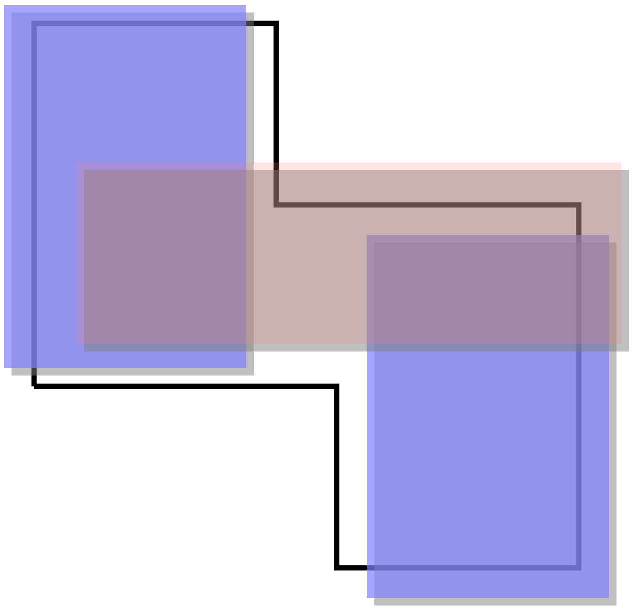
Covering Rectilinear Polygons by Rectangles

Jaromił Najman



Master Thesis in Mathematics

Institute for Operations Research
Rheinisch-Westfälische Technische Hochschule Aachen
Fields of Research in Approximation Algorithms
Prof. Dr. Marco Lübbecke

September 2014

Table of Contents

§1 Introduction

In this Master Thesis we consider the problem of covering rectilinear polygons by the minimum number of axis-parallel rectangles. This problem finds applications in the fabrication of DNA chip arrays [Hannenhalli et al. 2002], in VLSI design, where the rectilinear polygon is a chip that has to be covered by a huge number of rectangular transistors, see Fig. 1 ["Diopsis", Licensed under Creative Commons Attribution]. Other applications are data compression and in particular image compression, where large rectangular areas with the same color can be compressed into one pixel.

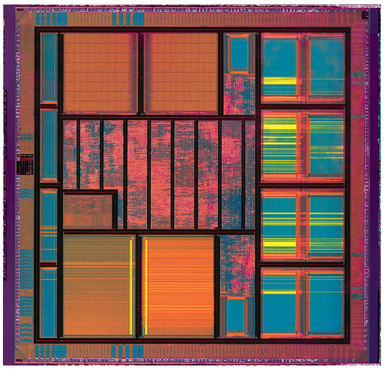


Fig. 1: VLSI chip with thousands of integrated transistors.

First we will need some essential definitions for understanding the problem.

— *Definitions* —

(1.1) Definition (Rectilinear Polygon)
A **rectilinear polygon** R is a finite set of unit squares, called pixels, on a two dimensional integer grid.

(1.2) Definition (Rectangle)
A **rectangle** is a rectangle in the usual sense formed by the union of unit squares (pixels).

The boundary of a rectilinear polygon R consists of horizontal and vertical **edges**.

(1.3) Definition (Polygon vertices)
Let R be a rectilinear polygon. A **vertex** of R is the intersection of a horizontal and vertical edge of R.

A **normal convex vertex** is the intersection of exactly two edges which form a 90° angle inside R.
A **concave vertex** is the intersection of exactly two edges which form a 270° inside R.

(1.4) Definition (Hole)
We define a **hole** in a rectilinear polygon R, as a hole in the common sense, i.e. a finite set of connected unit squares inside of R, that are not a subpart of R.

(1.5) Definition (Rectangle Cover)
A **rectangle cover** for a given rectilinear polygon R is a finite set of rectangles completely contained in R, whose union is equal to R. The rectangles may overlap.
A **minimum rectangle cover** is a rectangle cover containing the minimum number of rectangles needed to cover R.

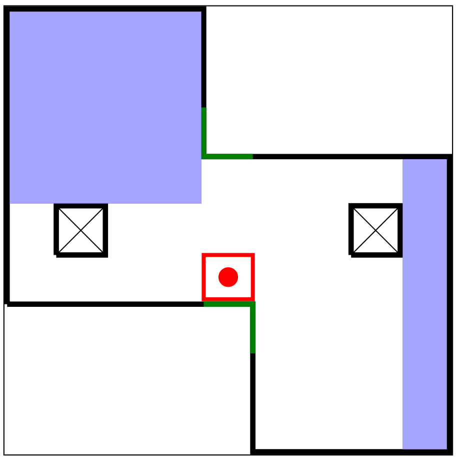


Fig. 2: An example showing a rectilinear polygon R with edges in black, two holes with a large X, one of the pixels in red, two rectangles in blue, and 8 vertices. The two vertices shaded green are concave and the other six are convex.

We can now define the **Rectangle Cover Problem**:

input: rectilinear polygon R
output: minimum number of rectangles, completely contained in R, needed to cover R completely

We denote the optimal number of needed rectangles to cover a given rectilinear polygon R by $\theta(R)$.

In the rectangle cover problem we only consider the number of needed rectangles and not their areas, since the rectangles may overlap. Thus picking the largest rectangles results in a lower or the same number of rectangles needed. Hence we can restrict our attention without loss of generality to inclusionswise maximal rectangles.

(1.6) Remark
Notice that every rectangle contained in the optimal solution covers a unique pixel that none of the other rectangles covers.

Moreover we will need the basic definition of an approximation algorithm and some basics from the fields of graph theory. More information in the fields of approximation algorithms can be found in [Garey and Johnson 1979] and for the fields of graph theory in [Ahuja,Magnanti 1993].

(1.7) Definition (Approximation Algorithm)
An approximation algorithm of factor c for a minimization problem Π

1. computes for every instance $I \in \Pi$ a feasible solution with value $\mathcal{A}(I) \leq c \cdot \text{OPT}(I)$
2. and has a running time that is polynomial in its input size $|I|$.

(1.8) Definition (Undirected Graph)
An **undirected graph** G is an ordered pair (V, E) consisting of a set of **vertices** V and a set E of **edges**, where every edge is contained in $V \times V$. Two vertices v_1, v_2 are called **adjacent** if $e = (v_1, v_2) \in E$.

(1.9) Definition (Clique)
A **clique** in a given graph $G = (V, E)$ is a subset $S \subseteq V$ where every two vertices are adjacent.

(1.10) Definition (Independent Set)
An **independent set** in a given graph $G = (V, E)$ is a subset $I \subseteq V$ where none of the vertices are adjacent.

— *Structure and current knowledge* —

In the first part of the thesis we will think of the polygon R as a union of finitely many pixels, i.e. 1×1 squares, also called polyomino in other literature. R can be associated with a visibility graph G described in [Maire 1994; Motwani et al. 1989a;

1989b; Schrijver 2003]: The vertex set of G is the set of pixels of R and two vertices are adjacent in G if and only if their associated pixels in R can be covered by a common rectangle. Notice that rectangles then correspond to cliques in G.
A set of pixels in R, no two of which can be covered by a common rectangle, is called an *antirectangle*. In G this corresponds to the problem of finding an *independent set*, we denote its maximum size by $\alpha(G)$ and by **(1.6)** it is an obvious lower bound on $\theta(G)$.

It was originally conjectured by Chvátal that $\theta = \alpha$. This is true for convex polygons [Chaiken et al. 1981] and a number of special cases. Szemerédi gave an example of $\theta \neq \alpha$, see Fig. 2, and Erdös then asked whether $\frac{\theta}{\alpha}$ is bounded by a constant. The best proven lower bound is $\frac{\theta}{\alpha} \geq \frac{8}{7}$ [Chaiken et al. 1981], which we are going to improve in the second section of the thesis.

The rectangle cover problem is $\mathcal{NP}$-hard for polygons with holes and even for those without holes [Masek 1979; Culberson and Reckhow 1994] and it is Max $\mathcal{SNP}$-hard [Berman and DasGupta 1997], what implies that there is no polynomial time approximation scheme for the rectangle cover problem. The rectangle cover problem can be interpreted as a special set cover problem, for which the standard greedy algorithm gives us an $\mathcal{O}(\log n)$ approximation, where n is the number of pixels in the polygon. [Anil Kumar and Ramesh 2003] improved this factor to $\mathcal{O}(\sqrt{\log n})$, where n is the number of edges in R. The open question remained:

Is there a constant factor approximation algorithm
for the rectangle cover problem?

asked in [Bern and Eppstein 1997], that we will answer in the third section by reanalysing the *Partition/Expand* algorithm from [Franzblau 1989], which she conjectured to be a factor 3 approximation algorithm.

§2 Lower bound improvement for $\frac{\theta}{\alpha}$

It was originally conjectured by Chvátal that $\theta = \alpha$, which is proven to be true for convex polygons [Chaiken et al. 1981]. Szemerédi gave an example with $\theta \neq \alpha$, see Fig. 3. The grey shaded rectangles have to be included in every optimal cover since they are the only ones covering the green pixels. The red pixels build an odd cycle of length 5. It implies that at most two pixels of this odd cycle can be independent but at least three rectangles are needed to cover the cycle. It implies that $\frac{\theta}{\alpha} = \frac{8}{7}$. There are several other examples where the case $\theta \neq \alpha$ is true, some of them not containing an odd cycle, see Fig. 4 [Chaiken et al. 1981] and others containing larger odd-length cycles, see Fig. 5. An example with $\frac{\theta}{\alpha} \geq \frac{21}{17} - \epsilon$ is mentioned in [Chaiken et al. 1981] but this example cannot be reconstructed from [Chaiken et al. 1981], and thus cannot be verified. The best proven lower bound is $\frac{\theta}{\alpha} \geq \frac{8}{7}$, which we are going to improve in this section.

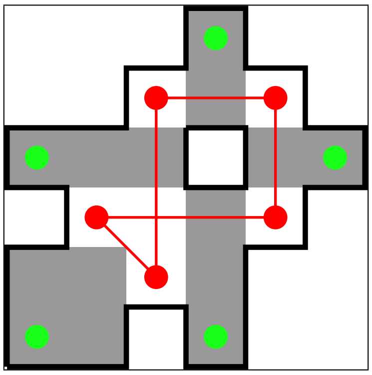


Fig. 3: Szemerédi's original counter example to $\theta \neq \alpha$.

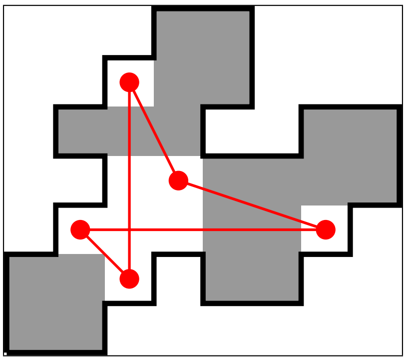


Fig. 4: Example with $\theta \neq \alpha$ without a hole.

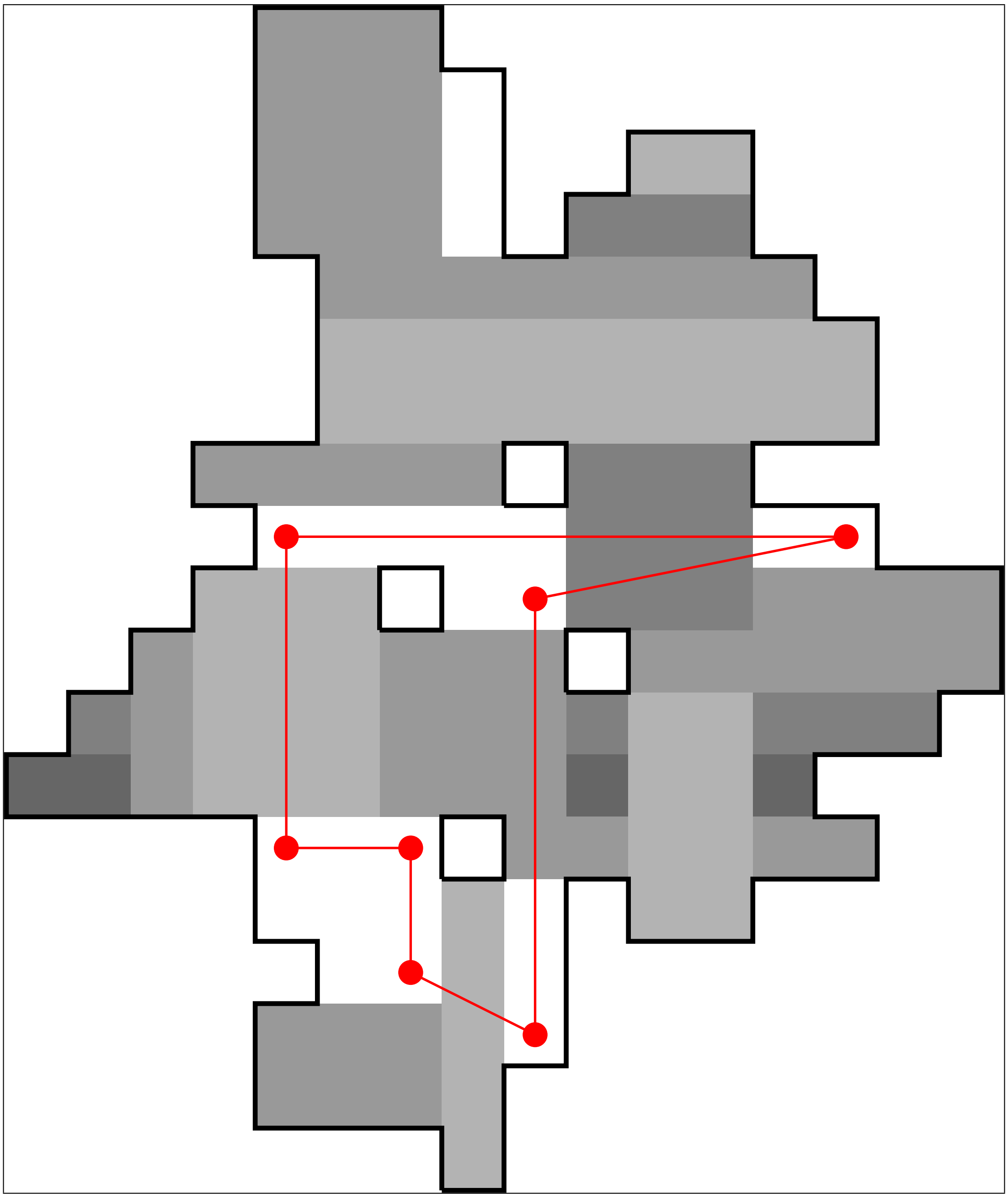


Fig. 5: A larger example with an odd cycle of length 7 with $\frac{\theta}{\alpha} = \frac{19}{18}$. The shaded rectangles represent all other rectangles used in the optimal cover, that do not contain the red pixels.

For this purpose we need to take a closer look at Szemerédi's example. If we extend the right horizontal stripe to the right and paste a copy of exact the same polygon to it, we recieve a new polygon which now contains two odd cycles of length 5, see Fig. 6. The rectangles that have to be contained in every optimal cover are shaded grey. We can see that those two polygons now share one common rectangle that has to be contained in every optimal rectangle cover, shaded darker grey here. For the optimal rectangle cover it implies that $\theta \geq 9 + 2 \cdot 3 = 15$ and for the independent set it implies that $\alpha \leq 9 + 2 \cdot 2 = 13$.

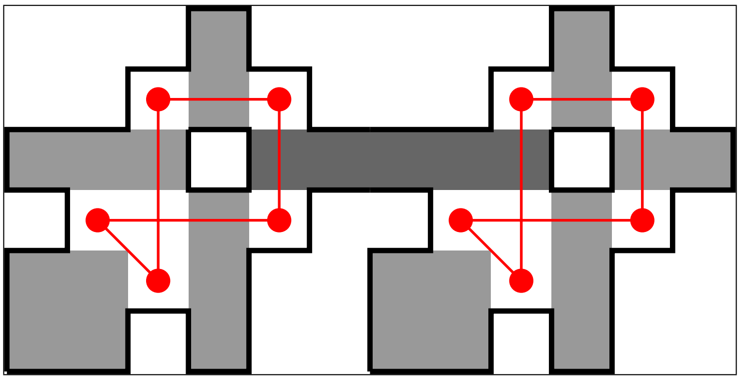


Fig. 6: The new constructed polygon out of 2 original Szemerédi's polygons. It is now $\frac{\theta}{\alpha} = \frac{15}{13} > \frac{8}{7}$.

Since $\frac{15}{13} > \frac{8}{7}$, we already improved the lower bound for $\frac{\theta}{\alpha}$ but we aren't finished yet. If we repeat the process and paste one more of the same polygon on the right of the new constructed one, we recieve a new polygon with three cycles of length 5, see Fig. 7. The grey shaded rectangles again represent the rectangles that have to be contained in every optimal cover. This implies $\theta \geq 13 + 3 \cdot 3 = 22$ and $\alpha \leq 13 + 3 \cdot 2 = 19$ and since $\frac{22}{19} > \frac{15}{13} > \frac{8}{7}$, we improved the bound again.

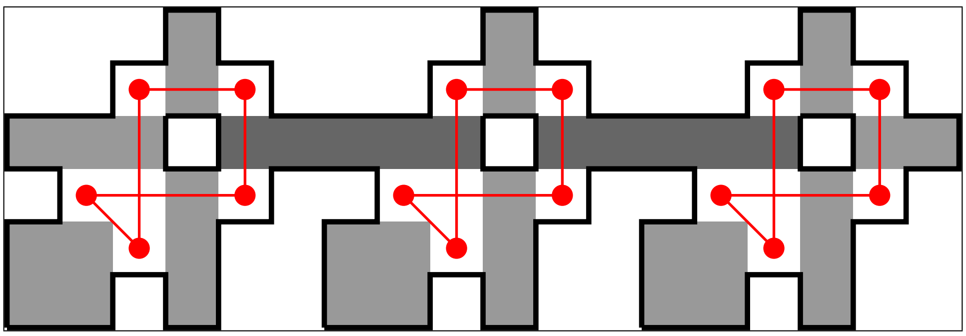


Fig. 7: The new constructed polygon out of 3 original Szemerédi's polygons.

To recieve a general bound we define a sequence $a_{n\in\mathbb{N}}$ of polygons that are constructed exactly the way we just did for all $n \in \mathbb{N}$. We use the $\frac{\theta}{\alpha}$ ratio as the value of each member of the sequence. The sequence has to start at $\frac{8}{7}$, so $a_1 = \frac{8}{7}$, and increase by 7 in the numerator and by 6 in the denominator. The sequence with these properties is $a_n = \frac{7n+1}{6n+1}, n \in \mathbb{N}$. The following lemma shows that this sequence is strictly monotonically increasing.

(2.1) Lemma
The sequence

$$a_n = \frac{7n+1}{6n+1}, n \in \mathbb{N}$$

is strictly monotonically increasing.

Proof
We show $a_{n+1} > a_n$.

$$\begin{aligned} \frac{7(n+1)+1}{6(n+1)+1} &> \frac{7n+1}{6n+1} \\ \Leftrightarrow (7(n+1)+1)(6n+1) &> (7n+1)(6(n+1)+1) \\ \Leftrightarrow (7n+8)(6n+1) &> (7n+1)(6n+7) \\ \Leftrightarrow 42n^2+55n+8 &> 42n^2+55n+7 \end{aligned}$$

This is obviously true for all $n \in \mathbb{N}$. □

Since the sequence is strictly monotonically increasing, the $\frac{\theta}{\alpha}$ ratio increases in our construction, thus the construction is correct in the sense of recieving a better lower bound. In the following lemma we show that the constructed sequence is bounded and therefore its limes is a new larger lower bound for $\frac{\theta}{\alpha}$.

(2.2) Lemma
It is

$$\lim_{n\to\infty} a_n = \frac{7}{6}.$$

Proof
It is

$$a_n = \frac{7n+1}{6n+1} = \frac{7+\frac{1}{n}}{6+\frac{1}{n}} \xrightarrow[n\to\infty]{} \frac{7}{6}.$$

We now obtained a new proven bound $\frac{\theta}{\alpha} \geq \frac{7}{6}$ but there is still space for improvement since we only expanded our example horizontally. What happens if we expand it vertically, i.e. we paste the whole construct once again on the top of itself, see Fig. 8 for an example with $n = 3$ and a pasted itself on its top. We count each vertical level of the construct with $m \in \mathbb{N}_0$ where $m = 0$ stands for the constructs we build before, i.e. only the horizontally expanded ones. For this example it is $\frac{\theta}{\alpha} = \frac{41}{35} > \frac{22}{19}$, where $\frac{\theta}{\alpha} = \frac{22}{19}$ is the fraction we got from our previous example, see Fig. 7.

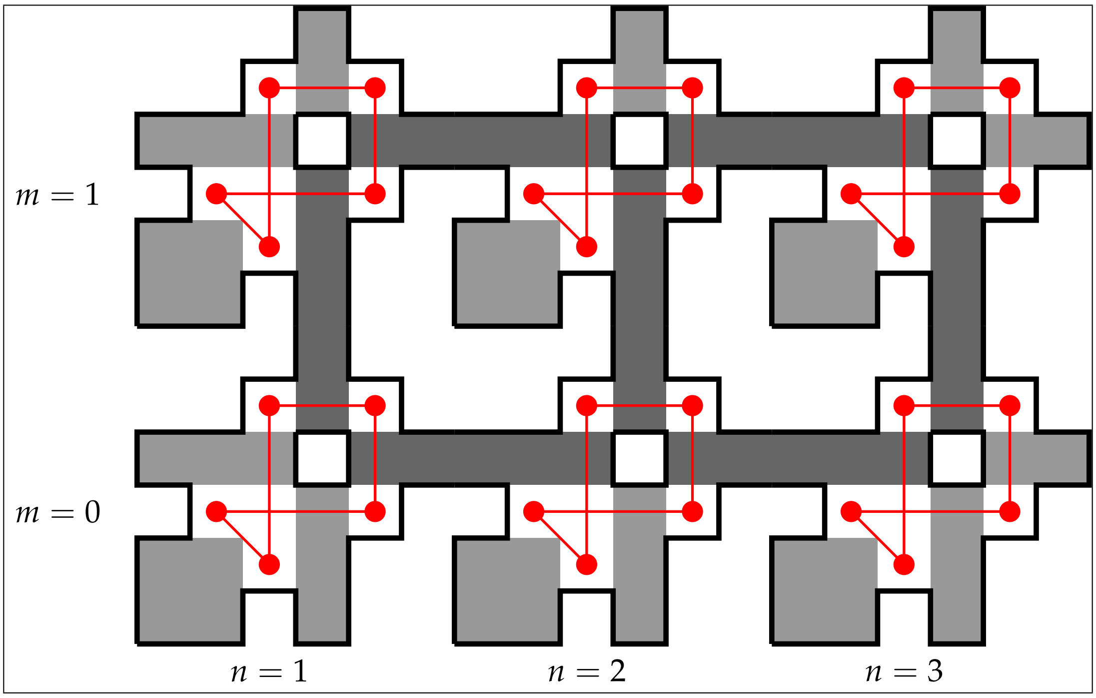


Fig. 8: The new constructed polygon out of 6 original Szemerédi's polygons for $n = 3$ and $m = 1$.

We can formulate this as a sequence again but this time we have got two variables $n \in \mathbb{N}$ and $m \in \mathbb{N}_0$, so we define the new sequence $a_{n \in \mathbb{N}, m \in \mathbb{N}_0}$. In addition to our first sequence we need to somehow express the vertical expansion of the construct in the new one. We recieve the same horizontal construct once again minus the n shared rectangles between two vertical levels. This leads to the sequence

$$a_{n,m} = \frac{7n + 1 + m(7n + 1 - n)}{6n + 1 + m(7n + 1 - n)} = \frac{7n + 1 + m(6n + 1)}{6n + 1 + m(5n + 1)}, n \in \mathbb{N}, m \in \mathbb{N}_0.$$

With the next two lemmas we improve this lower bound once again.

(2.3) Lemma
The sequence

$$a_{n,m} = \frac{7n+1+m(6n+1)}{6n+1+m(5n+1)}, n \in \mathbb{N}, m \in \mathbb{N}_0$$

is strictly monotonically increasing.

Proof
To show the monoty of the sequence, we will need to show 3 cases.
1. $a_{n+1,m} > a_{n,m}$

$$\frac{7(n+1)+1+m(6(n+1)+1)}{6(n+1)+1+m(5(n+1)+1)} > \frac{7n+1+m(6n+1)}{6n+1+m(5n+1)}$$
$$\Leftrightarrow m^2+2m+1 > 0.$$

This is true for $m \in \mathbb{N}_0$.

2. $a_{n,m+1} > a_{n,m}$

$$\frac{7n+1+(m+1)(6n+1)}{6n+1+(m+1)(5n+1)} > \frac{7n+1+m(6n+1)}{6n+1+m(5n+1)}$$
$$\Leftrightarrow n^2 > 0.$$

This is true for $n \in \mathbb{N}$.
3.$a_{n+1,m+1} > a_{n,m}$

$$\frac{7(n+1)+1+(m+1)(6(n+1)+1)}{6(n+1)+1+(m+1)(5(n+1)+1)} > \frac{7n+1+m(6n+1)}{6n+1+m(5n+1)}$$
$$\Leftrightarrow m^2+3m+n^2+n+2 > 0.$$

This is true for $n \in \mathbb{N}, m \in \mathbb{N}_0$. □

The new construction increases the $\frac{\theta}{\alpha}$ ratio as well by expanding the construct horizontally or vertically. We now need to show that it is bounded.

(2.4) Lemma
It is

$$\lim_{m\to\infty} \lim_{n\to\infty} a_{n,m} = \frac{6}{5}.$$

Proof
It is

$$\begin{aligned} a_{n,m} &= \frac{7n+1+m(6n+1)}{6n+1+m(5n+1)} = \frac{7+\frac{1}{n}+6m+\frac{m}{n}}{6+\frac{1}{n}+5m+\frac{m}{n}} \\ &\xrightarrow[n\to\infty]{} \frac{7+6m}{6+5m} = \frac{\frac{7}{m}+6}{\frac{6}{m}+5} \\ &\xrightarrow[m\to\infty]{} \frac{6}{5} \end{aligned}$$

□

With the horizontally and vertically expanded construct and the proved lemmas we improved the lower bound for $\frac{\theta}{\alpha}$ to $\frac{6}{5}$.

We tried to improve it once again by constructing similar examples but we were not able to find an example which could have been treated in a similar way and was simultaneously increasing the bound. We strongly assume that the mentioned bound $\frac{21}{17} - \epsilon$ [Chaiken et al. 1981] is constructed similarly. We can as well say that we proved a bound of $\frac{6}{5} - \epsilon$ for a small enough ϵ, since the instances are always of finite size. Probably the basic construct used in [Chaiken et al. 1981] is a different and a larger one, thus not the Szemerédi's $\frac{8}{7}$ example we used.

§3 Franzblau's algorithm reanalysed

In this section we will reanalyse the algorithm *Partition/Extend* from [Franzblau 1989] to find a better approximation ratio. We will repeat the algorithm and some of the most needed definitions here, a more detailed description can be found in [Franzblau 1989].

— *Franzblau's algorithm* —

(3.1) Definition (Chord (Franzblau 1989))
A **horizontal chord** of a rectilinear polygon R is a horizontal line segment completely contained in R, such that at least one endpoint is a concave vertex, both endpoints lie on the boundary, and no other point lies on the boundary. Every concave vertex determines a unique horizontal chord.

There are 3 types of chords Franzblau uses, which we as well need in our analysis. A **type 1 chord** contains exacly one concave vertex, a **type 2 chord** contains two concave vertices with opposite orientations and a **type 3 chord** connects two concave vertices with the same orientation, see Fig. 9.

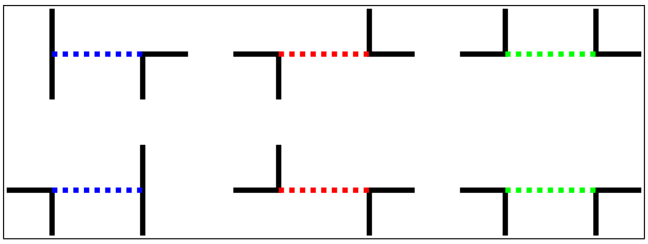


Fig. 9: Type 1 chord, type 2 chord and type 3 chord from left to the right.

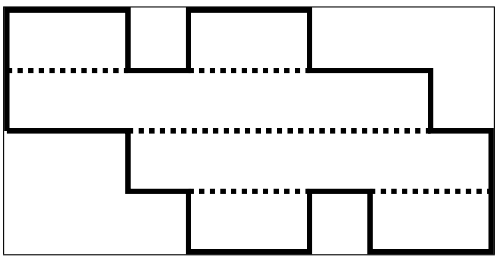


Fig. 10: Horizontal partition in the first step of the algorithm producing 6 rectangles.

We now present Franzblau's *Partition/Extend* algorithm:

ALGORITHM: *Partition/Extend*

Input: Rectilinear polygon R.
Output: Rectangle cover for R.

1. *Partition* R into rectangles by cutting along each horizontal chord of R, see Fig. 10.

2. *Extend* each rectangle vertically inside R until it is vertically maximal, i.e., touches the boundary on top and bottom. Delete any repeated rectangles, see Fig. 11.

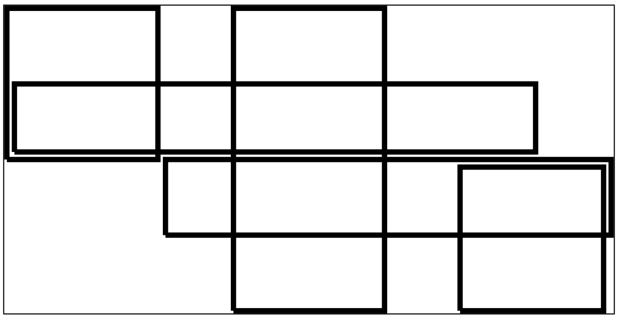


Fig. 11: The determined rectangle cover after the *Extend* part of the algorithm. The rectangles are shown slightly smaller for better understanding.

We call the number of rectangles obtained by horizontal partitioning only (step 1 of the algorithm) $p = p(R)$ for a rectilinear polygon R. Let $\overline{p} = \overline{p}(R)$ be the number of rectangles obtained by the algorithm *Partition/Extend* for a rectilinear polygon R. Franzblau showed the inequality $p \leq 2\theta + h - 1$, where h is the number of holes in a given rectilinear polygon R and this bound is tight, see Fig. 12.

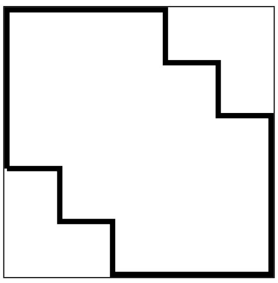


Fig. 12: An example where the inequality $p \leq 2\theta + h - 1$ is actually tight. Here is $\theta = 3$, there are no holes and $p = 5$.

Franzblau used another inequality for proving the above mentioned, namely

$$2p \leq 2\theta + c_1 + c_3,$$

where c_1 is the number of type 1 chords and c_3 is the number of type 3 chords in the rectilinear polygon R. Since it is $\overline{p} \leq p$ it holds that

$$2\overline{p} \leq 2\theta + c_1 + c_3.$$

Franzblau herself already encountered the problem created by type 3 chords. Namely if there are many of them creating the same rectangle in the *Extend* part of the algorithm, the above inequality is pretty weak, since we count too many type 3 chords regarding the $\overline{p}$ in the inequality, see Fig 13.

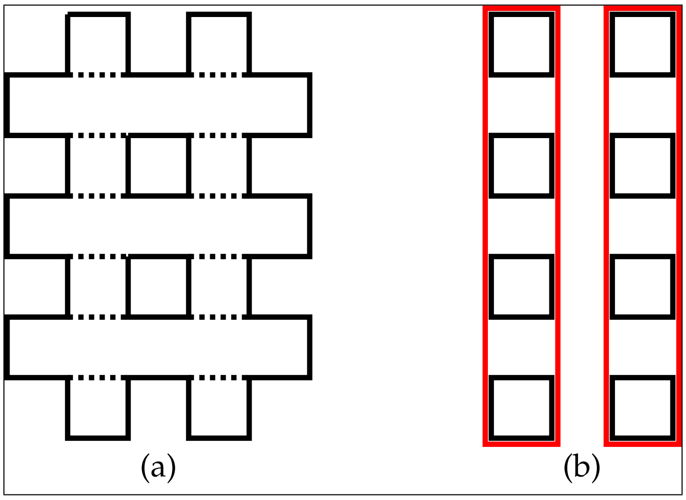


Fig. 13: *(a)*An other example where the inequality $p \leq 2\theta + h - 1$ holds.
*(b)*We see that all the dotted left type 3 chords will produce the same vertically expandable rectangle in the second part of the algorithm, the same with all the dotted right type 3 chords, so counting them all in the $2\overline{p} \leq 2\theta + c_1 + c_3$ inequality seems too much.

— Analysis of the chords —

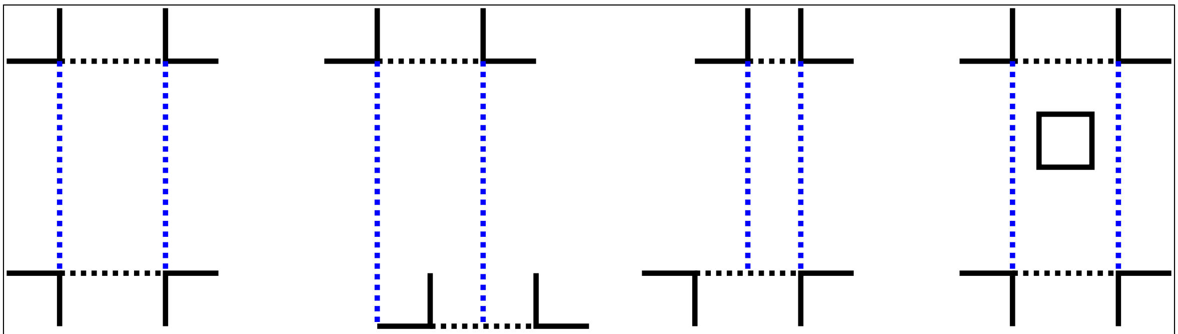
To improve this inequality we need to observe some properties type 3 chords have. Let $\mathcal{C}_3$ be the set of all type 3 chords in a given rectilinear polygon R.

(3.2) Definition

Two chords are called **alligned** if their right and left endpoints have the same horizontal coordinate, i.e. the chords have exactly the same width and there is no hole between them, i.e. they can be both covered by exactly one rectangle, see Fig. 14.

(3.3) Definition

A **chord-set** $\mathcal{S}$ is a set of chords of the same type with all of them beeing alligned and there is no chord not in the set, that is alligned to a chord in the set, i.e. the chord-sets have always maximum cardinality, see Fig. 15.



(a) (b) (c) (d)

Fig. 14: The blue dotted lines shall help in understanding the allignments.
*(a)*Two alligned type 3 chords.
*(b)*Two type 3 chords with the same width but different right and left point coordinates, hence they are not alligned.
*(c)*Two type 3 chords with the same right point coordinates but different left point coordinates, hence they are not alligned.
*(d)*Two type 3 chords with the same left and right coordinates, but a hole is in between them so they cannot be completely covered by exactly one rectangle, hence they are not alligned.

(3.4) Definition

A **chord-family** $\mathcal{F}$ is a family containing chord-sets such that

$$\mathcal{C}_3 = \bigcup_{\mathcal{S} \in \mathcal{F}} \mathcal{S}$$

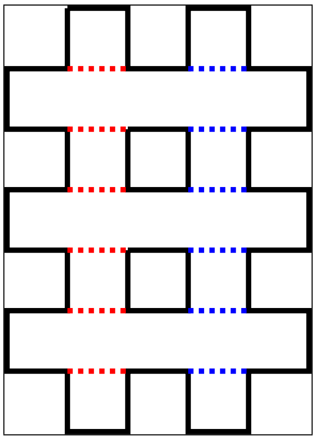


Fig. 15: The red dotted type 3 chords are all in the same chord-set $\mathcal{S}_1$ and the blue dotted type 3 chords are all in an other chord-set $\mathcal{S}_2$.

(3.5) Remark
Note that a type 3 chord can be contained in only one chord-set.

(3.6) Remark
Note that it is possible that each type 3 chord builds its own chord-set, because none of them are alligned. It is then

$$|\mathcal{F}| = |\mathcal{C}_3|$$

for the chord-family $\mathcal{F}$ of all type 3 chords.

A type 3 chord produces in the *Partition* part of the algorithm one wider and one sleeker rectangle below or above depending on the orientation of the concave vertices. Those sleeker rectangles at two alligned type 3 chords would vertically expand to the same rectangle in the second part of the algorithm. Regarding the *Extend* part of the algorithm all type 3 chords in a chord-set produce the same vertically extended rectangles and all of them except 1 will be deleted, see Fig. 13.

Let $\mathcal{F}_3$ be the chord-family of all type 3 chords in a given rectilinear polygon R. With the observations we can now replace the number of type 3 chords c_3 by the cardinality of $\mathcal{F}_3$, it is

$$2\overline{p} \leq 2\theta + c_1 + |\mathcal{F}_3|.$$

The first intuitional assumption would be $|\mathcal{F}_3| \leq \theta$ since every chord-set has to somehow be covered by at least one rectangle. This assumption was encouraged by experiments with randomly generated instances of rectilinear polygons, where $|\mathcal{F}_3|$ was always much lower than θ. This is not true in general, see Fig. 16 for an example.

Regarding this example it leads to the idea that $|\mathcal{F}_3|$ is bounded by $c \cdot \theta$ for a $c \in \mathbb{N}$, since adding new chord-sets leads to the need of more rectangles to cover them.

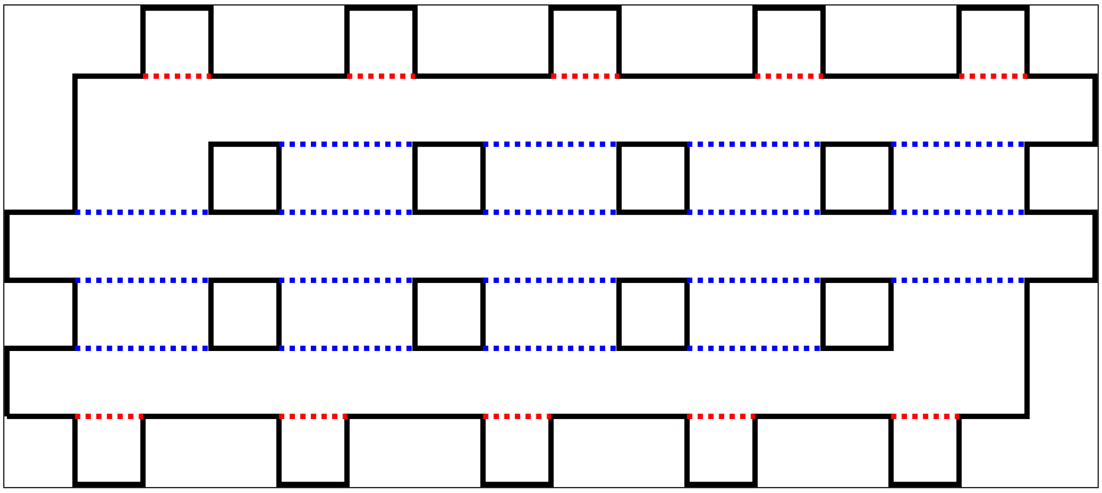


Fig. 16: An example where $\theta = 13 < 15 = |\mathcal{F}_3|$. The different chord-sets are the colored dotted lines, where every red chord builds his own set and every 3 or 4 alligned blue chords are in the same chord-set together.

In the following discussion, we will only need to consider the positioning of the type 3 chord-sets in between each other and checking how many of those mentioned sleeker vertically expanded rectangles, that go straight throught the chord, see Fig. 17, do we need to optimally cover all of the type 3 chord-sets.

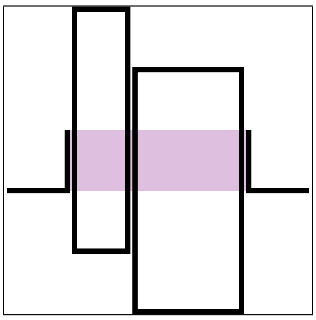


Fig. 17: The light violet area has to be covered by at least one of the mentioned sleeker vertical expanded rectangles in every optimal solution.

For this purpose we will need other observations and visualization of the chord-sets in a way we can somehow analyse. We will interpret a whole chord-set $\mathcal{S}$ as one straight horizontal line $\mathcal{S}_h$ with the left and right coordinates matching the left and right coordinates of the chords in the chord-set, see Fig. 18.

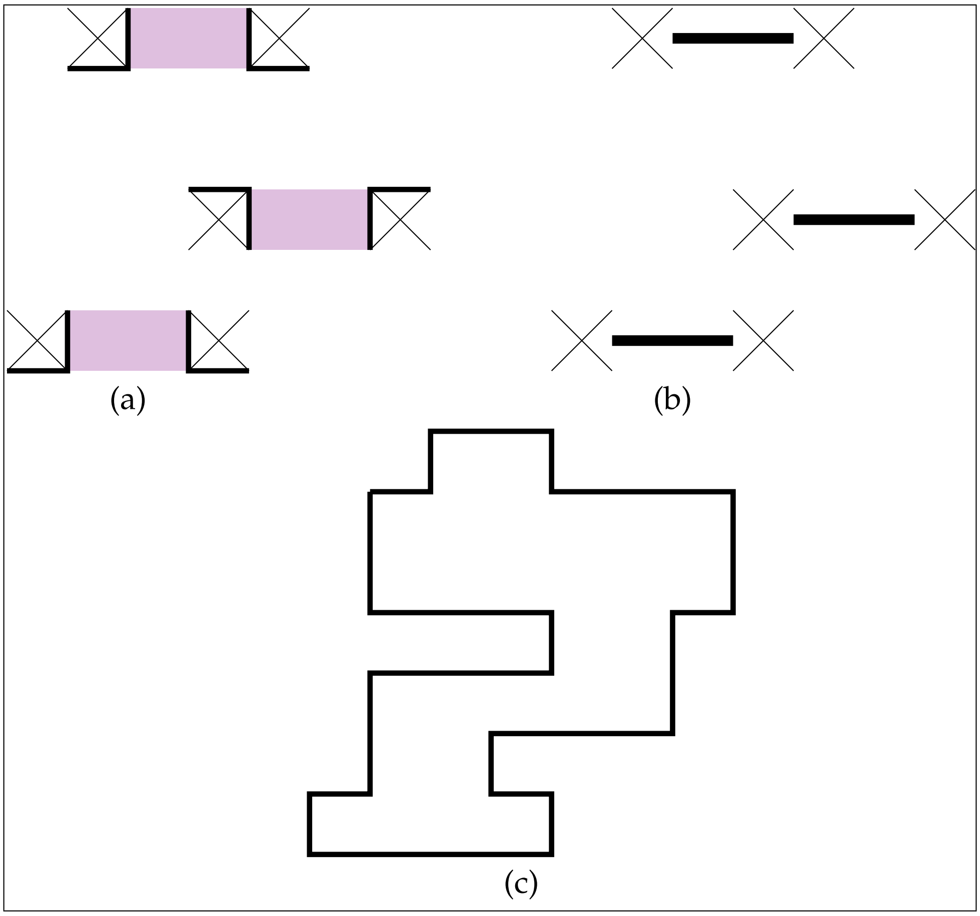


Fig. 18: Here is an example of our easier visualization of the chord-sets positions.
(a) 3 chords which build a chord-set for themselves, the big X are holes.
(b) The corresponding visualization of the chord-sets.
(c) An instance where this positioning of the chords is possible.

Notice that there are always at least two holes directly next to each chord-set, because of the way type 3 chords look like, and thus next to their corresponding horizontal lines, see Fig. 18 as well.

The only questionable data is their vertical positioning, since some chord-sets can be large and thus they expand vertically, moreover there can be chord-sets between other chord-sets, see Fig. 19.

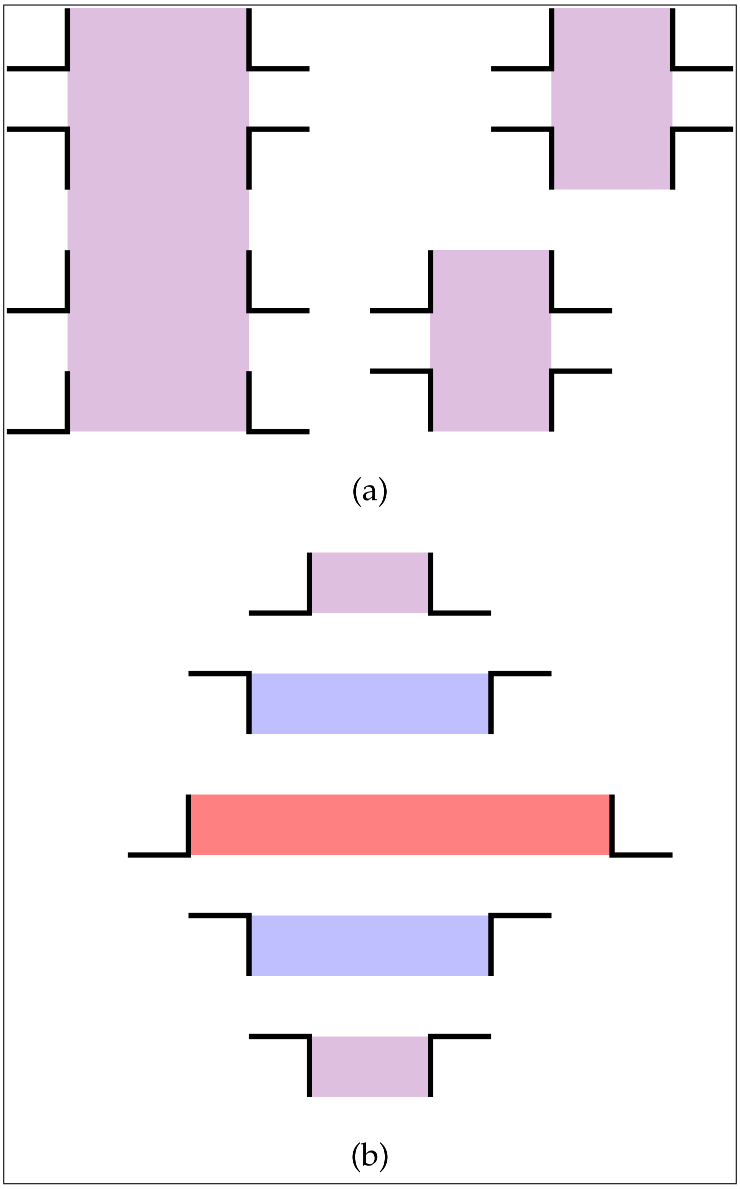


Fig. 19: Examples of larger chord-sets.
(a) Three chord-sets with cardinality >1.
(b) Three different chord-sets, but the red one is in between the blue and violet ones while the blue one is in between the violet one.

We can solve the vertical size problem easily by just putting the corresponding horizontal straight line $\mathcal{S}_h$ in the vertical middle of the chord-set $\mathcal{S}$, see Fig. 20. This way, we do not leave out any possibilities to cover the sets by rectangles and we do not create any new possibilities as well.

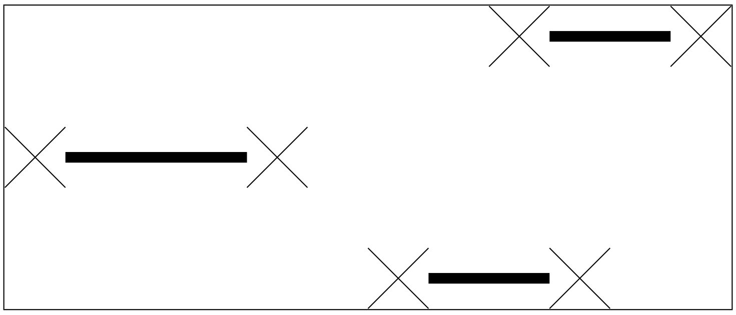


Fig. 20: The horizontal straight lines corresponding to the chord-sets in Fig. 18 *(a)*. Notice that we do not need to mark all holes since we only care about the mentioned sleeker vertical rectangles that go trought the horizontal straight lines.

To solve the problem with the chord-sets $\mathcal{A}$ beeing in between an other chord-set $\mathcal{S}$, we just position the corresponding horizontal line $\mathcal{S}_h$ above all the other horizontal lines $\mathcal{A}_h$ corresponding to the chords-sets beeing in between $\mathcal{S}$. Moreover we need to add additional holes for at least one chord of $\mathcal{S}$ under $\mathcal{A}_h$ to prevent creating new rectangle covering possibilities, see Fig. 21 and 22 for specific examples.

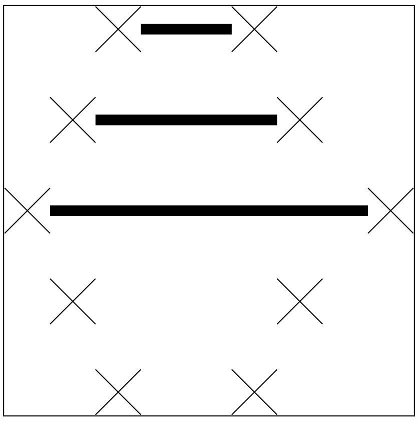


Fig. 21: The horizontal straight lines corresponding to the chord-sets in Fig. 18 *(b)*. Notice that this time we have to add additional holes under the lowest horizontal straight line, to prevent the creation of new possible rectangle coverings.

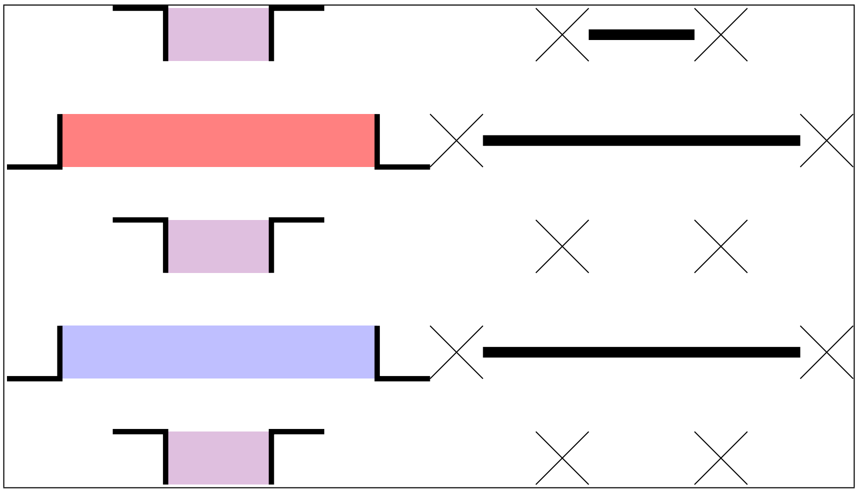


Fig. 22: An example of a larger chord set and two different chord sets in between on the left. Notice that the red and blue chord set are different since they cannot be both covered by exactly one rectangle.
On the right the corresponding horizontal straight lines and holes.

Using this construction, we will show the following theorem.

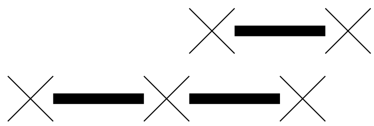
(3.7) Theorem
Let $|\mathcal{F}_3| = n \in \mathbb{N}$. There are at least $\lceil \frac{n}{2} \rceil$ rectangles needed to completely cover $\mathcal{F}_3$.

Proof
We will show this theorem by induction. First we show the basis step.
The cases $n = 1$ and $n = 2$ are trivial so we start at $n = 3$.
We need to show the assumption for every possible positioning of 3 chord-sets, since this shows that in any arbitrary instance containing these n chord-sets in this positioning, the assumption is true. We will show it for 10 positionings, all other possibilities are either symetrical or similar enough to be ignored, e.g. additional holes, one of the chord-sets positioned slithly more to the left, one of the chord-sets is wider etc.

1st positioning:

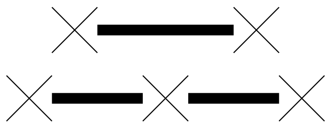
At least 3 rectangles are needed in every optimal covering.

2nd positioning:



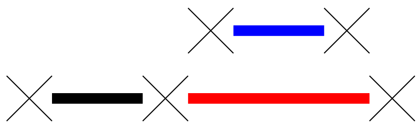
At least 3 rectangles are needed in every optimal covering.

3rd positioning:



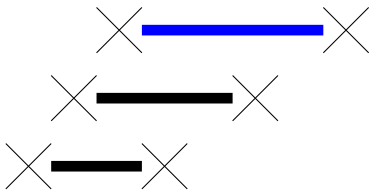
At least 3 rectangles are needed in every optimal covering.

4th positioning:



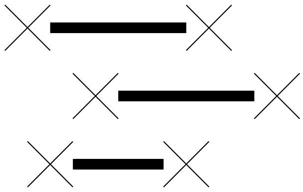
At least 3 rectangles are needed in every optimal covering. Notice that making the blue chord-set wider can lead to having only two chord-sets, while adding additional holes, e.g. to make the red chord-set be in between the blue chord set, does not lower the optimal number of rectangles needed to cover the chord-sets.

5th positioning:



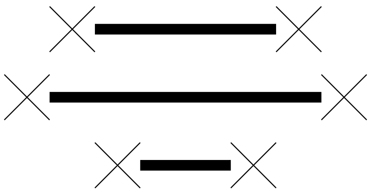
At least 3 rectangles are needed in every optimal covering. Notice that moving the blue chord-set slightly to the left does not lower the optimal number of rectangles needed to cover the chord-sets. Adding additional holes does not create new possibilities of covering rectangles.

6th positioning:



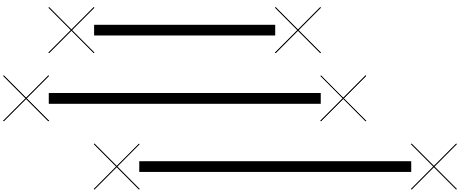
At least 3 rectangles are needed in every optimal covering.

7th positioning:



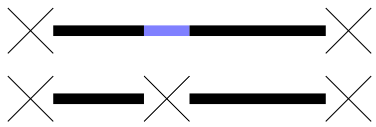
At least 3 rectangles are needed in every optimal covering. Notice that swapping the vertical positions of the chord-set will not affect the optimal needed number of rectangles. It is the same with adding holes and creating chord-sets in between chord-sets.

8th positioning:



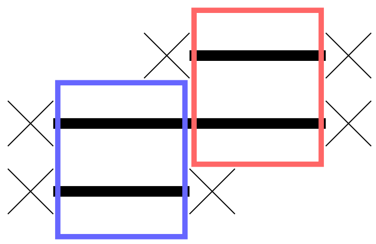
At least 3 rectangles are needed in every optimal covering. Notice that moving the chord-sets to the left or right will not lower the optimal number of needed rectangles to cover the chord-sets.

9th positioning:



At least 3 rectangles are needed in every optimal covering. It seems that there are only two rectangles needed here, but the small blue area above the middle hole is the reason why there are 3 rectangles needed.

10th positioning:



In this case at least $2 = \lceil \frac{3}{2} \rceil$ rectangles are needed. Notice that moving the chord-sets would in the worst case only increase the optimal number of needed rectangles to cover the chord-sets. Adding holes would lead to the same conclusion. It is indeed the only positioning for $n = 3$ where less than 3 rectangles are needed in an optimal cover.

Let the assumption be true for every $1 \ldots m \in \mathbb{N}$.
Let $|\mathcal{F}_3| = n = m + 1$.

Let x be the minimum right coordinate of a chord-set contained in $\mathcal{F}_3$. Notice that it is possible that there is more than only one chord-set with x as the right coordinate. Let $c \in \underline{n}$ be the number of chord-sets with x as the right coordinate. If $c = n$ then let y be the maximum left coordinate of a chord-set contained in $\mathcal{F}_3$ and $d \in \underline{n}$ be the number of chord-sets with y as the left coordinate. If $d = n$ we have a special case which we will solve later.

Let w.l.o.g. $c < n$ and x be the mentioned minimum right coordinate, since if $c = n$ and $d < n$, we can just flip the horizontal straight lines instance vertically to recieve $c < n$.

By induction we need at least $\lceil \frac{c}{2} \rceil$ rectangles to cover the chord-sets in any optimal solution. The side of the instance with coordinates larger than x are still completely

uncovered. We create a vertical line going from the top to the bottom of the instance straight throught x and call it the *split line*.
There may be some chord-sets that are split in 2 by the *split line* but the right part of these chord-sets can not be covered by the rectangles needed to optimally cover the c chord-sets on the left side of the *split line*, see Fig. 23.

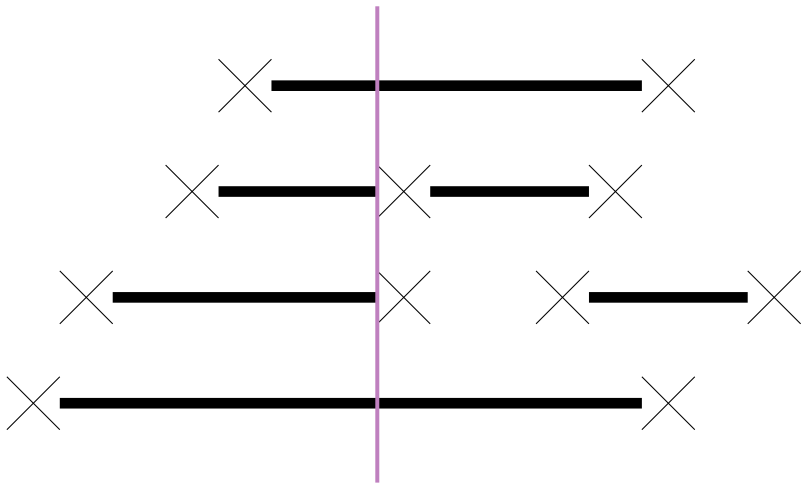


Fig. 23: The violet vertical line is the *split line* in this horizontal straight line instance with $c = 2$. We see two chord-sets split in two and their straight line parts on the right of the *split line* can not be covered by rectangles needed to optimally cover the two chord-sets with minimum right coordinate.

However straight line parts on the left of the *split line* of the chord-sets split by it, can be partially or even completely covered by the rectangles needed to cover the c chord-sets with minimum right coordinate.
We can now interpret the straight line parts on the right side of the *split line* as uncovered chord-sets, since we do not decrease the number of rectangles needed to cover the horizontal straight line instance on the right side of the *split line*.
To cover the right side of the instance, by induction we need at least $\lceil \frac{n-c}{2} \rceil$ rectangles. In total we need at least $\lceil \frac{n-c}{2} \rceil + \lceil \frac{c}{2} \rceil$ rectangles to cover the horizontal straight line instance. We need to show that

$$\left\lceil \frac{n-c}{2} \right\rceil + \left\lceil \frac{c}{2} \right\rceil \geq \left\lceil \frac{n}{2} \right\rceil.$$

We will show the four possible cases to prove the inequality.

1st case: $n = 2k, c = 2l\ , k, l \in \mathbb{N}$

$$\left\lceil \frac{2k-2l}{2} \right\rceil + \left\lceil \frac{2l}{2} \right\rceil = \lceil k-l \rceil + \lceil l \rceil = k \geq \left\lceil \frac{n}{2} \right\rceil$$

2nd case: $n = 2k, c = 2l+1\ , k \in \mathbb{N}, l \in \mathbb{N}_0$

$$\left\lceil \frac{2k-2l-1}{2} \right\rceil + \left\lceil \frac{2l+1}{2} \right\rceil = \left\lceil k-l-\frac{1}{2} \right\rceil + \left\lceil l+\frac{1}{2} \right\rceil = k-l-1+l+1 = k \geq \left\lceil \frac{n}{2} \right\rceil$$

3rd case: $n = 2k+1, c = 2l+1\ , k \in \mathbb{N}, l \in \mathbb{N}_0$

$$\left\lceil \frac{2k+1-2l-1}{2} \right\rceil + \left\lceil \frac{2l+1}{2} \right\rceil = \lceil k-l \rceil + \left\lceil l+\frac{1}{2} \right\rceil = k-l+l+1 = k+1 \geq \left\lceil \frac{n}{2} \right\rceil$$

4th case: $n = 2k+1, c = 2l\ , k \in \mathbb{N}, l \in \mathbb{N}$

$$\left\lceil \frac{2k+1-2l}{2} \right\rceil + \left\lceil \frac{2l}{2} \right\rceil = \left\lceil k-l+\frac{1}{2} \right\rceil + \lceil l \rceil = k-l+1+l = k+1 \geq \left\lceil \frac{n}{2} \right\rceil$$

The inequality holds and we only have to show the last special case mentioned before.
Let $c = n$ and $d = n$. In this case all chord-sets have the same left and right coordinates and between every two chord-sets there has to be a hole or else they would be alligned and create only one instead of two chord-sets, see Fig. 24.

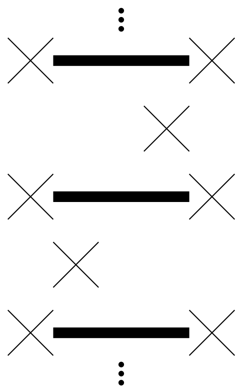


Fig. 24: The special case $c = n$ and $d = n$.

Let $\mathcal{S}$ be the lowest chord-set. There is no chord-set below $\mathcal{S}$ and there is at least one hole directly above it. Hence there is an area on the horizontal straight line that has to be covered directly under one of the holes. To cover this area there is at least one rectangle needed and this rectangle does not cover any parts of other horizontal straight lines in this instance. By induction we need at least $\lceil \frac{n-1}{2} \rceil$ rectangles to cover all the chord-sets above $\mathcal{S}$ and an additional rectangle to cover $\mathcal{S}$. Thus we need at least $\lceil \frac{n-1}{2} \rceil + 1$ rectangles. We show the only two possibilities for n.

1st case: $n = 2k \,, k \in \mathbb{N}$

$$\left\lceil \frac{2k-1}{2} \right\rceil + 1 = \left\lceil k - \frac{1}{2} \right\rceil + 1 = k - 1 + 1 = k \geq \left\lceil \frac{n}{2} \right\rceil$$

2nd case: $n = 2k + 1 \,, k \in \mathbb{N}$

$$\left\lceil \frac{2k+1-1}{2} \right\rceil + 1 = \lceil k \rceil + 1 = k + 1 \geq \left\lceil \frac{n}{2} \right\rceil$$

The inequality holds as well for this special case and hence we have proven the theorem by induction. □

The first question that comes in mind is, if it is possible to improve this theorem, i.e. why is it not possible to prove it for $\lceil \frac{21n}{40} \rceil$?

We will use the 10th positioning in the induction basis of the theorem to show that improving this theorem is not possible. Let $1 > a > \frac{1}{2}$ be the ratio that we want to improve to, i.e. we need at least $a \cdot n$ rectangles to cover n type 3 chord-sets.

We expand the 10th positioning by always adding a wide and a sleeker chord-set above the right sleeker chord-set such that we only need one more rectangle to cover both new chord-sets, see Fig. 25.

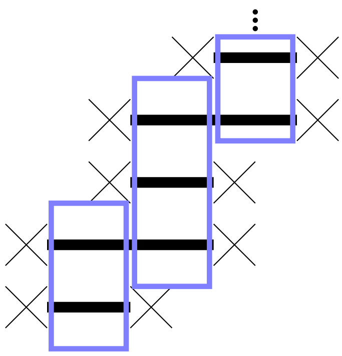


Fig. 25: The construct to show that the theorem can not be improved. We have 5 chord-sets covered by only 3 rectangles.

This construction leads to the sequence $a_n = \frac{3+2n}{2+n}, n \in \mathbb{N}$ and its limes is $\frac{1}{2}$. Thus the theorem cannot be improved.

Using the proven theorem we recieve:

$$\theta \geq \frac{|\mathcal{F}_3|}{2}$$
$$\Leftrightarrow 2\theta \geq |\mathcal{F}_3|$$

With the inequality $c_1 \leq 4\theta$ from [Franzblau 1989] we recieve:

$$2\overline{p} \leq 2\theta + 4\theta + 2\theta$$
$$\Leftrightarrow \overline{p} \leq 4\theta$$

This implies that the algorithm *Partition/Extend* is a constant factor approximation algorithm with ratio 4.

We can further improve this ratio. The theorem is adaptable for type 1 chords, i.e. for $|\mathcal{F}_1|$, as well since they have the same properties type 3 chord-sets have. The only difference in the visualization of type 1 chord-sets would be an additional hole on the left or right side of the straight horizontal line, since type 1 chords only contain one concave vertex. With this result we can improve the inequality to

$$\overline{p} \leq 3\theta$$

the ratio conjectured by Franzblau herself.

There is even one more property of type 1 chords we can use to improve this inequality. In the proven theorem we only used the rectangles that go vertically throught the chord. There is at least one horizontally expanded rectangle needed to cover the area below or above type 1 chord, see Fig. 26.

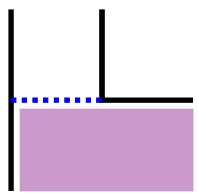


Fig. 26: The purple area below this type 1 chord has to be covered by at least one horizontally expanded rectangle.

At most four chords and thus chord-sets can share one of the rectangles needed to cover the purple area, see Fig. 27.

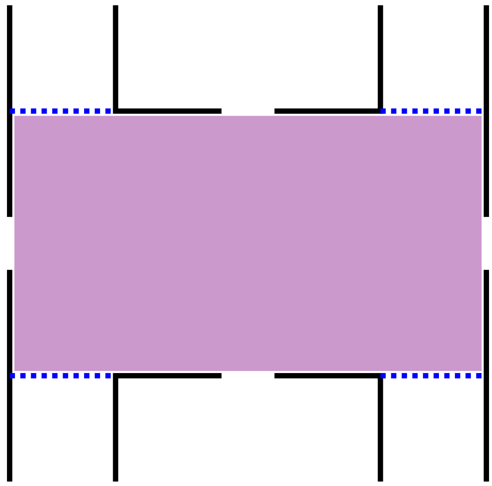


Fig. 27: At most four type 1 chords or chord-sets can be attached to the purple area in the middle at once.

Hence we need at least additional $\left\lceil \frac{|\mathcal{F}_1|}{4} \right\rceil$ rectangles to cover the areas below all type 1 chord sets. Thus we recieve:

$$\theta \geq \left\lceil \frac{|\mathcal{F}_1|}{2} \right\rceil + \left\lceil \frac{|\mathcal{F}_1|}{4} \right\rceil$$
$$\Leftrightarrow \frac{4}{3}\theta \geq |\mathcal{F}_1|$$

In total the resulting inequality is:

$$2\overline{p} \leq 2\theta + \frac{4}{3}\theta + 2\theta$$
$$\Leftrightarrow \overline{p} \leq \frac{8}{3}\theta$$

The result is that algorithm *Partition/Extend* from [Franzblau 1989] is a $\frac{8}{3}$-approximation algorithm.

§4 Conclusions and Outlook

— An upper bound for $\frac{\theta}{\alpha}$ —

We showed that there exists a constant approximation algorithm for the rectangle cover problem and increased the lower bound for $\frac{\theta}{\alpha}$. There is still the open question Erdös asked whether $\frac{\theta}{\alpha}$ was bounded by a constant. Let us discuss this question a bit further.

In the experiments using randomly generated instances, there was no instance containing an odd-cycle with a length larger than 7 and I did not find any example of such an instance in the literature. Let us assume that the largest possible odd-cycle in a rectilinear polygon instance has length 7 and that we obtain such an instance with the largest possible $\frac{\theta}{\alpha}$ ratio. The combinatorial possibilities of such instances with this largest $\frac{\theta}{\alpha}$ ratio are bounded by a constant since the rectilinear polygon instances are always two dimensional. If the $\frac{\theta}{\alpha}$ is not already larger than our improved bounds, we can try to use the construction showed in section 2 to create a sequence that possibly increases the bound. If it does not increase the bound we can try to combine the instances with the odd-cycles of length 7 with the instances with odd-cycles of length 5. With these new constructs we can build a sequence similar to the one in the second section which possibly inscreases the bound. It would mean that we can construct the worst possible instance in means of the $\frac{\theta}{\alpha}$ ratio and show that an upper bound exists, but as long as we do not show that the length of odd-length cycles in rectinlinear polygon instances are somehow bounded, it remains an open question.

— Linear programming approach —

In [Heinrich-Litan and Lübbecke 2007] it is said that the rectangle cover problem turns to be computationally very tractable. We can only agree with this statement. In the experiments all randomly generated instances could have been as well solved by iterative rounding to recieve an approximation algorithm of factor 2, i.e. there existed always a rectangle variable with solution value $\geq \frac{1}{2}$ in every solution of the experimental instances and after rounding this variable up to 1 and solving the instance again, there was an other rectangle variable $\geq \frac{1}{2}$ etc.

At first I wish for a way to create random instances with a given relevancy ratio, since it was very hard and time consumpting to create random instances with some significant solutions. Moreover I hope that this work will open a different point of view on the rectangle cover problem and instead of searching for an algorithm that instantly creates a bound for $\frac{\theta}{\alpha}$, we try to solve the problem geometrically by using already developed algorithms. This way it may be possible for us to observe other properties of the problem and use them to answer *Erdös'* question:

Is $\frac{\theta}{\alpha}$ bounded by a constant?

§5 References

Ahuja, Ravindra, K., Magnanti, Thomas, L., Orlin, James B. 1993. Network Flows - Theory, Algorithms and Applications. *Printice Hall, New Jersey*.

Anil Kumar, V. and Ramesh, H. 2003. Covering rectilinear polygons with axis-parallel rectangles. *SIAM J. Comput. 32*, 6, 1509-1541.

Berman, P. and DasGupta, B. 1997. Complexities of efficient solutions of rectilinear polygon cover problems. *Algorithmica 17*, 4, 331-356.

Bern, M. and Eppstein, D. 1997. Approximation algorithms for geometric problems. *See Hochbaum [1996]*, Chapter 8, 296-345.

Chaiken, S., Kleitman, D. Saks, M., and Shearer, J. 1981. Covering regions by rectangles. *SIAM J. Algebraic Discrete Methods 2*, 394-410.

Franzblau, D. 1989. Performance guarantees on a sweep-line heuristic for covering rectilinear polygons with rectangles. *SIAM J. Discrete Math. 2*, 3, 307-321.

Garey, Michael, R., Johnson, David, S. 1979. Computers and Intractability. *Bell Telephone Laboratories*.

Hannenhalli, S., Hubell, E., Lipschutz, R., and Pevzner, P. 2002. Combinatorial algorithms for design of DNA arrays. *Adv. Biochem. Eng. Biotechnol.*, 77, 1-19.

Heinrich-Litan, L. and Lübbecke,M.E. 2007. Rectangle Covers Revisited Computationally. *ACM J. Exp Algorithmics 11*, Article No. 2.6.

Marie, F. 1994. Polyominos and perfect graphs. *Inform. Process. Lett. 50*, 2, 57-61.

Masek, W. 1979. Some NP-complete set covering problems. *Unpublished manuscript*, MIT.

Motwani, R., Raghunathan, A., and Saran, H. 1989a. Covering orthogonal polygons with star polygons: The perfect graph approach. *J. Compt. System Sci. 40*, 19-48.

Motwani, R., Raghunathan, A., and Saran, H. 1989b. Perfect graphs and orthogonally convex covers. *SIAM J. Discrete Math. 2*, 371-392.

Schrijver, A. 2003. Combinatorial Optimization: Polyhedra and Efficiency. *Springer*, Berlin.

"Diopsis", Licensed under Creative Commons Attribution-Share Alike 3.0 via Wikimedia Commons, http://commons.wikimedia.org/wiki/File:Diopsis.jpg#mediaviewer/File:Diopsis.jpg

List of Figures

Statement of authorship

I hereby certify that this master thesis has been composed by myself, and describes my own work, unless otherwise acknowledged in the text. All references and verbatim extracts have been quoted, and all sources of information have been specifically acknowledged. It has not been accepted in any previous application for a degree.

Aachen, 25th September 2014