

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	1. Introduction	5
40	2. Definitions	7
41	3. Geometric Set Cover with segments	9
42	3.1. FPT for segments	9
43	3.1.1. Segments parallel to one of the axis	9
44	3.1.2. Segments in d directions	9
45	3.1.3. Segments in arbitrary direction	9
46	3.2. APX-completeness for segments parallel to axis	10
47	3.2.1. Definition of MAX-(3,3)-SAT problem	10
48	3.2.2. Reduction construction	11
49	3.2.2.1. Variable gadget	12
50	3.2.2.2. Or gadget	13
51	3.2.2.3. Clause gadget	14
52	3.2.2.4. Summary	16
53	3.2.3. Proofs of construction Lemma 3.2.1	16
54	3.2.3.1. Proof of Lemma 3.2.1	17
55	3.3. Weighted segments	17
56	3.3.1. FPT for weighted segments with δ -extensions	17
57	3.3.2. W[1]-completeness for weighted segments in 3 directions	19
58	3.3.3. What is missing	21
59	4. Geometric Set Cover with lines	23
60	4.1. Lines parallel to one of the axis	23
61	4.2. FPT for arbitrary lines	23
62	4.3. APX-completeness for arbitrary lines	23
63	4.4. 2-approximation for arbitrary lines	24
64	4.5. Connection with general set cover	24
65	5. Geometric Set Cover with polygons	25
66	5.1. State of the art	25
67	6. Conclusions	27

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 δ -expansions, which loosen the restrictions on geometric set cover. We
allow the objects to cover the points after δ -expansion and compare the result to the original
setting. This way we can produce both FPT and EPTAS for the rectangle set cover with
 δ -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
 δ -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.

100 Chapter 2

101 Definitions

102 Some definitions what geometric set cover is. \mathcal{P} – set of objects, \mathcal{C} – set of points. Choose
103 $\mathcal{R} \subset \mathcal{P}$ such that every point in \mathcal{C} is inside some element from \mathcal{R} and $|\mathcal{R}|$ is minimal.

104 In parametrized setting we only look among $|\mathcal{R}| \leq k$. In weighted settings there is some
105 $f : \mathcal{P} \rightarrow \mathbb{R}$ and we minimize $\sum_{R \in \mathcal{R}} f(R)$.

Chapter 3

Geometric Set Cover with segments

3.1. FPT for segments

3.1.1. Segments parallel to one of the axis

You can find this in Platypus book.

We'll show $\mathcal{O}(2^k)$ branching algorithm. Let's take point K that hasn't been covered yet with the smallest coordinate in lexicographical order. We need to cover K with some of the remaining segments.

We choose one of the 2 directions on which we will cover this point. In this direction we take greedily the segment that will cover the most points (there are points in \mathcal{C} only on one side of K in this direction, so all segments covering K in this direction create monotone sequence of sets – zbiory zstępujące).

3.1.2. Segments in d directions

The same algorithm as before but in complexity $\mathcal{O}(d^k)$.

3.1.3. Segments in arbitrary direction

Theorem 3.1.1 (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 $f(k) \cdot (nm)^c$ for some computable function f and constant c , and outputs a subfamily $\mathcal{R} \subseteq \mathcal{P}$ such that $|\mathcal{R}| \leq k$ and \mathcal{R} covers all points in \mathcal{C} .*

Proof. We will show such algorithm in FPT.

If there exist two segments a and b in \mathcal{P} , such that any point covered by a is also covered by b , then without loss of generality we can remove segment a from \mathcal{P} . We repeat this process until no such (a, b) pair exists.

Let us first assume that we reduced our instance to a kernel, where *any line* contains no more than k points.

Since any segment covers a set of colinear points, for such a kernel k segments can cover only at most k^2 points. Therefore, for the answer to be positive, the number of points has to be at most k^2 . The number of segments is now bounded by k^4 , since if we consider two *extreme* points covered by a given segment, then these pairs must be distinct, otherwise two segments would contain the same set of points. Since both the number of points and the

number of segments is bounded by a function of k , this instance can be easily solved in time $O(f(k))$.

It remains to show how to construct the kernel.

Assume there exists a line l containing points x_1, \dots, x_t , where $t \geq k + 1$. Note that a segment that does not lie on l can cover only at most one of the points x_i . Therefore, out of points x_1, \dots, x_{k+1} , at least one has to be covered by a segment that lies on l , let us fix x_i to be the first such point. Then, we can greedily choose a segment that lies on l , covers x_i , and also covers the largest number of points x_j for $j > i$.

Since we have at most $k + 1$ choices to branch over and each choice adds a segment to the constructed solution, we obtain an algorithm with complexity $O(k^k)$.

3.2. APX-completeness for segments parallel to axis

Let's analyze approximation of set cover with rectangles.

The question on the table is if there exists $(1 + \epsilon)$ -approximation scheme for set cover with rectangles.

Let's restrict this problem to some very easy setting: segments parallel to axes and allow $(1/2)$ -extension. Segments are basically degenerated rectangles with one side very narrow.

Theorem 3.2.1 (*axis-parallel segment set cover with $1/2$ -extension is APX-hard*). For sufficiently small $\epsilon > 0$, there does not exist an $(1 + \epsilon)$ -approximation scheme for unweighted geometric set cover with axis-parallel segments in 2D (even with $1/2$ -extension) (problem is APX-hard).

The problem of rectangle set cover doesn't have $(1 + \epsilon)$ -approximation scheme even in the above setting (so every less restricted setting also doesn't have approximation scheme).

Theorem 3.2.2 (*rectangle set cover is APX-hard*). For sufficiently small $\epsilon > 0$, there does not exist an $(1 + \epsilon)$ -approximation scheme for unweighted geometric set cover with rectangles (even with $1/2$ -extension) (problem is APX-hard).

We will prove it by taking a problem that is APX-complete (doesn't have $(1 + \epsilon)$ -approximation scheme). Such problem in this section will be MAX-(3,3)-SAT that we will define in detail below.

Given an instance I of MAX-(3,3)-SAT, we will construct an instance J of axis-parallel segment set cover problem, such that $(1 + c\epsilon)$ -approximation of J will approximate an I with $(1 + \epsilon)$ scheme for some constant $c > 1$. Therefore if there would exist general approximation scheme of the axis-parallel segment set cover problem, we would produce a general approximation scheme of MAX-(3,3)-SAT (that doesn't exist).

3.2.1. Definition of MAX-(3,3)-SAT problem

Here we define MAXSAT problem.

Theorem 3.2.3 [*Håstad, 2001*]

For any $\epsilon > 0$, it is NP-hard to distinguish satisfiable (3,3)-SAT formulas from $(7/8 + \epsilon)$ -satisfiable (3,3)-SAT formulas. Said equivalently, MAX-(3,3)-SAT is nonapproximable beyond the random assignment threshold on satisfiable instances.

175 **Lemma 3.2.1** *Given an instance of MAX-(3,3)-SAT with n variables and optimal result k ,*
 176 *we can construct an instance of axis-parallel segments in 2D, which optimal result (even with*
 177 *$1/2$ -extension) is exactly $15n - k$.*

178 We will provide a proof for Lemma in subsequent sections, but meanwhile let's prove
 179 Theorem 3.2.1 using Lemma 3.2.1 and Theorem ??.

180 **Proof of Theorem 3.2.1** Take any $0 < \epsilon < 1/(15 \cdot 8)$. Choose n sufficiently large, so that
 181 $\epsilon'(n)$ from Theorem ?? is not greater than ϵ .

182 Let's assume that there exists an $(1 + \epsilon)$ -approximation scheme for unweighted geometric
 183 set cover with axis-parallel segments in 2D. We will construct an algorithm distinguishing
 184 instances of MAX-(3,3)-SAT in Theorem 3.2.3. Take two instances to be distinguished and
 185 using Lemma 3.2.1 and name them satisfiable – S_1 and unsatisfiable – S_2 . Let's construct
 186 two instances of geometric set cover and name them respectively I_1 and I_2 .

187 Use $(1 + \epsilon)$ -approximation scheme for instances of geometric set cover, let's name the result
 188 of this approximation for an instance of problem I as $approx(I)$.

From definition of S_1 and S_2 we have:

$$OPT(S_1) = n$$

$$OPT(S_2) \leq (\frac{7}{8} + \epsilon'(n))n$$

189 From Lemma 3.2.1 we have:

$$OPT(I_1) = 14n$$

$$OPT(I_2) = 15n - (\frac{7}{8} + \epsilon'(n))n$$

190 Let's prove that $approx(I_2) > approx(I_1)$:

$$approx(I_2) \geq OPT(I_2) = 15n - (\frac{7}{8} + \epsilon'(n))n = 14n + (\frac{1}{8} - \epsilon'(n))n > 14n + (\frac{1}{8} - \epsilon)n >$$

$$> 14n + (15\epsilon - \epsilon)n = 14n + (14\epsilon)n = 14n(1 + \epsilon) = OPT(I_1)(1 + \epsilon) \geq approx(I_1)$$

191 Therefore, by using our supposed $(1 + \epsilon)$ approximation, it's possible to distinguish S_1
 192 from S_2 , since the approximation scheme will always return a smaller value for I_1 than for I_2 .
 193 This is a contradiction, hence the approximation scheme cannot exist.

194 3.2.2. Reduction construction

195 We will show reduction from MAX-(3,3)-SAT problem to geometric set cover with segments
 196 parallel to axis. Moreover the instance of geometric set cover will be robust to $1/2$ -extensions
 197 (have the same optimal solution after $1/2$ -extension).

198 The construction will be composed of 3 types of gadgets: variable gadgets, or gadgets,
 199 clause gadgets.

We define:

$$\mathcal{C} := \bigcup_{1 \leq i \leq n} C_variable_i \cup C_clause_i$$

$$\mathcal{P} := \bigcup_{1 \leq i \leq n} P_variable_i \cup P_clause_i$$

We will prove some properties of different gadgets. Every segment for gadget will only cover points in this gadget (won't interact with any different gadget), so we can prove lemmas *locally*.

TODO: y axis is increasing values downward on figures (not upwards like in normal).

3.2.2.1. Variable gadget

Points. Define points:

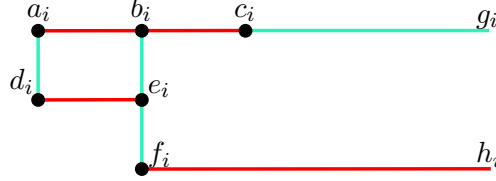


Figure 3.1: **Choose variable value gadget.** We denote set of points marked with black circle as $C_variable_i$ and need to be covered (are part of set \mathcal{C}). We denote set of red segments as x_i^{false} and set of blue segments as x_i^{true} .

TODO: inline $L = 12n$ after finishing these formulas

$$\begin{array}{llll} a_i = (-L, 4i) & b_i = (-\frac{2}{3}L, 4i) & c_i = (-\frac{1}{3}L, 4i) & d_i = (-L, 4i + 1) \\ e_i = (-\frac{2}{3}L, 4i + 1) & f_i = (-\frac{2}{3}L, 4i + 2) & g_i = (L, 4i) & h_i = (L, 4i + 2) \end{array}$$

Let's define

$$C_variable_i = \{a_i, b_i, c_i, d_i, e_i, f_i\}$$

Segments. Let's define

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

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

$$P_variable_i = x_i^{true} \cup x_i^{false}$$

Lemma 3.2.2 For any $1 \leq i \leq n$, points $C_variable_i$ can be covered using 3 segments from $P_variable_i$.

Proof. We can use set x_i^{true} or x_i^{false} .

Lemma 3.2.3 For any $1 \leq i \leq n$, points $C_variable_i$ can not be covered with less than 3 segments from $P_variable_i$.

Proof. There is independent set $\{d_i, f_i, c_i\}$ of size 3, therefore it can not be covered with less than 3 sets (segments).

Lemma 3.2.4 If both segments (c_i, g_i) and (f_i, h_i) are chosen, then the remaining points from $C_variable_i$ must be covered with 2 different segments from $P_variable_i$.

218 **Proof.** There is an independent set $\{a_i, e_i\}$ of size 2 in $P_variable_i - \{ (c_i, g_i), (f_i, h_i) \}$,
 219 therefore it can not be covered with less than 2 sets (segments).

220 3.2.2.2. Or gadget

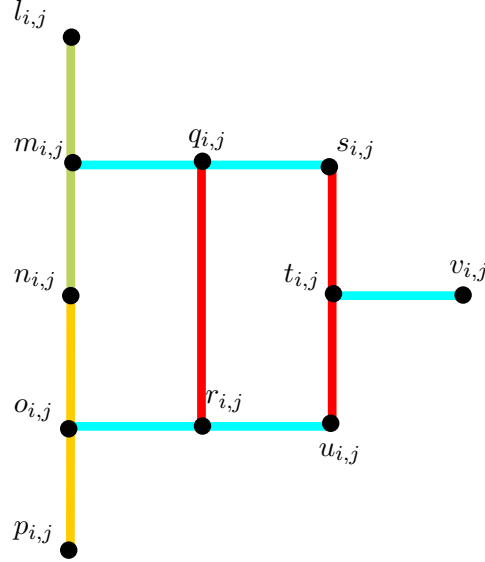


Figure 3.2: **Or gadget.** We denote these point as $or_gadget_{i,j}$. We denote set of red segments as $or_{i,j}^{false}$, set of blue segments as $or_{i,j}^{true}$, green and yellow segments as $or_move_variable_{i,j}$.

221 Points.

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

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

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

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

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

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

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

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

$$or_move_variable_{i,j} = \{(l_{i,j}, n_{i,j}), (n_{i,j}, p_{i,j})\}$$

226 Segments in or gadget:

$$P_or_gadget_{i,j} = or_{i,j}^{false} \cup or_{i,j}^{true} \cup or_move_variable_{i,j}$$

227 **Lemma 3.2.5** 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
 228 $C_or_gadget_{i,j} - \{x\} \cup \{v_{i,j}\}$ with 4 segments.

229 **Proof.** We can do that using one segment from $or_move_variable_{i,j}$ (chosen depending on
 230 the value of x) and all segments from $or_{i,j}^{true}$.

231 **Lemma 3.2.6** For any $1 \leq i \leq n, j \in \{0, 1\}$, we can cover points in $C_or_gadget_{i,j}$ with 4
 232 segments from $P_or_gadget_{i,j}$.

233 **Proof.** We can do that using $or_move_variable_{i,j}$ and $or_{i,j}^{false}$.

234 3.2.2.3. Clause gadget

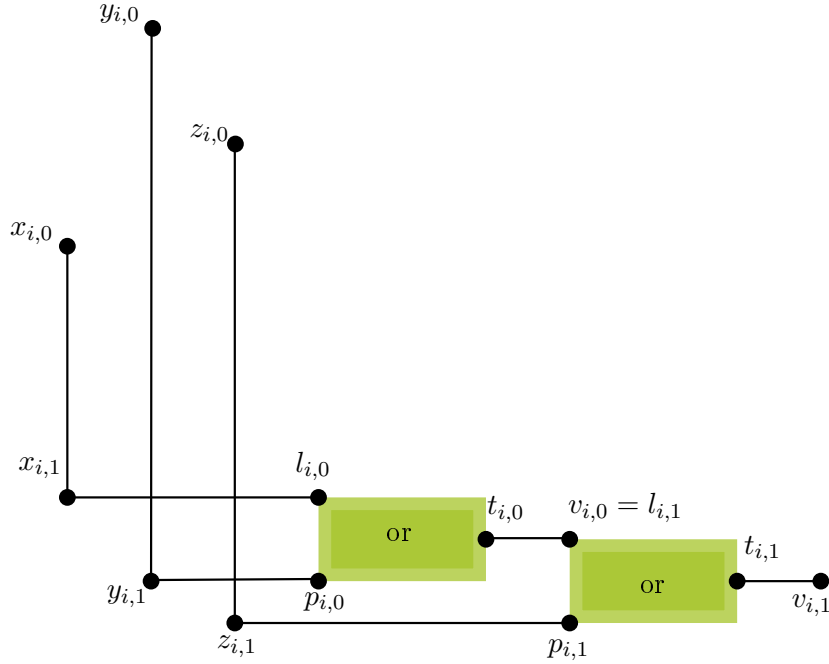


Figure 3.3: **Clause gadget.** We denote set of these points as C_clause_i . Every green rectangle is an or gadget. y -coordinates of $x_{i,0}$, $y_{i,0}$ and $z_{i,0}$ depend on the values of variables in the i -th clause.

235 **Points.** TODO: Rephrase it

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

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

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

$$C_clause_i = \text{move_variable}_i \cup C_or_gadget_{i,0} \cup C_or_gadget_{i,1} \cup \{v_{i,1}\}$$

Segments.

$$\begin{aligned}
P_clause_i &= \{(x_{i,0}, x_{i,1}), (y_{i,0}, y_{i,1}), (z_{i,0}, z_{i,1}), (x_{i,1}, l_{i,0}), (y_{i,1}, p_{i,0}), (z_{i,1}, p_{i,1}), \} \cup \\
&\cup P_or_gadget_{i,0} \cup P_or_gadget_{i,1}
\end{aligned}$$

Lemma 3.2.7 For any $1 \leq i \leq n$ and $a \in \{x_{i,0}, y_{i,0}, z_{i,0}\}$, points $C_clause_i - \{a\}$ can be covered using 11 segments from P_clause_i .

Proof. For $a = x_{i,0}$ (analogous proof for $y_{i,0}$): First we use Lemma 3.2.5 twice with excluded $x = l_{i,0}$ 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 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 3.2.6 and Lemma 3.2.5 with $x = p_{i,1}$, resulting with 8 segments $or_{i,0}^{false} \cup or_{i,1}^{true}$ 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 those using additional 3 segments: $\{(x_{i,0}, x_{i,1}), (y_{i,0}, y_{i,1}), (z_{i,1}, p_{i,1})\}$.

Lemma 3.2.8 Points C_clause_i can be covered with 12 segments from P_clause_i .

Proof. Using Lemma 3.2.6 twice we can cover $or_gadget_{i,0}$ and $or_gadget_{i,1}$ with 8 segments.

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

Lemma 3.2.9 For any $1 \leq i \leq n$, points $C_clause_i - \{x_{i,0}, y_{i,0}, z_{i,0}\}$ can not be covered using less than 11 segments from P_clause_i .

All points C_clause_i can not be covered with less than 12 segments from P_clause_i .

Proof of no cover with less than 12 segments. There is independent set of 12 points in $C_clause_i \supseteq \{x_{i,0}, y_{i,0}, z_{i,0}, l_{i,0}, p_{i,0}, q_{i,0}, u_{i,0}, v_{i,0} = l_{i,1}, p_{i,1}, q_{i,1}, u_{i,1}, v_{i,1}\}$.

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

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

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

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

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

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

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

Therefore C_clause_i must be covered with at least $3 + 4 + 4 = 11$ segments.

3.2.2.4. Summary

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

Lemma 3.2.10 Robustness to 1/2-extensions. *For every segment $s \in \mathcal{P}$, s and $s^{+1/2}$ cover the same points from \mathcal{C} .*

3.2.3. Proofs of construction Lemma 3.2.1

Lemma 3.2.11 *Given an instance of MAX-(3,3)-SAT of size n with optimal solution k . For instance of geometric cover, constructed according to Lemma 3.2.1, there exists a solution of weight $15n - k$.*

Proof. Let's name the assignments of the variables in MAX-(3,3)-SAT instance, that achieve the optimal solution, $y_1, y_2 \dots y_n$, Let's cover every variable gadget with solution described in Lemma 3.2.2, in the i -th gadget choosing the set of segments responsible for the value of y_i (true - x_i^{true} or false - x_i^{false}).

Cover every satisfied clause gadget with solution described in Lemma 3.2.7 and unsatisfied clause gadget with solution from Lemma 3.2.8.

This solution uses $3n + (11m + (m - k)) = 15n - k$ segments.

Lemma 3.2.12 *Given an instance of MAX-(3,3)-SAT of size n , and solution of size w to the instance of geometric cover, constructed according to Lemma 3.2.1, there exists a solution to MAX-(3,3)-SAT of size at least $15n - w$.*

Proof. Among $x_i^{true} \cup x_i^{false}$, we need to use at least 3 segments (Lemma 3.2.3). If we have chosen both segments (c_i, g_i) and (f_i, h_i) , then we have used at least 4 segments (Lemma 3.2.4).

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 (at least 2) clauses. If we have chosen none of the segments, choose any value.

To cover $\bigcup_{1 \leq i \leq n} C_variable_i$ 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) .

Among the segments responsible for the clause $C_i = x \vee y \vee z$ we need to use at least 11 segments (Lemma 3.2.9) and if we can cover it with 11 segments, then we have earlier chosen segment responsible for the value of variable x, y or z that satisfies C_i .

So we have at least 11 segments for satisfied clauses and at least 12 segments for unsatisfied clauses, so we cover it with at least $11n + b$ segments, where b is number of clauses where none of the variables x, y, z were chosen. If the segment responsible for value of x was taken, but this variable is set to have different value, then we have chosen segments for both x and $\neg x$ for this variable, so "we cheated" and this maybe clause is not met, but we assigned the value for this x_i that meets the most clauses, so for each of such "cheated" variables, at most one of the clauses isn't met.

So there are at most $a + b$ unsatisfied clauses in this instance, so we have shown the assignment with at least $n - (a + b)$ satisfied clauses.

$$w \geq 3n + a + 11n + b = 14n + a + b$$

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

3.2.3.1. Proof of Lemma 3.2.1

Given an instance of MAX-(3,3)-SAT of size n with optimal result k . Let's construct an instance of geometric cover, constructed in aforementioned manner.

Given the Lemma 3.2.11, we know the optimal solution for the constructed geometric cover is at most $15n - k$ and since the k is optimal solution for MAX-(3,3)-SAT, then according to Lemma 3.2.12 there doesn't exist a solution with cost less than $15n - k$.

3.3. Weighted segments

3.3.1. FPT for weighted segments with δ -extensions

Theorem 3.3.1 (FPT for weighted segment cover with δ -extensions). *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 a parameter k , runs in time $f(k) \cdot (nm)^c$ for some computable function f and constant c , and outputs a subfamily $\mathcal{R} \subseteq \mathcal{P}$ such that $|\mathcal{R}| \leq k$ and $\mathcal{R}^{+\delta}$ covers all points in \mathcal{C} .*

To solve this problem we will introduce kernel for slightly different problem: Weighted segment cover of points and segments. In shortcut: WSCPS.

Lemma 3.3.1 (Algorithm for kernel of WSCPS). *There exists an algorithm that given a family \mathcal{P} of n weighted segments (in any direction), a set of m_1 points \mathcal{C}_1 and m_2 segments \mathcal{C}_2 and a parameter k , runs in time $f(k) \cdot g(m_1, m_2) \cdot n^c$ for some computable functions f, g and constant c , and outputs a subfamily $\text{sol} \subseteq \mathcal{P}$ such that $|\mathcal{R}| \leq k$ and \mathcal{R} covers all points in \mathcal{C}_1 and all segments in \mathcal{C}_2 .*

Proof Only sketch for now.

We can compute dynamic programming $dp(A, B, z)$ – the best cost to cover at least whole segment A, B using at most z segments. A, B are all interesting points – ends of any segment given on the input or points given on the input. We can compute it in polynomial time.

Then we can create a new double weighted set (original weight, number of used segments from \mathcal{P}) – \mathcal{P}_2 that has only segments which never cover partially any segment from \mathcal{C}_2 (covers the whole segment or doesn't cover at all). In such \mathcal{P}_2 we can find solution \mathcal{R} where any 2 segments have empty intersection (don't cover each other and don't meet at the ends). Because if we had such solution, we can merge these two segments and such segment there's also in \mathcal{P}_2 .

In that case we can find kernel of \mathcal{P}_2 of size $k \cdot (m_1 + 2m_2)^2$, because we only need to take the best weight covering some subset of $\mathcal{C}_1 \cup \mathcal{C}_2$.

Lemma 3.3.2 Kernel in WSCPS. *TODO: formulate it properly*

For segment cover, there is a kernel of size $f(k)$ in WSCPS.

Claim 3.3.1 *If there are more than k lines with at least $k + 1$ points on them, then they can't be covered with k segments.*

Claim 3.3.2 *If there is more than k^2 points that don't lie on any line with more than k points on it, then they can't be covered with k segments.*

Claim 3.3.3 *For every long line L (with more than k points on them) we can choose $f(k)$ points on them, that if we cover all of these points with at most k segments, then the rest of the points with δ -extensions will be covered by segments in the direction of line L .*

Proof of Lemma 3.3.2. After applying the previous lemmas, we have at most $k^2 + k \cdot f(k)$ points that can be covered in any direction and for the rest of the points we can draw at most $k \cdot f(k)$ segments along their respective long lines that have to be covered by segments after δ -extensions.

Then we extend every available segment by δ -extension and we achieve the kernel in WSCPS for this instance of problem.

Lemma 3.3.3 *If all the points are covered with k segments and the biggest $2(1 + 1/\delta)^{k+1}$ spaces between points are filled, the whole segment is filled after δ -extensions of these segments.*

Proof. Let's name the $2(1+1/\delta)^{k+1}$ -st biggest space between points as y . We have guarantee that all segments of length $x > y$ are covered without δ -extensions.

Let's take one space between points that is not covered before δ -extension and we will prove it will be covered after δ -extensions. Let's assume it isn't.

This space has length x . Since it's uncovered, $x \leq y$.

Let's take side where the sum of lengths of segments covering the points is greater (left or right). Without loss of generality, let us assume it's right.

There are at most k segments to the right of this space between points. Name their lengths $l_1, l_2 \dots l_k$. If the point is covered in the other direction, the segment is degenerated to the point and $l_i = 0$. Name the space between endpoints of l_i and $l_{i+1} - x_i$. Of course, x_i is uncovered space between two points, therefore $x_i \leq y$.

TUTAJ BEDZIE PEWNIE RYSUNEK Z TYMI SUPER RZECZAMI DO PRZERW

Let's write equations meaning that i -th segment doesn't cover space x after δ -expansion.

$$l_1\delta < x \leq y \Rightarrow l_1 < y/\delta$$

$$l_2\delta < x + l_1 + x_1 < 2y + y/\delta \Rightarrow l_2 < 2y/\delta + y/\delta^2$$

$$l_3\delta < x + l_1 + x_1 + l_2 + x_2 < 3y + 3y/\delta + y/\delta^2 \Rightarrow l_3 < 3y/\delta + 3y/\delta^2 + y/\delta^3$$

From this we can "guess" induction $l_i < y((1 + 1/\delta)^i - 1)$

Trivially for $l_1 < y/\delta$.

Assume that for all $j < i$:

$$l_j < y((1 + 1/\delta)^j - 1)$$

$$\begin{aligned} l_i\delta &< x + \sum_{j=1}^{i-1} (l_j + x_j) < iy \sum_{j=1}^{i-1} l_j < iy + \sum j = 1^{i-1} y((1 + 1/\delta)^j - 1) = iy - (i - 1)y + \sum j = 1^{i-1} y(1 + 1/\delta)^j = y(1 + \sum_{j=1}^{i-1} (1 + 1/\delta)^j) = y(2 + \sum_{j=1}^{i-1} (1 + 1/\delta)^j - 1) = \\ &y(\sum_{j=0}^{i-1} (1 + 1/\delta)^j - 1) = y((1 + 1/\delta)^i / (1 - (1 + 1/\delta)) - 1) = y((1 + 1/\delta)^i \delta - 1) < y((1 + 1/\delta)^i \delta - \delta) \end{aligned}$$

Of course we also know that (since we have chosen the side with greater sum of the width of segments):

$$\sum_{i=1}^k l_i \geq 1/2 \cdot y \cdot 2(1 + 1/\delta)^{k+1} = y \cdot (1 + 1/\delta)^{k+1}$$

369 But $\sum_{i=1}^k l_i < \sum_{i=1}^k y((1 + 1/\delta)^i - 1) = y((1 + 1/\delta)^{k+1}/(1 - (1 + 1/\delta)) - k) = y((1 +$
370 $1/\delta)^{k+1}\delta - k) < y(1 + 1/\delta)^{k+1}$

371 Therefore the space must have been covered after δ -expansions.

372 3.3.2. W[1]-completeness for weighted segments in 3 directions

373 **Theorem 3.3.2** *W[1]-completeness for weighted segments in 3 directions.* Consider
374 the problem of covering a set \mathcal{C} of points by selecting k axis-parallel or right-diagonal weighted
375 segments with weights from a set \mathcal{P} with minimal weight. Assuming ETH, there is no algorithm
376 for this problem with running time $f(k) \cdot (|\mathcal{C}| + |\mathcal{P}|)^{o(\sqrt{k})}$ for any computable function f .

377 We will show reduction from grid tiling problem.

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

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

381 First let's choose any ordering of n^2 elements $\{1, n\} \times \{1, n\}$ and name this sequence

382 $a_1 \dots a_{n^2}$.

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

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

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:

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

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

383 Let's define $\epsilon = 0.1$. For a point $\{x, y\} = p$ we define points $p^L = \{x - \epsilon, y\}$, $p^R = \{x + \epsilon, y\}$,
384 $p^U = \{x, y - \epsilon\}$, and $p^D = \{x, y + \epsilon\}$.

Then we define:

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

385 **Segments.** Define horizontal segments.

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

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

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

$$\text{hor}_{\text{end}_{i,t}} = (h_{i,n,t}^R, h_{i,n,n^2}^R)$$

$$\begin{aligned} \text{verbeg}_{i,t} &= (v_{i,1,1}^U, v_{i,1,t}^U) \\ \text{verend}_{i,t} &= (v_{i,n,t}^D, v_{i,n,n^2}^D) \end{aligned}$$

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

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

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

386 TODO: explain that these segments are in fact diagonal

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

387 **Lemma 3.3.4** *If there exists solution for grid tiling, then there exists solution for our con-*
388 *struction using $2(k+1)k + k^2$ segments with weight exactly $2k \cdot (k(n^2 + 1) - 2 - 2\epsilon(k-1))$.*

Claim 3.3.4 *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*

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

389 *with segments in DIAG and the rest in VER or HOR and has weight $2k \cdot (k(n^2 + 1) -$*
390 *$2 - 2\epsilon(k-1))$.*

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

392 **Lemma 3.3.5** *If there exists solution for our construction using $2(k+1)k + k^2$ segments with*
393 *weight exactly $2k \cdot (k(n^2 + 1) - 2 - 2\epsilon(k-1))$, then there exists a solution for grid tiling*

394 **Proof.** This follows from Lemma 3.3.6, because we just take which points are covered with
395 *DIAG.*

396 **Claim 3.3.5** *Points p^L, p^R, p^U, p^D cannot be covered with DIAG.*

397 **Claim 3.3.6** *Points in $H \cup \{p^L : p \in H\} \cup \{p^R : p \in H\}$ cannot be covered with VER.*

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

399 **Claim 3.3.7** *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*
400 *DIAG, then some spaces between neighbouring points were covered twice.*

401 **Claim 3.3.8** *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*
402 *covered with DIAG, then one of them had to be also covered with a segment from HOR*
403 *(VER).*

404 **Proof.** Point v_{i,j,t_2}^L had to be covered with VER from Claims 3.3.5 and 3.3.6. And every
 405 segment in VER covering v_{i,j,t_2}^L , covers also v_{i,j,t_1}^L .

406 **Lemma 3.3.6** *If there exists solution for our construction with weight at most (exactly) $2k \cdot$
 407 $(k(n^2 + 1) - 2 - 2\epsilon(k - 1))$, then for every i, j there must be exactly one t such that $h_{i,j,t}$ ($v_{i,j,t}$)
 408 is covered with $DIAG$ and moreover if h_{i,j,t_1} and $h_{i,j+1,t_2}$ are uncovered, then $math_h(t_1, t_2)$.
 409 Analogically for v .*

410 **Proof.** Only k^2 points can be covered only in $DIAG$, the rest has to be covered with
 411 $VER \cup HOR$. Therefore every result must be at least $ALL_LINES - 2k^2\epsilon$, because only
 412 $2k^2$ spaces of length ϵ can be uncovered in this axis.

413 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
 414 between h_{i,j,t_1}^R and $h_{i,j+1,t_2}^L$, so $math_h(t_1, t_2)$ must be true.

415 3.3.3. What is missing

416 We don't know FPT for axis-parallel segments without δ -extensions.

417 Chapter 4

418 Geometric Set Cover with lines

419 4.1. Lines parallel to one of the axis

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

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

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

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

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

431 Non-weighted setting - Konig theorem + max matching

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

433 4.2. FPT for arbitrary lines

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

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

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

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

440 4.3. APX-completeness for arbitrary lines

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

444 Then for every edge in $(v, w) \in E(G)$ we put a point on crossing of lines for vertices v
445 and w . They are not pararell, so there exists exactly one such point and any other line don't
446 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.

471 Chapter 5

472 Geometric Set Cover with polygons

473 5.1. State of the art

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

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

479 Covering points with squares is W[1]-hard [Marx, 2005]. It can be proven that assuming
480 *SETH*, there is no $f(k) \cdot (|\mathcal{C}| + |\mathcal{P}|)^{k-\epsilon}$ time algorithm for any computable function f and
481 $\epsilon > 0$ that decides if there are k polygons in \mathcal{P} that together cover \mathcal{C} , *Theorem 1.9* in [Marx
482 and Pilipczuk, 2015].

⁴⁸³ Chapter 6

⁴⁸⁴ Conclusions

485 Bibliography

- 486 [Har-Peled and Lee, 2009] Har-Peled, S. and Lee, M. (2009). Weighted geometric set cover
487 problems revisited. *Journal of Computational Geometry*, 3.
- 488 [Håstad, 2001] Håstad, J. (2001). Some optimal inapproximability results. *J. ACM*,
489 48(4):798–859.
- 490 [Li and Jin, 2015] Li, J. and Jin, Y. (2015). A PTAS for the weighted unit disk cover problem.
491 *CoRR*, abs/1502.04918.
- 492 [Marx, 2005] Marx, D. (2005). Efficient approximation schemes for geometric problems? In
493 Brodal, G. S. and Leonardi, S., editors, *Algorithms – ESA 2005*, pages 448–459, Berlin,
494 Heidelberg. Springer Berlin Heidelberg.
- 495 [Marx and Pilipczuk, 2015] Marx, D. and Pilipczuk, M. (2015). Optimal parameterized algo-
496 rithms for planar facility location problems using voronoi diagrams. *CoRR*, abs/1504.05476.