $\delta = 1/2$ ), it follows from APX-completeness for segments with  $\delta$ -expansion in Section 3.2.

Covering points with squares is W[1]-hard [Marx, 2005]. It can be proven that assuming *SETH*, there is no  $f(k) \cdot (|\mathcal{C}| + |\mathcal{P}|)^{k-\epsilon}$  time algorithm for any computable function  $f$  and  $\epsilon > 0$  that decides if there are  $k$  polygons in  $\mathcal{P}$  that together cover  $\mathcal{C}$ , *Theorem 1.9* in [Marx and Pilipczuk, 2015].





<sup>642</sup> Chapter 6

<sup>643</sup> Conclusions



## 644 Bibliography

- 645 [Har-Peled and Lee, 2009] Har-Peled, S. and Lee, M. (2009). Weighted geometric set cover  
646 problems revisited. *Journal of Computational Geometry*, 3.
- 647 [Håstad, 2001] Håstad, J. (2001). Some optimal inapproximability results. *J. ACM*,  
648 48(4):798–859.
- 649 [Li and Jin, 2015] Li, J. and Jin, Y. (2015). A PTAS for the weighted unit disk cover problem.  
650 *CoRR*, abs/1502.04918.
- 651 [Marx, 2005] Marx, D. (2005). Efficient approximation schemes for geometric problems? In  
652 Brodal, G. S. and Leonardi, S., editors, *Algorithms – ESA 2005*, pages 448–459, Berlin,  
653 Heidelberg. Springer Berlin Heidelberg.
- 654 [Marx and Pilipczuk, 2015] Marx, D. and Pilipczuk, M. (2015). Optimal parameterized algo-  
655 rithms for planar facility location problems using voronoi diagrams. *CoRR*, abs/1504.05476.