Karlsruhe Institute of Technology

Transforming rectangles into squares

An introduction to the squarability problem

Bachelor Thesis of

Jonathan Klawitter

At the Department of Mathematics
Institute for Algebra and Geometry
Research Group on Discrete Mathematics †

And at the Department of Informatics
Institute of Theoretical Informatics
Chair of Algorithmics I *

Reviewer:	Dr. Martin Nöllenburg *
Second reviewer:	Prof. Maria Axenovich Ph.D. †
Advisor:	Dr. Torsten Ueckerdt †
Second advisor:	Dr. Martin Nöllenburg *

Duration:: 01. Juli 2014 – 31. October 2014

I declare that I have developed and written the enclosed thesis completely by myself, and have not used sources or means without declaration in the text.

Karlsruhe, 21. October 2014

.....

(**Jonathan Klawitter**)

Abstract

In this bachelor thesis we introduce the SQUARABILITY problem: When can a set of axis-aligned rectangles be transformed into squares without changing combinatorial properties? This means, that we do not allow to change whether, how and in which order the rectangles respectively squares intersect.

We use a sweep line algorithm to compute the combinatorial information from geometrically given rectangles. We give a full characterisation of triangle-free rectangle arrangements via enhanced intersection graphs. We define a mixed integer linear program to solve any instance of the SQUARABILITY problem. We give some exemplary instances, which indicate that the problem is in general not that easy to solve. However, we show that some classes of rectangle arrangements can always be transformed into squares in the desired way.

Zusammenfassung

In dieser Bachelorarbeit stellen wir das SQUARABILITY Problem vor: Wann kann man eine Menge von achsenparallelen Rechtecken in Quadrate transformieren ohne die kombinatorischen Eigenschaften ihrer Schnitte zu verändern? Das heißt wir erlauben nicht, dass bei der Transformation verändert wird ob, wie und in welcher Reihenfolge sich die Rechtecke bzw. Quadrate schneiden.

Wir benutzen einen Sweep-Line Algorithmus um die kombinatorischen Informationen aus geometrisch gegebenen Rechtecken zu berechnen. Mittels erweitertem Schnittgraph geben wir eine vollständige Charakterisierung von dreiecksfreien Rechtecksmengen an. Wir stellen ein Mixed Integer Linear Program auf um jegliche Instanzen des SQUARABILITY Problem lösen zu können. Des Weiteren zeigen wir einige Beispielinstanzen des Problems auf, welche verdeutlichen, dass das Problem im allgemeinen wohl nicht einfach zu entscheiden ist. Wir können jedoch auch einige Klassen von Rechtecksmengen angeben, welche immer auf die gewünschte Art und Weise in Quadrate transformierbar sind.

Contents

1. Introduction

We start directly with the problem, even though just informally for the moment.

SQUARABILITY *problem: Given a set of axis-aligned rectangles in the plane. Can we transform all rectangles into squares of any size without changing any of the combinatorial properties formed by intersections between them?*

The SQUARABILITY problem connects geometry and combinatorics. Loosely speaking, we try to change the geometric properties of rectangles, without altering the combinatorial properties of their relative positions to each other. More precisely, this means that, when we transform the rectangles into squares, we are not allowed to form new intersections, lose any intersection or even change the way two rectangles intersect. As an example, if rectangle A pierces rectangle B into the left side, then the corresponding square A' has to pierce square B' also into the left side. In other words, we ask if and when rectangle and square intersections can have equivalent combinatorial properties. A formal definition of the SQUARABILITY problem is given later with Definition 1.18, after we have stated more precisely what we mean with combinatorial properties.

Figure 1.1 depicts a set of rectangles where such a transformation into squares is possible. And to get a better understand of what is not allowed, Figure 1.2 gives some examples of sets of squares that are combinatorially different from the rectangles in Figure 1.1.

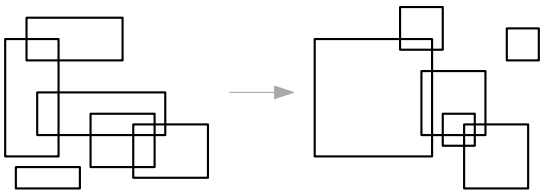

Figure 1.1: Transformation of a set of rectangles into a combinatorially equivalent set of squares.

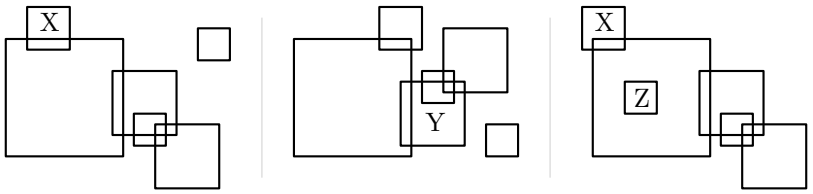

Figure 1.2: Three sets of squares, each combinatorially different from the set of rectangles in Figure 1.1.
Left: The square X intersects in a different way with the big one.
Middle: At square Y two intersections are on the opposite side.
Right: The former single square Z lies inside another and square X intersects a different corner.

When can such a transformation be done? Is it maybe always possible? Not surprisingly this is not the case. The smallest counter example consists of two rectangles, as shown in Figure 1.3. There two rectangles pierce completely through each other. Or one could describe it via sets. For two rectangles A and B seen as subsets of the plane $\mathbb{R}^2$ the set $A \backslash B$ is disconnected. Obviously this configuration is not possible for two squares.

We will see some more complex unsquarable examples later. This will show that the decision problem seems to be not that easy. Sadly, we can not say how hard it is in terms of computational complexity.

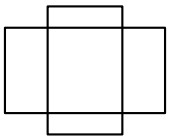

Figure 1.3: The smallest counterexample for a set of rectangles that can not be transformed into squares without changing the way they intersect.

But before we dive deeper into the problem, we will define some terminology, enabling us to talk more formally about this topic. As a start, we define types of intersections between rectangles and the problem instances of sets of rectangles we consider. Then, we will be able to formally define the Squarability problem. After that, at the end of this chapter, we outline what parts of the problem are investigated in this thesis. In Chapter 2 we motivate why we consider the combinatorial equivalence of rectangle and square intersections.

1.1 Rectangle arrangements

We consider axis-aligned rectangles in the plane.

Definition 1.1. *An axis-aligned **rectangle** in the plane is the point set* $[a, b] \times [c, d] \subset \mathbb{R}^2, a < b, c < d$.

Note, that this definition implies that the interior as well as the borders are included in the rectangle. Axis-aligned squares are defined analogously.

Definition 1.2. *An axis-aligned **square** in the plane is the point set* $[a, b] \times [c, d] \subset \mathbb{R}^2, a < b, c < d, (b - a) = (d - c)$.

1.1.1 Intersection types

Unlike most other work, we are interested in the exact way rectangles and squares intersect, not just whether or not they intersect at all. This includes the specific sides or corners which form the intersections and the order among multiple intersections.

Two rectangles can stand in different relations to each other. They can either be disjoint, one can contain the other or they can form one of three types of overlaps defined below. Throughout this thesis, we assume that no two rectangles are the same or share segments of their boundaries, as in the examples of Figure 1.4.

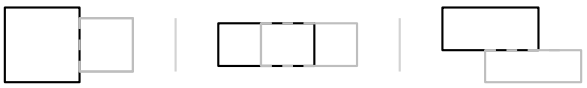

Figure 1.4: We do not allow rectangles to share border segments.

The first intersection type between two rectangles (or two squares) describes the situation where, informally, one pierces the side of the other. With the following Definition 1.3 we define this phrase and the intersection type.

Definition 1.3. *A **side-piercing intersection** between two axis-aligned rectangles $A = [a,b] \times [c,d]$ and $B = [a',b'] \times [c',d']$, with A piercing B, is an intersection of these two rectangles where*

1. *$[a,b] \subset [a',b']$ and $[c,d]$, $[c',d']$ overlap, i.e. either*
   a) *$c < c' < d < d'$ or*
   b) *$c' < c < d' < d$, or where*
2. *$[c,d] \subset [c',d']$ and $[a,b]$, $[a',b']$ overlap, i.e. either*
   a) *$a < a' < b < b'$ or*
   b) *$a' < a < b' < b$.*

So for example in Case 2a we say that A is piercing B into the left side. This is shown in Figure 1.5. Symmetrically, there are three other cases for side-piercing intersections at the right, upper and lower side. Requesting not to change the combinatorial properties of a side-piercing intersection during the squaring includes that the squares have to form a side-piercing intersection with the same case as the rectangles do.

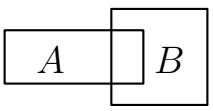

Figure 1.5: A side-piercing intersection with A piercing the left side of B.

The second intersection type describes the case where the two rectangles overlap with one corner each. Definition 1.5 gives a more precise characterisation and a name for this type.

Definition 1.4. *An **opposite-corner intersection** between two axis-aligned rectangles $A = [a, b] \times [c, d]$ and $B = [a', b'] \times [c', d']$ is an intersection of these two rectangles where $[a, b]$, $[a', b']$ as well as $[c, d]$, $[c', d']$ overlap, i.e. either*

1. $a < a' < b < b'$ *and*
   a) $c < c' < d < d'$ *or*
   b) $c' < c < d' < d$, *or*
2. $a' < a < b' < b$ *and*
   a) $c < c' < d < d'$ *or*
   b) $c' < c < d' < d$.

The name is self-explanatory since this is only possible if the two corners which lie inside the other rectangle are the opposite ones. For example in Case 1b the lower right corner of A and upper left corner of B can take these roles, as in Figure 1.6. Again, when considering the combinatorial properties of an opposite-corner intersection we mean also which roles the rectangles take and which corners are contained in which rectangles.

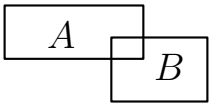

Figure 1.6: An opposite-corner intersection between A and B.

One rectangle completely containing the other is the third intersection type. The role the rectangles respectively squares take is again important.

Definition 1.5. *A **containing intersection** between two axis-aligned rectangles $A = [a, b] \times [c, d]$ and $B = [a', b'] \times [c', d']$ is an intersection of these two rectangles where $[a, b] \subset [a', b']$ and $[c, d] \subset [c', d']$.*

The fourth intersection type describes an intersection that can not occur with squares (Figure 1.7).

Definition 1.6. *An **unsquarable intersection** formed by two axis-aligned rectangles A and B is an intersection of these two rectangles where $[a', b'] \subset [a, b]$ and $[c, d] \subset [c', d']$.*

As mentioned before, this can be described in another way, too. Looking at A and B as sets, both $A \backslash B$ and $B \backslash A$ are disconnected. We call this the unsquarable intersection, since the two participating rectangles can not be squared such that the corresponding squares form the same intersection type.

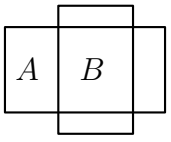

Figure 1.7: An unsquarable intersection formed by A and B.

It is not only important which intersections are formed, but also in which order.

Definition 1.7. *The **intersection order** of a rectangle A is the sequence of rectangle borders of the rectangles intersecting A counterclockwise along the border of A starting at the upper right corner of A.*

Figure 1.8 shows two rectangle arrangements with different intersection orders for A and A', even though the sequence of intersecting borders is the same for both. However, they belong to different rectangles. We can identify rectangle B respectively B', by the fact that they are the only rectangles which pierce into two other rectangles. On the left B comes counterclockwise after C, but on the right B' before C'.

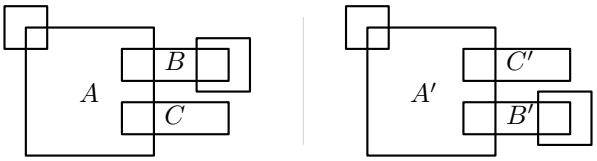

Figure 1.8: Two rectangle arrangements where the intersection orders of A and A' differ.

1.1.2 Rectangle arrangements

We refer to a set of rectangles as a geometrical object in its entirety as a rectangle arrangement. We do only allow the intersection types defined above which exclude cases where two rectangles share border segments. The same applies for square arrangements.

Definition 1.8. *A **rectangle arrangement** is a finite set $\mathcal{R}$ of axis-aligned rectangles in the plane $\mathbb{R}^2$, where two rectangles $A, B \in \mathcal{R}$ either intersect with a containing, side-piercing, opposite-corner or unsquarable intersection or are disjoint.*

Definition 1.9. *A **square arrangement** is a finite set $\mathcal{S}$ of axis-aligned squares in the plane $\mathbb{R}^2$, where two squares $A, B \in \mathcal{S}$ either intersect with a containing, side-piercing or opposite-corner intersection or are disjoint.*

With arrangements formally defined, we make some observations. We can scale a whole rectangle arrangement without changing the combinatorial properties. The size of single rectangles do not matter as long as intersections do not change as well. Furthermore, reflectional and rotational symmetry of a rectangle arrangement do not affect the SQUARABILITY problem. In order words, we are only interested in the combinatorial properties and the geometrical properties resulting out of them compulsorily. In order to record this formally we define equivalence classes.

Definition 1.10. *Two rectangle arrangements $\mathcal{R}$ and $\mathcal{R}'$ are **combinatorially equivalent** if a one-to-one mapping from $\mathcal{R}$ to $\mathcal{R}'$ exists which respects the intersection types and the intersection orders for every rectangles, up to reflections.*

*A **combinatorial equivalence class** of rectangle arrangements is the set of rectangle arrangements all combinatorially equivalent to one another.*

All results we present are invariant under this combinatorial equivalence. So, from now on and throughout this thesis, whenever we consider a rectangle arrangement we actually mean its equivalence class.

The different terms for the intersection types may seem superfluous but will make it easier for us to consider the different combinatorial properties which arise from them. The main combinatorial aspect behind a rectangle arrangement is its corresponding *intersection graph*. An intersection graph of a set of objects is a graph, that has the set itself as vertex set and two vertices are adjacent if and only if their corresponding objects have a nonempty intersection. In order to cover all combinatorial properties we will later (Chapter 4) describe a way to put the additional information arising from the intersection types and intersection orders into the intersection graph.

Definition 1.11. *A rectangle arrangement $\mathcal{R}$ is connected, if the union of $\mathcal{R}$ is a connected set.*

Or equivalently, if the intersection graph of $\mathcal{R}$ is connected. Transforming into squares or describing characteristics of a disconnected rectangle arrangement can be done for each connected component separately. Therefore, we assume from now on that without loss of generality the considered rectangle arrangements are connected.

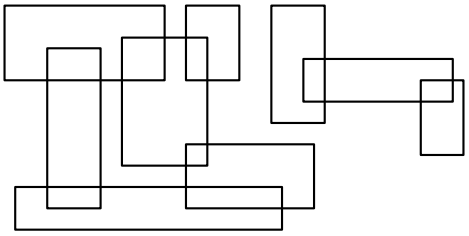

Figure 1.9: A disconnected rectangle arrangement, which consists of two connected components.

Similar to subgraphs in graph theory we define rectangle subarrangements and one important example.

Definition 1.12. *A rectangle arrangement $\mathcal{S}$ is **rectangle subarrangement** of a rectangle arrangement $\mathcal{R}$, shortly $\mathcal{S} \tilde{\subseteq} \mathcal{R}$, if a subset $\mathcal{S}' \subseteq \mathcal{R}$ exists which is combinatorially equivalent to $\mathcal{S}$.*

Definition 1.13. *The **cross** is the rectangle arrangement of two rectangles forming an unsquarable intersection.*

We say that a *cross-free rectangle arrangement* is a rectangle arrangement that contains no cross as rectangle subarrangement.

1.1.3 Restricted rectangle arrangements

We will later consider rectangle arrangements with further restrictions. One of these restrictions is, that in the rectangle arrangement no three rectangles pairwise intersect.

Definition 1.14. *A **triangle-free rectangle arrangement** is a rectangle arrangement $\mathcal{R}$, where no three rectangles share one point, i.e. $\forall X, Y, Z \in \mathcal{R} : X \cap Y \cap Z = \emptyset$.*

The name triangle-free arises from the fact that three rectangles X, Y, Z have nonempty intersection, i.e. $X \cap Y \cap Z \neq \emptyset$ if and only if their corresponding vertices in the intersection graph form a triangle. Note that is not the case for all geometric objects. However, for rectangles and therefore squares we can conclude this from the fact that they have Helly number 2. In addition to that, we will show with Lemma 4.1 that triangle-free rectangle arrangements have planar intersection graphs.

Observe further that in triangle-free rectangle arrangements containing intersections, with A containing B, can be ignored since they do not further influence the squarability question. Square B could just be placed as small as needed in any free space inside A at the end of the transformation process. This free space exists always, since otherwise further rectangles intersecting with A would cover it completely. This could only be possible if they share a border with A or each other, which we do not allow, or if A is enclosed by a third rectangle, a contradiction to being triangle-free. Therefore, from now on, if

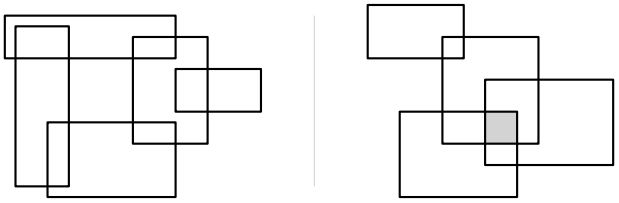

Figure 1.10: Left: A triangle-free rectangle arrangement.
Right: Since the gray area is covered by three rectangles, a non triangle-free rectangle arrangement.

not stated otherwise, we assume that triangle-free rectangle arrangements do not contain rectangles lying inside another one, i.e. no containing intersections.

Another possible restriction is to not allow both side-piercing and containing intersections in a rectangle arrangement. As we will see in Chapter 6, these intersections can easily inhibit a transformation from rectangles to squares.

Definition 1.15. *A rectangle arrangement $\mathcal{R}$ is called restricted to* ***opposite-corner intersections****, if all pairs of rectangles $A, B \in \mathcal{R}$ either form only an opposite-corner intersection or are disjoint.*

A third restriction we consider is that all rectangles in a rectangle arrangement can be pierced by one horizontal line.

Definition 1.16. *A rectangle arrangement $\mathcal{R}$ is called* ***line pierced rectangle arrangement*** *if a horizontal line l exists such that the intersection of each rectangle A with this line is not empty, i.e. $\forall A \in \mathcal{R} : l \cap A \neq \emptyset$.*

Obviously, the restrictions triangle-free, only opposite-corner intersections and line pierced are not mutually exclusive and can be combined.

1.2 The Squarability problem

We can now define what it means for a set of rectangles to be transformed into squares without changing the combinatorial properties arising from the intersections. As we mentioned, the different intersections include the roles participating rectangles take in them and which case of their definitions apply. For example, the piercing or the pierced rectangle in a side-piercing intersection are different roles. At which side or with which corners the intersections are formed correspond to the cases. Not changing the combinatorial properties includes the order of the intersection as well. For example, if a rectangle A is pierced by two others into the same side, the squares have to pierce the square of A in the same order and into the same side. Definition 1.17 provides a formal characterisation of whether a rectangle arrangement, as we call it, can be *squared*.

Definition 1.17. *A rectangle arrangement $\mathcal{R}$ can be* ***squared*** *or is* ***squarable*** *if a square arrangement $\mathcal{S}$ exists which is combinatorially equivalent to $\mathcal{R}$.*

If a rectangle arrangement can not be squared we also say it is *unsquarable*.

Now that we know what squarable formally means, we can state our decision problem.

Definition 1.18. *The* Squarability *problem is the decision problem of whether a rectangle arrangement is squarable.*

1.2.1 The class Squares

We can divide rectangle arrangements into the squarable and unsquarable ones.

Definition 1.19. *The class* Squares *is the set of all squarable rectangle arrangements.*

An easy example for a rectangle arrangement in Squares is a totally disconnected rectangle arrangement $\mathcal{R}$, meaning that no two rectangles intersect. Thus, we have that $\{\mathcal{R} \mid \mathcal{R} \text{ rectangle arrangement}, \forall X, Y \in \mathcal{R} : X \cap Y = \emptyset\} \subset$ Squares.

We will later see many unsquarable rectangle arrangements and obviously it is not possible to square a rectangle arrangement if it has an unsquarable rectangle subarrangement.

Theorem 1.20. *If* $\mathcal{S} \tilde{\subseteq} \mathcal{R}$ *and* $\mathcal{S} \notin$ Squares*, then* $\mathcal{R} \notin$ Squares*.*

We already know one important unsquarable rectangle arrangement, the cross rectangle arrangement.

Conclusion 1.21. *If a rectangle arrangement* $\mathcal{R}$ *contains the cross as rectangle subarrangement, then* $\mathcal{R} \notin$ Squares*.*

1.3 Outline

Let us briefly outline which aspects of the Squarability problem we consider in this bachelor thesis. In Chapter 2 we motivate the problem and give some related work on square and rectangle intersections. Then we go stepwise from geometrically given rectangles over a combinatorial description to solving the problem. First, in Chapter 3 we take a view on the algorithmic problem of extracting the combinatorial properties of geometrically given rectangles. Then, in Chapter 4 we show a way to describe rectangle arrangements solely combinatorially and getting back from such a description to a geometrical representation with a flow network. Thereby we focus on triangle-free rectangle arrangements. In Chapter 5 we present an algebraic solution model for the Squarability problem. At the end, in Chapter 6 we discuss the question of when a rectangle arrangement is squarable. We examine rectangle arrangements with the restrictions triangle-free, restricted to opposite-corner intersections and line pierced, as well as combinations of them. We give some more complex unsquarable rectangle arrangements and show some indications that it might be quite hard to solve the problem constructively.

2. Motivation

Before diving deeper into the SQUARABILITY problem, we would like to motivate this question. Why should one be interested in the combinatorial equivalence of rectangles and squares in the plane? Well, one direct answer is for the sake of curiosity and fundamental research. A possibly more satisfying answer is, that knowing when a rectangle arrangements can be transformed into a square arrangements with the same combinatorial properties, then some results on squares may be generalizable and also apply for those rectangle arrangements.

The first part of this chapter examines a more broadly studied equivalence of geometrical objects, namely between lines and pseudolines. Similar to our SQUARABILITY problem we discuss the STRECTCHABILITY problem asking whether pseudolines can be stretched to lines. We show some exemplary results which can be generalized from lines to pseudolines. In the second part of this chapter we take a look at graph colorings. More precisely, we quote results on the maximum chromatic number of intersection graphs in terms of their clique number. For square intersection graphs a linear upper bound is known, whereas for rectangle intersection graphs only a quadratic bound is known. This will serve us as an exemplary result which can be generalized to rectangle arrangements in SQUARES.

2.1 Pseudoline arrangements

In this section we look at arrangements of lines and pseudolines. A *line arrangement* is the partition of the plane by a set of distinct lines. The size of a line arrangement is the number of lines it consists of.

A *pseudoline* is a simple infinite curve which divides the plane into two connected components. A *pseudoline arrangements* is the partition of the plane by a set of distinct pseudo lines. Similar to line arrangements, any two pseudolines are only allowed to cross each other exactly once. Again the size is given by the number of curves.

If no three lines or pseudolines share one point the arrangement is called *simple*. So pseudoline arrangements are a generalization of line arrangements in terms of combinatorics and topology.

A good survey on the topic of pseudoline arrangement can be found in Chapter 5 of Handbook of Discrete and Computational Geometry by Goodman [TOG04, ch. 5].

2.1.1 Stretchability

Having a pseudoline arrangement at hand one can ask the question of whether the pseudolines can be stretched to lines without changing the combinatorics of the arrangement. This is called the STRETCHABILITY problem and pseudoline arrangements for which this is possible are called *stretchable*. Figure 2.1 shows a pseudoline arrangement and its stretched form, a line arrangement. By the way, we derived the notions rectangle arrangement and squarability from those terms.

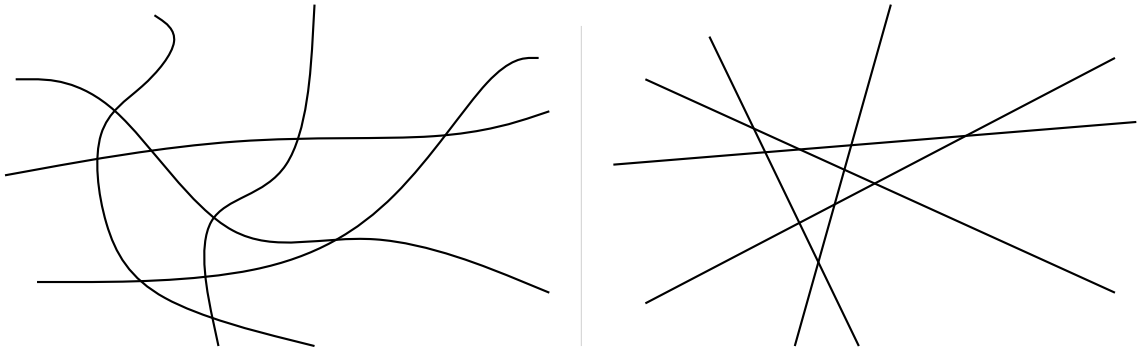

Figure 2.1: Left: A pseudoline arrangement.
Right: A combinatorial equivalent line arrangement, the stretched form.

Not every pseudoline arrangement is stretchable. One counterexample can be derived from the Pappus configuration shown in Figure 2.2 on the left. It can be constructed by pairwise connecting three points on one line, in the example A, B and C, with three points, X, Y, Z, on a second line, leaving out lines between the points in same order, like through A and X. Pappus' theorem says that geometry enforces the created crossing points P, Q and R to be collinear. Thus, a non-stretchable pseudoline arrangement can be achieved by replacing one line through three of the points by a pseudoline which only goes through two of them. One such example is shown in 2.2 on the right. Stretching this pseudoline would have enforce it to go through the third point, which would not be combinatorially equivalent.

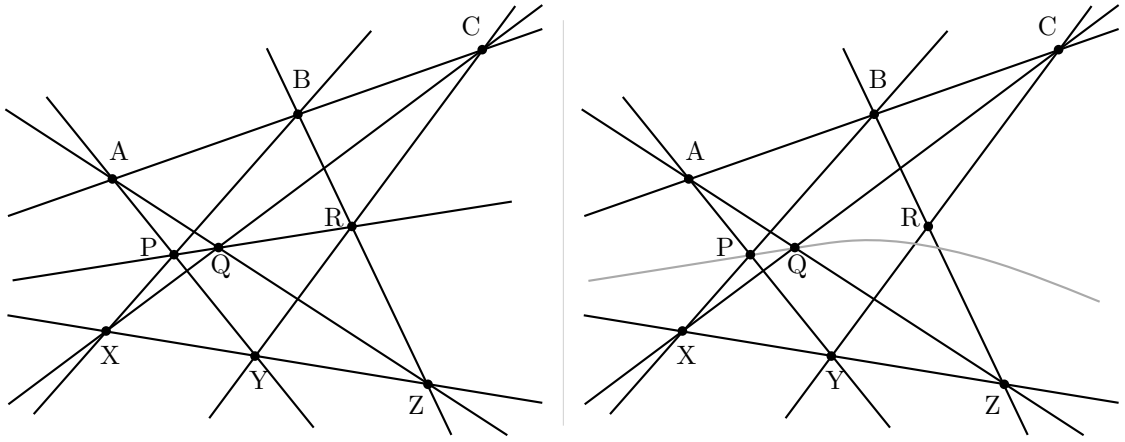

Figure 2.2: Left: The Pappus configuration.
Right: A non-stretchable pseudoline arrangement on the right.

Ringel [Rin56] converted this arrangement into a simple nonstretchable pseudoline arrangement. This example, the *non-Pappus arrangement*, is shown in Figure 2.3.

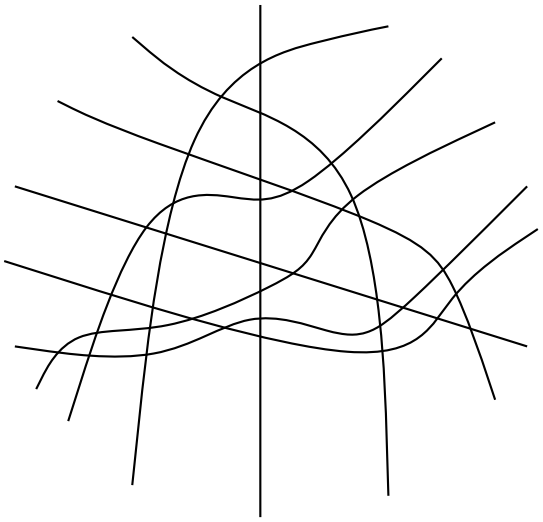

Figure 2.3: The non-Pappus arrangement, the simple nonstretchable arrangement of 9 pseudolines.

Goodman and Pollack [GP80] could show that these counterexamples are two of the smallest possible.

Theorem 2.1. *Every arrangement of 8 or fewer pseudolines is stretchable. [GP80]*

Since simple arrangements are more restricted than nonsimple ones, this also holds for all simple arrangements with at most 8 pseudolines. Moreover, Richter-Gebert [Ric89] proved that the non-Pappus arrangement by Ringel is the only simple nonstretchable one with 9 pseudolines.

Theorem 2.2. *Every simple arrangement of 9 pseudolines is stretchable, with the exception of the simple non-Pappus arrangement. [Ric89]*

Furthermore, Shor [Sho91] showed that in general the question of stretchability is $\mathcal{NP}$-hard.

2.1.2 Number of isomorphism classes

As line arrangements induce a cell complex covering the plane, we can consider two arrangements to be isomorphic or combinatorial equivalent if there is a one-to-one adjacency-preserving correspondence between the objects in their associated cell complexes [AdBMS98]. We can count the number of isomorphism classes for line and pseudoline arrangements of specific size. Table 2.1 shows some results on that.

With the previous theorems we can now generalize results on line arrangements to pseudoline arrangements. It is known how many isomorphism classes exist for line arrangements with at most 8 lines. Up to size 6 they are illustrated in Grünbaum's book Arrangements and Spreads from 1972 [Gr2]. Now, Theorem 2.1 yields that there are as many pseudoline arrangement isomorphism classes per size as line arrangements, up to size 8. Of course, the same holds for simple arrangements. Moreover, from Theorem 2.2 follows that there is only one isomorphism class of simple pseudoline arrangement with size 9 more than for simple line arrangements. In both cases we do not have to search for more pseudoline arrangements.

Isomorphism classes \ n	3	4	5	6	7	8	9	10
arrangements of n lines	1	2	4	17	143	4890		
simple arrangements of n lines	1	1	1	4	11	135	4381	312114
arrangements of n pseudolines	1	2	4	17	143	4890	461053	
simple arrangements of n pseudolines	1	1	1	4	11	135	4382	312356

Table 2.1: The number of isomorphism classes of line and pseudoline arrangements in relation to their size. [TOG04]

2.2 Maximum chromatic number of intersection graphs

One interesting problem started by Asplund and Grünbaum [AG60] is estimating the maximum chromatic number of an intersection graph depending on its clique number. For different geometric objects in different dimensions one is interested in lower and upper bounds on the maximum chromatic number. Thereby the geometric objects studied have varied widely, e.g. intervals, circles, squares, or convex bodies. The operations allowed to apply to these objects have also been differed. One may consider only translations or homothetic transformations (translation and scaling) of one origin object or of a whole set of various origin objects [KKN04], [Kos04], [Per03], [DJ12], [Hen98].

We consider only cases in 2 dimensions, upper bounds and where the objects are rectangles and squares. Looking only at translations of a given object would be the case of unit squares or equivalently translations of one origin rectangle in the plane. The more general case, considering homothets, fits our case of a set of squares which may have different sizes, but are still axis-aligned. In order to get to rectangles of variable sizes we have to consider the results on sets of homothetic copies of various origin objects.

We use the following notations. Le $\mathbb{F}$ be a family of graphs. We will consider families of intersection graphs over different geometric objects, like homothets of squares and rectangles, or only unit squares. Be further $\omega(G) = k$ the maximum clique number of considered graphs G in $\mathbb{F}$, i.e. the maximal size of subset of vertices in G all pairwise adjacent. Then the maximum chromatic number in terms of the maximum clique number k for a family of graphs $\mathbb{F}$ is denoted with $\chi(\mathbb{F}, k)$.

2.2.1 Linear bounds

Let $\mathbb{F}$ be the family of intersection graphs over convex objects in the plane, where we have one origin object and only allow translations in the plane. Kim and Kostochka [KKN04] proved that any graph $G \in \mathbb{F}$ with clique number $\omega(G) = k$ is $(3k-3)$-degenerate and therefore $\chi(\mathcal{T}, k) \leq 3k-2$. Thereby a graph is d-degenerate if every subgraph of it has a vertex with maximal degree d.

Interesting for us is the special case where $\mathbb{F}$ is the family of intersection graphs over unit squares. So, let now $\mathbb{F}$ be the family of these graphs. We then get by a result of Kim and Kostochka a linear bound on the chromatic number in terms of the clique number.

Theorem 2.3. *Let $\mathcal{U}$ be the family of intersection graphs over translations of the unit square, and $\omega(G) = k, k \geq 2$ be the maximum clique number of a graph $G \in \mathcal{U}$. Then $\chi(U, k) \leq 3k-2$, i.e. the maximum chromatic number of an intersection graph G over unit squares is at most $3k-2$. [KKN04]*

Going one step further we now consider homothets. Here translations and scaling of a origin object is allowed. Again Kim and Kostochka [KKN04] give us as a linear bound. Let $\mathcal{H}$ be the family of intersections graph over a family of homothetic copies of a convex

object in the plane. Further let again $\omega(G) = k, k \geq 2$, be the maximum clique number of a graph $G \in \mathcal{H}$. Then they could prove that $\chi(T, k) \leq 6k - 6$.

Again we consider a special case of this result for our interest. If the convex object is a square, without loss of generality axis-parallel, we look at intersection graphs over axis-parallel squares of different size.

Theorem 2.4. *Let $\mathcal{S}$ be the family of intersection graph over homothets of a square, and be $\omega(G) = k, k \geq 2$, the maximum clique number of a graph $G \in \mathcal{S}$. Then $\chi(\mathcal{S}, k) \leq 6k-6$, i.e. the maximum chromatic number of a intersection graph G over homothetic copies of a squares is at most $6k - 6$. [KKN04]*

2.2.2 Quadratic bounds

The intersection graphs of rectangles are often called box graphs. Thus, we denote the family of these graphs with $\mathcal{B}$. For $\chi(\mathcal{B}, k)$ the know upper bounds are, as far as we know, not linear. However, the known lower bounds are linear and can therefore not disprove that the upper bounds for rectangles may be linear as well. Note that bounds on the coloring number can not be given for rectangles by the degeneracy of the graph. One can easily construct a $K_{n,n}$ box graph.

Asplund and Grünbaum [AG60] proved that for triangle-free box graphs the maximum chromatic number is 6, i.e. $\chi(\mathcal{B}, 2) = 6$.

For $k \geq 3$ the best known upper bound is quadratic, which was proven by Hendel.

Theorem 2.5. *Let $\mathcal{B}$ be the family of intersection graphs of parallel boxes in the plane, i.e. axis-parallel rectangles. Further, let $\omega(G) = k, k \geq 3$, be the maximum clique number of a graph $G \in \mathcal{B}$. Then $\chi(\mathcal{B}, k) \leq 3k^2 - 2k - 1$, i.e. the maximum chromatic number of a intersection graph G over boxes in the plane is at most $3k^2 - 2k - 1$. [Hen98]*

This shows that the gap between the upper bounds for the chromatic number in terms of the clique number of square and rectangle intersection graphs is big. Since the squaring process does not change the combinatorics of a rectangle arrangement and therefore the chromatic number of its intersection graph, we can generalize the results from Theorem 2.4 and apply the linear upper bound on those rectangles arrangements which lie in Squares.

However, we know that rectangle arrangements with a cross are unsquarable and for those without a cross we can given a linear bound right away.

Lemma 2.6. *Let $\mathcal{B}'$ be the family of intersection graphs of rectangle arrangements without a cross and $\omega(G) = k, k \geq 3$, be the maximum clique number of a graph $G \in \mathcal{B}'$. Then $\chi(\mathcal{B}', k) \leq 4k - 3$.*

Proof. Let G be graph in $\mathcal{B}'$. Since we have no unsquarable intersection, we have that for every edge at least two corners of a rectangle lie inside another one. With n rectangles we have $4n$ corners. We get $|E(G)| \leq \frac{4n(k-1)}{2} = 2n(k-1)$, since every corner can lie in at most $k - 1$ different rectangles, where k is the maximum clique number, and every edge needs at least two corners inside another one. Together with the Handshake Lemma we get that G is $4(k-1)$-degenerate. Thus, G is $4k - 3$ colorable. □

Hence, the gap we can close for squarable rectangle arrangements for the upper bound on the chromatic number in terms of the clique number is not that big.

2.2.3 Other interesting results

Another interesting result is that the maximum chromatic number for intersection graphs of triangle-free rectangle arrangements is 3. This follows as a special case of a result by Perepelitsa [Per03]. He proved that every triangle-free intersection graph G of a finite number of compact connected sets A_i with boundaries that are piecewise differentiable Jordan curves and where for every i and j being $A_i \backslash A_j$ nonempty and connected is a plane graph. He then further used Grötzsch Theorem to prove that G is 3-colorable.

Well, rectangles are compact connected sets with piecewise differentiable Jordan curves as boundaries. Furthermore, in rectangle arrangements only the unsquarable intersection creates a pair of rectangles where $A_i \backslash A_j$ is disconnected. If we do not allow rectangles to lie completely inside each other, we get $A_i \backslash A_j$ is nonempty for all pairs of rectangles. Hence, if we avoid containing and unsquarable intersections, we can apply the result of Perepelista from above. However, containing intersections yield degree one vertices in the intersection graph and do therefore not interfere with the chromatic number.

Theorem 2.7. *The intersection graph of a triangle-free rectangle arrangement without unsquarable intersections has chromatic number at most 3.*

For box graphs in dimensions higher than 2 the chromatic number can not be bound, a result of Burling [Bur65]. This was quoted by Kostochka [Kos04], where he considers the problem of the maximum chromatic number for different types of intersection graphs in terms of the clique number. Fox and Pach [FP12] derived bounds depending both on the clique number and the size of the intersection graph.

3. Construction

3.1 Introduction

Besides the combinatorial aspects of rectangle arrangements and the SQUARABILITY problem, one may also consider the geometrical aspect. Doing so, we start with a given set of n rectangles in the plane and want to retrieve the combinatorial properties. We can assume that the rectangles are given only by their corner points. Moreover, we do not have any information about which rectangles intersect with each other or how they are ordered in the plane. So we start only with a bunch of rectangles given unsorted and only by points in the plane. In order to be able to get a combinatorial description of the rectangle arrangement, we first have to find all intersecting pairs and determine how they intersect.

This chapter is titled *construction*, because the rectangle arrangement as a problem instance of the SQUARABILITY problem is constructed from a set of geometrically given rectangles.

The aspect of finding the combinatorial characteristics or geometrical features of solely geometrically given objects is not uncommon in theoretical computer science, especially computational geometry. One example similar to our problem is the task of finding all intersections between a set of line segments in the plane. This problem arises for example when computing the union of different maps. These maps could be one for the forests of a country and one for the rainfall in the area. Searching for correlations between them one may want to find the union of the maps. Here the areas are mostly given as polygons which in turn are a list of line segments. Thus, the problem of merging the two maps breaks down to finding the intersections of line segments. [BCKO08, ch. 2]

Closely related is the construction of line arrangements, which we considered in the previous chapter. The construction problem can be generalized to any set of geometrical objects in the plane, for example, polygons. Algorithms for those problems are given by Shamos and Hoey [SH76], and later by Bentley and Ottmann [BO79], which make use of the algorithms for finding line segment intersections.

One straight forward algorithm is to simply check all pairs of rectangles for intersections. Even though this leads to the information we want to gather, the algorithm has an unnecessarily poor asymptotic behaviour, which is in $O(n^2)$. This follows directly from the fact that with n rectangles we have $\frac{(n-1)n}{2}$ unsorted pairs of rectangles, i.e. $O(n^2)$ pairs, and a check for intersections can be done in constant time, $O(1)$. The asymptotic behaviour may

seem good, since n rectangles can form up to $\frac{(n-1)n}{2}$ intersections and therefore finding all the intersections can sometimes not be done any better than in $O(n^2)$.

However, note that triangle-free rectangle arrangement can only have a linear number of intersections and therefore a quadratic behaviour seems not so good any more. In this chapter we give a brief overview of a sweep line algorithm which only needs $O(n \log n + k)$ time and $O(n + k)$ space, where k is the number of found intersections. So with $k \in O(n)$ for triangle-free rectangle arrangements, we get an asymptotic time $O(n \log n)$ and space $O(n)$.

There exists several other solutions for the problem of finding intersections between rectangles. For example Bentley and Wood [BW80] use a similar sweep line algorithm and Gutin and Wood [GW84] give a divide and conquer algorithm, both with the same asymptotic time behaviour as our algorithm.

3.2 Sweep line algorithm

Given a set of n axis aligned rectangles in the plane we want to find all intersections between them. The algorithm we present is a sweep line algorithm. So first, we have to explain what a sweep line algorithm is.

In a sweep line algorithm we sweep an imaginary line over the plane, for instance a horizontal line vertically from top to bottom. The line stops at some points, the so called events, which then are handled. These events could be the beginning or end of a geometrical object, an intersection point or critical point for the solution. The important invariant of the algorithm is that the part of the solution which lies in the area the line swept over is already computed at this point. During the sweep we keep the status of the sweep line in a data structure. This contains mostly the objects that cross the sweep line at this moment or have influence on the solution together with coming objects. The data structure usually keeps the objects sorted, such that inserting, deleting and finding objects can be done efficiently. Handling the events means updating this data structure, making checks between the geometrical objects currently in the data structure, or computing new events in the area not covered yet.

3.2.1 Reduction to one dimensional problem

Our algorithm for finding rectangle intersections is a vertical sweep line algorithm. For the sake of an easy description we assume that the rectangles are in general positions, i.e. no rectangle borders have the same x- or y-coordinate. Further, the algorithm has the top and bottom y-coordinate of rectangles as events. We will call these events `rectangle-start` and `rectangle-end` events. The first step of the algorithm is to sort them by their y-coordinate. In Figure 3.1 the situation of the sweep line is illustrated before the first event. Furthermore, for a fast access, the events have pointers to their corresponding rectangles.

Sweeping from top to bottom we proceed now as follows. When we come to a `rectangle-start` event, we insert the x-interval of the corresponding rectangle in the status data structure. This is an interval tree which we will discuss in the next subsection (3.2.2). At `rectangle-end` events we delete the corresponding interval from the interval tree. Thus, the interval tree contains all vertical intervals which intersect the sweep line at this very moment.

The crucial step for our algorithm now is, that at every `rectangle-start` event we also check if the added interval intersects with any other intervals currently in the interval tree. Since the interval tree only keeps the x-intervals of rectangles intersecting the sweep line

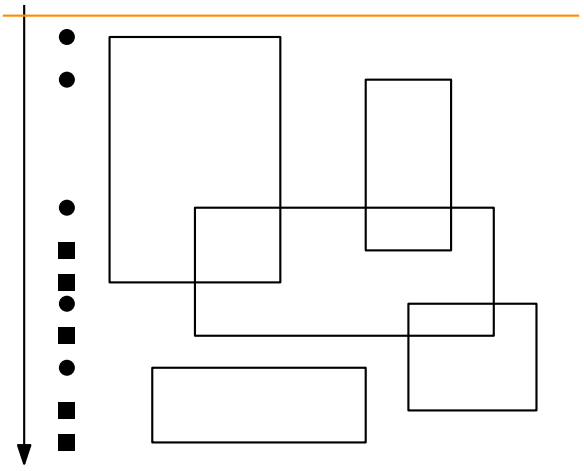

Figure 3.1: Situation in the sweep line algorithm for finding rectangle intersections before the first event. The `rectangle-start` events are marked with ● symbols, the `rectangle-end` events with ■.

at this y-position, an intersection of the intervals means that the corresponding rectangles intersect as well. We observe further that this algorithm finds in fact all intersections. When the x-interval $[a, b]$ of a rectangle $A = [a, b] \times [c, d]$ is added to the interval tree, we find all intersection between A and rectangles which start above A. The intersection of A with a rectangles $B = [a', b'] \times [c', d']$, which starts below, i.e. $d' < d$ is found when the $[a', b']$ is added to the interval tree. This happens before $[a, b]$ is removed from the interval tree. Thus, we have reduced the two-dimensional problem of finding rectangle intersections to finding interval intersections, a one-dimensional problem. Therefore, if we can solve this problem, we can find all intersections between axis-aligned rectangles. Our in other words, the correctness of our algorithm follows from this reduction and the invariant of the sweep line algorithm, that everything above the sweep line is already handled. The complete algorithm is given below in pseudo code.

Algorithm: FindIntersections
Data: axis-aligned rectangles R
Result: all intersections between rectangles in R
$Q \leftarrow$ new list
$T \leftarrow$ new interval tree
foreach *rectangle* r *in* R **do**
  add rectangle-start and rectangle-end event for r to Q
end
sort Q
foreach *event* q *in* Q **do**
  $I_x \leftarrow x-$interval of rectangle r for event q
  if q *is rectangle-start event* **then**
    search in T for intersections with I_x
    add I_x to T
  end
  if q *is rectangle-end event* **then**
    remove I_x from T
  end
end

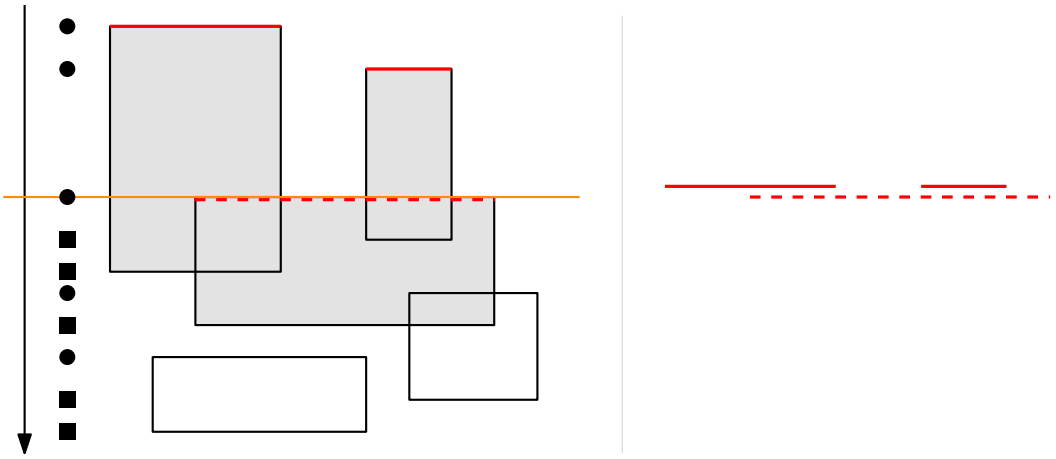

Figure 3.2: The sweep line algorithm at a `rectangle-start` event. Three rectangles, the grey ones, intersect with the sweep line. In the data structure on the right we have their corresponding intervals. The dashed interval for the newly added rectangle intersects with both other intervals, which is correct since the corresponding rectangle intersects with the two other rectangles as well.

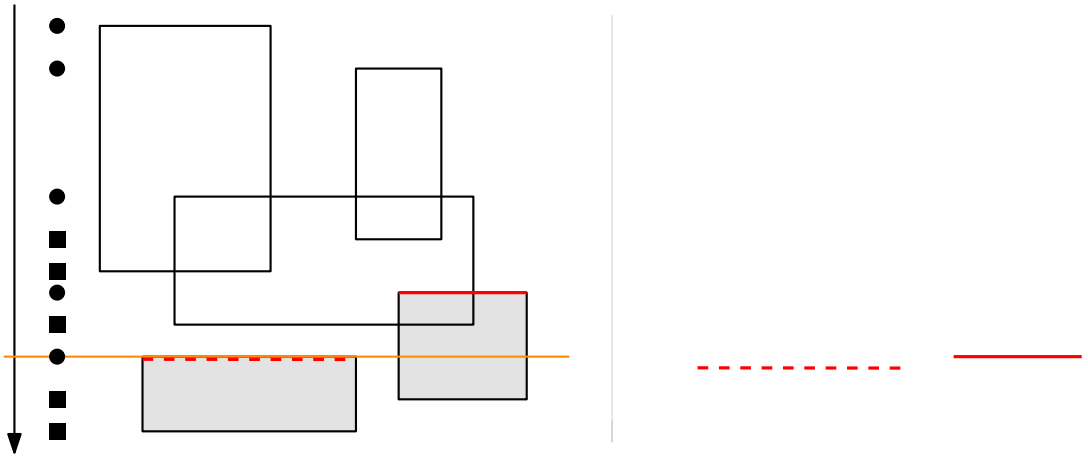

Figure 3.3: The algorithm at a later `rectangle-start` event. Since rectangles above the sweep line can not form any new intersections with rectangles below the sweep line, they are no longer in the status data structure.

Figure 3.2 shows the sweep line at a `rectangle-start` event, with the intervals currently in the data structure on the right. In Figure 3.3 the algorithm is visualized at a later point. There, the intersections between the rectangles starting above the sweep line are already found.

It is further worth mentioning, that if we found an intersection, we can easily compute how the two rectangles intersect. Determining the four intersection types containing, side-piercing, opposite-corner and unsquarable can be done done by comparing their corner coordinates and therefore in $O(1)$ time.

Before we look at the required time and space, we take a look at the interval tree data structure.

3.2.2 Data structure interval tree

Now we take a closer look at the data structure we use for storing the intervals during the sweep. It must be efficient for inserting and deleting intervals as well as for finding

intervals which intersect a given interval. One data structure providing this functionality is the *interval tree*.

A good explanation how interval trees work is given by de Berg et al. [BCKO08, ch. 10]. For the sake of completeness, however, we give a short overview of this description.

First of all, an interval tree is a binary search tree. We assume the intervals are all part of the x-axis. Then, a vertex represents the coordinate x, chosen as the end of one interval. Moreover, it stores all the intervals which intersect this point. In its left subtree all intervals which lie completely left of x are handled, in the right subtree those completely to the right of x. To get a balanced tree, the represented x-coordinate is recursively chosen as the median of all endpoints the tree or subtree stores.

Figure 3.4 shows an interval tree for seven intervals, where the left and right subtree consist only of one vertex.

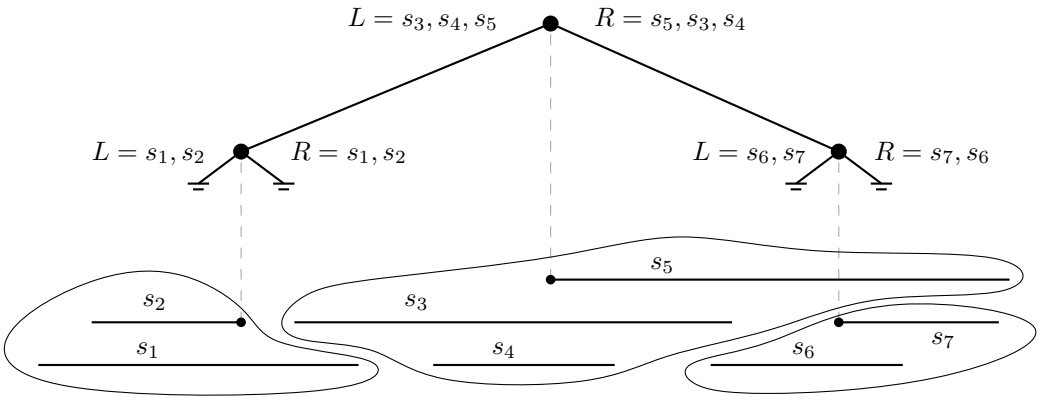

Figure 3.4: Interval tree for seven intervals. The root vertex is chosen as the median of all interval ends. In the left subtree are all intervals which lie completely left of it, on the right it is analogous. The lists at every vertex store the endpoints of the intervals which intersect the x-coordinate of the vertex with decreasing distance to it. This allows a fast search in the lists, only looking at the crucial intervals.

The intervals which contain the point x of a vertex v of the tree are stored as follows. We have two lists at each vertex, one for left endpoints of these intervals, and one for the right endpoints. In the left list L the endpoints are stored sorted by decreasing distance to the x, on the right in R accordingly. In the example given in the Figure 3.4 these lists are also illustrated. Note that any interval is only handled in one vertex.

Since a binary search tree on m elements needs at most $O(m)$ space and every interval has its endpoints only stored once in a list, the lists together need also only space $O(m)$. Hence, we can state the following.

Lemma 3.1. *An interval tree for m intervals requires space in $O(m)$.*

3.2.2.1 Finding intersecting intervals

The procedure for finding an interval now explains why the interval endpoints are stored in decreasing distance. Assume we search for all intervals that intersect a given query interval s, like in Figure 3.5 the interval s_8. At every vertex of the tree, there are three possibilities. The interval lies completely to the left, completely to the right or contains the point x stored at representative by this vertex. In the two first cases we proceed as follows. If the interval lies completely to the left, a recursive call on the left subtree is made. Moreover

we check the list L, if the intervals contain the right endpoint of s. Since the list is sorted with decreasing distance to x, we know that we only have to search through the list until we find the first interval which does not contain this endpoint. All later intervals would lie more to the right and could therefore not intersect with s. The case that s lies right of x is handled analogously.

In the third case, if s intersects x, we know that the query interval intersects all intervals in the two lists L and R. Hence we can report them all. Stopping here would not yield all requested intervals. We now have to check if the left point of s intersects with intervals which lie completely left of x, so in the left subtree of v. The same applies for the right point of s and the right subtree of v. Again we can use the sorted lists in the vertices of the subtrees to find all the intervals without looking at more than $O(1)$ list elements at each tree level.

Since the lists are sorted and the tree divides the intervals at every vertex recursively in three groups, the described procedure finds all intersections of s with our intervals in the tree.

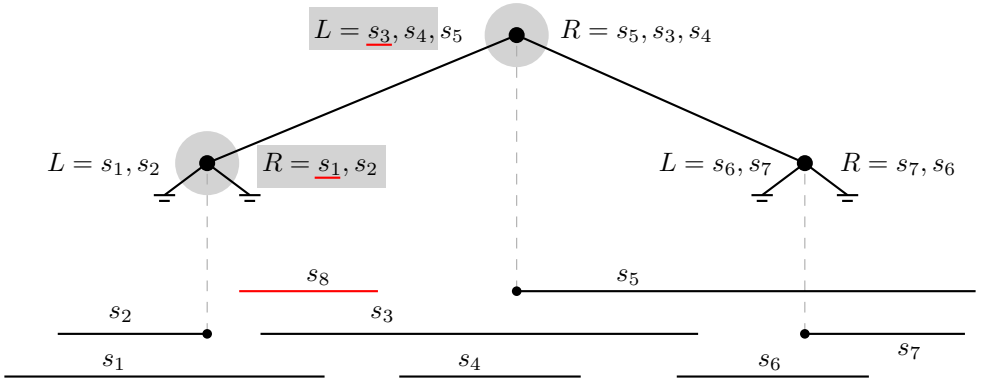

Figure 3.5: The interval tree from Figure 3.4 and a query interval s_8. The grey vertices and the lists are those, at which we have to take a look during the search for intersections. s_3 in the left list L of the root vertex is underlined red, marking that we found an intersection. As s_4 does not intersect, the search can stop here and does not have to look at s_5.

Now we look at the required time. A balanced binary search tree on m elements has height in $O(\log m)$. The checks in each vertex can be done in $O(1)$ time and the searching in the lists takes all together $O(k_i)$ time, where k_i is the number of found intervals intersecting with s.

Inserting and deleting an interval basically works the same way. It is worth mentioning that when scanning the left and right lists, we may have to look again at all k_i intervals which intersect with the handled interval. It is crucial to maintain that the tree is balanced, which we required by the choice of medians for handled intervals in a subtree. As de Berg et al. also mention [BCKO08, ch. 10], the balancing can be done by replacing the binary search tree with a red-black tree or other balanced binary search trees. This is possible without additional cost in asymptotic time for operations or space required by the tree. However, both has no influence on the asymptotic time behaviour for the operations. Hence we can conclude the following.

Lemma 3.2. *Insert and find intersection operations for an interval s in an interval tree of size m can be done in $O(\log m + k_i)$ time, where k_i is the number of intersections between s and other intervals in the tree. Delete operations for an interval s in an interval tree of size m can be done in $O(\log m)$ time.*

3.2.3 Complexity

With Lemma 3.2 and Lemma 3.1 we have statements about interval trees, which we will now use to say something about the required time and space of our sweep line algorithm. Let n be the number of rectangles.

The algorithm, given in Listing 1, first sorts all $2n$ events. This can be done in $O(n \log n)$ time. Handling each event requires one or two operations on the interval tree, which by Lemma 3.2 can be done in $O(\log m + k_i)$ time, where k_i is the number of intersections interesting for these operations as in Lemma 3.2. Since the tree can not contain more than n elements and the k_i sum up to overall number of intersections k, each step can be done in $O(\log n)$ time plus all together $O(k)$ time for all steps. Like mentioned before, determining the way two rectangles intersect can be done in constant time and has no further influence on the asymptotic time behaviour of the algorithm. With $O(n)$ many events we get the first part of the theorem below.

By Lemma 3.1 the interval tree needs $O(n)$ space and the intersections $O(k)$. Together we get the second part of the following theorem.

Theorem 3.3. *The sweep line algorithm to find all intersections between n rectangles needs $O(n \log n + k)$ time and $O(n + k)$ space, where k is the number of intersections.*

3.3 Conclusion

If we have no further information about the given rectangles, the sweep line algorithm may be no better than the naive variant, since it is possible that $O(k) = O(n^2)$. However, if we know that the rectangles are triangle free, we get that there can be at most $O(n)$ many intersections. This follows from the fact, that the corresponding intersection graph, which we examine extensively in the next chapter, is planar. Thus, the algorithm needs $O(n \log n)$ time and $O(n)$ space. This is way more satisfying then an $O(n^2)$ algorithm.

Figure 3.6 shows that it is possible for non triangle free rectangle arrangements, even for square arrangements, to have $O(n^2)$ intersections, even without containing, side-piercing and unsquarable intersections.

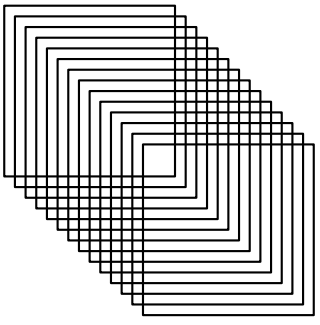

Figure 3.6: Possible arrangement for n squares to form $O(n^2)$ many intersections.

Besides, during the algorithm we can easily determine whether the arrangement is triangle free. Furthermore, we can not only determine the intersections but also compute the corresponding intersection graph. If the rectangles form a triangle-free rectangle arrangement, we can construct the corresponding embedding of the intersection graph as well. The only thing we have to do additionally is to follow the order of the intersections, when adding edges between corresponding vertices.

4. Characterisation of rectangle arrangements

In the last chapter we have shown how to retrieve combinatorial information from a geometric representation. Now, in this chapter we show a way to describe the resulting rectangle arrangement. Here we are only interested in the characteristic combinatorial properties not the actual geometrical drawing. We give a combinatorial description using an enhanced intersection graph for triangle-free rectangle arrangements. Furthermore, we prove what properties are required to give a full characterisation.

4.1 From rectangle arrangement to intersection graph

Let $\mathcal{R}$ be a triangle-free rectangle arrangement without a cross as rectangle subarrangement. Remember, that we consider equivalence classes of rectangle arrangements and do not care about the actual size of rectangles, only about the relations between them. We omit the unsquarable intersection for our characterisation, since it directly makes the whole rectangle arrangement unsquarable and therefore less interesting for us. However, for a triangle-free rectangle arrangement without it we can give a well-defined combinatorial description which captures all the characteristic combinatorial properties.

4.1.1 Intersection graph

The base of our combinatorial description is the intersection graph $G_{\mathcal{R}} = (V, E)$ induced by $\mathcal{R}$. This means that for every rectangle $A \in \mathcal{R}$ we get a vertex in the vertex set of $G_{\mathcal{R}}$, $v_A \in V(G_{\mathcal{R}})$, and for every pair of rectangles $B, C \in \mathcal{R}$ with nonempty intersection, i.e. $B \cap C \neq \emptyset$, we have an edge between the corresponding vertices $v_B, v_C \in V$.

We introduce directed and coloured multi-edges for the different intersection types. Thus $E(G_{\mathcal{R}})$ will be capable of describing the intersections. We use the embedding induced by $\mathcal{R}$ for $G_{\mathcal{R}}$ to fix the intersection orders.

A formal definition of the intersection graph is given later (Definition 4.2) after we have examined its planarity, the edge set and the embedding more detailed.

4.1.1.1 Planarity of intersection graph

Before we get more detailed, we prove that the induced embedding of an intersection graphs for a triangle-free rectangle arrangements has the following nice properties.

Lemma 4.1. *Every triangle-free rectangle arrangement without a cross has a planar intersection graph and induces a plane embedding.*

Proof. Be $\mathcal{R}$ a triangle-free rectangle arrangement without a cross and be $G_{\mathcal{R}}$ its intersection graph. Since containing intersections result in degree one vertices, they can be ignored for the planarity as well as for the characterisation. So we can assume $\mathcal{R}$ contains no containing intersection.

Since $\mathcal{R}$ is triangle-free and we do not care about containing intersections, we can place the vertex $v_A \in V(G_{\mathcal{R}})$ for any rectangle A at a point solely covered by A. Figure 4.1 illustrates that at v_A the adjacent edges are, induced by the arrangement, in a circular order.

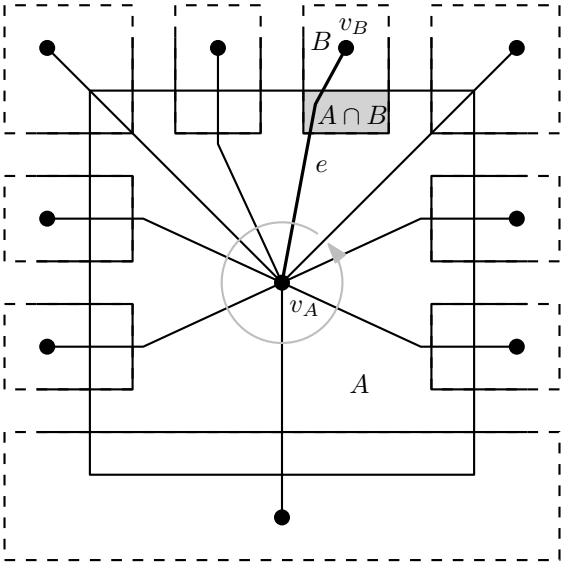

Figure 4.1: Induced edge order around vertex v of the intersection graph.

Let $e \in E(G_{\mathcal{R}})$ be the edge connecting the v_A and v_B. Since e only exists because $A \cap B \neq \emptyset$, we know that it can be drawn completely into A and B. Starting at v_a, going through the intersection $A \cap B$ and ending at v_B. Since A, B and e were chosen arbitrarily and by respecting the induced circular edge orders around vertices, we conclude that $G_{\mathcal{R}}$ with the induced embedding is plane. □

4.1.1.2 Directed and coloured edges for the intersections types

Whereas it is easy to describe the vertex set $V(G_{\mathcal{R}})$, we now use $E(G_{\mathcal{R}})$ to store the information about the intersection types.

Like mentioned in the proof of the previous lemma, we can ignore containing intersections. They have no effect on the embedding or any face. For a squaring of a triangle-free rectangle arrangement enclosed squares can be placed in any free space of the enclosing square. We also exclude the cross . This leaves us with only side-piercing and opposite-corner intersections. Ignoring the roles two rectangles can take in them, there are 2 different opposite-corner intersections and 4 different side-piercing intersections. Thus, we have to able to describe 6 different intersections. One way to distinguish all these intersections could be to just assign each of them a different color or number and give their corresponding edges in the intersection graph the same. We choose a more illustrative and by the geometry of the rectangles inspired approach.

We assign each corner of the rectangles a specific color (also represented by a number), as illustrated in Figure 4.2. We assign green, orange, blue and red starting at the upper right corner and continue counterclockwise.

- Upper right corner: **green = 1**
- Upper left corner: **orange = 2**
- Lower left corner: **blue = 3**
- Lower right corner: **red = 4**

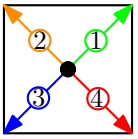

Figure 4.2: Colour encoding for the corners of a rectangle.

Only considering side-piercing and opposite-corner intersections we observer the following. In both intersection types exactly two corners of the participating rectangles lie inside the other one. This means that we can describe every pair of intersecting rectangles with two directed edges. So an edge in $E(G_{\mathcal{R}})$ becomes an edge pair. We direct an edge for each of the two corners inside the other rectangle and give it the colour of the corner.

We consider the example of Figure 4.3, simply an opposite-corner intersection. Rectangle A has its upper left corner inside another rectangle B. Thus we set an orange edge e_1 from $v_A \in V(G_{\mathcal{R}})$ directed to $v_B \in V(G_{\mathcal{R}})$. B has its lower right corner in A, inducing an edge e_2 in red towards v_A. The whole opposite-corner intersection is now described by the edge pair $\{e_1, e_2\}$.

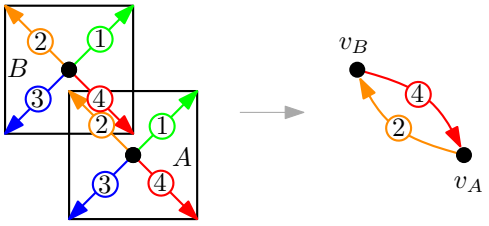

Figure 4.3: A opposite-corner intersection between A and B and the corresponding edge pair between v_A and v_B.

For the side-piercing intersection illustrated in Figure 4.4, where A pierces B into the right side, we get two edges both directed to v_B. One edge in orange for the upper left corner, one in blue for the lower left corner of A.

As already mentioned the edges of the intersection graph $G_{\mathcal{R}}$ are actually edge pairs, more precisely $E(G_{\mathcal{R}}) := \{\{e_1, e_2\} \mid \{e_1, e_2\} \text{ describes an intersection of } \mathcal{R}\}$. This implies that the edge pairs $\{e_1, e_2\}$ are always two directed and coloured edges between the same two vertices $v_A, v_B \in V(G_{\mathcal{R}})$ corresponding to the two rectangles $A, B \in \mathcal{R}$. Obviously this enables to encode all 6 different intersections possible by side-piercing and opposite-corner intersections.

4.1.1.3 Induced embedding for intersection order

Lemma 4.1 tells us that the intersection graph $G_{\mathcal{R}}$ is planar. Using edge pairs does not compromise this fact. The lemma shows also that the embedding induced by the

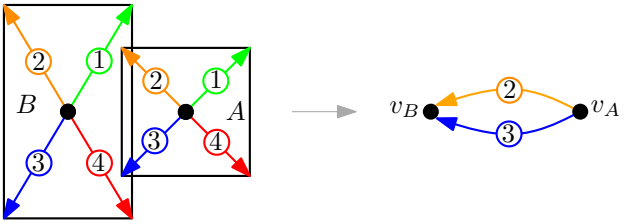

Figure 4.4: A side-piercing intersection between A and B and the corresponding edge pair.

arrangement serves well for this purpose. A plane embedding of a planar graph can be described by the circular order of the adjacent edges for every vertex. Thus, we set the embedding of $G_{\mathcal{R}}$ as the one induced by $\mathcal{R}$. This means that for every vertex $v_A \in V(G_{\mathcal{R}})$ its circular edge order is given by the intersection order of $A \in \mathcal{R}$.

Figure 4.5 shows an example, where just the left embedding of the intersection graph corresponds to the rectangle arrangement.

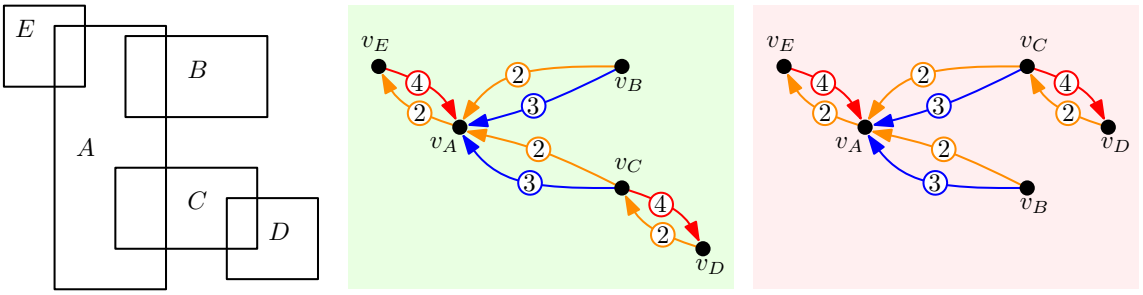

Figure 4.5: A rectangle arrangement $\mathcal{R}$ on the left and two graphs, which only differ in their embedding. Only the left graph with the induced embedding of $\mathcal{R}$ can be $G_{\mathcal{R}}$.

However, only a circular edge orders for every vertex is not sufficient. We defined the intersection order for a rectangle as the sequence of border intersections starting at its upper right corner. We have to define a start of the circular edge order for the edges at a vertex as well. Otherwise the graphs for two different triangle-free rectangle arrangement could be the same. Such a case is illustrated in Figure 4.6. This problem can only occur when a rectangle has multiple side-piercing intersections at the same side.

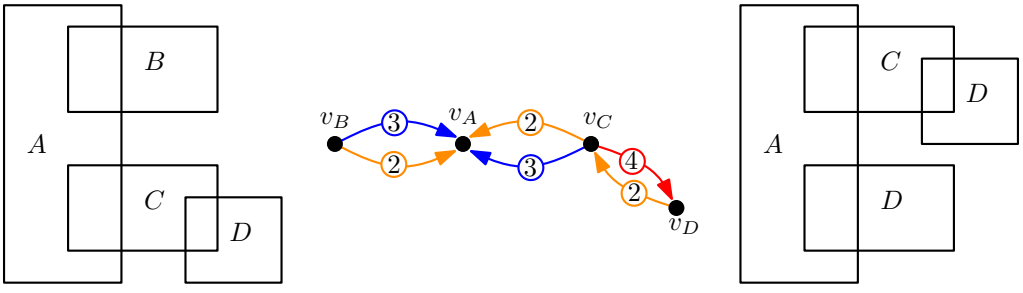

Figure 4.6: Two different (not combinatorially equivalent) triangle-free rectangle arrangements with a similar intersection graph. Only when considering a start for the circular edge orders they can be distinguished.

If we consider the edge order around a vertex we can ignore the fact that every edge is actually an edge pair. The two edges of an edge pair are always next to each other and non-crossing. Nevertheless we define orders between the two edges of an edge pair.

- If $\{e_1, e_2\}$ between v_A and v_B describes an opposite-corner intersection, then the outgoing edge is more counterclockwise then the incoming edge both at v_A and v_B.
- If $\{e_1, e_2\}$ between v_A and v_B describes a side-piercing intersection with A piercing B, then at v_A they are in the order of the corners, i.e. one of the four possible cases counterclockwise $1-2, 2-3, 3-4, 4-1$. At v_B they are of course in the opposite order.

4.1.2 Corner color intersection graph

We have now everything to define this enhanced intersection graph describing all combinatorial properties of a triangle-free rectangle arrangement.

Definition 4.2. *A **corner color intersection graph** of a triangle-free rectangle arrangement $\mathcal{R}$ with no cross is its corresponding intersection graph $G_{\mathcal{R}} = (V, E)$, with*

- $V := \mathcal{R}$ *as set,*
- $E := \big\{\{v_A v_B, v_B v_A\} \;\big|\; v_A v_B \in E_i, v_B v_A \in E_{i+2(4)}\big\}$
  $\cup \big\{\{v_A v_B, (v_A v_B)'\} \;\big|\; v_A v_B \in E_i, (v_A v_B)' \in E_{i+1(4)}\big\}$,
  where $v_A v_B \in E_i = E_{i(4)} \Leftrightarrow$ *corner i of A contained in B*
- *the embedding induced by $\mathcal{R}$ including starting points for the circular orders.*

It encodes the types of intersections determined by the piercing corners via the colours of its edges and their direction.

With respect to the fact that we consider equivalence classes of rectangle arrangements we can state that by construction every triangle-free rectangle arrangement has its own unique corner color intersection graph. So the following lemma follows directly from the previous definition.

Observation 4.3. *A triangle-free rectangle arrangement without a cross induces a unique corner color intersection graph.*

4.1.2.1 Other intersection types

Like mentioned before we can assume that we have no containing intersection. However, we could describe an enclosed rectangle in a triangle-free rectangle arrangement with a degree one vertex in $G_{\mathcal{R}}$, with the only edges being four multi-edges with each distinct colours all directed to the vertex of the enclosing rectangle.

The corner color intersection graph is not capable of describing an unsquarable intersection. Since no corner of the two rectangles lies inside the other one, there would be no edge in the graph.

4.1.2.2 Non triangle-free rectangle arrangements

We use the induced embedding for $G_{\mathcal{R}}$ to describe the intersection orders of rectangles. This explains why we only look at triangle-free rectangle arrangements. A rectangle arrangement which is not triangle-free is illustrated in Figure 4.7. We observe that its intersection graph has no unique embedding. The intersection order does not yield a natural edge ordering for the vertices and with that embedding, as it does for triangle-free rectangle arrangements.

Moreover, with three or more rectangles sharing one point the intersection graph can also be not planar at all.

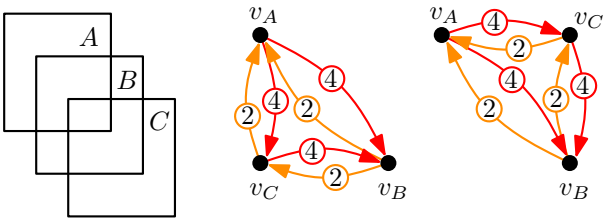

Figure 4.7: Three pairwise intersecting rectangles yielding no unique embedding for the intersection graph.

4.2 Properties of corner color intersection graphs

Our aim is to characterize the corner color intersection graph with a small set of combinatorial properties. First we list some necessary conditions. Even though we have seen most of the properties before we want to examine them from this point of view.

4.2.1 Basic properties

First of all, the graph G has a vertex set V, an edge set E and a plane embedding, the latter given by the order of the edges around every vertex. The planarity is needed since every triangle-free rectangle arrangement is planar by Lemma 4.1. Furthermore, if we want to interpret the graphs as corner color intersection graph of exactly one triangle-free rectangle arrangement, we have to look at the graphs and one of its designated plane embeddings. As we have seen, different embeddings, if legal, result in different rectangle arrangements.

Furthermore, G has to be triangle-free, but we will ensure this with a property of the edges. We can summarize this with the following lemma.

Observation 4.4. *A graph G can be a corner color intersection graph of a triangle-free rectangle arrangement with no cross only if G is plane and triangle free.*

4.2.2 Edge properties

Every edge is actually an edge pair of two directed and coloured edges. They can have only one of the four corner colours, but not every colour and direction combination is possible. Besides this, they also have to be in a geometrically logical order. These requirements are described in the following.

4.2.2.1 Legal edge pairs

Between two rectangles there are six different intersections possible, two opposite-corner intersections and four side-piercing intersections. Of course in all of them the two rectangles can take each role. Nevertheless, this results only in edge pairs, where the two edges have the same direction and neighbouring colours, e.g. red and green, for the side-piercing intersections. Or two edges with opposite directions and with opposite colours, e.g. red and orange, for the opposite-corner intersections.

Observation 4.5. *A graph $G = (V, E)$ can be a corner color intersection graph of a triangle-free rectangle arrangement with no cross only if the edges E consists of edge pairs of one of the two following types:*

- *Bidirectional edge $e = (vw, wv)$ in opposite colours, i.e. $|color(vw) - color(wv)| = 2$*
- *Unidirectional edge $e = (e_1, e_2)$ with neighbouring colours, i.e. $|color(e_1) - color(e_2)| = 1 \mod 2$*

4.2.2.2 Legal edge order

Now we take a look at the order of edge pairs around one vertex. This only makes sense since we have always the embedding of the graph in mind as well. So it is really important that the order of edges respectively edge pairs can have a geometrical equivalence. For example, an edge pair with an outgoing red edge at v can not lie between two edge pairs with both incoming blue and orange edges. Geometrically this would mean that the lower right corner of the rectangle for v would lie between two rectangles which both pierce the right side of the rectangle. This is of course not possible. Therefore we need some requirements for the order of these edge pairs around a vertex.

First of all, we can demand that both edges of an edge pair are directly next to each other and are non crossing. Even though it is not completely necessary but makes it easier for us to handle the graphs, we defined the order between two edges of an edge pair above in Section 4.1.1.3. For a bidirectional edge pair $e = (vw, wv)$ between v and w, i.e. of an opposite-corner intersection, we defined that both at v and w the incoming edge is more clockwise, so to the right, than the corresponding outgoing edge. For example at v an incoming red edge, $color(wv) = 4$, lies directly right of an outgoing orange edge, $color(vw) = 2$. For an unidirectional edge pair $e = (e_1, e_2)$ from v to w we defined that at v they are in the order of the corners,i.e. one of the four possible cases counterclockwise $1-2, 2-3, 3-4, 4-1$.

Secondly, we observe that among all pairs around one vertex, an outgoing edge in one particular colour can occur at most one time. Otherwise a corner of a rectangle would lie inside more than one other rectangle. A contradiction to the fact that we only consider triangle-free rectangle arrangements. Furthermore the outgoing edges have to be in the order of the corner colours, so in clockwise order blue, orange, green and red.

Third, we can demand the order of edge pairs with two incoming edges. We consider two different edge pairs adjacent to v with outgoing edges of neighbouring colours, say counterclockwise 2 and 3. Then the only edge pairs which can come in between them, are those with the two different colours, so only edge pairs of colors 1 and 4. If there is an edge pair at v with two outgoing edges, meaning that two corresponding corners of a rectangle form a side-piercing intersection, no edge pair of the two left colours can come in at v. To put it briefly for the remaining cases, if there are not all edge pairs with outgoing edges, then the unidirectional incoming edge pairs have the order as if the outgoing would also exist.

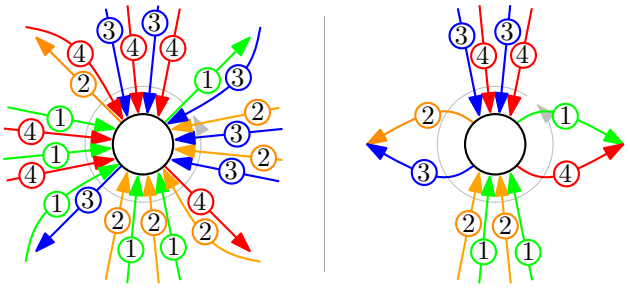

Figure 4.8: Two possible edge pair orders around a vertex. The gray arrow indicates the circular edge order starting at the top right corner.
Left: All four outgoing edges in different edge pairs and possible incoming edge pairs between them.
Right: Two completely outgoing edge pairs.

Last but not least, as discussed before we have to take the starting point for the circular orders into consideration as well.

So to put this all together we can only have edge pair orders of similar kind like those illustrated in Figure 4.8. All these necessary requirements a summarized with the following observation.

Observation 4.6. *A graph $G = (V, E)$ can be a corner color intersection graph of a triangle-free rectangle arrangement with no cross only if its edges respectively edge pairs have to have the following properties:*

- *The out degree in every colour of every vertex $v \in V$ is at most one.*
- *The edges around $v \in V$ every vertex are in the following order:*
  - *The outgoing edges are in the order of the corner colours, counterclockwise green, orange, blue and red respectively 1,2,3,4.*
  - *Between the two edges of an unidirectional outgoing edge pair can not be any incoming edge in between.*
  - *Incoming edges are either the second of bidirectional edge pair and then right of it, or are in an unidirectional edge pair in the order between the (maybe not existing) outgoing edges of the two different colours.*
- *If there are no outgoing edges and only incoming edge pairs of the same two colors, there is a dummy edge at v preventing the pairs to permute around v.*

4.2.3 Face properties

The described properties of the edges are not sufficient for our planar, triangle-free graph to have a corresponding triangle-free rectangle arrangement. Figure 4.9 shows a graph which has all the required properties stated above, meaning has legal edge pairs, out degree in every colour is at most one, has correct edge order, is planar and triangle-free. Nevertheless, the geometry does not allow the four rectangles for the four vertices to be intersecting in the way the edges demand it.

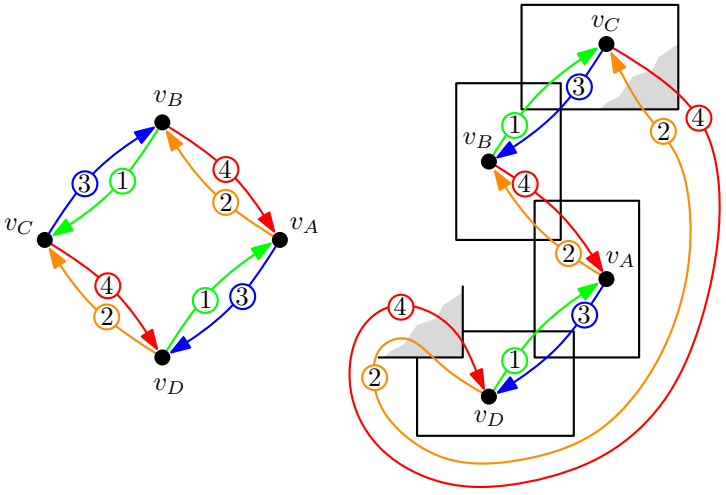

Figure 4.9: A planar and triangle-free graph with correct edge pairs and edge order, but still without corresponding geometrical representation in form of a triangle-free rectangle arrangement.

A face in a rectangle arrangement is a connected region not covered by rectangles. Since the rectangles are axis aligned, it is obvious that the border of a face in a rectangle arrangement is given by several "stairs". Figure 4.10 gives an example on that. Furthermore, the steps of adjacent stairs have to be given by adjacent corners. So there are 4 different stair

types and in order to close correctly the sequence of stairs has to be circular and can only switch between neighbouring colors. Moreover, the sequence can not go more than one time through all four colors in their order without going the same amount backwards. It is obvious that otherwise the stairs could not be the border of one face.

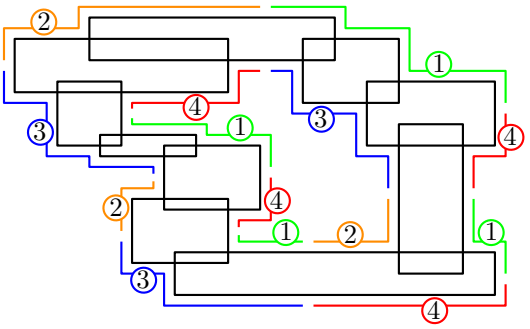

Figure 4.10: A planar and triangle-free graph with correct edge pairs and edge order, but still without corresponding geometrical representation in form of a triangle-free rectangle arrangement.

Thus, in addition to the previous properties, we need to bring the geometrical component of rectangle arrangements into the graph in a more global way, not just locally around one vertex. Since every face f of the graph correlates with a face $f_{\mathcal{R}}$ of the rectangle arrangement, f has to imply a correct description of $f_{\mathcal{R}}$. For this purpose we count the number of turns that are represented by the vertices in the graph along a face. On an inner face we go counterclockwise, both in the graph face as in the arrangement face, and on the outer face we go clockwise. This implies that one left turn in the graph should represent one stair change with increasing color, i.e. from 1 to 2 or equivalently. One right turn should represent a stair change with decreasing color. Two right turns should represent that at one rectangle two stair changes occur, what is only possible with decreasing color. And lastly, if the graph does not represent a turn at a vertex for a face, this should mean that in the arrangement the stair does not change.

Next, we define turns more precisely and when a graph has this required property for its faces. Then, in the following section, we show that it is not only necessary to have these face properties, but together with the basic and edge properties from above also sufficient for a graph in order to be a corner color intersection graph of a triangle-free rectangle arrangement.

4.2.3.1 Angle number

For a pair (f, v), consisting of a face f and vertex v incident to f, we define the *angle number* $\alpha(f, v)$ to describe the number of left turns the edge sequence along f takes at v. This could be $1, 0, -1$ or -2, where negative numbers mean right turns.

The angle number is determined by the colours and directions of the two edges of the two different edge pairs incident to the vertex and the face. Thereby it is important whether an edge is directed clockwise or counterclockwise along the face. We define a helping function c taking this into account in respect to the colors of the edges. Let $G = (V, E)$ be a graph satisfying basic and edge properties, $v \in V$ incident to a face f of G. Let e be an edge of an edge pair from E incident to v and f.

$$\mathrm{c}_{f,v}(e) := \begin{cases} color(e) \mod 4, & \text{if } e \text{ is directed counterclockwise along } f \\ color(e) + 2 \mod 4, & \text{if } e \text{ is directed clockwise along } f \end{cases}$$

Now let the two edges e_1 and e_2 be part of edge pairs, which are both incident to v and f, and with e_2 counterclockwise directly after e_1 along the border of f. This means if we go counterclockwise along the border of f, we first come to e_1 with f to its left and its partnering edges of its edge pair are to its right. Then comes v and third comes e_2, again with f to its left. Since a vertex can appear in the border of a face multiple times between different edges, the angle number is also depending on e_1 and e_2.

Definition 4.7. *The **angle number** $\alpha(f, v, e_1, e_2)$ for the face f at vertex v and e_1 and e_2 edges incident to v and f, with e_2 clockwise directly after e_1 in the circular edge order of v, is*

$$\alpha(f, v, e_1, e_2) := \begin{cases} 1, & \text{if } c_{f,v}(e_1) - c_{f,v}(e_2) \equiv 3 \mod 4, \\ 0, & \text{if } c_{f,v}(e_1) - c_{f,v}(e_2) \equiv 0 \mod 4, \\ -1, & \text{if } c_{f,v}(e_1) - c_{f,v}(e_2) \equiv 1 \mod 4, \\ -2, & \text{if } c_{f,v}(e_1) - c_{f,v}(e_2) \equiv 2 \mod 4. \end{cases}$$

If v is only incident to f once between the edges e_1 and e_2, then

$$\alpha(f, v) := \alpha(f, v, e_1, e_2).$$

Even though e_1 and e_2 are clearly needed for a formally correct definition, we will ignore them from now on, since it is always clear which edges are meant.

Figure 4.11 shows four cases where the angular number is 1. The illustrated graph parts contain only the edges incident to the face f and not their partner edges. However, all cases with corresponding rectangle arrangements are shown in the bottom row of the figure as well. As we can see they all have a stairs change with the color increasing by exactly 1 at the rectangle A_v for the vertex v. Moreover, one can easily see that rotating the colors both of e_1 and e_2 by the same amount does not change the angle number. Thus, an angle number of 1 describes a left turn along the face of the graph and a stairs change with color increasing by 1.

Figure 4.12 shows cases where the angular number is $-1, -2$ and 0. Equivalently to above, we can see that the angle number describes what we aimed for this cases as well. The angle number is 0 if they represent no turn and no stair change, -1 for a right turn, and -2 for reversal of the direction or two stair changes with decreasing color.

4.2.3.2 Angle sum

We can now count the number of left and right turns for a face by simple adding together the angle numbers along its border. An example is shown in Figure 4.13, which also illustrates that the stairs on the outer face only change at non zero angle numbers.

Let $G = (V, E)$ be a graph with satisfying basic and edge properties, f a face of G. We now define the angle sum of f.

Definition 4.8. *The **angle sum** $\alpha(f)$ of a face f is the sum over all angle numbers of every triple $(v, e_{1,2})$ of a vertex v and two edges along the border of f, i.e.*

$$\alpha(f) := \sum_{(v,e_1,e_2) \text{ along the border of } f} \alpha(f, v).$$

We can now tell when a face of the graph induces a correct face in the rectangle arrangement. This is the case when the face description by the edges corresponds to a valid description of a rectangle arrangement face. As we have seen, this applies if the colors

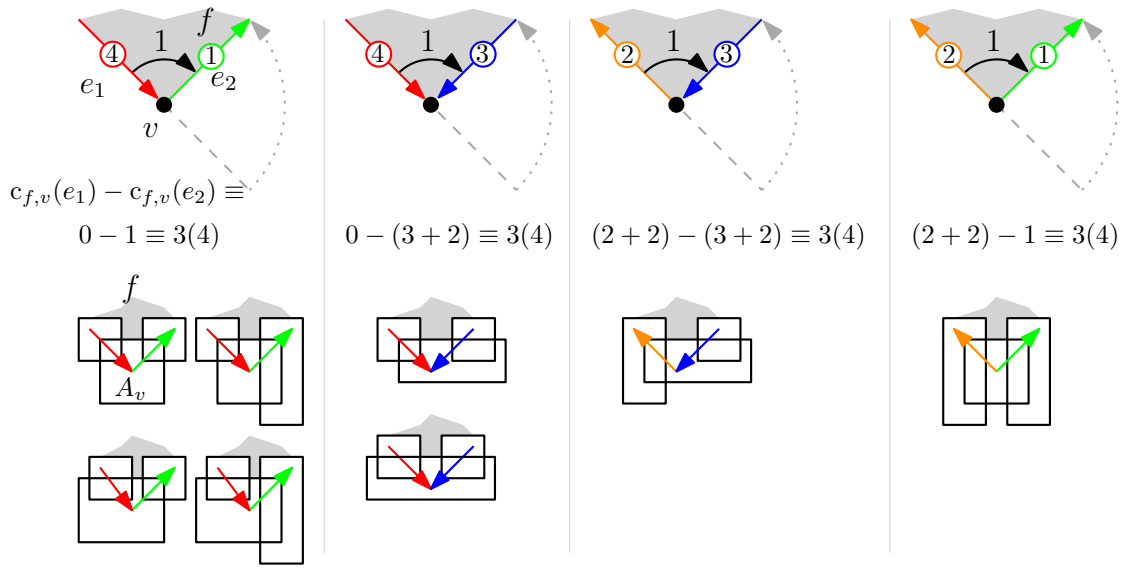

Figure 4.11: The four cases where e_1 and e_2 represent a left turn, i.e. $\alpha(f, v, e_1, e_2) = 1$. The first row shows parts of the graph for each case, the second row how the angle number is computed and the third row where this case appears in rectangle arrangements.

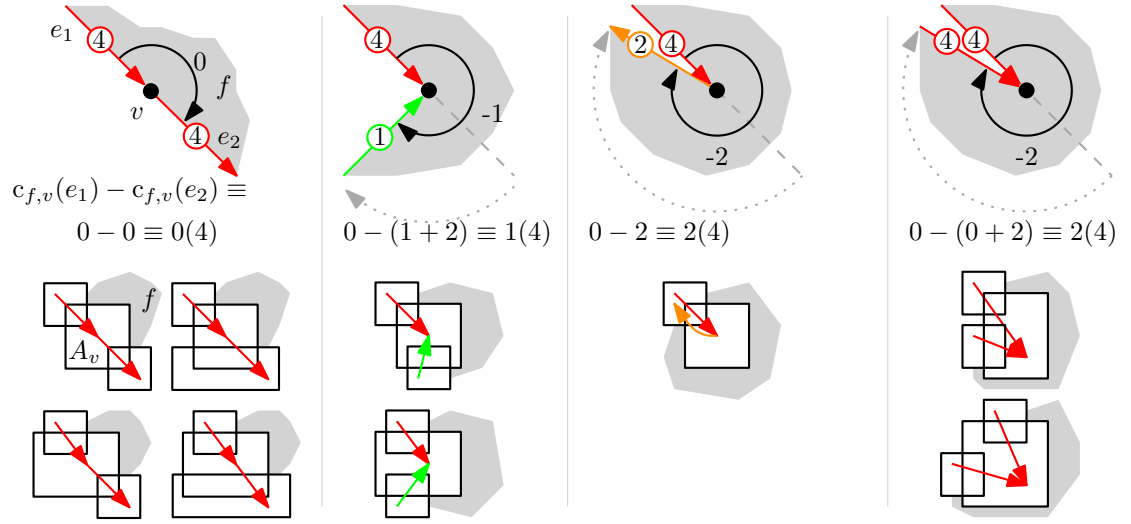

Figure 4.12: Examples for angular numbers of 0, −1 and −2.

of the stairs in the arrangement are traversed exactly once, and each stair change with decreasing color is compensated with a stair change with increasing color.

The angle numbers describe changes with increasing or decreasing stair colors in the rectangle arrangement. This means that a graph face must have an angle sum of exactly 4, since it has to increase the stair color 4 times more then decreasing it. One exception is the outer face. We traverse the outer face clockwise and thus its angel sum has to be −4. The stair colors has to decrease exactly 4 more than increase.

Observation 4.9. *A graph G can be a corner color intersection graph of a triangle-free rectangle arrangement with no cross only if $\alpha(f) = 4$ for every inner face f and $\alpha(f_0) = -4$ for the outer face of G.*

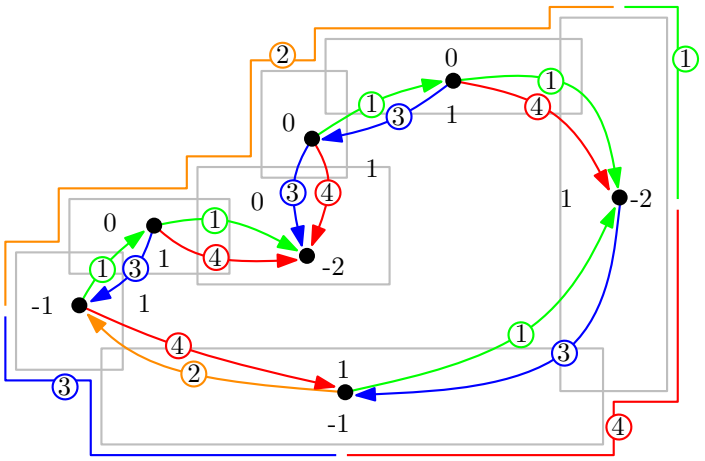

Figure 4.13: Angle sums of a graph, with $1+1-2+1+1+1+1=4$ for the inner face and $0+0+0-1-1-2=-4$ for the outer face.
On the outer face of the rectangle arrangement are the stairs drawn which only change at non zero angle numbers.

In this section we have defined some necessary properties for a graph to be a corner color intersection graph of a triangle-free rectangle arrangement. They are about planarity, edge pairs, edge order and angle sums of faces. We will now make our way to show that they are not only necessary but also sufficient.

4.3 From corner color intersection graph to rectangle arrangement

The last section of this chapter is dedicated to the proof the following theorem.

Theorem 4.10. *A graph $G = (V, E)$ is a corner color intersection graph and has a corresponding triangle-free rectangle arrangement without a cross if and only if it is plane, triangle-free, has legal edge pairs and edge order, angle sum 4 for every inner face and -4 for its outer face.*

This theorem summarizes the Observations 4.4, 4.5,4.6 and 4.9 and states that these described properties are not only necessary but also sufficient. This means that, if a graph is planar, triangle-free, with correct edge pairs and edge order, as well as correct angle sum for every face, then and only then it is a corner color intersection graph and we can find a triangle-free rectangle arrangement for it.

We have seen that every triangle-free rectangle arrangement with no cross can be described uniquely with a corner color intersection graph (Observation 4.3). Thus we already have one direction for the proof of Theorem 4.10. What we have to proof is the direction from graph to rectangle arrangement. We could easily observe that a graph has to be plane, triangle-free and have correct edge pairs and edge order. Figure 4.9 showed us that we need a property for the faces as well. Therefore we introduced the angle sum, which has to be 4 for every inner face and -4 for the outer face. So these properties are obviously all necessary. Thus, for the rest of this chapter we assume that G is a corner color intersection graph, meaning that it has all these requested properties of Theorem 4.10. We can further assume without loss of generality that G is connected.

We give a constructive proof for the direction from graph to arrangement. More precisely, we will use a flow network to construct a triangle-free rectangle arrangement to G. Before this, we do some preparations with G. We show that we can make some more assumptions making it make the proof aph easier to handle.

4.3.1 Preparations

We make some preparations for the constructive proof of Theorem 4.10. Therefore we make the with the corner color intersection graph G more regular. First, we will get rid of negative angle numbers at inner faces and of positive ones along the outer face. Afterwards, we make every face a little smaller and thus easier to handle.

4.3.1.1 Positive and negative regular faces

Given a corner color intersection graph G we define the following properties for its faces.

Definition 4.11. *A face is **positive regular**, if it has no negative angle numbers, and **negative regular**, if it has no positive angle numbers.*

Here, having no negative angle numbers means that for every vertex v adjacent to this face f $\alpha(f, v) \geq 0$. Note also, that angle number of zero is neither positive nor negative. Furthermore note, that for such a graph only the outer face can be negative regular, anyway.

Lemma 4.12. *A corner color intersection graph G can be extended to a corner color intersection graph $H \supseteq G$, such that every inner face of H is positive regular and the outer face of H is negative regular.*

Proof. At first we assume that the outer face f_0 is not negative regular, otherwise we proceed with the inner faces. Since G is a corner color intersection graph, we have that $\alpha(f_0) = -4$. Therefore we have at least two vertices at f_0 with negative angle number. And as f_0 is not negative regular, we can find a vertex v at f_0 with positive angle number, $\alpha(f_0, v) = 1$, such that the next vertex x in counterclockwise order, that has not angle number 0, has a negative angle number. Figure 4.14 shows this in a simplified way.

Starting at the vertex clockwise after x we sum up the angle numbers of the vertices in clockwise order along the face border. Then it is not 0 at v for the first time, but positive. We continue till we find the first vertex z where the sum is 0 or less again. Be e_x the edge adjacent to x and clockwise before x, i.e. e_x is not between x and v. In the same sense be e_z the edge adjacent to and after z. Without loss of generality be $\alpha(f_0, x) = \alpha(f_0, z) = -1$ and not -2 for one of them. Now, since the sum over the angle numbers from x to z is 0, but both have negative angle number, we know that $color(e_x) - 1 \equiv color(e_z) \mod 4$.

Moreover, assuming e_x is directed to x, we know that at x no outgoing edge e'_x with $color(e_x)$ exists. If e_x is directed away from x, then again no outgoing edge with color $color(e_x) - 2 \mod 4$ exists. (This is the same color in both cases.) Otherwise x would not have negative angle number. The same case distinction applies for e_z and z, i.e. we know that the outgoing edge e'_z with corresponding color is not used at z, for the same reason. Figure 4.17 illustrates the case where e_x is directed to x and e_z away from z. Thus, we can add a vertex y and connect it with bidirectional edge pairs to both x and z, with e'_x and e'_z part of the pairs.

As a result, we get that $\alpha(f_0, x) = \alpha(f_0, z) = 0$ and $\alpha(f_0, y) = -1$. We still have $\alpha(f_0) = -4$ by the choice of x and z and since v is no longer at f_0. So we increased $\alpha(f_0)$ for both x and z but decreased it with y and v. The newly create face f, bordered by the vertices from x over v to z and y, has correct angle sum as well. We have by construction $\alpha(f, x) = \alpha(f, y)\alpha(f, z) = 1$ as well as $\alpha(f, v) = 1$. The other angle numbers between x and z cancel each out by the choice of x and z.

Since v is no longer on the outer face, we have at least one vertex with positive angle number less on f_0. Hence, we can repeat this process until f_0 is negative regular.

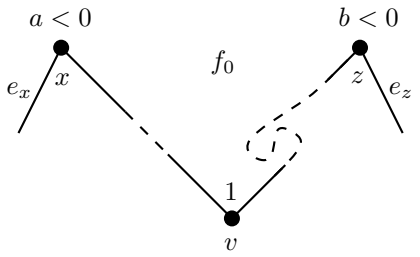

Figure 4.14: Simplified view of a piece of the edge sequence bounding the outer face f_0. Between v and z it may turn several times more to the left and right, till it is completely unfolded at z again. The vertices with angle number 0 for f_0 can be ignored, since they do not affect the logical colour ordering.

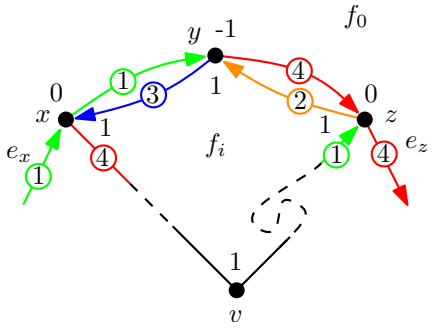

Figure 4.15: Added vertex y to take v with positive angle number out of the border of the outer face f_0. One possible case for the edges e_x and e_z as mentioned in the proof of Lemma 4.12.

Next, we can assume that the graph has an inner face that is not positive regular, otherwise we are done. So, let f be one of the inner faces with at least one negative angle number. In the following we show a way to decrease the absolute sum of all negative angle numbers of f by 1. For that matter we add a vertex y and some edges, such that the face is split into one positive regular face f_r and one face f' with the absolute sum of negative angle numbers one less than f. Again, since G is a corner color intersection graph, we have that $\alpha(f) = 4$. Hence there are more vertices with positive angle number than with negative. This implies that, ignoring vertices z with $\alpha(f, z) = 0$, there has to be a sequence of three vertices x, w, v in counterclockwise order, such that $\alpha(f, x) = a < 0$, $\alpha(f, w) = 1$, and $\alpha(f, v) = 1$. This case is shown in a simplified way in Figure 4.16. Without loss of generality we can assume that there are no vertices z with $\alpha(f, z) = 0$ in our sequence both between x and w and between w and v.

Now, we add a vertex y into f and connected it to v with an unidirectional edge pair directed towards v. We color the edges accordingly to the edge order at v. We also connect it with x with a bidirectional edge pair. Since $\alpha(f, x) < 0$ and by the choice of x, w and v, we know that no outgoing edge with the $color(xw) - 1 \mod 2$ exists, where xy is the edge adjacent to x and bordering f between x and w. Thus we use this edge for the bidirectional edge pair and color the second edge with the opposite color.

We denote the new face enclosed by y, x, w, v with f_r and with f' the one emerged from f, with the sequence x, y, v instead of x, w, v. It is $\alpha(f_r) = 4$, since $\alpha(f_r, x) = \alpha(f_r, w) = \alpha(f_r, v) = \alpha(f_r, y) = 1$ by the choices of x, w, v and the colors of the new edges. Furthermore, f_r is positive regular.

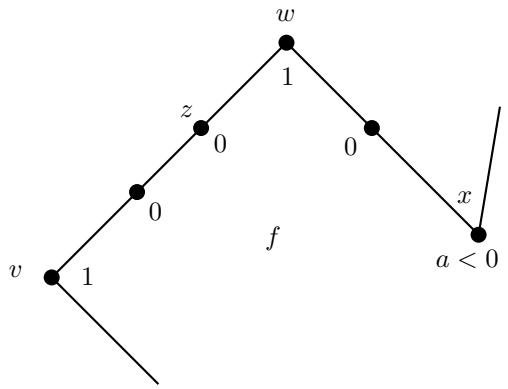

Figure 4.16: Simplified view of a piece of the edge sequence bounding the face f. Along the edges in counterclockwise order is the vertex sequence x, w, v with $\alpha(f, x) = a < 0$, $\alpha(f, w) = 1$, and $\alpha(f, v) = 1$. Between them some vertices z with $\alpha(f, z) = 0$.

For similar reasons we have $\alpha(f') = 4$. However, $\alpha(f', x) = \alpha(f, x) + 1$. Thus, we have decreased the amount of negative angle number by 1. Repeating this procedure we can make f' respectively the new resulting faces all positive regular.

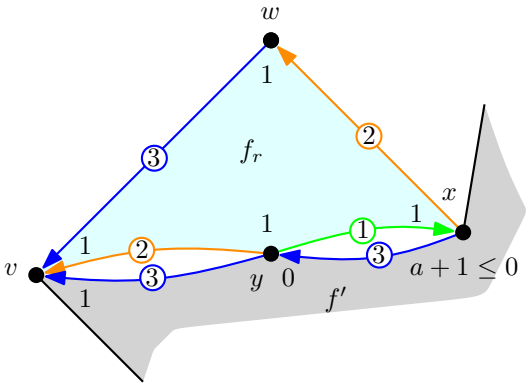

Figure 4.17: One case of the edges between x, w and v. Furthermore the inserted vertex y and edges forming the faces f_r and f' are pictured. For the sake of generality not all edges between the vertices are shown.

We have shown a way to make the outer face negative regular and every inner face positive regular, such that the new graph is a corner color intersection graph as well. Hence we can close this proof. □

4.3.1.2 Dividing the faces in smaller ones

In our next step towards the proof of Theorem 4.10 we divide every inner face, such that every newly emerged face is either bounded by a four or a five cycle. We also add some new faces and a bounding four cycle around the graph to get this property also for the outer face. The next lemma concretises this. But before that, note that an inner face of a four or five cycle in a corner color intersection graph is always positive regular.

Lemma 4.13. *A corner color intersection graph G can be extended to a corner color intersection graph $H \supseteq G$, such that every inner face of H is bound by either a four or five cycle.*

Proof. In the first step of this proof we apply Lemma 4.12 to our graph. So our new graph has only positive regular inner faces and a negative regular outer face f_0. Remember also that our graph is connected by assumption.

Next, we handle the outer face, before later we will proceed with the inner faces. If the outer face f_0 of our graph G is already enclosing a four cycle, we are done. So we consider the other case.

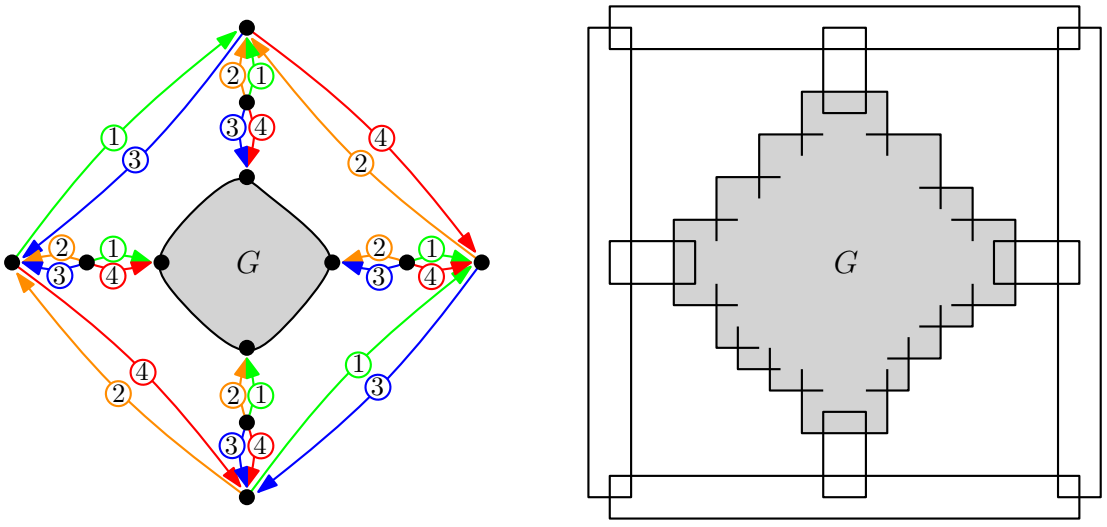

Figure 4.18: Additional surrounding four cycle plus four vertices connecting the graph to this cycle. On the right the correlating rectangles to this construction.

The construction described in the following is illustrated in Figure 4.18. To have a four cycle on the other face we just introduce a new 4-cycle and place the whole graph inside of it. We connect the four vertices with bidirectional edge pairs in the colours forming a legal cycle, with respect to the properties of a corner color intersection graph. To connect the graph with this cycle, we add four more vertices inside the cycle but outside of the old graph. Then we connect each with one vertex of the new cycle, both edges directed towards the vertex on the cycle in the colours of the two order edges incoming at this vertex. To connect the new vertices, and thereby the cycle, to the graph we connect each of the four vertices to one of the at most four and at least two vertices on the boundary of the graph with negative angle number. Remember that the graph was negative regular on the outer face. Again we direct both edges away from the single vertex. This implies that each of the vertices uses its two left outgoing colours for this edges. The choice of the partnering vertex on the old graph is determined by the colours used at the right turn in clockwise order at the vertices with negative angle number, and in respect to the properties of a corner color intersection graph. Note that since G is a corner color intersection graph, this choice is always unambiguous. Note also, that if a vertex had angle number -2, two of the vertices will be connected to it, ordered in respect to planarity and colours. It is easy to see that the four newly introduced inner faces are again all positive regular, and that the outer face is now enclosing a four cycle.

A geometrical explanation of this construction is that we add a cycle of four rectangles intersecting with one corner each to both its partners around our rectangle arrangement. Then we add four rectangles with side-piercing intersection to the four outer rectangles and to one of the rectangles of the old arrangement with a free corner on the outer face, i.e. where the corners on the outside change direction where they point to. This is illustrated in Figure 4.18 as well.

In order to have all inner faces f_i to be four or five cycles, we will again add some new vertices. Since our inner faces are all positive regular they have precisely four vertices with angle number one, say in counterclockwise order u, w, y, z, and the remaining with angle number 0. Then starting from w we pick the next vertex x in direction to y, maybe y itself. If there is no vertex between u and w, we do nothing. In the other case, if it exists let v be the second vertex on the path from w to u along f_i. Otherwise we set $v = u$. Now, for every vertex v' between v and u, including both v and u, we add one new vertex $m_{v'}$ and two bidirectional edge pairs, one from $m_{v'}$ to x and one from $m_{v'}$ to v'.

For the colours of the edges towards x we choose the two which are allowed by the legal edge orders to be incoming from f_i at y, the same for the edges from m_v to v with the colours allowed at u. It is easy to see that every new face, except maybe not f_i', the one containing z, is now bounded by a four or five cycle. This construction is shown in Figure 4.19.

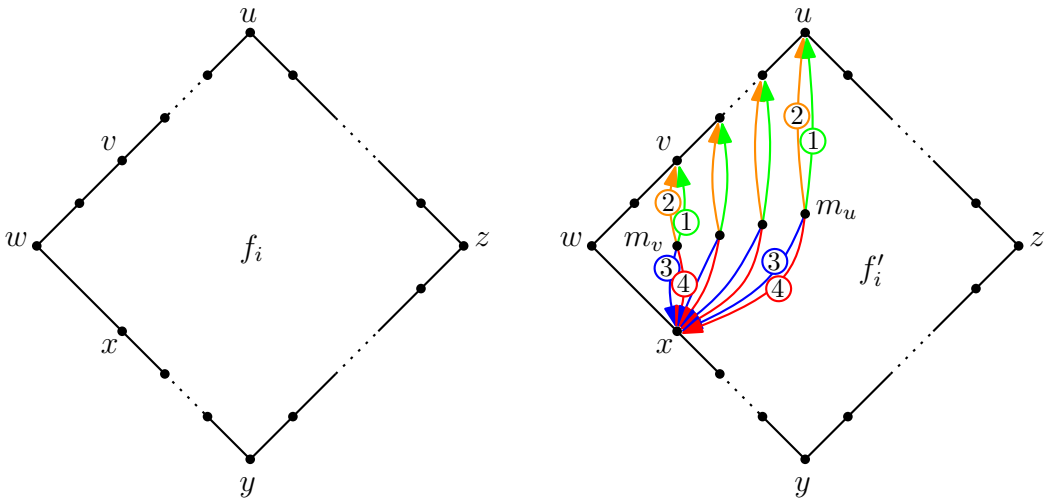

Figure 4.19: Starting situation for the construction of inner four and five cycles on the left. After the first run we get the situation on the right, where the number of vertices at f_i has been decreases by the number of vertices between w and u. Thereby some five cycles where introduced. The next runs repeat this with vertices between x and y, y and z, and z and u. Since the actual direction and colour of the edges along the face do not matter, they are only represented by a black edge.

We repeat this process for f_i' beginning at x and the next new partner vertex for all new vertices now u. Furthermore we do this whole construction also starting at z. So at the end we have systematically reduced the number of vertices in the inner face, till only u, y and two newly added vertices are left. Hence every inner face is now either bound by a five or four cycle. By the choices of the colours for the new edges we still have a corner color intersection graph. □

4.3.2 Flow network

We will now use a flow network to get from a graph to a rectangle arrangement. The basic idea is to spread the space of the plane as width through the graph, once for horizontal width and once for vertical height. For the horizontal flow this means, that we give width from the plane in which our triangle-free rectangle arrangement will lie towards the rectangles which lie most left. They themselves give this width to every rectangle they intersect with on their right side and also to every face to their right. This is continued till all width returns again to the outer face on the right. So, if we find two flows, a horizontal

and a vertical, which are positive in every vertex of the graph, thus in the corresponding rectangle, and in every intersection and every face, we can use the found widths to draw the rectangle arrangement.

First, we define the flow network formally. We do this for a corner color intersection graph with faces bound by four or five cycles and positive regular respectively negative regular outer face f_0. Afterwards, we show, that we can always find flows to spread width through them horizontally and vertically and how this results in a drawing of a corresponding rectangle arrangement. This will prove Theorem 4.10.

Given a corner color intersection graph G with every face bound by a four or five cycle and positive respectively negative regular, we define two flow networks, the horizontal $N_h = (V_N, E_h, s, t)$ and the vertical $N_v = (V_N, E_v, s, t)$. Here V_N is the vertex set, E_h respectively E_v the set of directed edges, s is the only source and t the only sink, both vertices $s, t \in V_N$.

4.3.2.1 Vertices

The width we want to spread through our flow network should go through every rectangle, every inner face and every intersection of two rectangles. So the vertices in our flow network have four different origins in our graph G. We denote the set of faces of the graph by $F(G)$.

First, we have two vertices (□ in the figures) for the outer face of G, the source s, which gives the width of the plane to the network, and the sink t receiving it all. Second, every inner face of G gets a vertex (□). Third, we have one vertex per vertex $v \in V(G)$ (●). And fourth, each edge of G, thus every edge pair, gets a vertex (■), since they represent the intersection of the rectangles and the corresponding intersection area.

$$V_N := V(G) \cup F(G) \backslash \{f_0\} \cup E(G) \cup \{s, t\}.$$

4.3.2.2 Edges

Whereas the vertices are the same for both the horizontal and the vertical width flow, of course, we have different edges between them for each network. So in the following we will only describe the edges E_h for the horizontal flow, for the vertical ones E_v it works analogously.

Before describing when to add edges precisely, we may look at this in a more geometrical way. We want to distribute width through our network. For the horizontal flow this means, that from every face left of a vertex, we will have flow towards it and outwards to every face at its right. Furthermore we have width coming from every edge pair representing an intersection on the left side of the rectangle towards a vertex. The same for outgoing edge pairs. To ensure that the intersections do not touch inside a rectangle, we divide the flow through the rectangle into three parts. One part for the free middle area of the rectangle and one each for the intersections with the upper side and the lower side of the rectangle. It is easy to see, that having the same width for all upper intersections and all lower intersections, does not reduce our possibilities to find a rectangle arrangement for the given combinatorial properties described by our graph. If we do this separation also for the vertical flow, we can guarantee that no two intersections inside a rectangle touch or intersect each other.

Figure 4.20 shows how the edges at one vertex correlate with the flow and the searched rectangle arrangement. In the following paragraphs we will describe how this corresponds to the graph. The flow for the outer face and its two vertices is illustrated in 4.21.

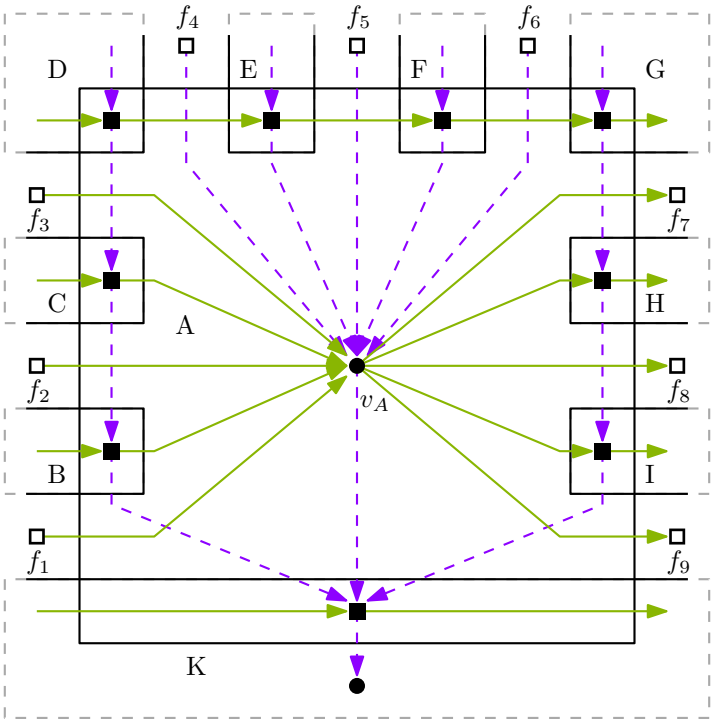

Figure 4.20: The horizontal flow (lime) and in the (dashed, violet) vertical flow around a vertex v_A (●) for the rectangle A and its neighbouring faces (□) and edge pairs (■).

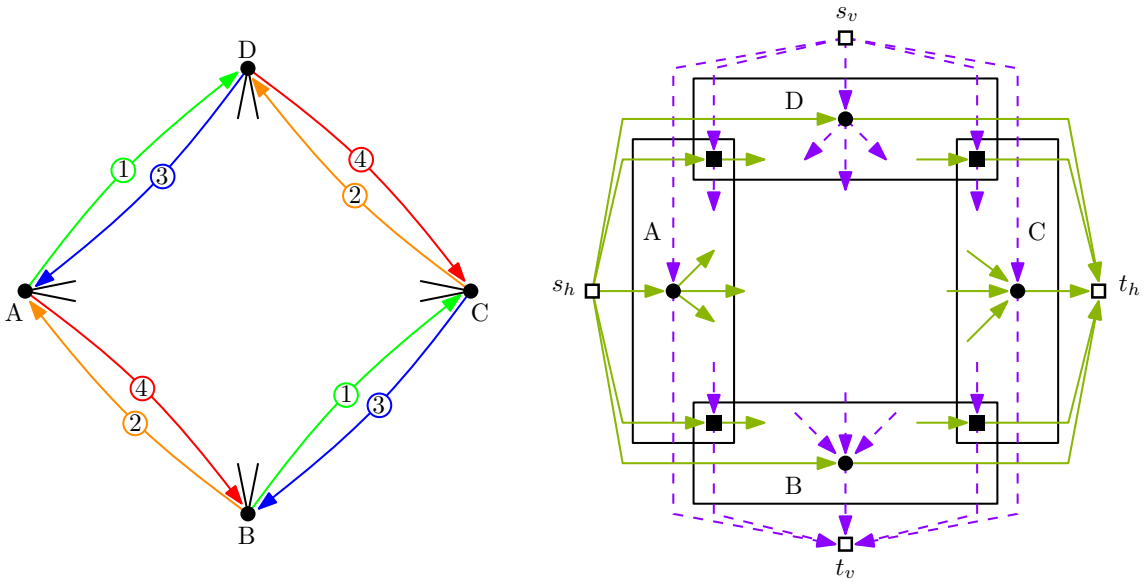

Figure 4.21: Flow for the outer four cycle of the graph. Both sink and source (□) are split for the horizontal flow (lime) and the vertical flow (dashed, violet) for illustrative reasons.

Edges between vertices and faces

Let's first consider the flow from faces to vertices. On the horizontal flow every rectangle should get the width of every face left of it and give it to every face right of it. However, in a graph there is no real left or right between faces and vertices. Therefore we define it for a corner color intersection graph , when a face f is *directly left* of a vertex.

Definition 4.14. *Let H be a corner color intersection graph and v a vertex of H. A face f of H is **directly left** of v, if f is incident to v in counterclockwise order between the outgoing edges in color orange and blue. If no edge pair with outgoing orange edge exists, the previous pair in the circular order is considered, and if no edge pair with outgoing blue exists, the next pair is considered.*

That a face is *directly right, above* or *below* a vertex is defined analogously.

Since in G every inner face is positive regular, we can make a nice observation.

Observation 4.15. *Every inner face f of G is in at most two of the four relations directly left, below, right or above to an incident vertex v. If f is in 2 of these relations with v, then these are not the opposite ones.*

A vertex with angle number 1 is in only one of these relation with a considered face and a vertex with angle number 0 in two of them, which are those of the previous and the next vertex with angle number 1 along the border of the face. Since there are exactly four different vertices with angle number one and we have four different relations, we directly get this observation. Furthermore this yields, that a face is only incident once to a vertex.

Observation 4.16. *Every face of G is incident to a vertex at most once.*

Let's get back to the flow. Now, the easiest case is, that a vertex has a orange-blue edge pair of outgoing edges. Then we do not have a face directly left of it. The second case is that we have one or more faces left of v. Then v should get flow from each of them. This is shown simplified in Figure 4.22 by the edges between $f_1 f, f_2, f_3$ (□) and v_a (●).

We collect all these edges in $E_{h,f\to v}$:

$$E_{h,F\to V} := \{(f,v) \mid f \in F(G), v \in V(G), f \text{ lies directly left of } v, f = s \text{ if } f = f_0\} \quad (4.17)$$

Next, we consider the edges from vertices to faces. Here we have to consider the equivalent cases with faces directly right of a vertex.

$$E_{h,F\leftarrow V} := \{(v,f) \mid v \in V(G), f \in F(G), f \text{ lies directly right of } v, f = t \text{ if } f = f_0\} \quad (4.18)$$

To sum up, we have an edge from the source s to all vertices of rectangles with the outer face directly left of them. Every vertex has edges from faces directly left of it and to faces directly right of it.4.22 these are all edges between boxes (□) for faces and circles (●) for the vertices.

$$E_{h,F\leftrightarrow V} = E_{h,F\to V} \cup E_{h,F\leftarrow V} \quad (4.19)$$

Edges between vertices and edge pairs

Similar to the faces a rectangle may get width from rectangles intersected by it or intersecting it at the left side. As mentioned before we split the flow inside a rectangle into three parts, through the centre and the lower and upper intersections. This means for the flow through towards the vertex we only add edges for either incoming red-green edge pairs or the outgoing orange-blue pair. Only one case is possible, but it may also be none. Then the only flow to the vertex comes from one left face.

$$\begin{aligned} E_{h,E\to V} := \{(e,v) \mid e \in E(G), v \in V(G), &(e \text{ red-green edge pair directed to } v) \\ &\vee (e \text{ orange-blue edge pair directed away from } v)\} \end{aligned} \quad (4.20)$$

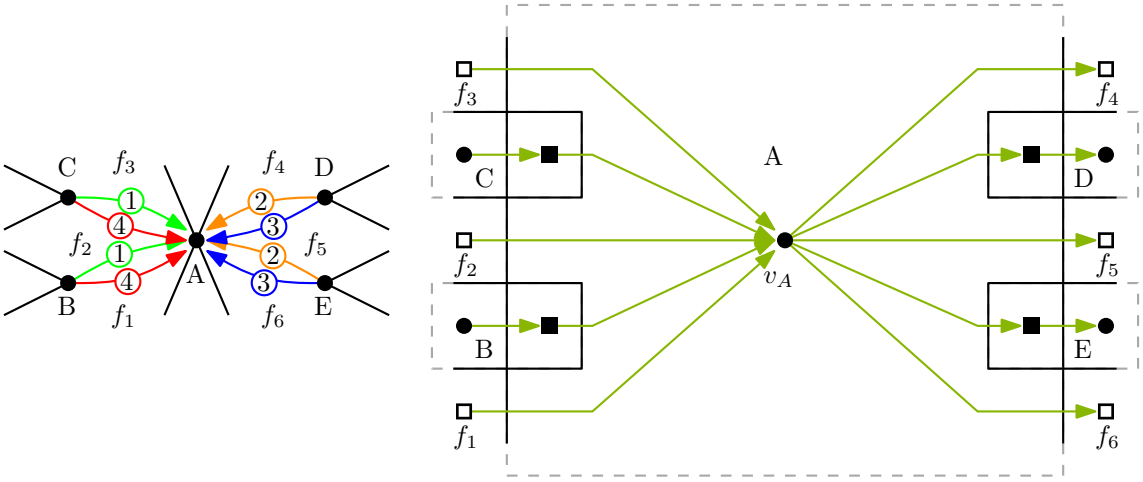

Figure 4.22: Part of a graph with vertex A in the center. On the right the corresponding flow from left edge pairs (■) and faces (□) through v_A (●) to the right edge pairs and faces.

Again we have the same situation on the right. Here we may have one edge to a outgoing red-green edge pair or one edge per incoming orange-blue edge pair.

$$E_{h,E\leftarrow V} := \{(v,e) \mid v \in V(G), e \in E(G), (e \text{ orange-blue edge pair directed to } v) \\ \vee (e \text{ red-green edge pair directed away from } v)\} \tag{4.21}$$

This is also illustrated with the left and right faces in Figure 4.22 by the edges from squares (■) for edge pairs and circles (●) for vertices and the other way round. Together they form the edge set $E_{h,E\leftrightarrow V}$.

$$E_{h,E\leftrightarrow V} = E_{h,E\rightarrow V} \cup E_{h,E\leftarrow V} \tag{4.22}$$

Edges between edge pairs and faces

Edges between edge pairs are along the upper and lower intersections of a rectangle. This means for the upper we have edges starting from the pair with outgoing orange edge along every pair with an incoming red edge, till the pair with outgoing green in clockwise order. Figure 4.23 illustrates this.

So first, with Definition 4.23 we define the edges from the edge pair with outgoing orange to the next blue-red ones at every vertex. With Definition 4.24 we define those between blue-red edge pairs and with Definition 4.25 we define the last possible edges in the upper row, from blue-red incoming edge pairs to bidirectional blue-green ones.

$$E_{h,E\rightarrow_{u1}E} := \{(e_1,e_2) \mid e_1,e_2 \in E(G), v \in V(G), \\ (e_1 \text{ edge pair with orange outgoing at } v), \\ (e_2 \text{ blue-red edge pair directed to } v), \\ (e_2 \text{ clockwise directly after } e_1 \text{ at } v)\} \tag{4.23}$$

$$E_{h,E\rightarrow_{u2}E} := \{(e_1,e_2) \mid e_1,e_2 \in E(G), v \in V(G), \\ (e_1 \text{ and } e_2 \text{ blue-red edge pairs directed to } v), \\ (e_2 \text{ clockwise directly after } e_1 \text{ at } v)\} \tag{4.24}$$

$$\begin{aligned} E_{h,E\to u3E} := \{(e_1,e_2) \mid\ & e_1,e_2 \in E(G), v \in V(G), \\ & (e_1 \text{ blue-red edge pair directed to } v), \\ & (e_2 \text{ edge pair with green outgoing at } v), \\ & (e_2 \text{ clockwise directly after } e_1 \text{ at } v)\} \end{aligned} \tag{4.25}$$

There is one more case we have to address, where the two edge pairs with outgoing orange and outgoing green exist, but no blue-red edge pair between them. Then of course the width has to be spread only between them in the upper row.

$$\begin{aligned} E_{h,E\to u4E} := \{(e_1,e_2) \mid\ & e_1,e_2 \in E(G), v \in V(G), e_1 \neq e_2 \\ & (e_1 \text{ edge pair with orange outgoing at } v), \\ & (e_2 \text{ edge pair with green outgoing at } v), \\ & (e_2 \text{ clockwise directly after } e_1 \text{ at } v)\} \end{aligned} \tag{4.26}$$

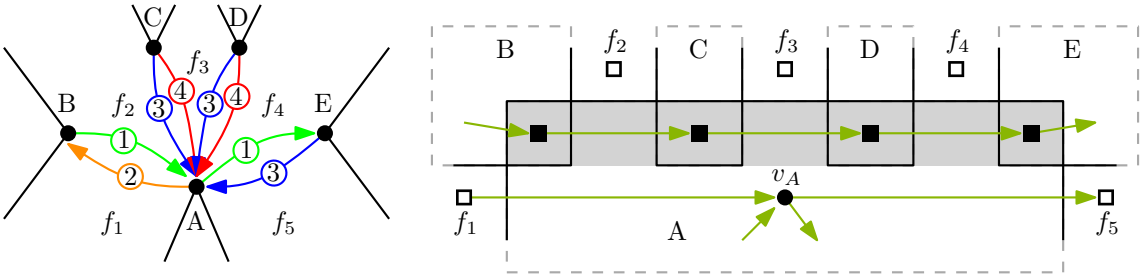

Figure 4.23: Part of a graph with edge pairs at vertex A which represent intersections in the upper row. In the flow network we get the upper row (gray area) for the flow through the upper edge pairs (■).

We have the same in counterclockwise order for lower intersections starting with outgoing blue, along incoming with orange-green, till outgoing with red. All together we get the set $E_{h,E\to E}$

$$\begin{aligned} E_{h,E\to E} := {} & E_{h,E\to u1E} \cup E_{h,E\to u2E} \cup E_{h,E\to u3E} \cup E_{h,E\to u4E} \\ & \cup E_{h,E\to l1E} \cup E_{h,E\to l2E} \cup E_{h,E\to l3E} \cup E_{h,E\to l4E} \end{aligned} \tag{4.27}$$

The special cases, where there is the edge pair with either both orange and green or both blue and red from w to v, is not handled for v. This intersection is managed in the lower respectively upper row of w.

If one of the edge pairs with outgoing colour does not exist, then we also have an edge from the upper or lower row to the adjacent face. For example if there is no pair with the outgoing green edge, but the upper row is not empty, we have an edge from the face, in which this not existing pair would be, to the next pair in described order. This could be an edge pair with incoming red edge or with outgoing orange edge. This case is shown in Figure 4.24.

Note, that the pair with outgoing blue and the one with outgoing red could be the same. Then we have no further edges in the network for this row. Furthermore, if there is no pair at all with the described pairs, we have no flow for the row, since it is not needed. By the fact that all our inner faces are positive regular, but this case would imply a negative

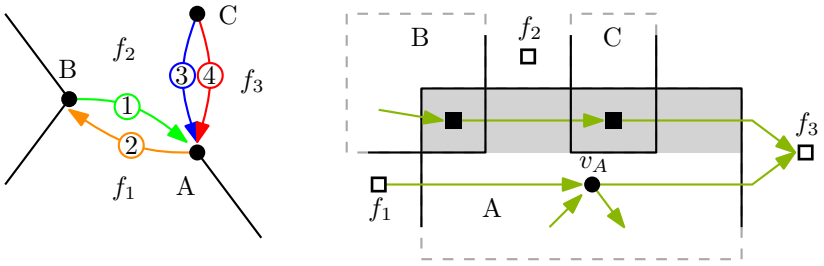

Figure 4.24: Flow through the upper row, where in the graph vertex A has no outgoing green edge. Thus we get an edge from the edge pair (■) between A and C to the face f_3 (□) in the horizontal flow network.

angle number, we see that this can only occur for vertices on the outer face. However, we define the possible edges for the upper row with Definition 4.28 and Definition 4.29.

$$\begin{aligned} E_{h,E\leftarrow_u F} := \{&(e,f) \mid e \in E(G), f \in F(G), v \in V(G) \\ &(v \text{ has no edge pair } e' \text{ with outgoing orange}), \\ &(e \text{ edge at } v \text{ which would be clockwise directly before } e'), \\ &(f \text{ directly right of } v \text{ and partly bound by } e)\} \end{aligned} \tag{4.28}$$

$$\begin{aligned} E_{h,E\rightarrow_u F} := \{&(f,e) \mid f \in F(G), e \in E(G), v \in V(G) \\ &(v \text{ has no edge pair } e' \text{ with outgoing green}), \\ &(e \text{ edge at } v \text{ which would be clockwise directly after } e'), \\ &(f \text{ directly left of } v \text{ and partly bound by } e)\} \end{aligned} \tag{4.29}$$

So together with the lower row we get all the edges in the flow between faces and edges of G with $E_{E\leftrightarrow F}$.

$$E_{h,E\leftrightarrow F} := E_{h,\leftarrow_u F} \cup E_{h,E\rightarrow_u f} \cup E_{h,\leftarrow_l F} \cup E_{h,E\rightarrow_l f} \tag{4.30}$$

Summary and vertical flow

Lets summarize all the edges we defined for the horizontal flow network. We have those between faces and an vertices, if they are adjacent in G and left or right of each other. Then there are those between edge pairs coming from left or right to a vertex. They form the middle row for a rectangle. The upper and lower row is given by the edges between upper and lower neighbouring edge pairs and maybe faces at the corner. So now we can define E_h of N_h.

$$E_h := E_{h,F\leftrightarrow V} \cup E_{h,E\leftrightarrow V} \cup E_{h,E\rightarrow E} \cup E_{h,E\leftrightarrow F} \tag{4.31}$$

The edges for the vertical flow network N_v are constructed the same way, only changing left with upper, right with lower, and the colours corresponding to that.

$$E_v := E_{v,F\leftrightarrow V} \cup E_{v,E\leftrightarrow V} \cup E_{v,E\rightarrow E} \cup E_{v,E\leftrightarrow F} \tag{4.32}$$

4.3.2.3 Finding a flow

So far we have defined the flow networks N_h and N_v for our prepared graph G with described properties. By their construction it is clear, that if we can find a flow, which is positive on all edges, we get the actual sizes to draw a corresponding triangle-free rectangle

arrangement. The horizontal flow through a vertex and its upper and lower row determines the rectangle's total height, the vertical its total width. The size of every border segment is given by the flow on the edge to the intersection which bounds this segment or the neighbouring face. Using three rows for each rectangles both in the horizontal and vertical flow ensures that every pair of intersections is disjoint.

The following lemma summarizes some important observations. Remember that we assume that G is connected.

Lemma 4.33. *The flow networks N_h and N_v for a corner color intersection graph G are connected, have the only source s and the only sink t and are acyclic.*

Proof. Obviously the network is connected if G is connected. Second, we observe that we really only have one source s and one sink t. Every vertex has at least either one face directly left of it or a corresponding unidirectional edge pair. The same applies for faces respectively edge pairs right, above and below a vertex. Every face except f_0 is directly left, right, above and below a vertex. Every edge pair has either the vertices which it is incident to as neighbours in the flow network, if it lies in direction of the flow, or otherwise a neighbouring edge pairs or faces. Therefore every vertex $v \in V_N \backslash \{s, t\}$ has at least one incoming and one outgoing edge. By definition s has only outgoing edges and t only incoming edges. Hence, s, t are the only source respectively source.

By construction are the edges through intersections and vertices only in one direction, from left to right respectively from top to bottom. The edges from edge pairs to faces or vice versa do only appear if a face is directly left, right, above or below the two vertices which are connected by this edge pair. By Observation 4.16 we know that a vertex is incident to a face of G at most once. Together we get, that the edges of the network are all directed from vertices representing things left of those represented by the vertices the edges end. This may either be given by left and right relations between faces and vertices or edge pairs in G, or by the circular orders of vertices in G for edges between edge pairs and vertices. Since s is leftmost and t rightmost in the network, respectively topmost and bottommost, we get that the flow network is acyclic. □

We can now can give the two flows easily. We do this for the horizontal flow f_h. The idea is to simply give all flow incoming at a vertex in equal parts to outgoing edges. The previous Lemma 4.33 directly shows that the following is well defined.

Lemma 4.34. *The flow networks N_h and N_v have nowhere zero flow.*

Proof. By Lemma 4.33 are the flow networks connected, acyclic and have only one source and one sink. Hence, is the flow given by the following equations nowhere zero.

$$\forall (s, v) \in E_h : f_h((s, v)) := \frac{1}{\deg(s)}$$

$$\forall (v, w) \in E_h, v \neq s : f_h((v, w)) := \frac{1}{\deg_{out}(v)} \cdot \Big(\sum_{(u,v) \in E_h} f_h(u, v) \Big)$$

□

This closes the proof of Theorem 4.10. Using the flow network we obtain from a corner color intersection graph a size insensitive triangle-free rectangle arrangement. This shows that the graph actually is a corner color intersection graph. Hence, we now have a full characterisation of triangle-free rectangle arrangements given exactly by corner color intersection graphs .

Sadly, the flow network does not work to solve the squarability question for a triangle-free rectangle arrangement. Here we would have to tangle the horizontal and vertical flow networks. The three rows for every rectangle used in both directions would have to have the same flow amount. At the same time the horizontal height and vertical width can not mix. As far as we know, there is no construction which would allow to synchronise two flows for some vertices or edges. It also seems impossible to compute the first flow and then set up the second flow network with suitable capacities. There are triangle-free rectangle arrangement in which the decisions we may make for the sizes in the first flow would not allow for a second flow.

5. Algebraic solution model

Now that we have seen how rectangle arrangements can be described combinatorially, we want to come back to the question of whether a rectangle arrangement $\mathcal{R}$ lies in Squares or not.

As a possible approach to solve the problem we discuss an algebraic model. We use the coordinates of the rectangle corners as variables and add some constraints describing the intersections and the fact, that we want the rectangles to be squares. The program may then give us concrete values for the coordinates and, thus, show that the arrangement is squarable, or prove the opposite.

The program we present to solve the Squarability problem is a *mixed integer linear program (MIP)*. Such programs consist of *variables*, *linear constraints* and *linear objective function*. Mixed integer means that some variables are constrained to be integers whereas others can be real numbers. We are only interested in whether or not a rectangle arrangement is squarable and do not care about the actual size. Therefore we do not give an explicit objective function. Any feasible solution will be sufficient. The constraints are linear inequalities on constant factors and the variables. With variable vector x, constant vector b and matrix A the constraints of a MIP constraints may be of the form $Ax \leq b$. However, we forgo bringing the presented constraints into such a standard form and use a more descriptive version.

We start with the description of the rectangles. Then, we show the required constraints both for intersecting and disjoint rectangle pairs. Aiming for a square arrangement we also have constraints for the square property.

5.1 Variables and constraints

Let $\mathcal{R}$ be a rectangle arrangement and $\epsilon > 0$. We design the MIP such that any feasible solution describes a square arrangement, combinatorially equivalent to $\mathcal{R}$. We use ϵ to describe the minimal size of a square and the margin between two squares. The solution will be within a drawing area $[0, N] \times [0, N] \subset \mathbb{R}^2$, with $N > 0$ sufficiently large. Of course, N depends on the one hand on ϵ and on the other hand on $\mathcal{R}$. Furthermore, we can observe that, if we chose N large enough, restricting the drawing area does not influence the solvability of the Squarability problem and the MIP.

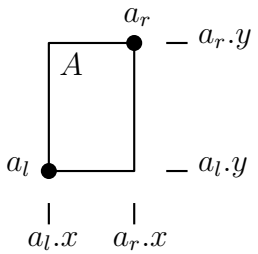

Figure 5.1: Identification of rectangle A via lower left and upper right corner.

5.1.1 Rectangle description

Each axis-aligned rectangle A can be described by its lower left corner $a_l := a_l(x, y)$ and upper right corner $a_r := a_r(x, y)$, i.e. $A = [a_l.x, a_r.x] \times [a_l.y, a_r.y]$. This is illustrated in Figure 5.1. So, for each rectangle $A \in \mathcal{R}$ we get four variables.

$$\mathit{rectangleVariables}(A) := \{a_l.x, a_l.y, a_r.x, a_r.y \in [0, N]\} \tag{5.1}$$

Their left-right and above-below relations yield for every rectangle $A \in \mathcal{R}$ the set of two constraints $\mathit{rectangle}(A)$.

$$\mathit{rectangle}(A) := \{a_l.x \leq a_r.x + \epsilon,\ a_l.y \leq a_r.y + \epsilon\} \tag{5.2}$$

5.1.2 Intersection constraints

For each intersection between two rectangle we need to describe the vertical and horizontal order of their two defining corners. If two rectangles intersect, then, for each dimension, only four orderings of their corners exist. They may either be alternating or enclosing.

5.1.2.1 Horizontal or vertical overlap

An alternating order may be the result of an opposite-corner intersection. In order to describe this in the program we define two sets of conditions, $\mathit{horizontalOverlap}(A, B)$ for the x-dimension and $\mathit{verticalOverlap}(A, B)$ for the y-dimension. We also determine the first named rectangle being more left, respectively below the second one.

$$\mathit{horizontalOverlap}(A, B) := \{a_l.x \leq b_l.x + \epsilon,\ b_l.x \leq a_r.x + \epsilon,\ a_r.x \leq b_r.x + \epsilon\} \tag{5.3}$$

$$\mathit{verticalOverlap}(A, B) := \{a_l.y \leq b_l.y + \epsilon,\ b_l.y \leq a_r.y + \epsilon,\ a_r.y \leq b_r.y + \epsilon\} \tag{5.4}$$

So for example the opposite-corner intersection represented in Figure 5.2 (a) would yield the constraints $\mathit{horizontalOverlap}(A, B) \cup \mathit{verticalOverlap}(B, A)$.

5.1.2.2 Horizontal or vertical enclosure

The second possible order could be induced by a side-piercing intersection, say on the right side. Then the two corresponding corner y-coordinates of the piercing rectangle would lie between those of the pierced. As this can occur both vertically and horizontally, we get again two sets of constraints. We define those sets, $\mathit{horizontalInside}(A, B)$ and $\mathit{verticalInside}(A, B)$, with the first named rectangle being the one enclosed the second.

$$\mathit{horizontalInside}(A, B) := \{b_l.x \leq a_l.x + \epsilon,\ a_r.x \leq b_r.x + \epsilon\} \tag{5.5}$$

$$\mathit{verticalInside}(A, B) := \{b_l.y \leq a_l.y + \epsilon,\ a_r.y \leq b_r.y + \epsilon\} \tag{5.6}$$

A side-piercing intersection on the right side, as the one shown in Figure 5.2 (b), can now be described with the constraints of a horizontal overlap and a vertical enclosing order. So here we get the constraints $horizontalOverlap(A, B) \cup verticalInside(A, B)$.

Those sets of constraints can also be used to describe a containing and unsquarable intersection. As Figure 5.2 (c) illustrates, this would yield the constraints $horizontalInside(A, B) \cup verticalInside(A, B)$ for the containing intersection.

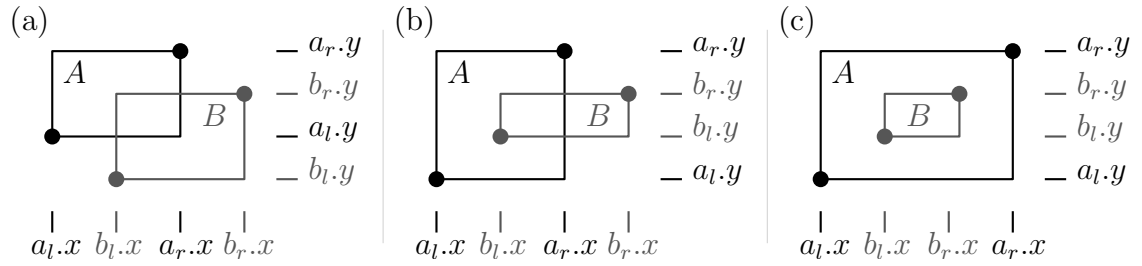

Figure 5.2: Resulting coordinate order for (a) opposite-corner intersection, (b) side-piercing intersection, (c) containing intersection. They corners can either alternate or enclose each other.

5.1.2.3 Intersection constraints set

For every intersection of two rectangles $A, B \in \mathcal{R}$ we can define the set of constraints as a union of overlaps or enclosing relations.

$$\begin{aligned} intersect(A, B) := \{ & X \cup Y \mid X \in \{horizontalOverlap, horizontalInside\}, \\ & X \text{ describing the intersection between } A \text{ and } B \text{ horizontally}, \\ & Y \in \{verticalOverlap, verticalInside\}, \\ & Y \text{ describing the intersection between } A \text{ and } B \text{ vertically} \end{aligned} \tag{5.7}$$

5.1.3 Independence constraints

It is not sufficient to describe only the intersections between the rectangles. The program has also to guarantee that two squares in the solution are disjoint if and only if their rectangles are disjoint in $\mathcal{R}$. Moreover, we have to retain the intersection order for each rectangle.

For the cases shown in Figure 5.3 we know that the squares A and B have to be independent and the intersection order of C tells us in which way. Here we have that A has to be left of B. However, in the general case, depicted by Figure 5.4, we do not know if one square will lie completely left, right, above or below another. The only thing we know is, that they have to be disjoint and one of these cases has to apply. So in order to claim their independence we have to ensure that we can separate them with either a horizontal or a vertical line.

5.1.3.1 Unrestricted independence

At first we consider the general case, where it is not fixed in which way two squares are separated, either by a horizontal or a vertical line. Later in the second case, where we know in which relation two squares are, we add some constraints to those of the general case.

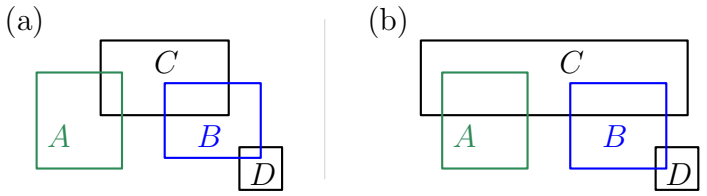

Figure 5.3: Non-intersecting rectangles have to be disjoint in the solution as well.
(a): A must and can only be completely left of B.
(b): A must be completely left of B. A could also be completely right of B respecting the intersection types, but not the intersection order.

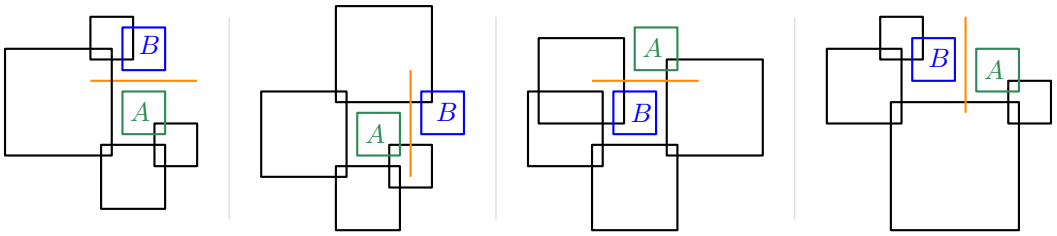

Figure 5.4: The squares A and B have to be disjoint. This can be achieved in four different ways.

For each pair of disjoint rectangles we add four binary variables. More precisely, for A and B independent we add Boolean variables $X_{lr}(A,B), X_{rl}(A,B)$ describing a left-right or right-left relation and $X_{ba}(A,B), X_{ab}(A,B)$ describing a below-above or above-below relation. By left-right relation between A and B we mean that they can be separated by a vertical line l such that A is left of l and B right of l. The three other relations are analogously defined.

$$\begin{aligned} \mathit{indepBoolean} :=& \{X_{lr}(A,B), X_{rl}(A,B), X_{ba}(A,B), X_{ab}(A,B) \in \{0,1\} \\ & \mid A,B \in \mathcal{R}, A \cap B = \emptyset\} \end{aligned} \tag{5.8}$$

We use these Boolean variables to describe the four possible separations. For example, we describe a left-right relation $\mathit{leftRight}(A,B)$ between a disjoint pair A and B with $X_{lr}(A,B)$ as follows. If $X_{lr}(A,B) = 1$ then A has to be left of B. If $X_{lr}(A,B) = 0$ then A does not have to be left of B. So we can define $\mathit{leftRight}(A,B)$ with the inequality below. Thereby let $M \geq N + \epsilon$, where N denotes the size of the drawing area as defined above.

$$\mathit{leftRight}(A,B) := \{(1 - X_{lr}(A,B)) \cdot M + (b_l.x - a_r.x) \geq \epsilon\} \tag{5.9}$$

We see that if $X_{lr}(A,B) = 1$, then the first part $(1 - X_{lr}(A,B)) = 0$. Hence $(b_l.x - a_r.x)$ has to be greater or equal than ϵ. Otherwise this constraint would not be fulfilled. But, $(b_l.x - a_r.x) \geq \epsilon$ means that B lies right of A as desired.

If $X_{lr}(A,B) = 0$, the first part is obviously M. From $a_r.x, b_l.x \in [0,N]$ follows $|b_l.x - a_r.x| \leq N$. By the choice of $M \geq N + \epsilon$, we know that $(1 - X_{lr}(A,B)) \cdot M + (b_l.x - a_r.x) \geq M - N \geq \epsilon$. Thus, the constraint is met in the second case as well and A and B can be in any horizontal relation.

Analogously to (5.9) we define the opposite horizontal case $\mathit{rightLeft}(A,B)$ with $X_{rl}(A,B)$, and for the y-coordinates the two cases $\mathit{belowAbove}(A,B)$ and $\mathit{aboveBelow}(A,B)$ with $X_{ba}(A,B)$ respectively $X_{ab}(A,B)$.

Separating the two rectangles either horizontally or vertically, at least one of the four corresponding boolean variables has to be true, i.e. 1. This yields another constraint *indepOr*.

$$\mathit{indepOr}(A,B) := \{X_{lr}(A,B) + X_{rl}(A,B) + X_{ba}(A,B) + X_{ab}(A,B) \geq 1\} \tag{5.10}$$

If one or more of the variables $X_{lr}(A,B), X_{rl}(A,B), X_{ba}(A,B), X_{ab}(A,B)$ are 1, then $\mathit{indepOr}(A,B)$ is met and we know that A and B can be separated in at least one way.

So the complete independence between two disjoint rectangles A and B can be described by four Boolean variables *indepBoolean* (5.8) and the following set of constraints, which we will refer to as $\mathit{independent}(A,B)$.

$$\begin{aligned}\mathit{independent}(A,B) := \mathit{indepOr}(A,B) \cup \mathit{leftRight}(A,B) \cup \mathit{rightLeft}(A,B) \\ \cup \mathit{belowAbove}(A,B) \cup \mathit{aboveBelow}(B,A)\end{aligned} \tag{5.11}$$

5.1.3.2 Restricted independence

The second case we have to consider is where we know in which horizontal or vertical order two disjoint rectangles have to be. Like in Figure 5.3 (a) there is sometimes only one way in which two squares could be separated. However, this is covered by the general case. It is also possible that two squares could be in different horizontal or vertical order to be disjoint. For example in Figure 5.3 (a) the squares C could be above or left of D, or both. This is covered by the general case as well. If we have have only one option or do not care in which way two squares are separated, we can look at them as completely independent. Any solution chooses the only possible or at least one of the possible cases as desired.

However, this is not always the case. For example in Figure 5.3 (b) A could be left or right of B, but only in the first case the intersection order of C would be respected. Obviously, this case can only happen between two disjoint squares both in the same side-piercing intersection with a third square. Between two squares in distinct side-piercing or opposite-corner intersections with a third square it is either fixed how they separated or it does not matter for the intersection order.

So if $A, B \in \mathcal{R}$ are disjoin and both in a side-piercing intersection of the same case with C, then we simply set the one variable of the four Boolean variables $X_{lr}(A,B)$, $X_{rl}(A,B)$, $X_{ba}(A,B)$, $X_{ab}(A,B)$, which describes the relation between A and B in the intersection order of C, to 1. Then the constraints of the general case (5.9) ensure that A and B are correctly separated. So let $A, B, C \in \mathcal{R}, A \cap B = \emptyset, A, B$ piercing rectangles in same side-piercing intersection with C with A before B in the intersection order of C. Then $\mathit{indepSidePiercing}(A,B,C)$ is defined as follows.

$$\mathit{indepSidePiercing}(A,B,C) := \begin{cases} X_{lr}(A,B) = 1, & A,B \text{ piercing } C \text{ from below} \\ X_{rl}(A,B) = 1, & A,B \text{ piercing } C \text{ from above} \\ X_{ba}(A,B) = 1, & A,B \text{ piercing } C \text{ from right} \\ X_{ab}(A,B) = 1, & A,B \text{ piercing } C \text{ from left} \end{cases} \tag{5.12}$$

For A and B in Figure 6.6 (b) this would mean $X = X_{lr}(A,B) = 1$. Then by $(1 - X_{lr}(A,B)) \cdot M + (b_l.x - a_r.x) \geq \epsilon$ we know that in any solution A is left of B.

5.1.4 Square constraints

We aim to square the rectangles. So, even if noted shortly, but nevertheless important, we also need the $\mathit{square}(A)$ constraints. They claim that both sides of each rectangle have to be of same length. Be $A \in \mathcal{R}$.

$$\mathit{square}(A) := \{a_r.x - a_l.x = a_r.y - a_l.y\} \tag{5.13}$$

5.2 The program

For a given rectangle arrangement $\mathcal{R}$ the MIP consists of the variables for each rectangle and each disjoint pair:

- $\forall A \in \mathcal{R} : \mathit{rectangleVariables}(A)$ (5.1)
- $\forall A, B \in \mathcal{R} : A \cap B \neq \emptyset : \mathit{indepBoolean}(A, B)$ (5.8)

The constraints are targeted at the square properties, the intersections and independence properties:

- $\forall A \in \mathcal{R} : \mathit{rectangle}(A) \cup \mathit{square}(A)$ (5.2),(5.13)
- $\forall A, B \in \mathcal{R} : A \cap B \neq \emptyset : \mathit{intersect}(A, B)$ (5.7)
- $\forall A, B \in \mathcal{R} : A \cap B = \emptyset : \mathit{independent}(A, B)$ (5.11)
- $\forall A, B, C \in \mathcal{R} : A \cap B = \emptyset, A, B$ in same side-piercing intersection with C : $\mathit{indepSidePiercing}(A, B, C)$ (5.12)

The MIP respects all intersections, the intersection order per square, all disjoint pairs and every rectangle is transformed to a square. Thus, the program considers all combinatorial properties. We claim that by construction this program yields a square arrangement combinatorially equivalent to $\mathcal{R}$ if and only if one exists.

Theorem 5.14. *Given a rectangle arrangement $\mathcal{R}$, the MIP defined above has a solution if and only if $\mathcal{R}$ lies in* SQUARES.

5.2.1 Example cross rectangle subarrangement

Even if we already know that a rectangle arrangement with a cross rectangle subarrangement can not be squared, we can describe a program for it. So we consider the cross rectangle arrangement $\mathcal{R}$ which only consists of the two rectangles forming the unsquarable intersection. We can see in Figure 5.5 that $\mathit{intersect}(A, B)$ would consist of $\mathit{horizontalInside}(A, B) \cup \mathit{verticalInside}(B, A)$.

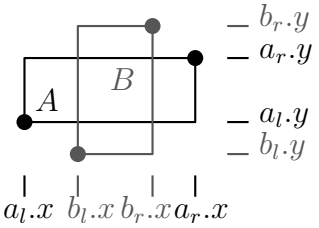

Figure 5.5: The cross rectangle arrangement and the resulting corner orders.

From the *square* constraints we get that $a_r.x - a_l.x = a_r.y - a_l.y = x$ and $b_r.x - b_l.x = b_r.y - b_l.y = y$. By $\mathit{horizontalInside}(A, B)$ we get $y + 2\epsilon \leq x$, thus $y < x$. But by $\mathit{verticalInside}(B, A)$ follows $x + 2\epsilon \leq y$ respectively $x < y$. As expected we get a contradiction. Hence, again we see a rectangle arrangement with a cross rectangle subarrangement is unsquarable.

5.3 Related work

Integer programming, quadratic programming and similar programs, like ours, are a common technique in computer science to describe and solve problems. So hardly suprising, there exist also several other work which uses algebraic solution models to solve problems concerning intersection graphs [KM94, vLvL06, vLvL11, HS02]. We outline two examples.

Van Leeuwen and van Leeuwen [vLvL11] used an algebraic model to prove that every intersection graph of certain convex polygons has an integer representation of polynomial size. They show the equivalence between solutions of their model and a representations of the corresponding intersection graph. Since they can always find a polynomial size solution, they showed that an integer representation of polynomial size exists. Furthermore they conclude that the recognition problem for those convex polygon intersection graphs is in $\mathcal{NP}$ and that a representation can be found algorithmically in exponential time.

A graph $G = (V, E)$ is a tolerance graph if there is a set $I = \{I_v \mid v \in V\}$ of closed real intervals and a set $\tau = \{\tau_v \mid v \in V\}$ of positive real numbers called tolerances such that $(x, y) \in E \Leftrightarrow |I_x \cap I_y| \geq min\{\tau_x, \tau_y\}$. Hayward and Shamir [HS02] used a system of inequalities to describe the linear orders over interval endpoints, tolerant points, bounded tolerance and interval lengths for every tolerance graph. They showed that every feasible solutions of the system yields a representation of the corresponding graph. Like van Leeuwen, they showed that there is always a polynomial size integer solution and therefore representation. Thus, they conclude the tolerance graph recognition problem is in $\mathcal{NP}$.

6. The squarability problem

In this chapter we want to examine the SQUARABILITY problem a little bit more precisely. Recall, that the problem asks whether a rectangle arrangement can be transformed into a square arrangement without changing any combinatorial properties. We described these properties by the intersection types and the intersection order per rectangle (Definition 1.10).

The previous chapter shows a way to find solutions for instances of the SQUARABILITY problem using a mixed integer program. Now we want to find characteristics of rectangle arrangements, which may tell us directly whether or not the rectangle arrangement can be squared. As we have seen with Conclusion 1.21, one such characteristics for rectangle arrangements is to not have a cross . Even with restrictions the problem does not seem easy. So we spare the general case and only look at special cases of rectangle arrangements. First we consider triangle-free ones, then those restricted to opposite-corner intersections. Last we combine both with line pierced rectangle arrangements.

6.1 Triangle free rectangle arrangement

First, we consider triangle-free rectangle arrangements. These are the rectangle arrangements where no three rectangles share a point or equivalently where the intersection graph is triangle-free. We show some properties that can prevent a triangle-free rectangle arrangement from being squarable. These properties lead to necessary but not sufficient conditions for triangle-free rectangle arrangement in order to be squarable.

6.1.1 The "smaller than" relation

When considering the squaring of a rectangle arrangement, we observe that every side-piercing intersection between two rectangles induces a "smaller than" relation for their two corresponding squares. The piercing square has to be smaller than the pierced one, in order to maintain the combinatorial properties. We define this relation and emphasise the observation with Observation 6.2.

Definition 6.1. *Let $\mathcal{R}$ be a rectangle arrangement, $A, B \in \mathcal{R}$ rectangles. Then we say A and B form a "smaller than" relation with A smaller than B, denoted by $A \prec B$, if in every square representation of $\mathcal{R}$ the square for A is smaller than the square for B.*

Observation 6.2. *A side-piercing intersection with A piercing B induces a "smaller than" relation $A \prec B$.*

"Smaller than" relations do not arise solely from side-piercing intersections. It is rather possible that the geometry of a rectangle arrangement induces that in every squaring one square has to be smaller than another. Figure 6.1 shows such a triangle-free rectangle arrangement, where a "smaller than" relation is induced not by a side-piercing intersection, but the geometry.

Observation 6.3. *The geometry of a rectangle arrangement can induce a "smaller than" relation between two of its rectangles.*

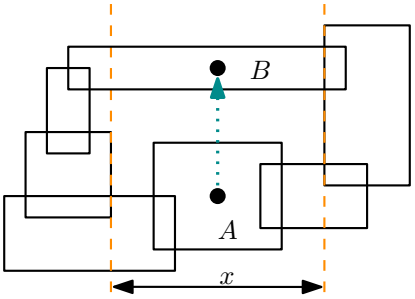

Figure 6.1: A rectangle arrangement $\mathcal{R}$ with a "smaller than" relation induced by its geometry. Enforced by the intersections is the square A is smaller than x in every squaring of $\mathcal{R}$. However, the square B has to be wider and is therefore bigger than x. Hence, the geometry induces a "smaller than" relation $A \prec B$.

Furthermore, we observe that "smaller than" relations can be implied by transitivity. If we have $A \prec B$ and $B \prec C$, we get $A \prec B \prec C$. "Smaller than" relations are obviously also transitive, i.e. $A \prec B \prec C \Rightarrow A \prec C$. This is problematic as "smaller than" relations can form a cycle, meaning $A \prec B \prec \ldots \prec C \prec A$. By the transitivity follows that $A \prec A$ and this is impossible. So we get the necessary condition, that a rectangle arrangement in order to be squarable may not induce a cycle of "smaller than" relations.

Observation 6.4. *If a rectangle arrangement $\mathcal{R}$ contains a cycle of "smaller than" relations, then $\mathcal{R} \notin$ SQUARES.*

Figure 6.2 shows that this is indeed possible. Moreover, the example of Figure 6.2 (a) indicates that such a cycle can be arbitrarily long.

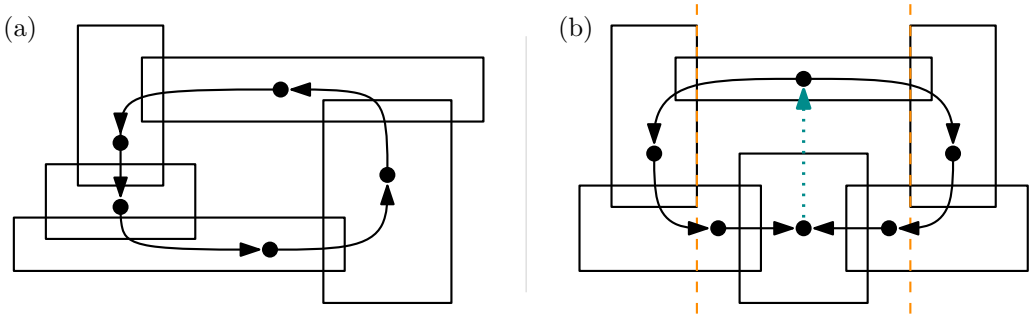

Figure 6.2: Two unsquarable rectangle arrangements since the "smaller than" relations form a unresolvable cycle. On the left all "smaller than" relations arise from side-piercing intersections, in the right one is induced by the geometry.

Detecting "smaller than" relations induced by side-piercing intersections is easy. However, it is not that obvious how to find "smaller than" relations induced by geometry. Moreover, it is also possible that a triangle-free rectangle arrangement induces a cycle of "smaller

than" relations which are all induced by its geometry. Figure 6.3 shows such a rectangle arrangement. This means, that in order to find potential induced "smaller than" relation cycles, we have to check between any two rectangles whether they induce a "smaller than" relation.

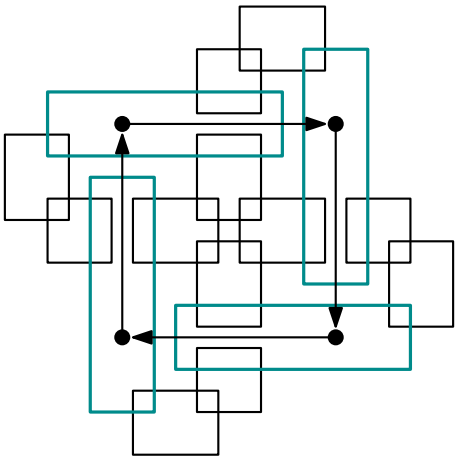

Figure 6.3: An unsquarable triangle-free rectangle arrangement, with a cycle of "smaller than" relations induced solely by geometry.

6.1.2 Variable "smaller than" relations

Obviously it is possible that a square A of a triangle-free rectangle arrangement is only in some square arrangements smaller than a square B. This gets interesting if further either A is smaller than B or a square X smaller than a square Y.

We consider another example. Figure 6.4 (a) shows a counterclockwise spiral of rectangles. The rectangle Z induces in a squaring that the square for X has to be smaller than the one for Y, i.e. $X \prec Y$. Therefore we get a "smaller than" relation cycle and the triangle-free rectangle arrangement is unsquarable. The interesting thing about this example is, that without Z the spiral could be squared by rolling it clockwise, as shown in the Figure 6.4 (b).

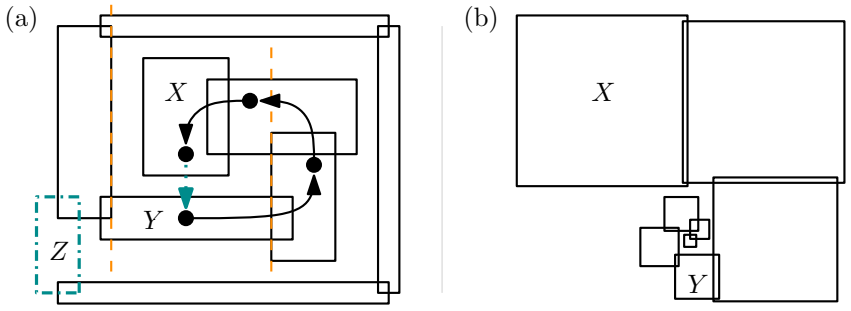

Figure 6.4: Again indicated by the orange lines we get a "smaller than" relation shown by the cyan arrow. Together with the side-piercing intersections we get an unresolvable cycle. Note, that the rectangle Z is crucial. Without it, the spiral can be unrolled and the rectangle arrangement squared like shown on the right.

We can now create another interesting triangle-free rectangle arrangement. This is shown in Figure 6.5. We take two of the spirals from Figure 6.4 and connect them. Thereby the left spiral with A and B is rolled counterclockwise and the right clockwise. However, in order to square the rectangle arrangement, as we have seen, the left spiral has to be rolled clockwise, and the right counterclockwise. Assume we roll the right spiral counterclockwise

and allow X to be bigger than Y. Then the left spiral can not be rolled clockwise at the same time. This induces that A has to be smaller than B. So, we induce either $A \prec B$ or $X \prec Y$. Thus the triangle-free rectangle arrangement is unsquarable, since each of them closes a "smaller than" cycle.

Definition 6.5. *Two rectangle pairs A, B and X, Y of a rectangle arrangement $\mathcal{R}$ form a pair of coupled "smaller than" relations, denoted by $A \prec B \otimes X \prec Y$, if in every squaring of $\mathcal{R}$ either A is smaller than B or X is smaller than B.*

The example shows that it is not sufficient to check statically for induced "smaller than" relations. Quite the contrary, depending on pairs of coupled "smaller than" relations, we have to check if a combinatorially equivalent square arrangement exists, such that no cycle of "smaller than" relations is induced.

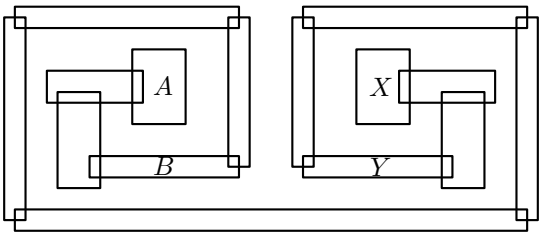

Figure 6.5: Like in Figure 6.4 the spirals have to be unrolled in order to square the rectangle arrangement. But since this can not be done for both at the same time, the rectangle arrangement is unsquarable.

In addition, the example shows also that it is not sufficient for a triangle-free rectangle arrangement to be squarable, if its intersection graph is a tree or even just a path.

6.1.3 Yet another reason for unsquarability

Not only cycles of "smaller than" relations, normal or coupled ones, prevent triangle-free rectangle arrangement from being squarable. It is also possible that a group of squares has to be of specific large size, but there is not enough space for them.

We consider the unsquarable triangle-free rectangle arrangement $\mathcal{R}$ shown in Figure 6.6. We assume for the sake of contradiction that $\mathcal{R}$ is squarable. Then in every squaring of $\mathcal{R}$ the square S has to be bigger than A and B, due to the side-piercing intersections. Now, each of the two pairs M, M' and N, N' has to bridge the distance x between A and B. However, x is bigger than S and therefore also bigger than A and B. If now M bridges $y < x$ of the gap, then N has to be strictly smaller than $x - y$ in order to fit into A. On the other side M' has to be of size strictly bigger than $x - y$ to close the gap. Then again, this induces that N' has to be strictly smaller than $x - (x - y) = y$ to fit in B too. Together $N < (x - y)$ and $N' < y$ are strictly smaller than $x = (x - y) + (y)$ and can therefore not close the gap. Hence $\mathcal{R}$ is unsquarable.

This example shows that size relations do not only occur between single squares but also between groups of them. Hence, to solve the SQUARABILITY problem it is not sufficient to find all normal and coupled "smaller than" relations and then check if they form a cycle.

6.2 Restricted to opposite-corner intersections

Note that each example of an unsquarable arrangement above contains a side-piercing intersection. Therefore, the second special case we consider are triangle-free rectangle arrangements restricted to opposite-corner intersections. We recall Definition 1.15 that

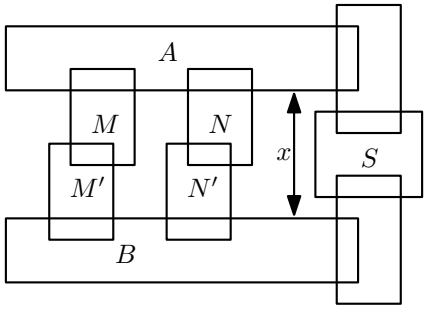

Figure 6.6: An unsquarable triangle-free rectangle arrangement. Either the pairs M, N and M', N' do not fit into A respectively B, or they are not large enough to connect A and B.

these are rectangle arrangements with no containing and side-piercing intersection and no cross.

We observe that in a triangle-free rectangle arrangement restricted to opposite-corner intersections each rectangle can intersect with at most four other rectangles, since it can only intersect with each of its corner with at most one other rectangle. This yields that its intersection graphs has a maximal degree of at most 4. Since we have one or more topmost rectangles we know further that the minimal degree can be at most 2.

Observation 6.6. *The intersection graph of a triangle-free rectangle arrangement restricted to opposite-corner intersections has maximal degree at most 4 and minimal degree at most 2.*

For rectangle arrangements with this restrictions we can prove the following theorem.

Theorem 6.7. *If the intersection graph of a rectangle arrangement $\mathcal{R}$ restricted to opposite-corner intersections is a tree (or forest), than $\mathcal{R} \in$ Squares, i.e. $\mathcal{R}$ is squarable.*

Proof. So, let $\mathcal{R}$ be a triangle-free rectangle arrangement $\mathcal{R}$ restricted to opposite-corner intersections, such that $G_{\mathcal{R}}$, its intersection graph, is a tree. Without loss of generality $\mathcal{R}$ is connected. Picking any $v \in V(G_{\mathcal{R}})$ as the root of the tree, be h the height of the tree.

Now, as we do not care about the sizes of the squares, we can embed $\mathcal{R}$ into a rectangle arrangement $\mathcal{R}'$, similar to the one shown in Figure 6.7. Let $\mathcal{R}'$ be the triangle-free rectangle arrangement restricted to opposite-corner intersections with intersection graph $G_{\mathcal{R}}{}'$ a tree rooted at w, such that every leaf has distance h to the root and every inner non-leaf vertex has degree 4. Then $\mathcal{R}'$ can be constructed as follows.

Let the starting square for the root vertex be of size x_0. We set the size x_1 for its four children, one for each corner, to be a quarter of it, $x_1 = \frac{x_0}{4}$, and place them with a quarter of their own size in the first rectangle. This step is repeated for every level k of the tree at the new free corners with squares of size $x_k = \frac{x_{k-1}}{4}$. Since $\sum_{i=1}^{h} \frac{1}{4^i} \leq \frac{1}{3}$, we know that surely no rectangles accidentally intersect at and between any level.

By Observation 6.6 the intersection graph $G_{\mathcal{R}}$ of $\mathcal{R}$ has maximal degree 4. Since we picked the height for $\mathcal{R}'$ as the height of $G_{\mathcal{R}}$ for root v, we can now map v to w and its children respectively. Obviously, leaving out the squares of $\mathcal{R}'$ to which no square of $\mathcal{R}$ was mapped yields a squaring of $\mathcal{R}$. □

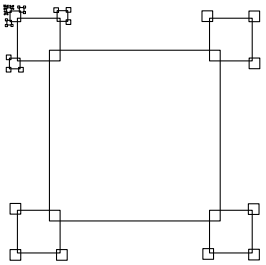

Figure 6.7: A square arrangement with intersection graph being a tree and recursively smaller sizes for the squares, such that no unintended intersections happen.

However, we are not yet able to go one step further, from tree intersection graphs to outerplanar intersection graphs. Moreover, we could neither prove nor disprove that triangle-free rectangle arrangements restricted to opposite-corner intersections or even all rectangle arrangements restricted to opposite-corner intersections are squarable.

6.3 Line pierced rectangles

Now we address another special case of rectangle arrangements. We consider the case that all rectangles are pierced by one horizontal line. This means they all share a y-coordinate. With Definition 1.16 we called them line pierced rectangle arrangements. Figure 6.8 gives a first example.

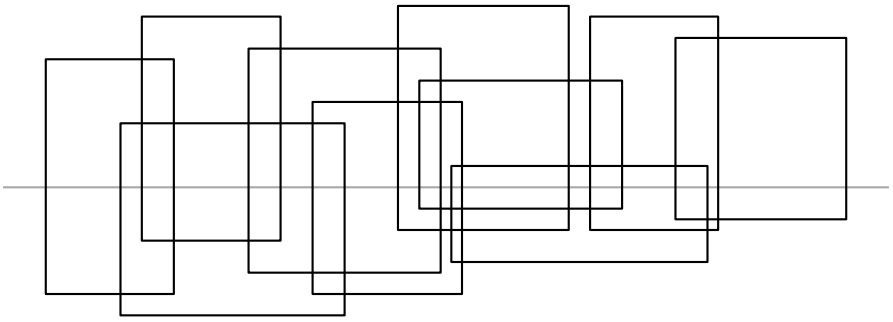

Figure 6.8: A line pierced rectangle arrangement, i.e. all rectangles can be pierced by one horizontal line.

We observe, that in a line pierced rectangle arrangement it is not possible that two rectangle share a x-coordinate without intersecting each other. Otherwise they could not be both pierced by one horizontal line. This means, that we can determine whether two rectangles intersect by looking at their spanned x-intervals. More formally, $A = [a, b] \times [c, d]$ and $B = [a', b'] \times [c', d']$ intersect if and only if $[a, b]$ and $[a', b']$ intersect. This directly proves the following lemma. An interval graph is the intersection graph of a set of intervals on the real line.

Observation 6.8. *The intersection graph of a line pierced rectangle arrangement $\mathcal{R}$ equals the interval graph, given by the horizontally spanned intervals of the rectangles in $\mathcal{R}$.*

6.3.1 Line pierced triangle-free rectangle arrangements

We add again the restriction that the rectangle arrangement is triangle-free. When we now consider the SQUARABILITY problem , i.e. asking whether we can square a line pierced triangle-free rectangle arrangement without changing its combinatorial properties, we have

to decide if this includes that the resulting triangle-free square arrangement has to be line pierced as well.

If we require it to be line pierced, then we can give a small counterexample which is not squarable under this terms.

Lemma 6.9. *Not every line pierced triangle-free rectangle arrangement without a cross can be squared such that the resulting square arrangement is line pierced as well.*

Proof. The smallest (in terms of the number of rectangles) line pierced triangle-free rectangle arrangement which is squarable in general, but not squarable into a line pierced square arrangement is shown in Figure 6.9.

Lets assume for the sake of contradiction that in a squaring the outer square has size 1 and the line shall pierce it at height x. Than the left smaller rectangle as a square has to have size at least $y > x$ in order to be pierced by the line too. The second one on his part has to have size $z > 1 - x$. Together they would have a width $y + z > 1 - x + x = 1$. Hence, they would not fit both together into the big square next to each other. □

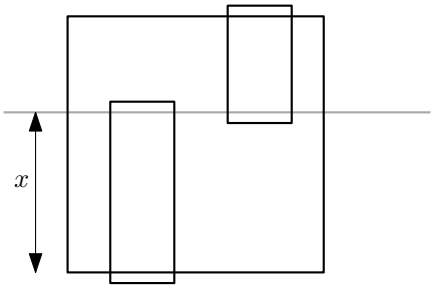

Figure 6.9: A line pierced triangle-free rectangle arrangement which can not be squared into a line pierced square arrangement.

If we do not require the squared arrangement to be also pierced by a line, it behaves quite different. In fact, then every line pierced triangle-free rectangle arrangement without a cross can be squared. To prove this we make another observation with the following lemma. A graph is called a caterpillar, if it is a tree consisting only of one central path and legs, which are leaf vertices with distance exactly one to this path.

Lemma 6.10. *The intersection graph of a line pierced triangle-free rectangle arrangement is a caterpillar.*

Proof. To prove this we first give the central path and the legs. After that, we prove that they cover all vertices. We use the corner color intersection graph introduced in Chapter 4 to describe the line pierced triangle-free rectangle arrangement. Remember that each edge of this graph is actually an edge pair. However, the edge pairs are only used to describe the intersection types and the caterpillar and tree properties are of course on the more general edges.

So, let $\mathcal{R}$ be a line pierced triangle-free rectangle arrangement, $G_{\mathcal{R}}$ its corner color intersection graph and l the piercing horizontal line. First, we determine the central path P. Again, we assume $\mathcal{R}$ is connected. The idea is, that the path goes from left to right along the rectangles, which intersect at the line l. Therefore, $\mathcal{R}$ contains a leftmost rectangle $A = [a, b] \times [c, d]$ and the vertex $v_A \in V(G_{\mathcal{R}})$ is the first on P. The next rectangle $B = [a', b'] \times [c', d']$ on the right of A, i.e. where $a < a' < b < b'$, yields the second vertex

v_B for P. We see, that A and B have to be in a left or right side-piercing intersection or an opposite-corner intersection. Thus, we can build P from left to right along these intersections. In Figure 6.10 A and B form an opposite-corner intersection.

Every side-piercing intersection on top or bottom, a containing or an unsquarable intersection yields one leg of the caterpillar. It is given by the corresponding vertex which is not on P, like C and D in Figure 6.10.

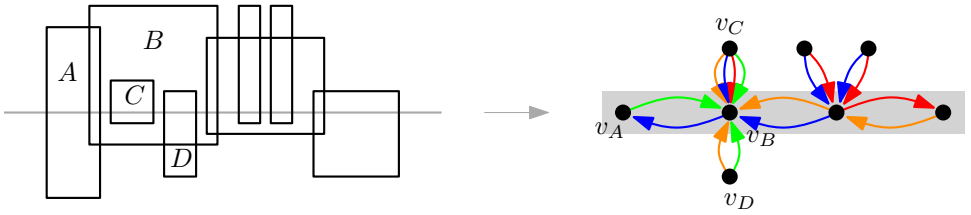

Figure 6.10: A line pierced triangle-free rectangle arrangement and its resulting corner color intersection graph, a caterpillar. The central path is highlighted with a grey area and the vertices outside of it are the feet of the legs.

Now, we have to prove that, on the one hand, this covers all rectangles and, on the other hand, this yields a caterpillar. So assume for the sake of contradiction, that a rectangle E is not covered by the central path or the legs. However, this would mean, that E does not stand in an opposite-corner or side-piercing intersection with a rectangle on its left, as well as it is in a side-piercing intersection from top or bottom or is contained in another one. However, since $\mathcal{R}$ is connected, this is impossible.

In order to prove that we get a caterpillar, we show that the P is indeed a path and all other vertices, the legs, are in distance 1 to P. Since $\mathcal{R}$ is triangle-free and line pierced, we can observe that a rectangle F can be part of only one side-piercing or opposite-corner intersection on its right. Otherwise $\mathcal{R}$ would not be triangle-free or the two partners on the right of F could be separated by a horizontal line k and therefore not both can be pierced by l. See Figure 6.11 for this.

Second, we observe that if a rectangle D stands in a top or bottom side-piercing intersection with a rectangle B on the path, it can not intersect with any other rectangle H. Otherwise either D and H would intersect or could again be separated by a horizontal line. This is shown in Figure 6.11 as well. However, since G is connected, this implies that P is indeed a path and the other vertices stand in distance 1 to P. Hence, $G_{\mathcal{R}}$ is a caterpillar. □

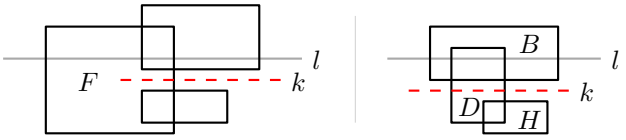

Figure 6.11: The two impossible cases described in the proof of Lemma 6.10.

Lemma 6.10 enables us to prove the following theorem nicely.

Theorem 6.11. *Every line pierced triangle-free rectangle arrangement without a cross $\mathcal{R}$ is squarable, i.e. $\mathcal{R} \in$ SQUARES.*

Proof. By Lemma 6.10 the corner color intersection graph $G_{\mathcal{R}}$ of $\mathcal{R}$ is a caterpillar and thus we have a central path P. So, in order to square $\mathcal{R}$, in the first step we square the

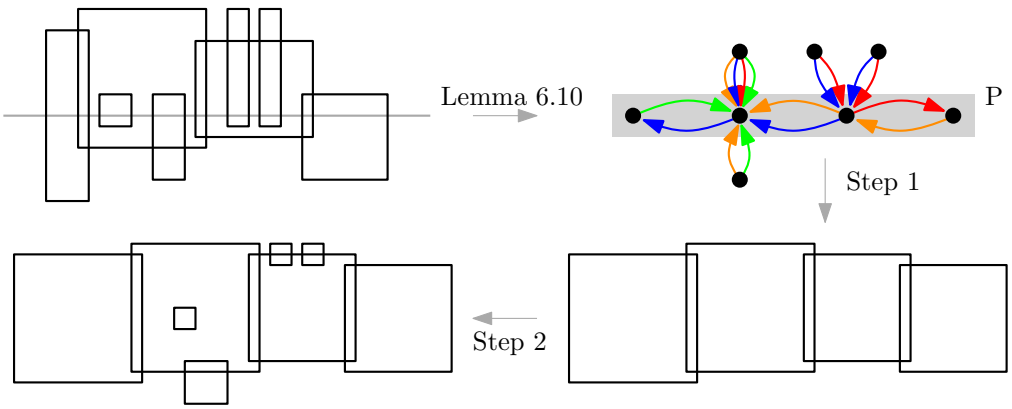

Figure 6.12: Picture to the proof of Theorem 6.11. Simply squaring the rectangles from left to right and making those inside, piercing from above or below as small as needed.

rectangles on P. In the second step we add the rectangle for the legs. The whole process is illustrated in Figure 6.12.

Step 1: Let $\epsilon > 0$ be sufficiently small. Starting with rectangle A on the path P and neighbours B on P, we set the square $A' = [a, b] \times [c, d], (b - a = d - c)$ for A and $B' = [a', b'] \times [c', d'], (b' - a' = d' - c')$ for B, such that they intersects horizontally with ϵ, i.e. $a' = b' - epsilon$, and vertically the way the rectangles intersect. If A side-pierces B than $c' + \epsilon = c$ and $d' - \epsilon = d$, and vice versa. If A and B are in an opposite-corner intersection, with, say, A lower than B, than $c' + \epsilon = c$ and $d' + \epsilon = d$. Repeating this process, we get a chain of squares as the squaring of the rectangles on P.

Step 2: For every leg of $G_{\mathcal{R}}$ we place a square sufficiently small with the correct intersection to the chain of squares from step 1. If we choose the size of these squares small enough, as well as ϵ small enough, it is obviously possible since then there is enough space. □

6.3.2 Line pierced rectangle arrangements

If we remove the restrictions of being triangle-free for the rectangle arrangements, we can give again a necessary condition in order to be squarable. We already discussed that side-piercing intersections induce "smaller than" relations for the corresponding squares. Since we are no longer triangle-free, it is now possible to find rectangle arrangement which induce "smaller than" relations in their squaring. Figure 6.13 shows such a line pierced rectangle arrangement.

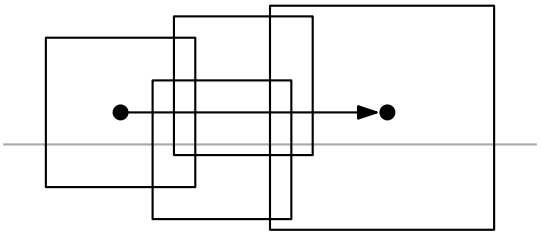

Figure 6.13: A line pierced rectangle arrangement with a "smaller than" relation induced by its geometry.

Like for triangle-free rectangle arrangements, we have of course that line pierced rectangle arrangements, in order to be squarable, may not contain a cycle of "smaller than" relations. Figure 6.14 and Figure 6.15 give examples how this could be possible for line

pierced rectangle arrangements, with and without "smaller than" relations induced by their geometry. It is further wort mentioning, that the intersection graph for the arrangement in Figure 6.14 a) has a maximal clique number of only three.

Being line pierced restricts the rectangle arrangement. However, like we have seen with the previous examples being non triangle-free enables many new configurations. The unsquarable line pierced rectangle arrangement in Figure 6.16 below makes no exception to that. This examples shows also that in order to be squarable it is not sufficient for a line pierced rectangle arrangement to have no cycle of "smaller than" relations.

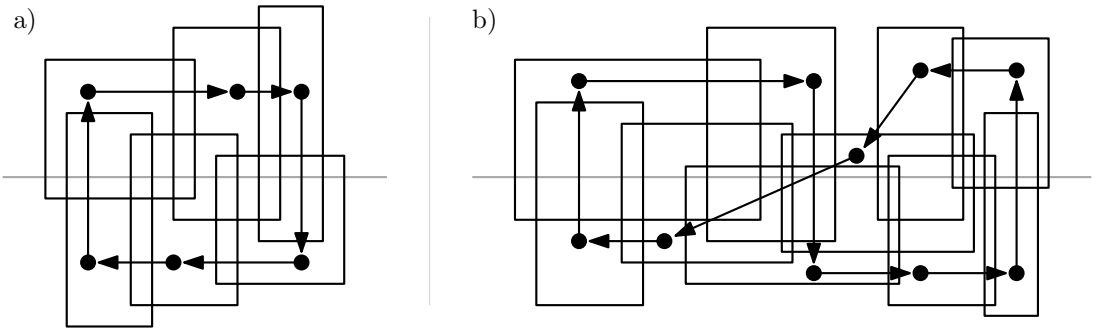

Figure 6.14: Two unsquarable line pierced rectangle arrangement due to a cycles of "smaller than" relations, all induced by side-piercing intersections.

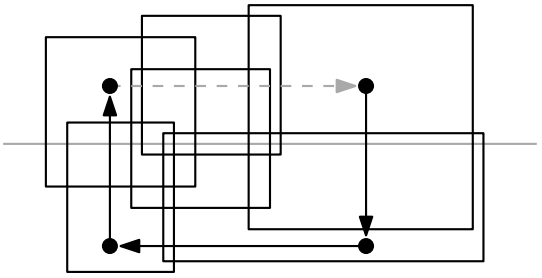

Figure 6.15: An unsquarable line pierced rectangle arrangement due to a cycle of "smaller than" relations. Here one relation is induced by the geometry like in Figure 6.13.

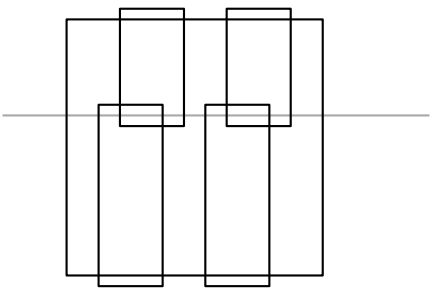

Figure 6.16: An unsquarable line pierced rectangle arrangement without a "smaller than" relations cycle.

6.3.3 Line pierced and restricted to opposite-corner intersections

For the last special case of line pierced rectangle arrangements, we do no longer request them to be triangle-free, but restrict them to opposite-corner intersections. This restriction yields some nice properties. Being restricted to opposite-corner intersections means there are no containing, side-piercing or unsquarable intersections. By the definition of opposite-corner intersections (Definition 1.5) two intersecting rectangles $A = [a, b] \times [c, d]$ and $B = [a', b'] \times [c', d']$ have alternating corners, horizontally and vertically, i.e. $a < a' < b < b'$ and $c < c' < d < d'$ or equivalently. For the y-coordinate we can conclude, that the order of the rectangle borders above and below the piercing line are the same.

Observation 6.12. *At every y-coordinate of a line pierced rectangle arrangement restricted to opposite-corner intersections is the order of the upper rectangle borders and the lower rectangle borders the same and they are parted by the piercing line.*

We make two observations for the x-coordinate. First, the rectangles are strictly ordered horizontally. A rectangle which starts more left than another, ends also before this other one. Otherwise their x-coordinates would not alternate. So by choosing the left border of the rectangles we get an total ordering. Second, as a conclusion of the first, we have that the rectangle which starts most left also ends most left.

Observation 6.13. *Let $\mathcal{R}$ be a line pierced rectangle arrangement restricted to opposite-corner intersections. Be $A \in \mathcal{R}$ the leftmost rectangle, i.e. its left border is leftmost among all left sides of rectangles in $\mathcal{R}$. Then the right side of A is leftmost among all right borders of rectangles in $\mathcal{R}$.*

Restricting line pierced rectangle arrangements to opposite-corner intersections is a harder restriction than requesting them to be triangle free in respect to the Squarability problem. Then, we can not only square them, we can also pierce the resulting square arrangement by a line.

Theorem 6.14. *Every line pierced rectangle arrangement restricted to opposite-corner intersections is squarable, such that the resulting square arrangement can be pierced by a line.*

We prove an even stronger statement. It is possible to only use unit squares for the squaring.

Theorem 6.15. *Every line pierced rectangle arrangement restricted to opposite-corner intersections is squarable, such that the resulting square arrangement consists only of unit squares and can be pierced by a line.*

Proof. We prove this by induction on the number n of rectangles in a line pierced rectangle arrangement $\mathcal{R}$ restricted to opposite-corner intersections. For $n = 1, 0$ it is trivial. So be $n > 1$.

Let A' be the leftmost rectangle of $\mathcal{R}$ and $\mathcal{R}' = \mathcal{R} \backslash \{A'\}$. So the size of $\mathcal{R}'$ is $n - 1$. By induction hypothesis follows, $\mathcal{R}'$ is squarable such that the resulting square arrangement consists only of unit squares and can be pierced by a line l. Now we show how to insert the unit square A for the rectangle A' into the squaring of $\mathcal{R}'$, such that result is a squaring of $\mathcal{R}$ fulfilling the requirements. This means that A has to intersect with all squares corresponding to the rectangles intersected by A' and A is pierced by l. First we give the horizontal placement of A, then the vertical. As we will see, both are just one dimensional problems.

Horizontal placement of A: Be $A_x := [a_n, b_n]$ the x-interval of A and $I := \{[a_i, b_i] \mid i = 1, \ldots, n-1\}$ the set of x-intervals of all squares in the squaring of $\mathcal{R}'$. Be further I sorted from right to left and $I_A := \{[a_i, b_i] \mid i = n-1-k, \ldots, n-1\} \subseteq I$ the k intervals for which the squares intersect with A. By Observation 6.8 we have to place the A_x such that the resulting interval graph together with I yields the intersection graph of $\mathcal{R}$. Since A' is the most left rectangle of $\mathcal{R}$ and by Observation 6.13, we know that $b_n < b_i \; \forall i = 1, \ldots, n-1$. Moreover, all intervals including A_x have length 1, since they come from unit squares. This means it is possible to set a_n most left while still b_n is right of all a_i for squares A has to intersect with. Otherwise the b_n could not be left most. Figure 6.17 illustrates this facts. All together, we can set $A_x = [a_n, b_n]$ such that $a_n < a_i \; \forall i = 1, \ldots, n-1$, $b_n > a_i \; \forall i = 1, \ldots, k$ and $b_n < b_i \; \forall i = 1, \ldots, n-1$.

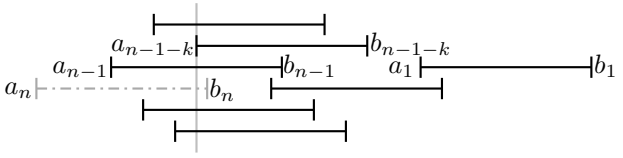

Figure 6.17: Placement for the $x-$interval of the unit square added in the induction step.

Vertical placement of A: We know by Observation 6.12 that the order of upper and lower endpoints of the y-interval is the same, and again all intervals have length 1. We place the y-interval A_y of the unit square A, such that it fits in the order given by $\mathcal{R}'$ at b_n and that its endpoints lie below respectively above the piercing line. It is necessary and sufficient to place A_y in respect to the order at b_n, since left of b_n are only less intervals to consider. Figure 6.18 illustrates this vertical placement. □

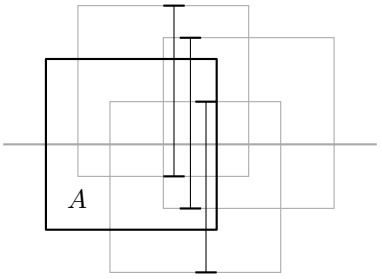

Figure 6.18: Placement for the y-interval of the unit square added in the induction step.

7. Summary and outlook

In this bachelor thesis we introduced the SQUARABILITY problem, a problem connecting geometry and combinatorics. The problem can be motivated by the fact that the similar STRETCHABILITY problem for pseudolines and lines is broadly studied. Furthermore, many results on square arrangements can be immediately generalized for squarable rectangle arrangements.

In Chapter 3, as a first step, we showed a sweep line algorithm to compute the combinatorial information from geometrically given rectangles. This can be done in $O(n \log n + k)$ time and space, where k is the number of intersections.

In Chapter 4 we introduced the concept of enhanced intersection graphs, namely corner color intersection graphs, to represent and characterize triangle-free rectangle arrangements. The main aspect of these graphs is that the intersection types between rectangles are encoded by the color and orientation of the edges. Each edge represents a corner of a rectangle which lies inside another rectangle. Triangle-free rectangle arrangements have a plane intersection graph and thus a plane corner color intersection graph. However, this property is lost without the restriction to triangle-free arrangements. We leave it an open problem to characterize also non triangle-free rectangle arrangements.

Open problem 7.1. *How can rectangle arrangements be characterized combinatorially?*

In Chapter 5 we defined a mixed integer linear program which can be used to decide whether a given rectangle arrangement is squarable. The program considers all pairs of rectangles, both intersecting and disjoint. The number of considered disjoint pairs could be reduced by using transitive relations between several rectangles. An implementation of the program should be straight-forward.

In Chapter 6 we have shown several examples indicating that the SQUARABILITY problem is not even for all classes of restricted rectangle arrangements easy to solve. We have considered triangle-free rectangle arrangements, for which we could give some necessary conditions in order to be squarable. However, we could not prove what properties are sufficient for a triangle-free rectangle arrangement to be squarable.

Open problem 7.2. *When are triangle-free rectangle arrangements squarable? What are sufficient conditions for a triangle-free rectangle arrangement in order to be squarable?*

We have shown that all unsquarable rectangle arrangements contain either side-piercing, containing or unsquarable intersections. Therefore we considered rectangle arrangements restricted to opposite-corner intersections in Section 6.2. We could easily prove that these arrangements are always squarable, if the intersection graph is a tree. However, we could neither prove nor disprove that fewer restrictions for the intersection graph result in unsquarable arrangements. Thus, we state the following open problems.

Open problem 7.3. *Are all rectangle arrangements restricted to opposite-corner intersections where the intersection graph is outerplanar squarable?*

Open problem 7.4. *Are all triangle-free rectangle arrangements restricted to opposite-corner intersections squarable?*

Open problem 7.5. *Are all rectangle arrangements restricted to opposite-corner intersections squarable?*

As a third special case we considered line pierced rectangle arrangements in Section 6.3. Again we could not give sufficient conditions for the arrangements to be squarable. However, we could prove that triangle-free line pierced rectangle arrangements and line pierced rectangle arrangements restricted to opposite-corner intersections are always squarable.

Open problem 7.6. *When are line pierced rectangle arrangements squarable?*

Ultimately also the question of computational complexity of Squarability remains open. We could not answer in which class the Squarability problem is. This means we can not tell whether the problem is in $\mathcal{P}$, $\mathcal{NP}$, $\mathcal{NP}$-complete or $\mathcal{NP}$-hard. Building gadget arrangements for problem reductions turned out to be quite difficult and our efforts remained without success.

Open problem 7.7. *What is the computational complexity of the* Squarability *problem?*

Bibliography

[AdBMS98] P. Agarwal, M. de Berg, J. Matousek, and O. Schwarzkopf, "Constructing levels in arrangements and higher order voronoi diagrams," *SIAM Journal on Computing*, vol. 27, no. 3, pp. 654–667, 1998.

[AG60] E. Asplund and B. Grünbaum, "On a coloring problem." *Mathematica Scandinavica*, vol. 8, pp. 181–188, 1960.

[BCKO08] M. d. Berg, O. Cheong, M. v. Kreveld, and M. Overmars, *Computational Geometry: Algorithms and Applications*, 3rd ed. Santa Clara, CA, USA: Springer-Verlag TELOS, 2008.

[BO79] J. Bentley and T. Ottmann, "Algorithms for reporting and counting geometric intersections," *Computers, IEEE Transactions on*, vol. C-28, no. 9, pp. 643–647, Sept 1979.

[Bur65] J. P. Burling, "On coloring problems of families of prototypes," 1965.

[BW80] J. L. Bentley and D. Wood, "An optimal worst case algorithm for reporting intersections of rectangles," *Computers, IEEE Transactions on*, vol. C-29, no. 7, pp. 571–577, July 1980.

[DJ12] A. Dumitrescu and M. Jiang, "Coloring translates and homothets of a convex body," *Beiträge zur Algebra und Geometrie / Contributions to Algebra and Geometry*, vol. 53, no. 2, pp. 365–377, 2012.

[FP12] J. Fox and J. Pach, "Coloring k_k-free intersection graphs of geometric objects in the plane," 2012.

[GP80] J. E. Goodman and R. Pollack, "On the combinatorial classification of nondegenerate configurations in the plane," *Journal of Combinatorial Theory, Series A*, vol. 29, no. 2, pp. 220 – 235, 1980.

[Gr2] B. Grünbaum, *Arrangements and Spreads*, ser. Conference Board of the Mathematical Sciences regional conference series in mathematics. Conference Board of the Mathematical Sciences, 1972.

[GW84] R. H. Guting and D. Wood, "Finding rectangle intersections by divide-and-conquer," *Computers, IEEE Transactions on*, vol. C-33, no. 7, pp. 671–675, July 1984.

[Hen98] C. Hendler, "Schranken für Färbungs- und Cliquenüberdeckungszahl geometrisch represäntierbarer Graphen," 1998.

[HS02] R. B. Hayward and R. Shamir, "A note on tolerance graph recognition," 2002.

[KKN04] S.-J. Kim, A. Kostochka, and K. Nakprasit, "On the chromatic number of intersection graphs of convex sets in the plane," *Electron. J. Combin*, vol. 11, no. 1, p. R52, 2004.

[KM94] J. Kratochvil and J. Matousek, "Intersection graphs of segments," *Journal of Combinatorial Theory, Series B*, vol. 62, no. 2, pp. 289 – 315, 1994.

[Kos04] A. Kostochka, "Coloring intersection graphs of geometric figures with a given clique number," *Contemporary mathematics*, vol. 342, pp. 127–138, 2004.

[Per03] I. Perepelitsa, "Bounds on the chromatic number of intersection graphs of sets in the plane," *Discrete Mathematics*, vol. 262, no. 1–3, pp. 221 – 227, 2003.

[Ric89] J. Richter, "Kombinatorische Realisierbarkeitskriterien fur orientierte Matroide," *Mitt. Math. Sem. Gießen*, vol. 194, pp. 1–112, 1989.

[Rin56] G. Ringel, "Teilungen der Ebene durch Geraden oder topologische Geraden." *Mathematische Zeitschrift*, vol. 64, pp. 79–102, 1956.

[SH76] M. I. Shamos and D. Hoey, "Geometric intersection problems," in *Foundations of Computer Science, 1976., 17th Annual Symposium on*, Oct 1976, pp. 208–215.

[Sho91] P. W. Shor, "Stretchability of pseudolines is np-hard," *Applied Geometry and Discrete Mathematics–The Victor Klee Festschrift (P. Gritzmann, B. Sturmfels, eds.), DIMACS Series in Discrete Mathematics and Theoretical Computer Science, Amer. Math. Soc., Providence, RI*, vol. 4, pp. 531–554, 1991.

[TOG04] C. Toth, J. O'Rourke, and J. Goodman, *Handbook of Discrete and Computational Geometry, Second Edition*, ser. Discrete and Combinatorial Mathematics Series. Taylor & Francis, 2004.

[vLvL06] E. J. van Leeuwen and J. van Leeuwen, "On the representation of disk graphs," Utrecht University, Tech. Rep., 2006.

[vLvL11] ——, "Convex polygon intersection graphs," in *Graph Drawing*, ser. Lecture Notes in Computer Science, U. Brandes and S. Cornelsen, Eds. Springer Berlin Heidelberg, 2011, vol. 6502, pp. 377–388.

List of Figures