Master's Thesis

Rotation Systems and Good Drawings

conducted at the
Institute for Software Technology
Graz University of Technology, Austria

by
Jürgen Pammer, 0730592
Telematics Master's Programme, F 066 411

Supervisor:
Assoc.Prof. Dipl.-Ing. Dr.techn. Oswin Aichholzer

Graz, September 16, 2014

Statutory Declaration

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

date (signature)

Abstract

This master's thesis deals with *good drawings* of graphs, that is, drawings that are restricted to certain crossing properties. In particular, edges in such drawings are allowed to have at most one point in common, a mutual endvertex or a proper crossing.

We recall in detail some properties of such good drawings of the complete graph K_n on n vertices and the relationship to *rotation systems*, that is, the cyclic order of all edges emanating from the vertices in the graph drawing. This involves the notions of *isomorphism* and *weak isomorphism* of graph drawings that are subsequently used to distinguish between essentially different drawings of K_n under both definitions. We present an algorithm to enumerate all good drawings of the complete graph $D(K_n)$ for small n, and compile a database of all drawings under both kinds of isomorphism.

Additionally, we take a closer look at a special kind of good drawings, so called *thrackles*. These are good drawings where every pair of edges has exactly one point in common. It was conjectured in the 1970's by John Conway that such drawings cannot have more edges than vertices; however, until today no proof could be given. We summarize most of what is known about thrackles and give computational results similar to the ones concerning complete graphs, including the fact that a possible counterexample to the conjecture of Conway would be required to have at least 13 vertices.

At the end of the thesis, some further results on topics related to good drawings are presented.

Zusammenfassung

Diese Masterarbeit beschäftigt sich mit *"Good Drawings"* von Graphen, also Zeichnungen die auf bestimmte Kreuzungseigenschaften beschränkt sind. Im Speziellen sind die Kanten solcher Zeichnungen darauf beschränkt, maximal einen gemeinsamen Punkt aufzuweisen. Dieser kann ein gemeinsamer Knoten oder eine Kreuzung der Kanten sein.

Die Eigenschaften solcher "Good Drawings" des vollständigen Graphen K_n auf n Knoten und der Zusammenhang mit *"Rotation Systems"*, also der zyklischen Reihenfolge der ausgehenden Kanten um die Knoten der Zeichnung des Graphen, werden im Detail zusammengefasst. Dies beinhaltet die Begriffe des *Isomorphismus* und des *schwachen Isomorphismus* von "Good Drawings", welche dazu genutzt werden, um zwischen im wesentlichen unterschiedlichen Zeichnungen, im Sinne der jeweiligen Definition, zu unterscheiden. Es wird ein Algorithmus angegeben, der alle unterschiedlichen "Good Drawings" $D(K_n)$ des vollständigen Graphen für kleine n enumeriert und eine Datenbank aller Zeichnungen unter beiden Varianten des Isomorphismus erstellt.

Zusätzlich wird eine spezielle Art von "Good Drawings" genauer betrachtet, die sogenannten *"Thrackles"*. Dabei handelt es sich um "Good Drawings", bei denen jedes Paar von Kanten genau einen Punkt gemeinsam haben muss. In den 1970er Jahren vermutete John Conway, dass solche Zeichnungen nicht mehr Kanten als Knoten aufweisen können, jedoch gibt es bis heute keinen Beweis dafür. Der Großteil der diesbezüglich bekannten Resultate wird zusammengefasst und die durch ein Computerprogramm berechneten Ergebnisse, ähnlich denen des vollständigen Graphen, werden präsentiert. Dies umfasst die Tatsache, dass ein mögliches Gegenbeispiel zur Vermutung von Conway mindestens 13 Knoten haben muss.

Am Ende der Arbeit werden zusätzliche Ergebnisse, die in Zusammenhang mit "Good Drawings" stehen, angeführt.

Acknowledgements

At this point I would like to thank my supervisor, Oswin Aichholzer, for his incredible support and patience throughout writing this thesis. I am especially grateful to him for giving me the opportunity to attend two research workshops in “Ratsch an der Weinstraße” and the final conference of the EuroGIGA project in Berlin during that time.

Furthermore, I express my gratitude to Thomas Hackl, Alexander Pilz, and Birgit Vogtenhuber from the Institute for Software Technology at Graz University of Technology for always being willing to discuss problems and subjects related to my thesis. I especially thank Thomas and Alex for additionally taking a thorough look at my write-up, in turn, helping me enhance the quality of this work.

I also highly appreciate the attendees of the first research workshop in fall 2013. I would like to thank Oswin Aichholzer, Luis Felipe Barba, Thomas Hackl, Alexander Pilz, and Birgit Vogtenhuber, who were involved in compiling the notes on the content elaborated throughout the workshop. The notes are integrated into this document in the form of a chapter on related topics (Chapter 6). In addition, I highly appreciate all people who also attended the workshop and contributed to the presented material: Michael Hoffmann, László Kozma, Vincent Kusters, Raimund Seidel, Emo Welzl, and Manuel Wettstein.

At this point I would also like to mention Richard Coombs who allowed me to use his very nice drawing of a torus that I adapted in order to create Figure 5.5.

Finally, I wish to thank my parents Regina and Manfred. My education at university and this thesis would have never been possible without their support and interest throughout the last years.

Contents

1 Introduction

An abstract graph is a representation of objects called *vertices* that are related to each other. This relation is indicated by connections between such vertices, called edges. While there are many different kinds of graphs, for instance graphs with directed or weighted edges, the simplest form is a set of vertices together with a set of undirected edges, that link pairs of vertices to each other.

A common way of visualizing a graph is to draw the vertices as dots on a sheet of paper and connect them by lines or curves that represent the edges. It is clear that an abstract graph can be visualized like that in many different ways. The discipline of graph drawing examines certain properties of such representations in the Euclidean plane. For reasons to be discussed in a later chapter of this thesis, we will consider so-called *good drawings* of graphs that essentially have the property that any pair of edges meets at most once, either in a common vertex or in a proper crossing.

One important property of graph drawings is the way in which the drawn edges intersect each other, and one of the most famous problems is to determine whether an abstract graph can be drawn in such a way that none of its edges cross, i.e., determining whether a graph is *planar*. A famous result by Kazimierz Kuratowski states that a graph is planar if and only if it does not contain a subgraph that is a subdivision of either the complete graph K_5 or the complete bipartite graph $K_{3,3}$. The 'only if' part of the theorem is quite intuitive, since the said graphs themselves cannot be drawn without crossings. The other part takes a bit more effort to prove.

Apart from the above result, there has been quite a lot of research concerning properties of non-planar graphs. While it is computationally easy to verify whether a graph has a plane drawing or not, it is incredibly hard to pin down the exact minimum number of crossings that necessarily occur in any drawing of the graph.

Definition 1.1. The crossing number $cr(G)$ of a graph G is the minimum number of crossings among all drawings of G.

The Hungarian mathematician Paul Turán first came to think of the crossing number of a special class of graphs, namely complete bipartite graphs, while working in a Hungarian labour camp during World War II. The following quote is from an article Turán himself wrote [67] and describes the origin of the said problem:

> *In July 1944 the danger of deportation was real in Budapest, and a reality outside Budapest. We worked near Budapest, in a brick factory. There were some kilns where the bricks were made and some open storage yards where the bricks were stored. All the kilns were connected by rail with all the storage yards. The bricks were carried on small wheeled trucks to the storage yards. All we had to do was to put the bricks on the trucks at the kilns, push the trucks to the storage yards, and unload them there. We had a reasonable piece rate for the trucks, and the work itself was not difficult; the trouble was only at the crossings. The trucks generally jumped the rails there, and the bricks fell out of them; in short this caused a lot of trouble and loss of time which was rather precious to all of us (for reasons not to be discussed here).*

To formalize the situation in a mathematical way, consider the kilns to be one class of vertices in a complete bipartite graph, and the storage yards to be the other. The edges connecting the vertices of opposite classes represent the railway tracks and the goal is to minimize the number of crossings among the edges in a drawing of the complete bipartite graph $K_{m,n}$. Due to obvious reasons, this special (and most prominent) variant of the crossing number problem is known as the *brick factory problem.*

A first attempt to find a solution was made by Kazimierz Zarankiewicz [73]. He supposedly gave a formula for any m, n to determine the minimum number of crossings in any drawing of $K_{m,n}$. His proof, however, contained a serious flaw pointed out by Kainen and Ringel (see Guy [32], Erdős and Guy [20]). Still, his result is a valid upper bound for $cr(K_{m,n})$.

Theorem 1.1 (Zarankiewicz)**.** *The crossing number $cr(K_{m,n})$ of the complete bipartite graph $K_{m,n}$ satisfies the following inequality:*

$$cr(K_{m,n}) \leq \lfloor \tfrac{n}{2} \rfloor \lfloor \tfrac{n-1}{2} \rfloor \lfloor \tfrac{m}{2} \rfloor \lfloor \tfrac{m-1}{2} \rfloor$$

Theorem 1.1 is known to be valid since Zarankiewcz gave a construction to obtain exactly that many crossings for any complete bipartite graph. Unfortunately, his attempts to show that no other drawing can have fewer crossings have failed and the erroneous proof could not be fixed until today. On the other hand, equality of the formula could be established for several special cases, which also support the conjecture that in Theorem 1.1 indeed equality holds. See Kleitman [41] who showed equality for $\min\{m, n\} \leq 6$, and also Vogtenhuber [68] and Woodall [72] for further results.

Conjecture 1.2 (Zarankiewicz)**.** *The crossing number $cr(K_{m,n})$ of the complete bipartite graph $K_{m,n}$ satisfies the following equality:*

$$cr(K_{m,n}) = \lfloor \tfrac{n}{2} \rfloor \lfloor \tfrac{n-1}{2} \rfloor \lfloor \tfrac{m}{2} \rfloor \lfloor \tfrac{m-1}{2} \rfloor$$

The equivalent problem for drawings of complete graphs is said to have its origins in a less mathematical environment. Most resources credit the first examinations of complete graph drawings with a minimal number of crossings to the British artist Anthony Hill. He experimented with drawings for small n and came up with crossing-minimal examples for up to $n = 9$. Together with Frank Harary, he later published a paper that also contained conjectured values for the crossing number for $n \leq 10$, where they also explicitly mentioned the case where edges are represented by straight-line segments only. The first paper to be published mentioning a conjecture similar to that of Zarankiewicz in the case of complete bipartite graphs was, however, due to Guy.

Conjecture 1.3 (Guy [32], Harary and Hill [33])**.** *The crossing number $cr(K_n)$ of the complete graph K_n satisfies the following equality:*

$$cr(K_n) = Z(n) = \tfrac{1}{4} \lfloor \tfrac{n}{2} \rfloor \lfloor \tfrac{n-1}{2} \rfloor \lfloor \tfrac{n-2}{2} \rfloor \lfloor \tfrac{n-3}{2} \rfloor$$

While a general resolution of the Harary-Hill conjecture probably requires a great advancement, there are certain classes of good drawings for which the statement could be verified. Definitions 1.2 to 1.5 can e.g. be found in Ábrego et al. [1].

Definition 1.2. In a *2-page drawing* of a graph all vertices are placed on a line and the edges are, except for their endvertices, entirely contained in one of the halfplanes defined by that line.

Definition 1.3. In a *cylindrical drawing* of a graph, there are two concentric circles that host all the vertices, and no edge is allowed to intersect these circles, other than at its endvertices.

Definition 1.4. A drawing is *monotone* if each vertical line intersects each edge at most once.

Definition 1.5. A drawing is x-bounded if by labelling the vertices $v_1, v_2, ..., v_n$ in increasing order of their x-coordinates, for all $1 \leq i < j \leq n$ the edge $v_i v_j$ is contained in the strip bounded by the vertical line that contains v_i and the vertical line that contains v_j.

Ábrego et al. [1] established that every 2-page drawing has at least $Z(n)$ crossings. Later they extended their result to a broader class of drawings (see [1]). They introduced the concept of *shellability* and proved that for drawings with certain shelling properties Conjecture 1.3 holds.

Definition 1.6 (Ábrego et al., [1])**.** A drawing D of K_n is *s-shellable* if there exists a subset $S = \{v_1, v_2, ..., v_s\}$ of the vertices and a region R of D with the following property. For $1 \leq i < j \leq s$, if D_{ij} denotes the drawing obtained from D by removing $v_1, v_2, ...v_{i-1}, v_{j+1}, v_{j+2}, ..., v_s$, then for all $1 \leq i < j \leq s$, the vertices v_i and v_j are on the boundary of the region of D_{ij} that contains R.

Theorem 1.4 (Ábrego et al., [1]). *Let D be an s-shellable drawing of K_n , for some $s \geq \lfloor n/2 \rfloor$. Then D has at least $Z(n)$ crossings.*

It was shown that all of the different types of drawings from Definitions 1.2 to 1.5 are indeed shellable for some $s \geq \lfloor \frac{n}{2} \rfloor$. This settles the Harary-Hill conjecture for these restricted classes of good drawings.

Theorem 1.5 (Ábrego et al., [1]). *The crossing number of 2-page, cylindrical, monotone, and x-bounded drawings is at least $Z(n)$.*

An interesting observation concerning shellability is that it is a property determined solely by the circular order of the edges emanating from the vertices in the drawing. This means that every realizable good drawing with the same rotation system has the same properties concerning shellability [1]. Further, see [8] where an interesting relationship between shellability and monotonicity of drawings is given.

1.1 Rotation Systems

Throughout this thesis we will make use of the circular order of edge incidences because it does not only determine shellability, but also has an interesting relation to crossing properties of some classes of graphs. We will denote this information as the *rotation system* of a graph; however, terminology is not consistent throughout literature and some authors refer to it as the "rotation scheme".

We provide a summary of the historical usage of rotation systems, as defined in [48]. Later in this work, we adjust it slightly in order to fit the needs for drawings of graphs, particularly in the plane.

Definition 1.7 ([48, p. 90]). For each vertex $v \in V(G)$ of a graph G let π_v denote a cyclic permutation of edges incident with v. We may call such an ordering the *rotation* at v and denote the collection of rotations at each vertex of G by $\pi = \{\pi_v : v \in V(G)\}$, a *rotation system* for G.

1.1.1 Embedding graphs in surfaces

Rotation systems were first used by Heffter [38] in 1891 for embedding graphs in orientable surfaces. Embedding in this context means to draw the graph on a surface without crossings. We can distinguish between two kinds of surfaces, namely orientable and non-orientable ones. Informally speaking, a surface is *orientable* if an object cannot be moved around on the surface in a closed curve such that it ends up at its starting point looking like its mirror image. If the said transformation is possible, we call such a surface *non-orientable*. For orientable surfaces the *genus* is the number of handles one needs to attach to the sphere to obtain its homeomorphism

[1] Personal communication: Oswin Aichholzer, Thomas Hackl, Alexander Pilz, Birgit Vogtenhuber

type. The genus of a graph is the minimum genus of a surface where this graph can be embedded without crossings. Examples of orientable surfaces are the sphere with genus 0 and the torus with genus 1. The projective plane and the Klein bottle are, for instance, non-orientable surfaces both having non-orientable genus 1. Definitions in the non-orientable case are similar. For details refer to [31]. Heffter could show that the genus $\gamma(K_n)$ of the complete graph on n vertices obeys the following inequality for all n:

$$\gamma(K_n) \geq \left\lceil \frac{(n-3)(n-4)}{12} \right\rceil \tag{1.1}$$

Further, he could prove that for $n \leq 12$ Equation 1.1 is in fact an equality, and conjectured this to be true for all n; however, it was not until 1968 that a proof could be established [61].

Theorem 1.6 (Ringel and Youngs, 1968)**.** *For $n \geq 3$ the genus of the complete graph on n vertices $\gamma(K_n) = \lceil (n-3)(n-4)/12 \rceil$.*

Note that a similar result could be established for the non-orientable genus of K_n [57], namely $\bar{\gamma}(K_n) = \lceil (n-3)(n-4)/6 \rceil$. Furthermore, Ringel has also used rotation systems to prove similar results for the complete bipartite graph $K_{m,n}$ [59].

Theorem 1.7 (Ringel, 1965)**.** *For all $m, n \geq 2$ the orientable genus $\gamma(K_{m,n}) = \lceil (m-2)(n-2)/4 \rceil$, and the non-orientable genus $\bar{\gamma}(K_{m,n}) = \lceil (m-2)(n-2)/2 \rceil$.*

Throughout literature (see e.g. [31, 48, 66]) the usage of rotation systems is independently credited to Edmonds [18] in 1960. He used the last sentence of the following theorem explicitly.

Theorem 1.8 ([48])**.** *Suppose that G is a connected multigraph with at least one edge that is cellularly embedded in an orientable surface $\mathcal{S}$. Let π be the rotation system of this embedding, and let $\mathcal{S}'$ be the surface of the corresponding 2-cell embedding of G. Then there exists a homeomorphism of $\mathcal{S}$ onto $\mathcal{S}'$ taking G in $\mathcal{S}$ onto G in $\mathcal{S}'$ (in such a way that we induce the identity map from G onto its copy in $\mathcal{S}'$). In particular, every cellular embedding of a graph G in an orientable surface is uniquely determined, up to homeomorphism, by its rotation system.*

Speaking of homeomorphism, it is convenient to define equivalence for two rotation systems π and π', and finally state an interesting corollary (also presented in [48]).

Definition 1.8. The rotation systems $\pi = \{\pi_v | v \in V(G)\}$ and $\pi' = \{\pi'_v | v \in V(G)\}$ are said to be equivalent if they are either the same or for each vertex $v \in V(G)$ we have that $\pi'_v = \pi_v^{-1}$.

Corollary 1.9. *Suppose that we have a cellular embedding of a connected multigraph G in orientable surfaces $\mathcal{S}$ and $\mathcal{S}'$ with rotation systems π and π', respectively. Then there is a homeomorphism $\mathcal{S} \to \mathcal{S}'$ whose restriction to G induces the identity if and only if π and π' are equivalent.*

It should be remarked here that the notion of rotation systems can be extended in order to adapt the preceding theorem and corollary for the case of non-orientable surfaces. This was first achieved by Ringel [60] in 1977 and Stahl [63] in 1978.

1.1.2 The map colour theorem

In 1890 Heawood conjectured an interesting relation between the genus $\gamma(\mathcal{S})$ of an orientable surface $\mathcal{S}$ and its *chromatic number* $\chi(\mathcal{S})$. The latter is defined as the maximum chromatic number of any graph embeddable in $\mathcal{S}$, where the chromatic number of a graph is the minimum number of colours required in order to assign colours to each vertex of the graph such that no adjacent vertices have the same colour. Evidently, it suffices to only consider simple graphs since replacing multiple edges by a single one does not alter the chromatic number of the underlying graph.

While many special cases could be proven by various authors (including Ringel), it was not until 78 years later that Ringel and Youngs could, with extensive use of rotation systems, establish Heawood's conjecture as a theorem.

Theorem 1.10 (Ringel and Youngs, [61])**.** *For any orientable surface $\mathcal{S}_\gamma$ with orientable genus $\gamma \geq 1$ the chromatic number $\chi(\mathcal{S}_\gamma) = \lfloor (7 + \sqrt{1 + 48\gamma})/2 \rfloor$.*

Interestingly, the above theorem states that $\gamma \geq 1$. This explicitly excludes planar graphs embedded in the sphere $\mathcal{S}_0$, while setting $\gamma = 0$ yields a chromatic number of 4 by the formula in Theorem 1.10. This case is known as the famous Four Colour Theorem and was finally verified in 1976 by Appel and Haken [5] with heavy assistance of computers. Based on the same approach, a simpler proof was published in 1997 by Robertson et al. [62]; however, it is still computer assisted.

In Theorem 1.10 the genus γ can be replaced by the Euler characteristics c of a surface in order to also include non-orientable surfaces. With $c = V - E + F$, where V, E, and F denote the number of vertices, edges, and faces of a graph embedded in an arbitrary surface, respectively, we get the following equation valid for all surfaces, except for the Klein bottle.

Theorem 1.11 (Ringel and Youngs, [61])**.** *For any surface $\mathcal{S}_c$ with Euler characteristics c, except for the Klein bottle, the chromatic number $\chi(\mathcal{S}_c) = \lfloor (7+\sqrt{49 - 24c})/2 \rfloor$. The Klein bottle $\mathcal{N}_1$ has chromatic number $\chi(\mathcal{N}_1) = 6$.*

The special case of the Klein bottle was already settled by Franklin [23] in 1934. He could prove that for every embedding of a graph in the Klein bottle, six colours suffice.

1.1.3 Graphs in the plane

Apart from being useful for characterizing surfaces, rotation systems have (more recently) also been used specifically for graphs in the plane (or, equivalently, the sphere).

Donafee and Maple [17] presented an algorithm to determine planarity of a graph and a given rotation system. In many applications it is of advantage that their method also considers the circular ordering of edges around each vertex (e.g. genetics, VLSI, communication design, network optimization). Additionally, the algorithm is also valid for non-simple graphs, which many others are not.

Definition 1.9. The crossing number $cr(G)$ of a graph G is the minimum number of edge crossings in any drawing of G in the plane.

It is known since the 80's that the problem of determining $cr(G)$ of a graph G is **NP**-complete [27], and in 2008 Pesmajer et al. [55] could show that it still remains to be **NP**-complete for graphs with fixed rotation systems. Moreover, the same is true for another interesting problem.

Definition 1.10. A graph G is said to be 1-planar if there exists a drawing of G in the plane such that each edge is crossed at most once.

NP-completeness of the problem of testing 1-planarity was shown by Korzhik and Mohar [42], and Auer et al. [7] gave a proof that the situation remains the same given rotation systems. Both results are interesting considering that simply testing planarity is well-known to be possible in linear time. Additionally, Auer et al. [7] showed that the crossing number problem for 1-planar graphs remains **NP**-complete even given a fixed rotation system.

For the previously presented problems, the use of rotation systems made no difference in terms of complexity; however, for upward planarity it does.

Definition 1.11. A directed graph G is *upward planar* if it can be drawn in the plane without crossings and in such a way that the curves representing the edges of G are monotonically increasing in y-direction.

It is known to be **NP**-complete to test upward planarity [28]. Given a rotation system this question can be answered in linear time [11, 16].

1.2 Further Problems Related to Crossing Properties

Aside from relaxing planarity by allowing each edge to be crossed at most once, there are many other problems concerning edge crossings that have been considered.

One among these is the problem of determining the number of disjoint edges, meaning pairs of arcs drawn in such a way that they do not share a point, in good drawings of the complete graph $D(K_n)$. A first result was published by Pach, Solymosi and Tóth [50] and stated that the number of disjoint edges in any $D(K_n)$ is $\Omega(\log(n)^{1/6})$. Pach and Tóth [53] improved this bound to $\Omega(\log(n)/\log(\log(n)))$, and subsequently posed the problem whether it is true that there exists a constant $c > 0$, such that every good drawing of the complete graph with n vertices has at least n^c disjoint edges (see Problem 4 in Chapter 9 of [13]). The question was first answered in the affirmative by Suk [64] in 2012. Later Fulek and Ruiz-Vargas [26] gave a simpler proof of the same result that the number of disjoint edges in any $D(K_n)$ is $\Omega(n^{1/3})$.

A similar problem with respect to crossing properties of complete graph drawings is presented in the following. An edge that is not properly intersected by any other edge of the drawing $D(K_n)$ is called *uncrossed* or *unavoidable*. The latter term is appropriate because such an edge is necessarily part of any maximal planar subdrawing of $D(K_n)$. Harborth and Mengersen [35] showed that for $n \leq 7$ in any good drawing of the complete graph there exist uncrossed edges. See Table 1.1 for the exact minimum number of uncrossed edges $h(n)$ in any $D(K_n)$.

n	2	3	4	5	6	7	≥ 8
$h(n)$	1	3	4	4	3	2	0

Table 1.1: Minimum number of unavoidable edges in any $D(K_n)$.

As the last entry in the above table suggests, there is always a $D(K_n)$ for $n \geq 8$, where each edge is crossed at least once. At first this may seem easy to achieve, but due to the restrictions imposed on the way edges are allowed to cross in good drawings, one needs to make such a construction with care. For details refer to Section 6.6.

Aside from investigating such planar structures in graph drawings, there is another class of graphs with certain well-defined crossing properties of the edges that deserves thorough examination. Namely, graphs that can be drawn in the plane in such a way that each pair of edges crosses or has a vertex in common. This can be regarded as the inverse of planarity. A long standing conjecture by John H. Conway (see [71]) states that any such good drawing can have at most as many edges as vertices. Despite a lot of research in this direction, the conjecture could neither be proved nor disproved since it was first stated in the 1960's. For further details refer to Chapter 5.

1.3 Outline of the Thesis

We start in Chapter 2 by recapitulating basic graph theoretic and topological preliminaries that are required throughout this thesis. Notations therein are taken largely from standard textbooks on graph theory and are partly adapted when considered adequate.

In Chapter 3 we cover properties of good drawings in detail and introduce rotation systems in this context. Afterwards we establish the connection between the rotation system of drawings of the complete graph and their crossing properties, hence, the weak isomorphism of drawings. We also describe an extended version of a rotation system to further apply to a more general form of isomorphism that takes the order of crossings along the edges into account.

The information elaborated is then used in the practical part of this thesis (Chapter 4). We describe an algorithm to enumerate all good drawings of K_n for small n under both forms of isomorphism defined in the preceding chapter. We present the results of the implementation of the algorithm together with a first attempt to visualize the obtained drawings.

Chapter 5 is dedicated to Conway's thrackle conjecture and summarizes most of what is known to date. We present results achieved so far towards resolving the conjecture and give insight in what is known about the properties of counterexamples, in the case that one does exist. We use an algorithm to obtain all non-isomorphic thrackle drawings, similarly used for enumerating good drawings of the complete graph. The algorithm is further adapted to only enumerate certain kinds of graphs that are drawn as thrackles in order to be able to verify that the conjecture is true for $n \leq 12$.

In Chapter 6 we deal with topics related to the contents of the thesis that were not covered in detail while this thesis was conducted. Most of the content provided originated from a workshop held in Ratsch in fall 2013. The following people were involved in work on the presented topics: Oswin Aichholzer, Luis Felipe Barba, Thomas Hackl, Michael Hoffmann, László Kozma, Vincent Kusters, Alexander Pilz, Raimund Seidel, Birgit Vogtenhuber, Emo Welzl, and Manuel Wettstein.

The final chapter summarizes the main contents of this work and recalls open problems that can be subject of future work.

2 Preliminaries

This chapter recapitulates basic definitions and notations that are frequently used throughout this thesis.

2.1 Basic Definitions

Basic definitions are largely taken from [12].

Definition 2.1 ([12, p. 2])**.** A *graph* G is an orderer pair $(V(G), E(G))$ consisting of a set $V(G)$ of *vertices* and a set $E(G)$ of *edges*, together with an *incidence function* ψ_G that associates with each edge of G an unordered pair of (not necessarily distinct) vertices of G.

Definition 2.2 ([12, p. 2])**.** We denote the *order*, i.e., the number of vertices, and the *size*, i.e. the number of edges, of a graph G by $n = v(G) = |V(G)|$ and $m = e(G) = |E(G)|$, respectively.

Definition 2.3 ([12, p. 2])**.** If e is an edge and u and v are vertices such that $\psi_G(e) = \{u, v\}$, then e is said to *join* u and v, and the vertices u and v are called the *ends* of e.

Note that Definition 2.1 admits *parallel edges*, that is, edges that join the same pair of vertices, and edges joining one and the same vertex, so called *loops*. However, it is often meaningful to only consider a more restricted class of graphs.

Definition 2.4 ([12, p. 3])**.** A *simple graph* G is a graph without parallel edges and loops.

Definition 2.5 ([12, p. 5])**.** A graph G is *connected* if, for every partition of its vertex set into two nonempty sets X and Y, there is an edge with one end in X and one end in Y; otherwise G is *disconnected*.

Figure 2.1 shows several examples of graphs. The one to the left is not simple because it has parallel edges joining the vertices u and v. Additionally, there is a loop at w. The graph in the centre and the graph to the right, on the other hand, are both simple. Also, despite being drawn in different ways, these two representations depict the same underlying graph.

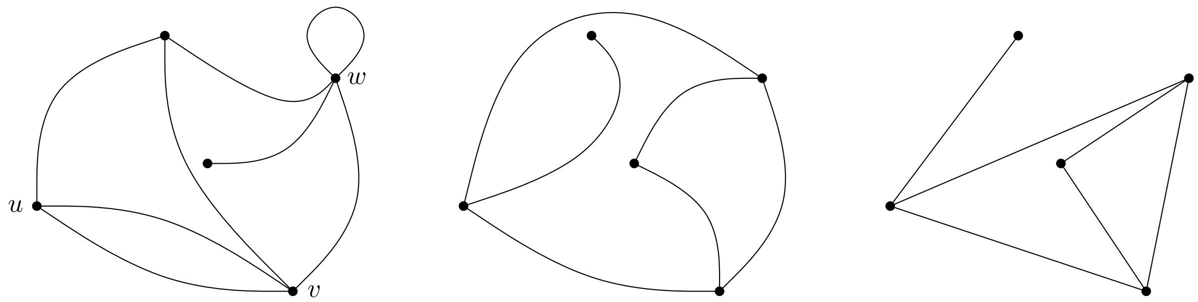


Figure 2.1: Three different graphs on the same set of five vertices.

Although graphs are a much more abstract concept than simple pencil drawings on a sheet of paper, many definitions are inspired by this way of thinking about graphs.

Definition 2.6 ([12, p. 3])**.** The ends of an edge are said to be *incident* with the edge, and vice versa.

Definition 2.7 ([12, p. 3])**.** Two vertices which are incident with a common edge are *adjacent*, as are two edges which are incident with a common vertex.

Definition 2.8 ([12, p. 3])**.** Two distinct adjacent vertices are *neighbours* and the set of neighbours of a vertex v is denoted by $N_G(v)$.

2.2 Special Classes of Graphs

The most frequently used graph throughout this work is a graph where each vertex is joined to any other vertex by exactly one edge.

Definition 2.9 ([12, p. 4])**.** A *complete graph* K_n is a simple graph on n vertices in which any two vertices are adjacent.

An apparent reason why such graphs are of importance is that any other simple graph on n vertices is clearly a subgraph of K_n.

Definition 2.10 ([12, pp. 4,10])**.** A *k-partite graph* is one whose vertex set can be partitioned into k subsets, or *parts*, in such a way that no edge has both ends in the same part. Such a graph is *complete* if any two vertices in different parts are adjacent. In particular, if $k = 2$ with a partition into two vertex sets, we call them *bipartite* or *complete bipartite graphs.*

We now define some other classes of graphs that frequently occur throughout this thesis.

Definition 2.11 ([12, p. 4])**.** A *star* S is a complete bipartite graph $K_{m,n}$ with either $m = 1$ or $n = 1$. We call a star with k edges a k-*star* and denote it by S_k.

Definition 2.12 ([12, p. 4]). A *path* P is a simple graph whose vertices can be arranged in a linear sequence in such a way that two vertices are adjacent if they are consecutive in the sequence, and are nonadjacent otherwise. We call a path on k edges a *k-path* and denote it by P_k.

Definition 2.13 ([12, p. 4]). A *cycle* C on three or more vertices is a simple graph whose vertices can be arranged in a cyclic sequence in such a way that two vertices are adjacent if and only if they are consecutive in the sequence. We call a cycle on k vertices a *k-cycle* and denote it by C_k.

Definition 2.14 ([12, p. 99]). A graph that does not contain a cycle is called *acyclic.* A *tree* T is a connected acyclic graph.

In a tree any two vertices are connected by exactly one unique path. Since any graph with a minimum degree of at least two contains a cycle, any tree must have at least one vertex of degree one. Moreover, the number of edges in a tree is well defined.

Theorem 2.1 ([12, p. 100]). *If T is a tree, then $e(T) = v(T) - 1$.*

Definition 2.15 ([12, p. 46]). A *wheel* W is obtained by joining a single vertex v to all vertices of a k-cycle C_k. Edges joining v to C_k are *spokes* and we call a wheel with k spokes a *k-wheel* and denote it by W_k.

2.3 Graph Drawings and Planar Graphs

Since it generally gives some insight to think about graphs as drawn on a sheet of paper, and a large part of this thesis deals with such drawings, it is required to give some more precise definitions on this topic.

Definition 2.16 ([48, pp. 4-5]). In a *drawing* of a graph G vertices are represented by points in the plane and edges are simple (polygonal) curves joining the points corresponding to their ends. Curves representing the edges are allowed to cross each other, but their interiors do not contain any vertices of the graph.

The above definition clearly allows to draw one and the same abstract graph in arbitrarily many different ways. However, a natural endeavour when illustrating a graph in such a way is probably avoiding edges that cross each other, if that is somehow possible.

Definition 2.17 ([12, pp. 243-244]). A graph is said to be *embeddable in the plane*, or *planar*, if it can be drawn in the plane so that its edges intersect only at their ends. Such a drawing is called a *plane embedding* and will often be referred to as a *planar graph.*

When talking about drawings of graphs in the plane, it is necessary to point out some intuitively obvious things, that; however, are quite delicate to prove rigorously.

Theorem 2.2 (Jordan Curve Theorem, [48, p. 25])**.** *Any simple closed curve C in the plane divides the plane into exactly two arcwise connected components. Both of these regions have C as the boundary.*

Theorem 2.3 (Jordan-Schönflies Theorem, [48, p. 25])**.** *If f is a homeomorphism of a simple closed curve C in the plane onto a closed curve C' in the plane, then f can be extended to a homeomorphism of the entire plane.*

The meaning of the Jordan Curve Theorem is obvious. Whenever we draw a simple closed curve on a sheet of paper, it will partition the paper into two parts, the inside and outside, where the closed curve serves as the boundary of the partition. The Jordan-Schönflies Theorem is fundamental for defining isomorphism of graph drawings as we will do later on. In other words it states that any simple, closed curve in the plane is homeomorphic to the unit circle.

3

Good Drawings and Rotation Systems

The general definition of graph drawings (Definition 2.16) clearly has one big downside. It makes no restrictions on how complicated one can make a drawing. As an example see Figure 3.1. All three pictures show drawings of the complete graph K_4. The graph to the left, although being an absolute valid graph drawing, seems to be unnecessarily complicated. One and the same pair of edges crosses many times which can easily be avoided as illustrated in the centre drawing; however, the arcs representing the edges still appear to be too arbitrary. In this case even straight line segments suffice to capture the key property of the edges drawn red and green, namely that they cross.

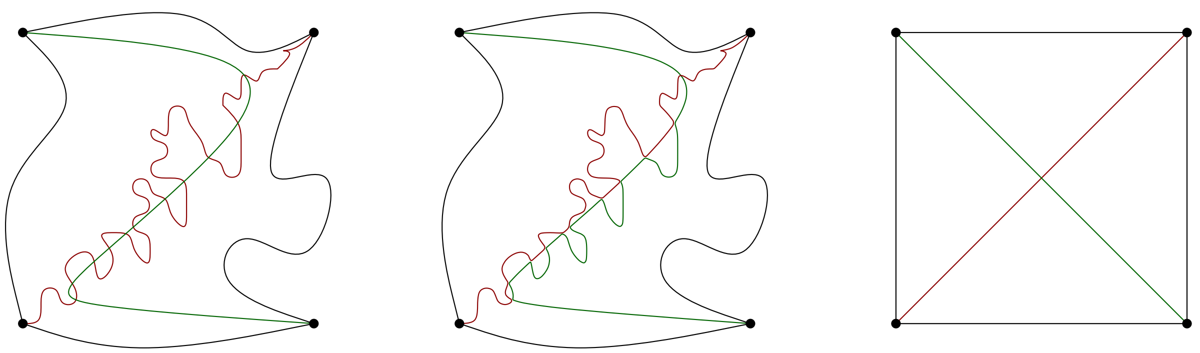
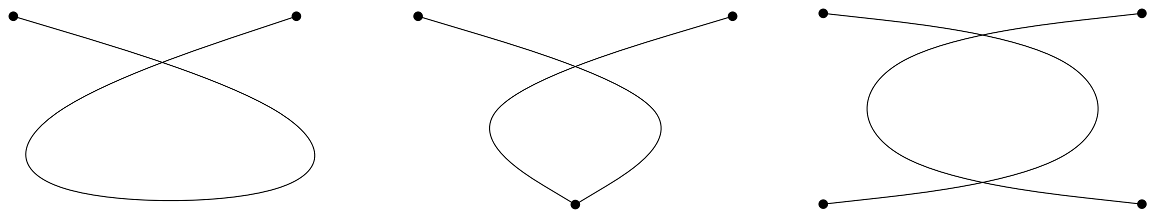


Figure 3.1: A drawing, good drawing, and geometric drawing of K_4.

The simplification of reducing the number of crossings on the pair of edges from above might require some more justification. An important observation is that this can, in fact, always be done. Additionally, it can be guaranteed that such a redrawing does not produce additional crossings on other edges. It is therefore meaningful to define such restricted drawings of graphs.

Definition 3.1. A *good drawing* $D(G)$ of a graph G is a drawing of the graph where each vertex is represented by a distinct point in the plane (or equivalently on the sphere) and edges are drawn as simple arcs connecting the corresponding endpoints (vertices). Additionally, no arc is allowed to pass through any vertex except for its endpoints, and every pair of arcs is allowed to cross at most once, either at their common endpoint, or in the interior of the edges.

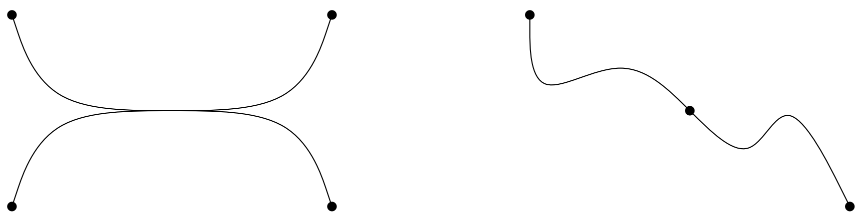
In Figure 3.2 invalid crossings in good drawings are depicted. An edge drawn such that it crosses itself (Figure 3.2(a)) contradicts that arcs are required to be simple. Furthermore, incident edges are clearly not allowed to cross again (Figure 3.2(b)) because any pair of arcs can have at most one point in common, whether it is an endpoint or a proper crossing (Figure 3.2(c)).



(a) Self-crossing edge. (b) Incident edges. (c) Double-crossing.

Figure 3.2: Forbidden crossings in good drawings.

We should also point out some subtleties concerning degenerate drawings such as arcs drawn in a way that they "touch" in one point (Figure 3.3(a)), but do not properly intersect each other. Meaning, they have a point in common, but one edge does not continue on the opposite side of the other edge right after this point. Furthermore, we restrict vertices to not lie on any arcs. This, of course, excludes endpoints of the arcs and the corresponding vertices (Figure 3.3(b)).



(a) Edges touching, but not properly crossing. (b) A vertex lying in the interior of an edge.

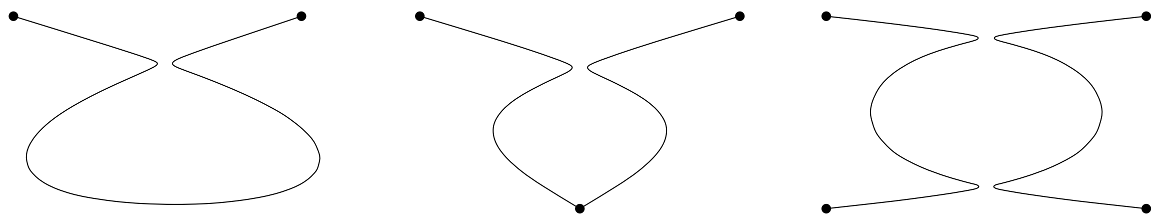
Figure 3.3: Degenerate cases forbidden in good drawings.

All these cases would invalidate many of the results presented in the following; however, apart from convenience there are problems concerning graph drawings where the said restrictions are very appropriate. Consider, for instance, the crossing number problem. In short, we are interested in a drawing of an abstract graph, that minimizes the overall number of edge crossings.

Suppose we have any of the cases shown in Figure 3.2. We can always redraw them as done in Figure 3.4.

In general, it is possible to redraw any edges that intersect multiple times in such a way that they cross at most once. For details on how such redrawings can be done, see [25].

Observation 3.1. *Any drawing of a graph G with pairs of edges mutually intersecting more than once, can be redrawn as a good drawing of the same graph with at most as many crossings as the original one.*



(a) Redrawing a self-crossing edge. (b) Redrawing incident edges. (c) Redrawing a double-crossing.

Figure 3.4: Forbidden crossings made good.

For the sake of clarity, there are a few things to note about the above definition. Throughout literature the used terminology is by far not unique. In their joint work Pach and Tóth use "topological graph" in order to denote a good drawing in the sense of Definition 3.1 (e.g. [50] and [53]). Jan Kynčl uses a similar term, but makes a difference between "topological graphs" and "simple topological graphs". Where the first does not restrict the number of crossings a pair of edges can have, the latter is precisely the same as a good drawing used here. Other authors use, for simplicity, only the word "drawing" (see e.g. [34] and [35]), but it seems appropriate to keep the terminology used, for instance, by Paul Erdős for our purpose (e.g. [20], [6], and [3]).

The above implies that, for problems such as determining the crossing number of a graph G, it suffices to only determine the crossing number of all good drawings of the graph.

However, as in the geometric case, every good drawing of the complete graph on any 4-tuple of vertices admits at most one crossing. This immediately gives that the maximum number of crossings in any $D(K_n)$ is $\binom{n}{4}$.

3.1 Comparison to Geometric Drawings

Apparently, geometric drawings of graphs are a special case of good drawings. The latter clearly give much more freedom in the way a graph can be drawn. Several well known results for straight-line drawings do either not apply for good drawings, or are incredibly hard to show.

Consider a geometric drawing $\bar{D}(G)$ of a simple graph G. It is easy to see that in case G is not a complete graph the drawing can be extended to a drawing $\bar{D}(K_n)$ by adding the missing edges. Note that there is no ambiguity in the way these edges can be drawn. The newly added edges are simply the straight-line segments connecting non-adjacent vertices in G. For good drawings; however, such an extension to a good drawing of the complete graph is not always possible.

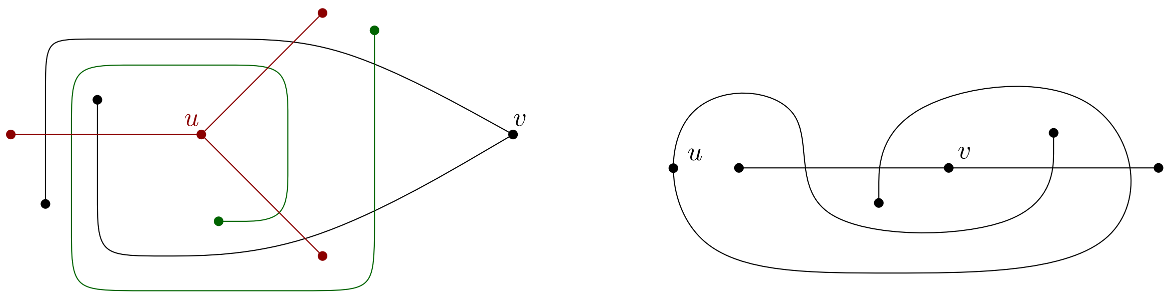


Figure 3.5: Two good drawings that are not extendible to good drawings of the complete graph. The graph on the right is taken from [43].

Figure 3.5 depicts two such examples, where the one to the right was published by Jan Kynčl [43]. In both cases it is not possible to connect the vertices u and v by an edge while preserving the properties of good drawings.

In a *maximal plane drawing* of a graph no further edge can be added without violating planarity. In the case of straight line drawings, it is well known that every maximal plane drawing has exactly $3n - 3 - h$ edges, where h is the number of vertices on the convex hull. For a triangular hull (triangular outer face), this gives $3n - 6$ edges; however, the example in Figure 3.6 shows that this is in general not the case with good drawings. For this graph on $n = 5$ vertices, one needs to remove either of the two edges drawn with a bend. But there is also the choice of either drawing the two dashed lines, or the one that is intersected by them, in order to obtain a maximal plane graph. These two graphs clearly differ in the number of edges. A *maximum plane drawing* of a graph has at least as many edges as any other maximal plane drawing of the same graph.

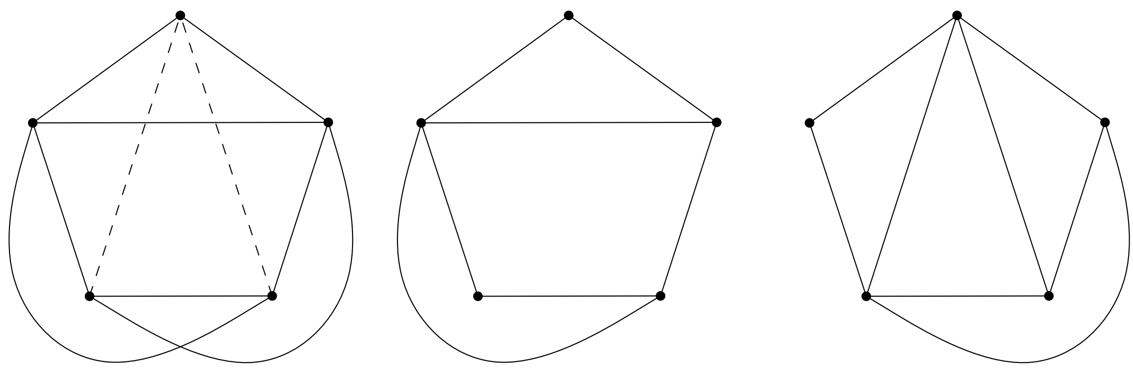
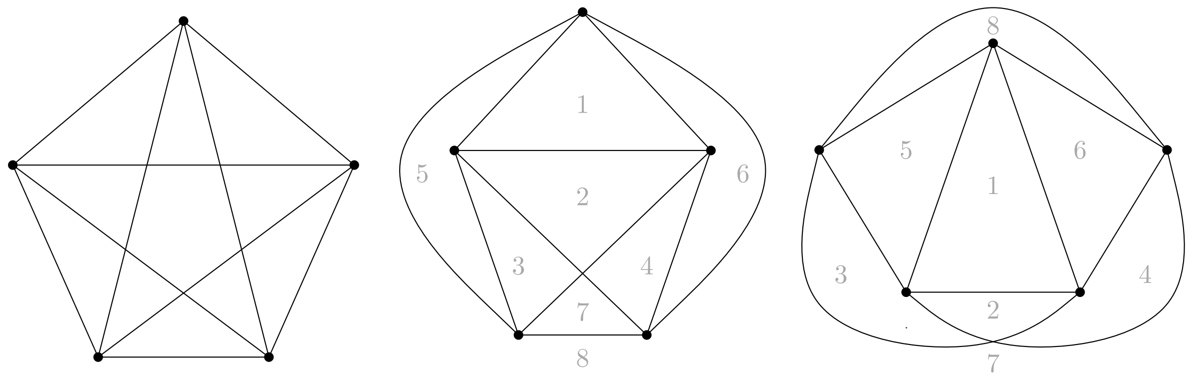


Figure 3.6: Maximal plane drawings having a different number of edges.

In Chapter 4 we will present some problems related to good drawings of complete graphs. For some of them, the solutions to the corresponding problems in the straight-line case are very easy to solve or even trivial, whereas for good drawings, they turn out to be incredibly difficult.

3.2 Isomorphism of Graph Drawings

It is natural to ask for a way to distinguish between different drawings of graphs. To make this clear, consider the examples in Figure 3.7. All three are drawings of one and the same abstract graph, namely K_5; however, informally speaking, nobody would say that these are equal. Comparing the graph on the left and the others, it is obvious that these are quite different representations. First of all the number of edges crossing between the graphs is different. Thus, the plane is subdivided into a different number of cells. But what about the one in the centre and that to the right? They both have exactly one crossing. But obviously there are differences. The number of bent edges, for instance, is not the same. Furthermore, the crossing appears between straight-line edges in the one drawing and between curvy edges in the other. However, they do have quite a lot in common. Consider the labels placed in the cells of Figures 3.7(b) and 3.7(c). They are chosen in such a way that it is easy to see that both drawings give a decomposition of the plane with the same facial structure. Each cell is bounded by the same cells in the same cyclic order. We can also map vertices onto each other such that they are incident to the same cells, also in the same cyclic order.



(a) $D(K_5)$ with five crossings. (b) $D(K_5)$ with one crossings. (c) $D(K_5)$ with one crossings.

Figure 3.7: Different drawings of K_5.

In order to be more precise and make the above differences clearer, we define two forms of isomorphism for good drawings of graphs.

Definition 3.2 ([45])**.** Two good drawings $D(G)$ and $D(H)$ are said to be *weakly isomorphic* if there exists an incidence preserving one-to-one correspondence between $V(G)$, $E(G)$ and $V(H)$, $E(H)$ such that two edges of $D(G)$ cross if and only if the corresponding two edges of $D(H)$ do.

Definition 3.3 ([45])**.** Two good drawings $D(G)$ and $D(H)$ are said to be *isomorphic* if there exists a homeomorphism of the sphere which transforms $D(G)$ into $D(H)$.

As we will see shortly, for graphs with fewer than six vertices, weak isomorphism as defined above implies the isomorphism. For $D(K_n)$ with $n \geq 6$ we will see in Section 4.1.4 examples of drawings that are weakly isomorphic, however, not isomorphic.

In Section 3.3 we will additionally give an equivalent definition for both kinds of isomorphism in terms of rotation systems.

3.3 Rotation Systems

Rotation systems as defined in Chapter 1 were used for problems concerning the crossing-free embedding of abstract graphs on arbitrary surfaces. We will give a similar definition for good drawings in the plane (or on the sphere) that will, as we see later, contain information about the structure of the drawing. Since terminology is not always consistent throughout literature, we will denote the following as a rotation system to clearly distinguish between the two different contexts.

Definition 3.4 ([43]). The *rotation* of a vertex v in a good drawing $D(G)$ is the clockwise cyclic order of the edges incident with v. The rotation $\rho(v)$ of a vertex v is represented by a cyclic sequence of the vertices adjacent to v. The *rotation system* $\mathcal{R}(D(G))$ of $D(G)$ is the set of rotations of all its vertices.

As we will see in the following, it makes sense to extend the above definition to capture the properties of crossings in good drawings.

Definition 3.5 ([43]). The *extended rotation system* $\mathcal{R}_e(D(G))$ of a good drawing of a graph is the set of all rotations of the vertices and crossings of $D(G)$. Similar as for the rotation of vertices, the rotation of a crossing represents the cyclic order of the four segments of the two edges involved in the crossing.

We should note here that every crossing can have exactly two different rotations. Furthermore, we say that two (extended) rotation systems of good drawings of the same abstract graph G are *inverse* if the rotations of each vertex (and crossing) of the two drawings are inverse cyclic permutations. This is, for instance, the case for the drawings of K_4 depicted in Figure 3.8. The rotation systems of the drawing to the left and those of the other two drawings are inverse. It is also easy to verify that all drawings are isomorphic. The same pair of edges crosses, and all drawings exhibit the same facial structure. While the drawing in the centre appears to be quite different, the picture to the right is simply the mirror image of the leftmost one. In this case isomorphism is obvious, since when seen in a mirror, the only thing changing is the cyclic direction in which the rotations are considered.

In fact, Kynčl [44] [43] showed that Definition 3.3 can also be expressed in terms of extended rotation systems.

Definition 3.6 ([43]). Two good drawings $D(G)$ and $D(H)$ of connected graphs G and H are *isomorphic* if:

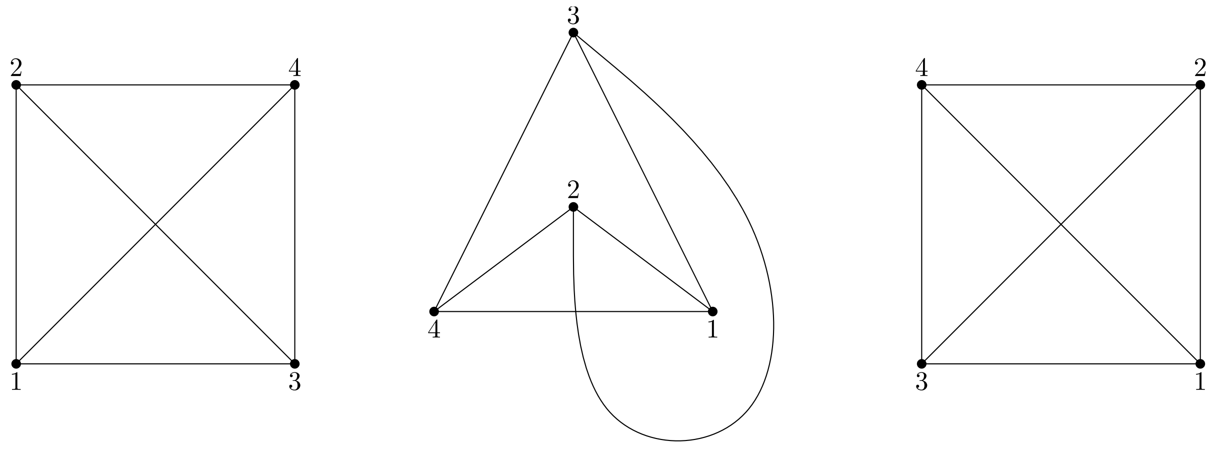


Figure 3.8: Good drawings of K_4 with inverse rotation systems.

(1) $D(G)$ and $D(H)$ are weakly isomorphic.

(2) the order of the crossings along each edge in $D(G)$ is the same as the order of the crossings along the corresponding edge in $D(H)$.

(3) the extended rotation systems $\mathcal{R}_e(D(G))$ and $\mathcal{R}_e(D(H))$ are either the same or inverse.

The above implies that for each face of $D(G)$ there is a corresponding face in $D(H)$, and the vertices and crossings appear in the same or inverse cyclic order when traversing the boundaries of the faces. As a consequence of the Jordan-Schönflies Theorem (Theorem 2.3), Definitions 3.3 and 3.6 are equivalent. It should be mentioned here that for simplicity, the above definition is restricted to drawings with only one connected component. Although Kynčl [43] gave a similar definition for good drawings with several connected components, Definition 3.6 is adequate here, since we will only consider connected graphs within this context.

3.3.1 Rotation Systems for Drawings of the Complete Graph

For good drawings of complete graphs, the rotation systems capture even more information regarding isomorphism. Pach and Tóth [54] proved that two good drawings of the complete graph on n vertices $D(K_n)$ with the same rotation systems are weakly isomorphic. Gioan [29] showed that the converse is also true, namely that weakly isomorphic graphs have either the same or inverse rotation systems. We recall the proof of the following proposition as presented in [44].

Proposition 3.2 ([44])**.**

(i) Two good drawings of the complete graph with the same rotation system are weakly isomorphic.

(ii) If two good drawings of the complete graph are weakly isomorphic, then their rotation systems are either the same or inverse.

Proof. Clearly, both parts of the proposition hold for $D(K_n)$ with $n \leq 3$. We will start by proving the first part for $n \geq 4$.

(i) The crossing properties of a good drawing of the complete graph $D(K_n)$ are fixed by the crossing properties of all drawings $D(K_4)$ of complete subgraphs on four vertices. Thus it suffices to show that whether a pair of edges of $D(K_4)$ crosses or not is determined by its rotation system $\mathcal{R}(D(K_4))$. For this purpose we consider labelled graphs and notice that there are exactly four non-isomorphic good drawings of K_4. They are depicted in the first row of Figure 3.9. Each drawing can be represented by two mutually inverse rotation systems (see the drawings in the second row). It is also easy to verify that two different rotation systems that are not mutually inverse are rotation systems for two non-isomorphic graphs.

This proves part one of Proposition 3.2 and the second part for $n = 4$, since all four non-isomorphic $D(K_4)$ are shown and their rotation systems are fixed, except for inversion when for instance considering their mirror images.

(ii) We show part two for $n \geq 5$. The idea is to prove the result for $n = 5$ and later extend it to good drawings of complete graphs with an arbitrary number of vertices.

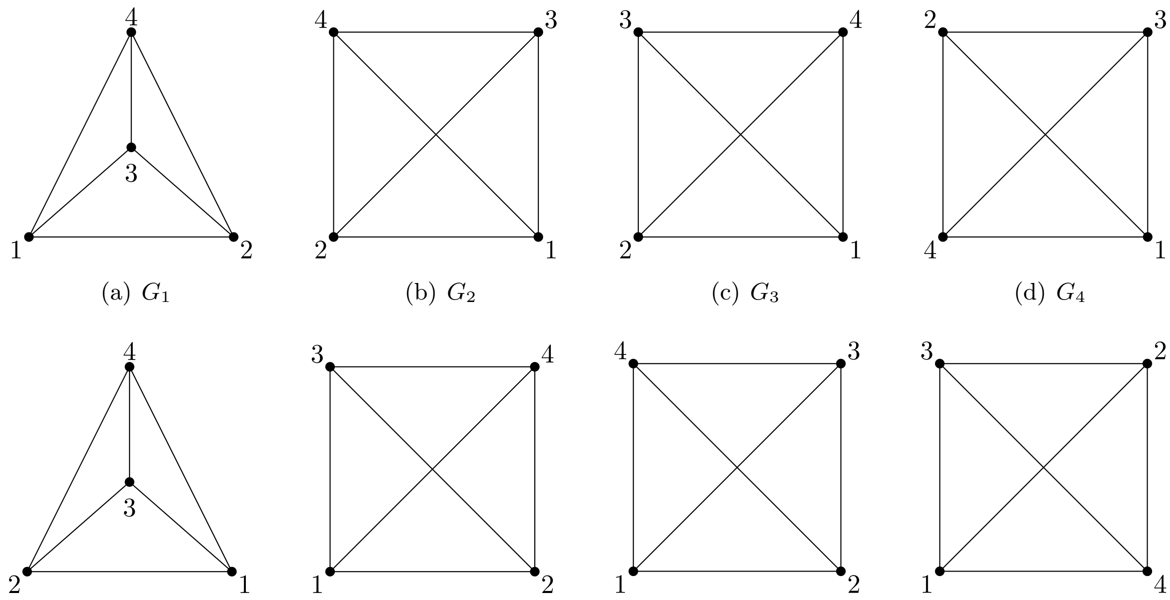
In Figure 6.1 on page 57 all five non-isomorphic good drawings of K_5 can be seen (see also [44]). All five drawings have different pairs of edges crossing, this again implies that for $n = 5$ all weakly isomorphic drawings are also isomorphic.

Consider now a good drawing $A = D(K_n)$ with vertices $\{1, 2, ..., n\}$ and $n \geq 6$. Suppose we have two five vertex subgraphs of A, say B and C with vertex sets $\{1, 2, 3, 4, 5\}$ and $\{1, 2, 3, 4, 6\}$, respectively. Then the rotation system $R(B)$ uniquely determines the rotation system $R(C)$. Without loss of generality, let the rotation of vertex 1 in B be $\rho(1) = (2, 3, 4, 5)$. This implies that the rotation of 1 in the four vertex subgraph that both B and C have in common (the subgraph on vertices $\{1, 2, 3, 4\}$) is $(2, 3, 4)$. This is, however, only the case in one of the two possible, mutually inverse rotation systems of C. Thus the rotation of vertex 1 in C is fixed, as well as the whole rotation system of C.

It is easy to see that we can repeatedly use the above argument and obtain that the rotation system of every complete five vertex subgraph of A is fixed. So finally, we need to argue that this uniquely determines the rotations of every vertex in A. However, the cyclic order of all three element subsets in the rotation of a given vertex is fixed, which implies that the cyclic order of the entire rotation is unique.

We have shown that any good drawing of K_n can only have two mutually inverse rotation systems, and so the rotation systems of any two weakly-isomorphic drawings are either the same or inverse. □

The first part of the proof contains a rather obvious property that is still worth to be mentioned. For each vertex of K_4, there are two possible rotations, so in total there are sixteen possible rotation systems. However, each of the four non-isomorphic $D(K_4)$ gives rise to only two rotation systems (mutually inverse ones). So eight of the possible rotation systems are in fact not rotation systems of good drawings of K_4. Moreover, if a rotation system contains such a non-realizable rotation system as the rotation system of a four-vertex subset, then the entire



(a) G_1 (b) G_2 (c) G_3 (d) G_4

(e) $\tilde{G}_1$ (f) $\tilde{G}_2$ (g) $\tilde{G}_3$ (h) $\tilde{G}_4$

Figure 3.9: The eight valid rotation systems of K_4.

rotation system is not realizable.

Actually, it is possible to characterize the rotation systems that are realizable as good drawings of the complete graph on four vertices by a simple parity condition. The rotation at each vertex i is represented as a triple $\rho(i) = (j, k, l)$ with $i \in \{1, 2, 3, 4\}$ and $j, k, l \in \{1, 2, 3, 4\} \setminus i$ such that $j = \min\{j, k, l\}$. The rotation $\rho(i)$ is called *negative* if $k > l$, and *positive* otherwise. By examining all eight valid rotation systems in Table 3.1, one can verify that a rotation system is a rotation system of a good drawing of K_4 if and only if the number of negative rotations is even.

$D(K_4)$	$R(D(K_4))$	$\mathcal{R}(D(K_4))$
G_1	((2,4,3),(1,3,4),(1,4,2),(1,2,3))	((2,3,4),(1,4,3),(1,2,4),(1,3,2))
G_2	((2,4,3),(1,4,3),(1,2,4),(1,2,3))	((2,3,4),(1,3,4),(1,2,4),(1,2,3))
G_3	((2,3,4),(1,3,4),(1,2,4),(1,2,3))	((2,3,4),(1,3,4),(1,2,4),(1,2,3))
G_4	((2,3,4),(1,4,3),(1,4,2),(1,2,3))	((2,3,4),(1,3,4),(1,2,4),(1,2,3))
$\tilde{G}_1$	((2,3,4),(1,4,3),(1,2,4),(1,3,2))	((2,3,4),(1,4,3),(1,2,4),(1,3,2))
$\tilde{G}_2$	((2,3,4),(1,3,4),(1,4,2),(1,3,2))	((2,3,4),(1,3,4),(1,2,4),(1,2,3))
$\tilde{G}_3$	((2,4,3),(1,4,3),(1,4,2),(1,3,2))	((2,3,4),(1,3,4),(1,2,4),(1,2,3))
$\tilde{G}_4$	((2,4,3),(1,3,4),(1,2,4),(1,3,2))	((2,3,4),(1,3,4),(1,2,4),(1,2,3))

Table 3.1: Valid rotation systems for $D(K_4)$.

We have recapitulated that for good drawings of the complete graph, the weak isomorphism classes can be determined solely from the rotation system. In the case of isomorphism for complete graph drawings, we furthermore note that the rotations of the crossings are also determined by the rotation system. This becomes clear when again having a look at the different rotation systems in Figure 3.9. Each 4-tuple of vertices produces at most one crossing, and for each rotation system there is only one possible rotation for that crossing. Hence, we do not require the extended rotation system, but only the order of the crossings along the edges. We will make use of the gathered results in the enumeration algorithm presented in Section 4.1.

4

Drawings of the Complete Graph and Related Problems

This chapter deals with good drawings of the complete graph K_n on n vertices. In Section 3.3.1 we have already pointed out the role of rotation systems for such drawings. We will use this knowledge to give algorithms to enumerate all different drawings $D(K_n)$ under the notions of weak isomorphism and isomorphism. These algorithms are implemented and used to build a database containing all the drawings, which can subsequently be used to verify statements for small n and gain more insight in problems concerning good drawings of the complete graph. Several such problems are presented in the remainder of the chapter.

4.1 Enumerating Drawings of the Complete Graph

In order to enumerate good drawings of the complete graph for small n, we will make use of the properties of rotation systems as presented above. For enumerating weakly isomorphic drawings, it is sufficient to determine all different realizable rotation systems of K_n. To obtain all non-isomorphic drawings, however, the different realizations have to be taken into account. We define a fingerprint that can be used to distinguish between different drawings in both manners and give an enumeration algorithm.

4.1.1 Fingerprint for Graph Drawings

For the purpose of determining whether two good drawings of K_n are (weakly) isomorphic, we need to define a standard form, a so called *(weak) fingerprint*. Two drawings must have the same standard form if and only if they are (weakly) isomorphic. We will present a fingerprint suitable for both kinds of isomorphism and explain how these can be determined algorithmically in the context of our enumeration algorithm.

In case of weak isomorphism, we have already established that two drawings of the complete graph K_n are weakly isomorphic if and only if their rotation systems are either the same or inverse. So for a *weak fingerprint* we can simply use a canonical form of the rotation system. A convenient choice is the rotation system that gives the lexicographically smallest integer sequence among all possible rotation systems of the drawing. In general the weak fingerprint is composed of the individual rotations of vertices 1 to n, while all rotations are represented by the cyclic permutations read in the same cyclic direction (clockwise or counter clockwise) starting at the vertex with lowest index. Furthermore, this implies that the integer sequence obtained from the rotation of vertex 1 is always $(2, ..., n)$. We can therefore omit this part and our weak fingerprint has the following structure:

$$FP_{RS} = ((\rho_{min}(2)), (\rho_{min}(3)), ..., (\rho_{min}(n))) \tag{4.1}$$

In the above equation $\rho_{min}(i)$ denotes the minimal sequence obtained from the rotation of vertex i under a labelling of the vertices that minimizes the entire fingerprint including the rotation of vertex 1. In our algorithm, however, we will use a reverse approach. Instead of generating all possible drawings of K_n and filtering duplicates in terms of weak isomorphism, we will generate all possible sequences that can represent a weak fingerprint, and afterwards check for realizability as a good drawing of the complete graph $D(K_n)$. The realizable weak fingerprints represent all different classes of weakly isomorphic drawings of K_n.

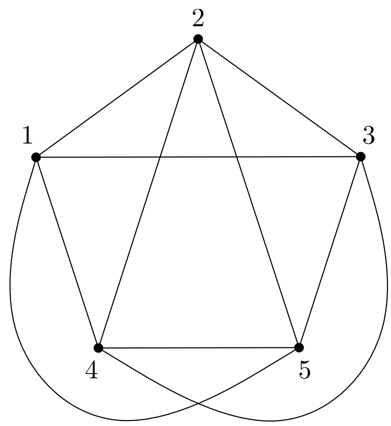


Figure 4.1: Labelling of $D(K_5)$ with three crossings giving minimal fingerprint.

As an example consider Figure 4.1. The labellings are chosen such that the rotation system and hence the weak fingerprint is lexicographically minimal. So in this case we can read the weak fingerprint directly off the rotations as depicted. We have that $\rho(1) = (2, 3, 4, 5)$, $\rho(2) = (1, 3, 5, 4)$, $\rho(3) = (1, 2, 4, 5)$, $\rho(4) = (1, 2, 5, 3)$ and $\rho(5) = (1, 4, 2, 3)$. Thus we get for the weak fingerprint $FP_{RS} = 1354124512531423$.

For the general fingerprint used to distinguish between *non-isomorphic* drawings (so drawings that are neither isomorphic nor weakly isomorphic) we will need to include some more information of specific drawings. Since isomorphism also implies weak isomorphism, we can simply use the weak fingerprint as the first part of the general fingerprint. Furthermore, we need to include the information about the order of crossings along the individual edges. We consider all edges (i, j) with $i, j \in V(G)$ and $i < j$ according to the labelling induced by the weak fingerprint. Order them lexicographically, and since the number of crossings on each edges is determined by the rotation system of $D(K_n)$, it suffices to list the indices of edges crossing (i, j) in the order encountered from i to j:

$$FP = (FP_{RS}, c(e_1), c(e_2), c(e_3), ...) \tag{4.2}$$

By $c(e_i)$ we denote the indices of edges crossing the edge e_i in the order obtained by going from the lower index vertex to the higher index vertex. We should note some subtleties concerning the fingerprint used here. The weak fingerprint can, in general, be obtained from several labellings of the graph. These *symmetric labellings* could, however, give different sequences for the edge crossings $c(e_1), c(e_2), c(e_3), ...$ To obtain a unique fingerprint in the case of isomorphism, we need to check which of those symmetric labellings gives the lexicographically smallest such sequence of edge crossings.

We will determine all different realizations of a weak fingerprint by a simple backtracking algorithm. For each good drawing obtained by the algorithm, we test the predetermined symmetric labellings and store the lexicographically smallest sequence $c(e_1), c(e_2), c(e_3), ..., c(e_{|E(K_n)|})$ as a fingerprint. The different such sequences obtained after checking all possible realizations represents the different isomorphism classes within a given weak isomorphism class.

Again, take a look at Figure 4.1. The labels as depicted do not only give the minimal rotation system, but also minimize the whole fingerprint including the crossings. Ordering edges lexicographically gives us the following indices for the edges: $e_0 = (1, 2)$, $e_1 = (1, 3)$, $e_2 = (1, 4)$, $e_3 = (1, 5)$, $e_4 = (2, 3)$, $e_5 = (2, 4)$, $e_6 = (2, 5)$, $e_7 = (3, 4)$, $e_8 = (3, 5)$ and $e_9 = (4, 5)$. This order gives us the following crossing information. $c(e_1) = (5, 6)$, $c(e_3) = (7)$, $c(e_5) = (1)$, $c(e_6) = (1)$ and $c(e_7) = (3)$. We finally get for the entire fingerprint $FP = 1354124512531423\underline{567113}$, where the underlined part represents the information about the crossings.

Note that we can simply omit those edges without crossings, since whether an edge is crossed and especially how many times can be determined from the preceding rotation system part of the fingerprint (the weak fingerprint).

4.1.2 Extending Rotation Systems

Now that we are able to obtain unique representations of graph drawings under both weak isomorphism and isomorphism, we present an algorithm to obtain all different (weak) isomorphism classes of $D(K_n)$ from all (weak) isomorphism classes of $D(K_{n-1})$.

The algorithm takes a realizable weak fingerprint of a good drawing of the complete graph on $n-1$ vertices as input. It is easy to reconstruct the corresponding rotation system. We extend the rotation system by an n^{th} vertex such that the rotation $\rho(1) = (1, 2, ..., n)$. In the remaining rotations, we add the new vertex in any possible way. Those rotation systems we obtain are certainly not guaranteed to be realizable as a good drawing of K_n. While realizability itself is tested separately, we can pre-check realizability by determining whether the new rotation system contains any non-realizable 4-tuple. Of course, the absence of non-realizable 4-tuples does not imply realizability of the corresponding rotation system, since there are rotation systems that do not contain such 4-tuples, but are still not realizable as a drawing $D(K_n)$. However, in case a non-realizable subset is contained, the entire rotation system can definitely not be realized. This allows us to filter out lots of cases and avoid testing these by the quite costly backtracking method of checking realizability. If this first test is passed, we then determine the weak fingerprint by permuting labellings accordingly and store the fingerprint in a database, if it was not yet encountered before by a different extension.

After the Algorithm was ran on every realizable rotation system of the complete graph on $n-1$ vertices, we compiled a database of possibly realizable weak fingerprints for n vertices. To filter out non-realizable ones, we use the procedure described in the following.

4.1.3 Realizability of Rotation Systems

We will use a simple backtracking algorithm for generating all non-isomorphic good drawings $D(K_n)$ with the same underlying rotation system $\mathcal{R}(D(K_n))$. For simply testing realizability, we can use the exact same algorithm and terminate in case the first valid realization is found.

For representing a drawing in the algorithm, we basically use the *half-edge data structure*, a widely used data structure for planar embeddings of graphs introduced by Weiler [69] in 1985 (see also [39] for details). Here the complete graph drawing is represented by directed twin segments that are linked to their successors, predecessors, and twin segments, such that following segments along their successors or predecessors will trace out a cell of the drawing in a certain direction. Each of the two twin segments is part of one of the two cells bounded by the corresponding part of the edge bounds. Segments point from a vertex or crossing in the drawing to another vertex or crossing. See Figure 4.2 for a detail of a graph embedding. The solid black circle depicts an actual vertex of the drawing, while the white circles are the crossings. If we follow the segment e in either direction, we trace out the boundary of the shaded cell.

First we need to observe that, contrary to geometric drawings, for good drawings the actual position of the vertices is of no importance. We can therefore use the following idea to guarantee that our algorithm produces realizations of the given rotation system, if realizable.

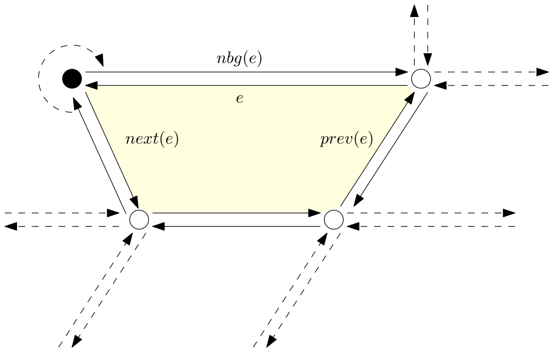
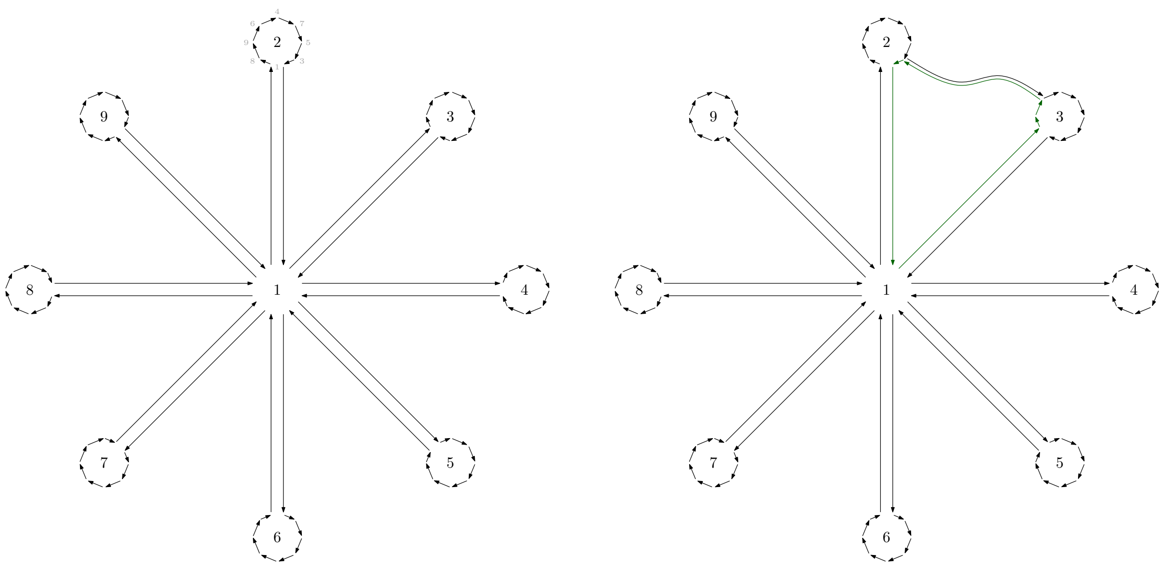


Figure 4.2: Half-edge data structure.

We start with a star as depicted in Figure4.3. The dummy segments around the outer vertices are introduced in order to force the drawing to obey the given rotation system. The numbers in grey around the topmost vertex in Figure 4.3(a) represent the rotation of vertex 2. By setting the rotations of vertices 2 to n accordingly, we can guarantee our final drawing (if realizable) to have the desired rotation system. The dummy segments differ from normal segments only in their property that they are never intersected during the algorithm and therefore do not need twin segments. In other words, these segments act as attachment points for segments of new edges that are added. In Figure 4.3(b) it is shown how a new edge is attached to the dummy segments. The edges that trace out the newly created face are highlighted. These are, in fact, the only modification we make to the standard half-edge data structure.



(a) Data structure as initialized.

(b) A first edge added.

Figure 4.3: Initial star of the algorithm as represented by in the data structure.

The function *init()* in Algorithm 1 takes the rotation system as input and constructs that star accordingly. This includes three arrays, namely segments[], start[][], and blocked[][][][].

The container segments[] is realized as a stack and contains the objects representing the segments with the information as required by the half-edge data structure. This is a meaningful approach since it guarantees that newly added segments lie on top of the stack after being added in each step of the recursion.

By using the two dimensional array start[][], we can conveniently access those segments that define the starting point of the sequence of segments going from vertex i to vertex j. This means that the entry start[i][j] contains the index of the dummy segment in segments[] where the edge from i to j is attached to.

Finally, blocked[a][b][c][d] contains the information whether the edge from a to b can intersect the edge from c to d. Since we want to realize the rotation system as a good drawing, we set the entries of edges that share a vertex to *true* in *init()*. This guarantees our algorithm to not intersect such edges again. We also obtain whether two vertex disjoint edges must not intersect from the rotation system and determine this for all pairs of edges beforehand. During the recursive steps, we keep this four dimensional matrix consistent by setting the corresponding entries *true* after intersecting segments and change them to *false* again, when the recursion returns and the intersection is undone (see Algorithm 2).

Algorithm 1 realizeRS($\mathcal{R}$)

1: init($\mathcal{R}$)
2: nextStep(start[1][2], start[2][1], 1, 2)

We then proceed by adding connections among vertices recursively until we (maybe) end up with a drawing of the complete graph. The main recursive routine is listed in Algorithm 2. It is called after *init()* for the connection between vertices 1 and 2 (all edges between vertex 0 and all the others were already created before).

The function is called with parameters s and t which are the indices of the starting segments of the source vertex and the target vertex, respectively. Furthermore, a and b denote the indices of the vertices that are about to be joined by an edge.

The outer loop (starting at line 2) traces the cell that the starting segment is adjacent. We do so until we return to where we started. Within the loop we check if the target segment was reached and add the required segment. In this case we either find a realization of the given rotation system, or need to further step into the recursion to add missing edges (lines 3 to 12).

Additionally, we check for all segments encountered in the walk around the boundary of the cell, if they need to be intersected (lines 13 to 19). Whenever we encounter a segment that is not the target segment, it is checked to see if it is blocked. If so, we simply skip it. Otherwise, it is intersected and the required entries in blocked[][][][] are set to *true*. Afterwards we go deeper into the recursion (line 16) and after returning, we undo the previous intersection and allow the respective edge to be crossed again.

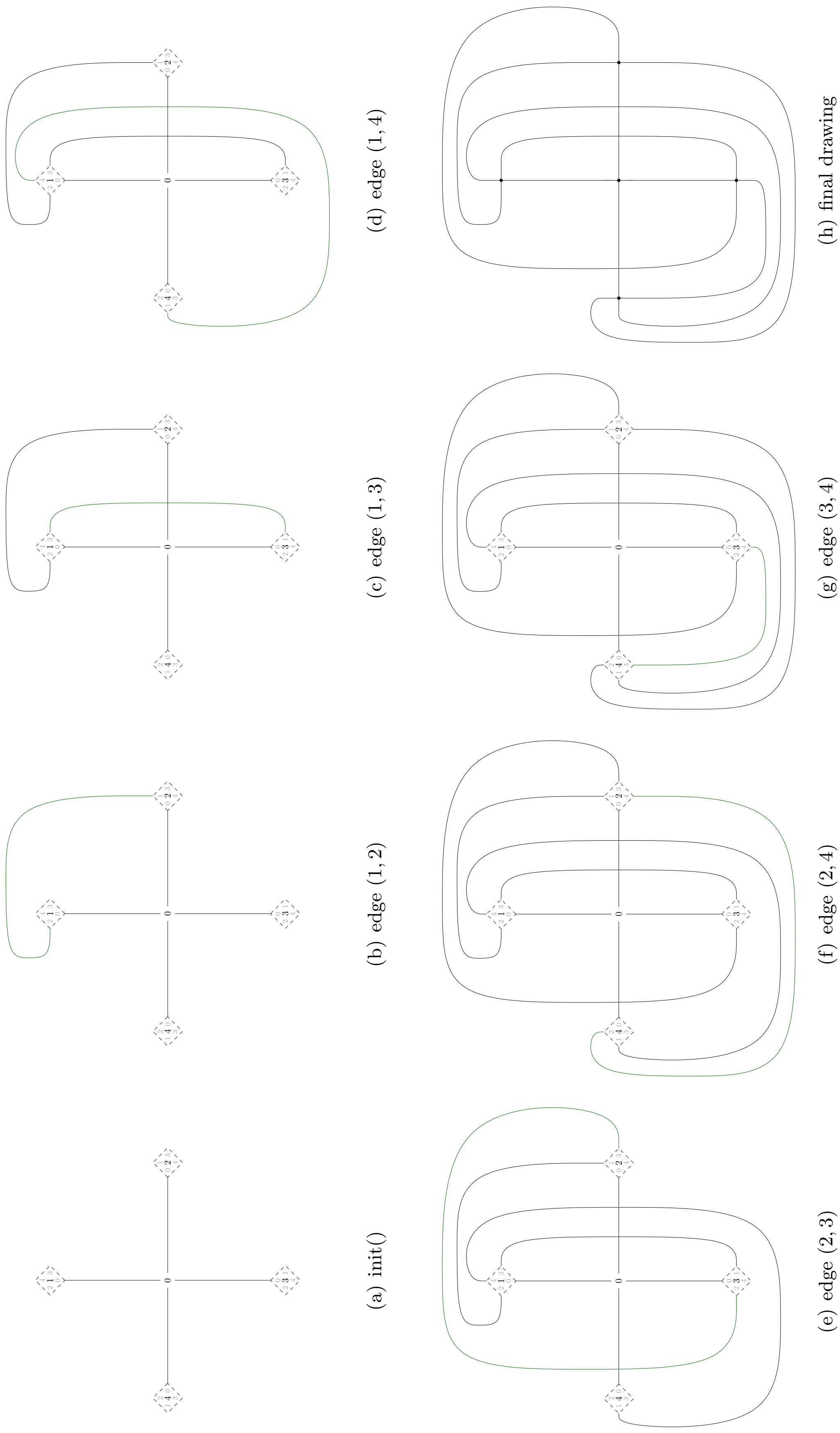
It is elementary to convince oneself that the described procedure produces only realizations as valid good drawings and also all possible orders of crossings. Of course, these different crossing orders need not imply different isomorphism classes. Depending on whether we only use the algorithm for checking realizability, or enumerating all non-isomorphic good drawings,

Algorithm 2 nextStep(s, t, a, b)

```
seg ← segments[s].nxt
while seg ≠ s do
    if seg = t then
        addSegment(s,t,a,b)
        if b < n − 1 then
            nextStep(start[a][b + 1], start[b + 1][a], a, b + 1)
        else if a < n − 2 then
            nextStep(start[a + 1][a + 2], start[a + 2][a + 1], a + 1, a + 2)
        else
            foundRealization()
        end if
    end if
    if blocked[a][b][segments[seg].a][segments[seg].b] = FALSE then
        blocked[a][b][segments[seg].a][segments[seg].b] ← TRUE
        intersect(–)
        nextStep(segments[seg].nbg, t, a, b)
        undoIntersect(–)
        blocked[a][b][segments[seg].a][segments[seg].b] ← FALSE
    end if
    seg ← segments[seg].nxt
end while
```

we either stop in *foundRealization()* with the result that the input rotation system is realizable, or we calculate the fingerprint for the good drawing obtained at that point (as desribed in Section 4.1.1). We leave out details here, since implementation is not very sophisticated and it would only bloat this thesis with unnecessary technicalities.

Figure 4.4 shows an example of how the algorithm finds a certain realization of a rotation system of $\mathcal{R}(D(K_5))$. Edges added in the individual steps shown in Figures 4.4(b) to 4.4(g) are highlighted in green. The drawing obtained and depicted in Figure 4.4(h) is isomorphic to the one used to illustrate the fingerprint in Figure 4.1.



(a) init()

(b) edge $(1, 2)$

(c) edge $(1, 3)$

(d) edge $(1, 4)$

(e) edge $(2, 3)$

(f) edge $(2, 4)$

(g) edge $(3, 4)$

(h) final drawing

Figure 4.4: Example of $D(K_5)$ as produced by Algorithm 1.

4.1.4 A Database of Rotation Systems and Drawings

The algorithm was used to set up a database of all realizable rotation systems of K_n and all non-isomorphic $D(K_n)$ for $3 \leq n \leq 8$. The numbers obtained are listed in Table 4.1. It is remarkable how fast these numbers grow.

n	# of realizable RS	# of non-isomorphic $D(K_n)$
3	1	1
4	2	2
5	5	5
6	102	121
7	11556	46999
8	5370725	502090394

Table 4.1: The number of realizable RS and non-isomorphic drawings of K_n.

As it was mentioned in the proof of Proposition 3.2, the weak isomorphism implies the isomorphism of two good drawings of K_n for $3 \leq n \leq 5$. This is, in contrast, not the case any more for $n \geq 6$. In fact, there are several rotation systems that admit two non-isomorphic drawings, and even one with three. The later has the following rotations: $\rho(1) = (2, 3, 4, 5, 6)$, $\rho(2) = (1, 3, 4, 5, 6)$, $\rho(3) = (1, 3, 4, 6, 5)$, $\rho(4) = (1, 5, 2, 6, 3)$, $\rho(5) = (1, 4, 2, 6, 3)$ and $\rho(6) = (1, 2, 4, 3, 5)$. In Table 4.2 the orders of the crossings along the edges are listed for the three drawings D_1, D_2, and D_3. Clearly, the edges involved in the crossings along a certain edge do not differ among these realizations, since this is already determined by the rotation system. However, there are different crossing orders along the edges. The underlined entries in the table indicate changes in these crossing orders with respect to D_1.

Since the above rotation system is a minimal example with multiple non-isomorphic drawings, we should take a look at how these differences in the crossing orders are produced. Consider a triangular face in a good drawing of K_n that is bounded by the three edges e_1, e_2, and e_3. Without loss of generality we move the edge e_1 across the intersection of the edges e_2 and e_3. Obviously, such a *triangle switch* can always be done without affecting the remainder of the drawing and in particular the vertex rotations. See [29] and Figure 4.5 for an illustration. This, however, means that for any of the three edges involved the order of the crossings with the other two edges exchanges. Note that this can produce a different drawing under the notion of isomorphism, but it does not necessarily need to be the case. We could, for instance, obtain the exact same crossing orders by relabelling vertices accordingly.

The three drawings D_1 to D_3 can be seen in Figures 4.6(a) to 4.6(c). Here triangle switches indeed produce non-isomorphic drawings. The changes in drawings D_2 and D_3, with respect to the first drawing D_1, are indicated by the highlighted triangular faces. If we compare these to the crossing orders in Table 4.2, we can see that each triangle switch induces changes on three edges as already mentioned before.

The results in Table 4.1 indicate that this maximum number of non-isomorphic realizations

edge	crossings D_1	crossings D_2	crossings D_3
12			
13	26 25 24 56	26 25 $\underline{56\ 24}$	26 25 $\underline{56\ 24}$
14	26	26	26
15	26 34 46 24	26 34 46 24	$\underline{34\ 26}$ 46 24
16	34	34	34
23	56	56	56
24	13 56 35 15	$\underline{56\ 13}$ 35 15	$\underline{56\ 13}$ 35 15
25	13	13	13
26	13 14 15 34	13 14 15 34	13 14 $\underline{34\ 15}$
34	56 16 26 15	56 16 26 15	56 16 $\underline{15\ 26}$
35	24	24	24
36			
45			
46	15	15	15
56	24 13 23 34	$\underline{13\ 24}$ 23 34	$\underline{13\ 24}$ 23 34

Table 4.2: Orders of crossings in the three non-isomorphic drawings.

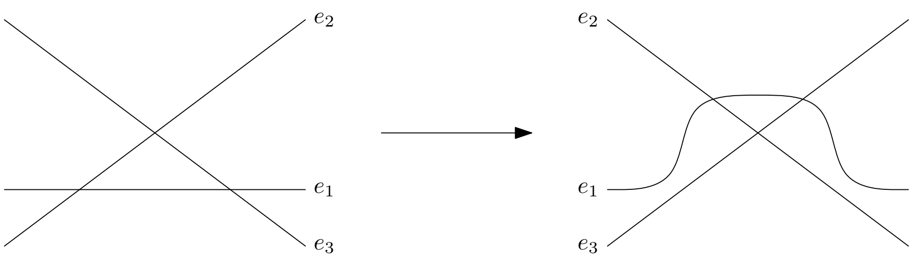
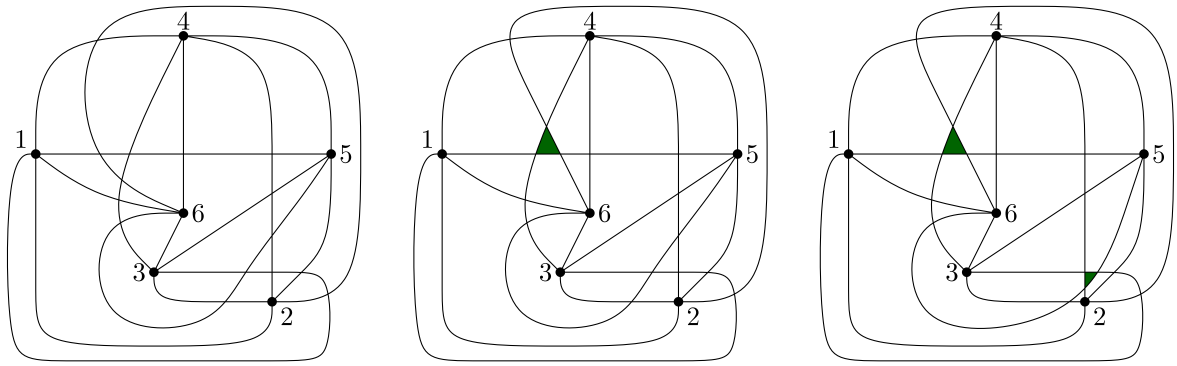


Figure 4.5: Local triangle switch in a good drawing as in [29].

per rotation system grows dramatically as n gets larger. For $3 \leq n \leq 8$ we list the exact numbers in Table 4.3.

n	max. # of non-isomorphic realizations per RS
3	1
4	1
5	1
6	3
7	57
8	46571

Table 4.3: The maximum numbers of non-isomorphic $D(K_n)$ per rotation system.



(a) First drawing. (b) Second drawing. (c) Thrid drawing.

Figure 4.6: Unique rotation system with three non-isomorphic $D(K_6)$.

4.1.5 Visualization

The database contains all information required to reconstruct a drawing uniquely. Especially for larger examples, it is very hard to construct the full drawing by hand, given the information about vertex rotations and crossings. In order to support visualization, a tool in form of an extension to the graphics editor IPE[1] was created. It takes a single entry from the drawings database as input, determines the *planarized graph* $\tilde{G}$ (i.e., the graph where all crossings are replaced by pseudo vertices resulting in a planar graph), and calculates a straight-line embedding of $\tilde{G}$. Removing the pseudo vertices in the embedding gives a drawing with edges represented by a sequence of straigt-line segments, comprising the desired drawing.

By a well-known result of Whitney [70], the embedding is guaranteed to be unique in case the planarized graph is 3-connected. Although the planarized graph of a $D(K_n)$ strongly appears to be 3-connected, we could not give a general proof for that. So, in addition, one needs to check in the drawing produced by the IPELET whether the required crossing orders along the edges are fulfilled.

As Figures 4.7 and 4.8 indicate, this simple method can only be used to give an idea of how the drawing could look like. The picture to the left in Figure 4.7 was directly created with the use of the IPE extension. One can see immediately that this drawing is not at all appealing compared to the drawing to the right, which is a way nicer isomorphic drawing. The example in Figure 4.8 shows that things clearly get even worse for larger graphs. Although not being drawn too small, it already is incredibly hard to follow the edges between the vertices in the picture.

[1] http://ipe7.sourceforge.net

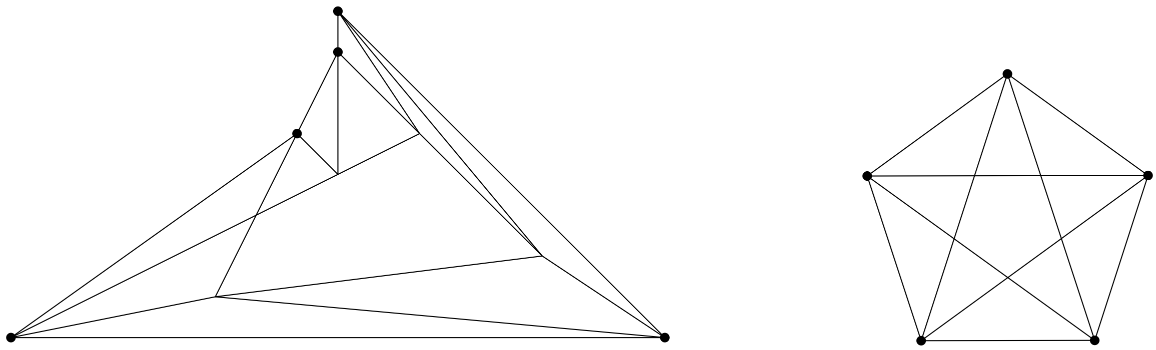


Figure 4.7: Isomorphic drawings of K_5 created with IPELET and by hand.

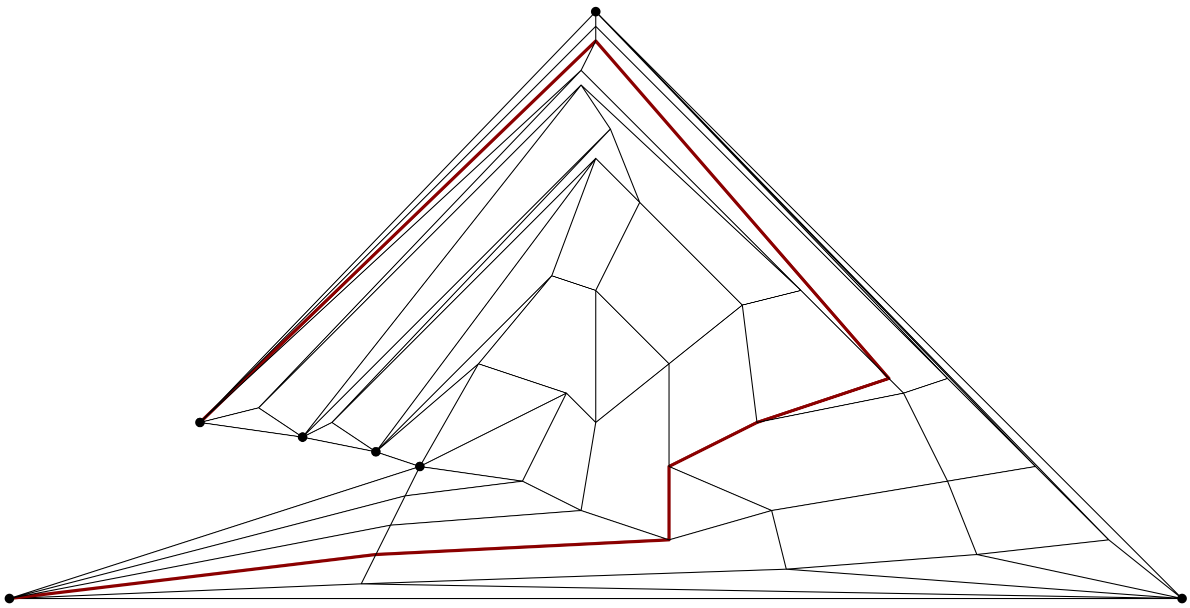


Figure 4.8: Drawing of K_7 created with IPELET.

4.1.6 Related Work

For n up to seven, Rafla [56] used an algorithm to determine the weak isomorphism classes of $D(K_n)$. Table 4.1 confirms his results. He also mentioned that, for $n = 6$, previously Uytterhoeven had found 100 different drawings and Beckelin 123, while only 96 of them were not weakly isomorphic. Both determined the drawings by hand which makes it extremely hard to not overlook or multiply count some of the drawings.

In 1990 Gronau and Harboth [30] have found all 121 non-isomorphic drawings $D(K_6)$. These results could be confirmed by Volker Leck [46] who enumerated all non-isomorphic drawings of connected graphs on 6 vertices, among them K_6.

We should note that our method of checking realizability of rotation systems is very simple, but not efficient. Clearly, we use a non-polynomial time algorithm. Kynčl [45] showed that realizability of complete abstract topological graphs (that is a complete graph together with which pairs of edges cross) as good drawings is in P. Since the pairs of edges that cross can easily be determined from the rotation system, realizability of rotation systems as good drawings of K_n can be verified in polynomial time. In fact, he initially determines a rotation system for the abstract graph in the algorithm he presents.

Furthermore, the algorithm only gives a realization if possible, so it is also not suitable for determining all non-isomorphic drawings of a given rotation system.

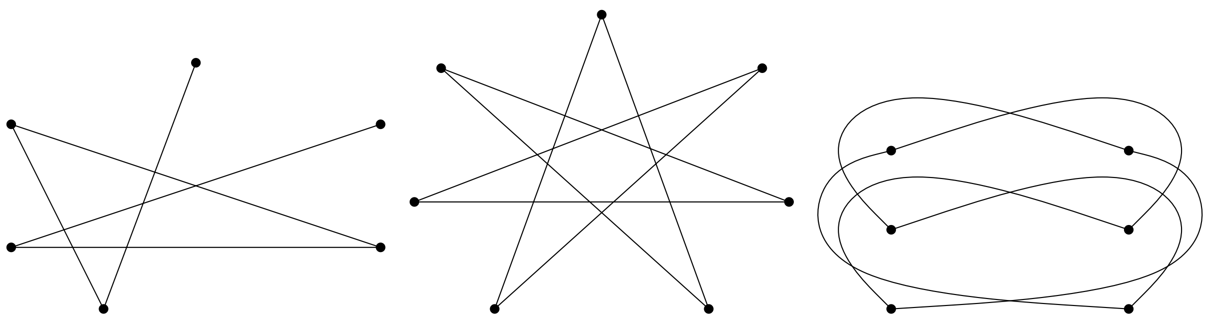
5

Conway's Thrackle Conjecture

This chapter deals with a certain class of good drawings which was brought to attention by John H. Conway in the 1960s, namely thrackles.

Definition 5.1. A *thrackle* is a good drawing of a graph where each pair of edges either share an endpoint or cross exactly once. An abstract graph G is said to be *thrackleable* if it can be drawn as a thrackle.

As an example consider the graph P_6. Since this graph is thrackleable, it can be drawn appropriately. See Figure 5.1(a) for a drawing of P_6 as a thrackle.



(a) Thrackle embedding of P_6. (b) The n-gram on 7 vertices. (c) Thrackle embedding of C_6.

Figure 5.1: Thrackle embeddings of P_6, C_7 and C_6.

Note that the above example is in addition drawn with edges represented as straight-line segments. Yet, not every abstract graph can be thrackled in this manner. Take for instance C_n the cycle on n vertices. As already mentioned by Woodall [71], there is a thrackle embedding of

C_n only using straight-line segments for n being odd. The example he uses is the *n-gram* which can be seen in Figure 5.1(b). It is easy to verify that this construction can always be extended by two additional vertices.

In contrast it is never possible to obtain a straight-line thrackle embedding of even cycles. As a brief argument, consider that for any edge of the cycle, all remaining vertices need to alternatingly lie in the two half-planes defined by the line passing through the vertices of the chosen edge. Anyhow, as Figure 5.1(c) depicts, it is possible to draw C_6 as a thrackle using curved lines to represent edges.

In fact, there is such a thrackle embedding of any even cycle except for C_4. To see this we use the following trick mentioned on Stephan Wehner's web page dedicated to thrackles [1]. Take any P_3 in the cycle and replace it by a P_5 as shown in Figure 5.2. By drawing the new edges sufficiently close to the one that is removed, one can guarantee to preserve the properties of thrackles during such a modification.

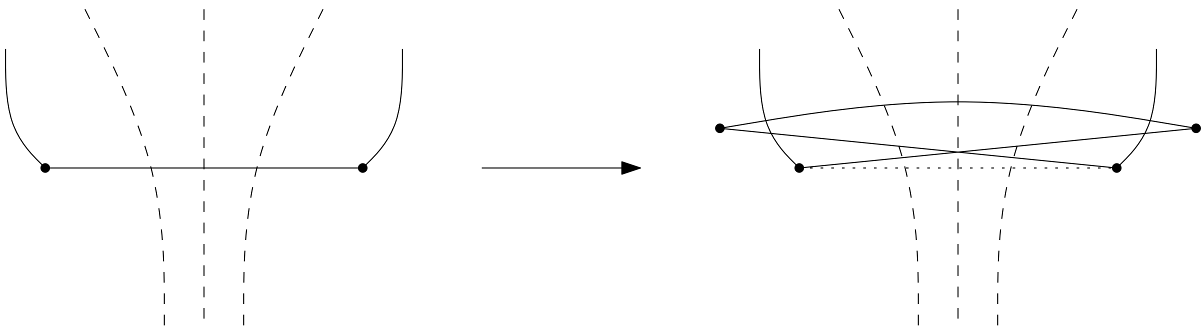


Figure 5.2: Extending a thrackle by two vertices.

Definition 5.2. A *cycle thrackle* on n vertices is a thrackle drawing of the cycle graph C_n.

Definition 5.3. A *path thrackle* on n vertices is a thrackle drawing of the path P_n.

Together with the existence of a thrackle embedding of C_6 and the trivial fact that C_3 is thrackleable, we can state the following observation concerning the two classes of thrackles defined above.

Observation 5.1. *Every cycle graph C_n for $n \neq 4$ is thrackleable. As a consequence there exists a thrackle embedding for every path P_n.*

The second part of Observation 5.1 follows immediately by deleting an arbitrary edge of the cycle. For P_4 it is straightforward to verify that it is thrackleable too. In addition this tells us that for any n there exists a thrackle having as many edges as vertices.

Apart from the cycles and a not hard to find substitute for C_4, there is an even easier construction to show this, which in addition verifies the second part of Observation 5.1 for linear thrackles. Consider the triangle C_3, which is clearly a thrackle with the same number of edges

[1] www.thrackle.org (accessed September 16, 2014)

as vertices. Now successively add edges incident to the same vertex of the triangle and intersecting the remaining edge. Clearly, we obtain a thrackle of the desired kind and this can be done by only using straight-line edges (Figure 5.3). On the previously mentioned web page this construction is called the n-ray.

We can therefore easily construct thrackles with $|V| = |E|$ and there are many other such examples aside from the aforementioned kind. However, after trying for a while using pen and paper, one will find it intriguingly hard to add one more edge to such a construction. John H. Conway first believed that this is not possible at all and we state his conjecture in the words used by Douglas R. Woodall [71] in one of the first publications on this topic.

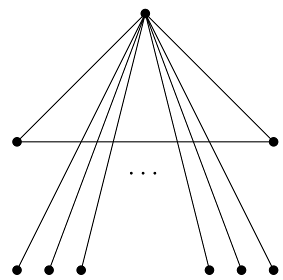


Figure 5.3: The n-ray.

Conjecture 5.2 (Conway's thrackle conjecture)**.** *A thrackleable graph with n vertices cannot have more than n edges.*

Despite many efforts the thrackle conjecture still remains unproven; however, the statement could be verified for several subclasses of thrackles which we will briefly summarize in the following. Some bounds on the number of edges are discussed and after recapitulating what is known to date about the structure of possible counterexamples, we will present computational results obtained in the course of this work.

5.1 Resolved Variants

Although a resolution of the thrackle conjecture appears to be out of reach at the time, there are several restricted versions of the problem that could already be settled.

5.1.1 Straight-line Thrackles

A first variant that will be covered in detail here is to restrict edges to straight-line segments.

Definition 5.4. A *straight-line thrackle* is a thrackle that is drawn in such a way that all its edges are represented by straight-line segments.

The answer to the thrackle conjecture for the straigt-line case was already given in the affirmative before Conway even stated the question for good drawings in general. The first proof can be attributed to Paul Erdős.

His idea for the proof originated in a related question posed some years earlier by Heinz Hopf and Erika Pannwitz [40] in the annual report of the German Mathematical Society[1]. They asked for a proof of the following statement: Let $p_0, p_1, ..., p_{n-1}, p_n = p_0$ be a set of n points in the plane s.t. the distance between any pair is less or equal to 1 and for successive pairs equality holds. This is possible if n is odd and impossible if $n >= 4$ is even.

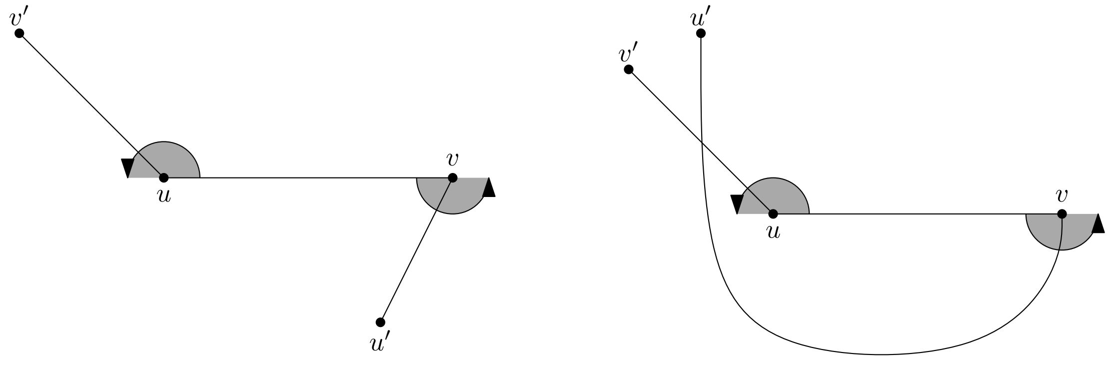
[1] Jahresbericht der Deutschen Mathematiker-Vereinigung

Several proofs were submitted in a later volume of the report ([65], [21] and [10]). Erdős [19] noticed that the maximum distance between any two points in the plane can occur at most n times. He credited this observation to Hopf and Panwitz directly, although it seems that the most helpful hints appear in Fenchel's proof of the problem. By pointing out that whenever two pairs of points p_1p_2 and p_3p_4 are both at maximum distance r, the lines $\overline{p_1p_2}$ and $\overline{p_3p_4}$ necessarily intersect. Otherwise the maximum distance among these four points must exceed r. It is not too hard to finish the proof of the said statement and thus of the thrackle conjecture in the straight-line case; however, we will give a different, particularly nice proof due to Micha Perles [51].

Theorem 5.3 (Erdős)**.** *The number of edges in any straight-line thrackle does not exceed its number of vertices.*

Proof. A *pointed* vertex v in a straight-line thrackle is a vertex where all its incident edges lie one half-plane determined by a line passing through v. We call the edge encountered first while rotating the line through v by 180° in clockwise direction *leftmost.* Clearly, every vertex has only one such edge and the claim is that after removing all leftmost edges at the pointed vertices, then there is no more edge remaining in the graph.

Suppose the contrary is the case and an edge uv remains after the deletion operations as described above. This implies that the edges were neither removed as leftmost edge at u nor as leftmost edge at v. Hence, at each vertex u and v there exists an additional edge that is reachable from the edge uv by rotating it 180° counter-clockwise around the respective vertex. See Figure 5.4(a) where the possible angles for the two edges uv' and vu' are shaded. This implies that the two said edges cannot intersect in a straight-line drawing since they emanate into opposite half-planes, which contradicts the properties of thrackles and concludes the proof of Theorem 5.3. ☐



(a) Argument used in Perles' proof. (b) Why it fails for x-monotone thrackles.

Figure 5.4: The argument used by Perles' proof in the straight-line and x-monotone case.

5.1.2 Outerplanar and x-Monotone Thrackles

We will now consider two other interesting special cases of thrackles for which the thrackle conjecture could be settled in the affirmative.

Definition 5.5. An *outerplanar* thrackle is a thrackle with the vertices placed on a circle and all of its edges are contained entirely in the interior of the circle.

To prove the thrackle conjecture in the outerplanar case Cairns and Nikolayevsky [15] first showed that every such thrackle with minimum degree of at least two is in fact a single cycle of odd length. To establish this, they used the same proof as Perles did for the straight-line case.

Theorem 5.4. *The number of edges in any outerplanar thrackle does not exceed its number of vertices.*

Definition 5.6. An *x-monotone* thrackle is a thrackle with its edges intersected at most once by every vertical line.

For x-monotone thrackles the proof used in the above two cases breaks down. See Figure 5.4(b) where both edges uv' and vu' intersect invalidating the counting argument used in Perles' proof. Pach and Sterling [51] instead imposed a partial order on the edges which allowed them to verify the conjecture for the x-monotone case.

Theorem 5.5. *The number of edges in any x-monotone thrackle does not exceed its number of vertices.*

As a counterpart to these three cases, it is worth mentioning a setting where the thrackle conjecture is no longer valid. If we move from embeddings in the plane (or equivalently the sphere) to an embedding on the torus, it is already possible to construct a counterexample with five vertices and six edges [71]. See Figure 5.5 for a thrackle drawing on the surface of a torus. Also it is possible to embed the 4-gon as thrackle.

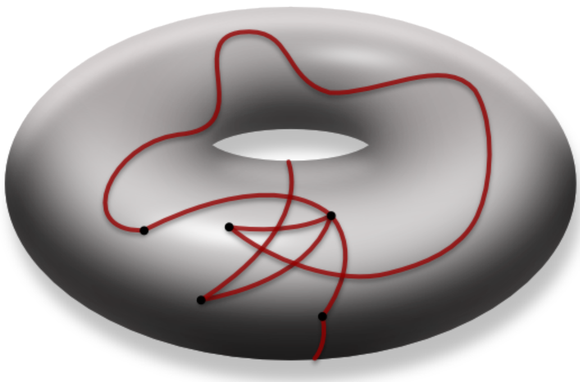


Figure 5.5: Embedding of a thrackle with 5 vertices and 6 edges on the torus as in [71].

5.1.3 Generalized Thrackles

Another variant of the problem is to not impose restrictions on the curves representing the edges of a thrackle, but instead on their crossing properties. In particular, an interesting problem arises when the number of crossings between any pair of edges is no longer forced to be exactly one.

Definition 5.7. A *generalized* thrackle is a drawing of a graph where any two edges share an odd number of points. Again these points may be a common vertex or proper intersections.

Also in this version of the problem it is not immediately apparent that the thrackle conjecture is false. Shortly after his first work on the subject, Woodall posed it as an open problem whether the number of edges in a generalized thrackle can become larger than the number of vertices.

Lovász, Pach, and Szegedy [47] proved that a bipartite graph can be drawn as a generalized thrackle if and only if it is planar. It is well known that planar bipartite graphs can have up to $2n-4$ edges. Thus, there are generalized thrackles with roughly twice as many edges as vertices. Cairns and Nikolayevsky [14] could then show a bound of $2n-2$ edges and gave examples that this bound is sharp. Furhtermore, they showed a similar, more general result, namely that any bipartite graph can be drawn as a generalized thrackle on a closed orientable connected surface if and only if it can be embedded in that surface.

5.2 Bounds on the Number of Edges

Since proving that any thrackle has at most as many edges as vertices seems to be incredibly hard, a natural thing to do is to establish reasonable bounds on the number of edges.

For a long time, the only bound known was not even linear in the number of vertices. It relies on the fact that there are no cycles of length four in any thrackleable graph. The rest of the proof can be finished using methods taught in basic lectures on graph theory. See for instance Exercise 2.1.15 in [12] where also the most relevant hints for establishing the following proof are given.

Theorem 5.6. *The number of edges in any thrackle is at most $O(n^{\frac{3}{2}})$.*

Proof. Observe that no thrackle can contain a non-thrackleable graph as a subgraph. In particular, this implies that any thrackle must be free of C_4's. Let G be a graph on n vertices and m edges. For any vertex $v \in V(G)$ let d_v denote the degree of v. Suppose that G does not contain C_4 as a subgraph, we show that m is at most $O(n^{\frac{3}{2}})$.

Consider a certain type of labelled subgraphs. A *cherry* $C(a,b,c)$ is the graph $K_{1,2}$ with both a and c connecting to the *centre* b of the cherry (see Figure 5.6). We count the number of labelled cherries $|C|$ in G. Every vertex v is the centre of $d_v^2 - d_v$ such cherries. On the other hand, any ordered pair of vertices (u,w) can have at most one common centre. Otherwise

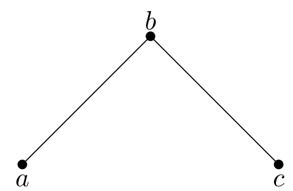


Figure 5.6: $C(a,b,c)$.

G contains a cycle of length four. With both observations we can count/bound the number of cherries contained in G.

$$|C| = \sum_{v \in V(G)} d_v^2 - d_v \leq n(n-1)$$

With the following well known relation between the root-mean-square and the arithmetic mean, we can further simplify the equality.

$$\begin{aligned}
\sqrt{\frac{d_1^2 + ... + d_n^2}{n}} &\geq \frac{d_1 + ... + d_n}{n} \\
\frac{d_1^2 + ... + d_n^2}{n} &\geq \frac{(d_1 + ... + d_n)^2}{n^2} \\
d_1^2 + ... + d_n^2 &\geq \frac{(d_1 + ... + d_n)^2}{n}
\end{aligned}$$

Together with $\sum\limits_{v \in V(G)} d_v = 2m$ we can come up with a quadratic inequality for the number of edges in graphs without cycles of length four.

$$\begin{aligned}
\frac{1}{n}\left(\sum_{v \in V(G)} d_v\right)^2 - \sum_{v \in V(G)} d_v &\leq n(n-1) \\
\frac{(2m)^2}{n} - 2m &\leq n(n-1) \\
m^2 - \frac{mn}{2} - \frac{n^3 - n^2}{4} &\leq 0
\end{aligned}$$

Solving the inequality yields the desired bound on the maximum number of edges in any thrackleable graph.

$$m \leq \frac{1}{4}(n + \sqrt{n^2(4n-3)}) = O(n^{\frac{3}{2}})$$

□

A first substantial improvement to the above bound was given by Lovász et al. [47]. They showed that any bipartite thrackle is planar. Additionally, they noticed that bipartite planar graphs without cycles of length four can have at most $\lfloor \frac{3n}{2} \rfloor - 3$ edges. Since it is possible to make any graph bipartite by removing at most half of its edges, this yields a bound of at most $3n-7$ edges for any thrackleable graph. By further exploiting properties of thrackles, the bound was eventually reduced.

Theorem 5.7 (Lovász, Pach, Szegedy, [47]). *Every thrackle of n vertices has at most $2n-3$ edges.*

The above bound was further reduced by Cairns and Nikolayevsky [14]. The interesting thing about their proof is that their argument was based entirely on generalized thrackles (see Definition 5.7). For the improvement in the case of thrackles, as in the proof of Theorem 5.6, they only used the property that no thrackle can contain C_4 as a subgraph.

Theorem 5.8 (Cairns, Nikolayevsky, [14]). *Every thrackle of n vertices has at most $\frac{3}{2}(n-1)$ edges.*

The result in Theorem 5.8 represents the to date best known bound achieved without computer assistance. The best result, however, comes from a very interesting method presented by Fulek and Pach [24]. By exploiting certain properties of the structure of possible counterexamples to the thrackle conjecture (see Section 5.3), they gave an algorithm for any $\varepsilon > 0$ terminating in $e^{O((1/\varepsilon^2)\ln(1/\varepsilon))}$ steps to decide whether the number of edges are not more than $(1+\varepsilon)n$ for all thrackles with $n \geq 3$. In case the algorithm fails to verify the said statement, it produces a counterexample to the conjecture.

If one takes a look at the runtime of their algorithm, it becomes clear that it will take a considerable amount of computing time to lower ε substantially. Anyway, the result they could achieve stands as the best known to date bound on the number of edges of thrackles.

Theorem 5.9 (Fulek, Pach, [24]). *Every thrackle of n vertices has at most $\frac{167}{117}n$ edges.*

5.3 The Structure of a Minimal Counterexample

Woodall [71] already mentioned that a counterexample to Conway's conjecture, if it exists, would consist of two cycles that are joined by a path, share a path, or meet in exactly one vertex. Stephan Wehner also mentioned this on his webpage where the corresponding cases are called *dumbbell*, *theta* and *figure-8*. We will, however, use the general notion of a dumbbell as introduced by Fulek and Pach [24] which captures all three cases at once.

Definition 5.8. Given three integers $c', c'' > 2, l \geq 0$, the dumbbell $DB(c', c'', l)$ is a simple graph consisting of two disjoint cycles of length c' and c'', connected by a path of length l. For $l = 0$, the two cycles share a vertex. The definition extends to negative values of l the following way. For any $l > -min(c', c'')$, let $DB(c', c'', l)$ denote the graph consisting of two cycles of lengths c' and c'' that share a path of length $-l$.

We will now give a proof that a minimal counterexample must be of the above form.

Theorem 5.10. *Suppose that G is a minimal thrackleable graph with more edges than vertices, then G is a dumbbell.*

We will first prove two lemmas of which we will make use of in the proof of Theorem 5.10.

Lemma 5.11. *Suppose that G is a minimal counterexample to the thrackle conjecture, then G*

(i) is connected.

(ii) has $|V(G)| + 1$ edges.

(iii) does not contain vertices of degree one.

Proof. (i) Suppose that G is not connected. To form a counterexample, at least one connected component is required to have more edges than vertices. This contradicts the minimality of G.

(ii) Trivially, $|E(G)| > |V(G)|$ must hold. In case $|E(G)| > |V(G)|+1$ we remove an arbitrary edge on a cycle. Removing a single edge on a cycle can never disconnect a graph and we obtain a graph with one edge less that is still a counterexample to the thrackle conjecture, contradicting minimality. We can do so until $|E(G)| = |V(G)| + 1$.

(iii) We remove a degree one vertex together with the edge incident to it. The resulting graph remains connected and since the number of vertices and the number of edges was reduced by one, it still has more edges than vertices. Thus G was not minimal. □

Lemma 5.12. *Every minimal counterexample G can be obtained by adding two edges to a path thrackle.*

Proof. By Lemma 5.11 (i) and (ii) G is connected and has $n+1$ edges. This means in particular that the sum of degrees in G is exactly $2n + 2$. Due to Lemma 5.11 (iii), every vertex has at least degree 2. This means that there can either be one vertex with degree 4, or two vertices with degree 3.

Case a: Suppose we have one vertex v of degree four. The induced subgraph on v looks locally like a star S_4 (see Figure 5.7 (left)). Since we cannot have vertices of degree one, each of the four vertices around v needs to be connected to exactly one of the other vertices adjacent to v by an edge or a sequence of edges. Since all vertices but v have degree 2, these edges or sequences of edges are necessarily vertex disjoint. We remove two of the edges incident to v that do not lie on the same sequence of edges as described before. A connected graph remains with all vertices having degree 2, except for two vertices of degree 1. This graph clearly resembles a path thrackle.

Case b: Consider that there are two vertices v_1 and v_2 with degree 3 in the graph. Both of these vertices are locally a star S_3 as depicted in Figure 5.7 (right). With the same argument as before, we know that all of the vertices a to f need to be connected to another vertex of that set by an edge or a sequence of edges. Again, all these sequences of edges are mutually disjoint, since all other vertices have degree 2. In particular, this means that there is at least one edge or sequence of edges connecting two vertices from the different S_3 subgraphs. This ensures that the thrackle remains connected after removing two edges in the following manner. Remove one edge adjacent to v_1 and one edge adjacent to v_2 and ensure that they are both not part of the sequence of edges mentioned above. We then obtain a connected graph where the degrees of v_1 and v_2 were lowered to 2 and the two degree one vertices lost a neighbour each. Again, what remains is a path thrackle. $\square$

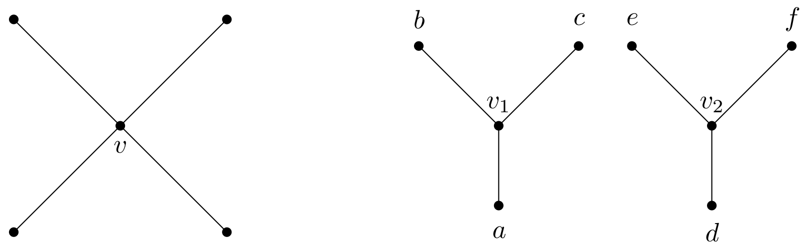


Figure 5.7: Degree four vertex and two degree 3 vertices.

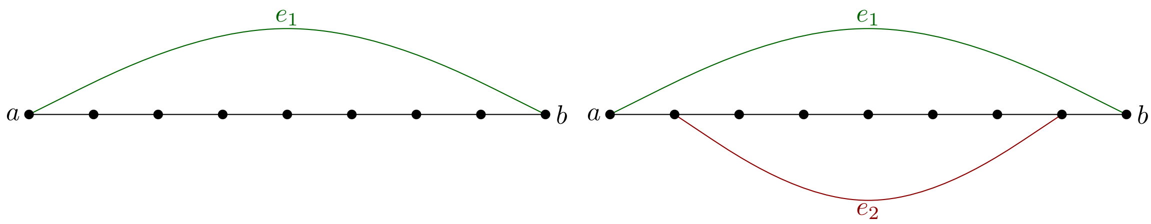
Proof of Theorem 5.10. As stated in Lemma 5.12, we can always construct the minimal counterexample by adding two edges to a path thrackle. Since a minimal counterexample cannot have any vertex with degree 1, we need to cover the end vertices, say a and b, of the path thrackle with the two edges we are going to add. The order in which we introduce the new edges is of no importance. We will therefore assume that the first edge e_1 we add is not incident to fewer of the end vertices than the second edge e_2. After that we will necessarily obtain a graph with exactly one cycle C. When adding the second edge, we produce a second cycle and only the following cases can occur.

Case a: Both endpoints of e_2 are on C. This case can only occur if the edge e_1 connected both vertices a and b and we were left with a cycle thrackle. We clearly construct a dumbbell $DB(c', c'', l)$ with $l < 0$. See Figure 5.8(a).

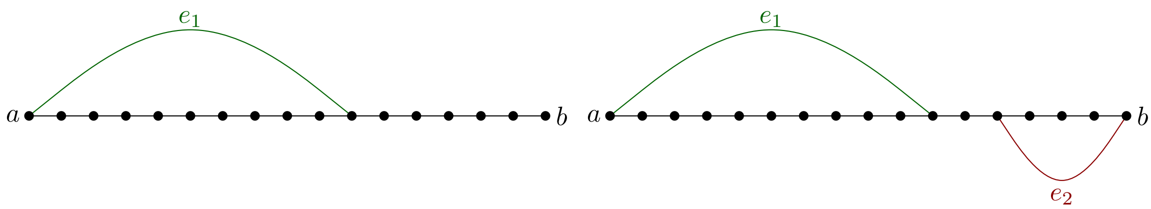
Case b: Both endpoints of e_2 are not on C. Without loss of generality, we can assume that the edge e_1 is only connected to a, but not to b. For otherwise we were left with a cycle thrackle alone, which makes it impossible to add an edge that is disjoint from the cycle C. So for e_2 we can only have one of the endpoints connecting to b and the other somewhere to the path attached to C, but not part of it. This gives two cycles that are joined by a path, so a dumbbell of the form $DB(c', c'', l)$ with $l > 0$ as it can be seen in Figure 5.8(b).

Case c: One endpoint of e_2 lies on C and the other does not. Again this case only occurs if the first edge left a cycle with a path attached. b is the only vertex with degree one when adding the second edge, which means that b is necessarily an endpoint of e_2. For the second endpoint of e_2, we are only left with vertices of C. This can either be the vertex where the path is attached to the cycle or any other vertex on the cycle. In both cases the final construction resembles a dumbbell $DB(c', c'', l)$ with $l = 0$ and $DB(c', c'', l)$ with $l < 0$, respectively. So either two cycles sharing a single vertex or a path of length $-l$. Both cases are depicted in Figure 5.8(c).

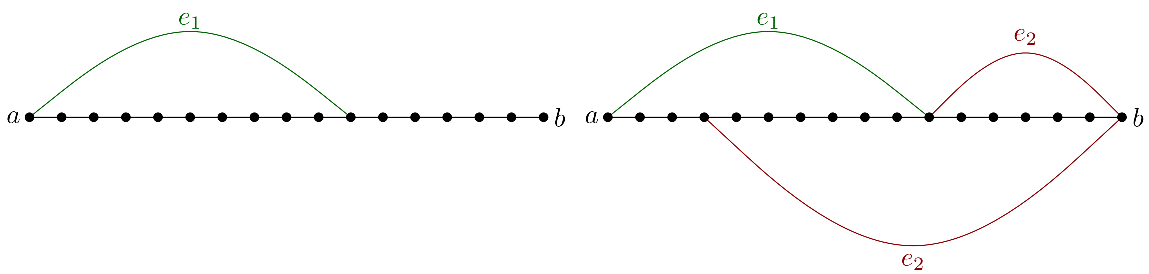
In either of the cases, we obtain a minimal counterexample with exactly two cycles of the form of a dumbbell as defined above. By Lemma 5.11 (iii) it follows that such a structure comprises the entire minimal counterexample. $\square$



(a) Case a.



(b) Case b.



(c) Case c.

Figure 5.8: Examples of adding e_1 and e_2 in the three cases of the above proof.

It is moreover possible to show that any dumbbell can be transformed to a dumbbell of the form $DB(c', c'', 0)$ with c' and c'' being even, hence two even cycles that share a vertex.

Woodall [71] mentioned this fact while Fulek and Pach [24] gave more details on how to perform these transformations. At the heart of their method is the so-called *Conway doubling* procedure. It is possible to transform a sequence of vertices and edges in a thrackle in such a way that the resulting drawing is still a thrackle, but the said vertices and edges are doubled. The original vertices are replaced by pairs of vertices lying sufficiently close together, and the edges are replaced by two edges following the original one as depicted in Figure 5.9. The dashed edges on the left side show that we can also avoid duplicating the first vertex of the path, which we will use later on.

Suppose now that we have a counterexample of the form $DB(c', c'', l)$ with $l \neq 0$. Meaning, two cycles that are connected by or share a path. In either of these cases one can use the doubling procedure as described above and transform the thrackle to a dumbbell of two cycles meeting in precisely one vertex. As an example see Figure 5.10. We can in both cases eliminate the path between or shared by the cycles such that we obtain two cycles meeting in a single vertex. It can also be shown that in case one of the two cycles has odd size, Conway doubling can be used to transform the counterexample to the form of two even cycles sharing a vertex. However, the obtained examples do not need to be minimal any longer.

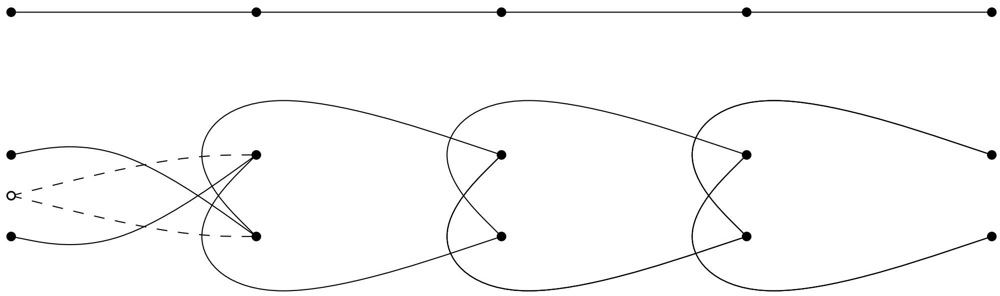


Figure 5.9: Illustration of Conway doubling.

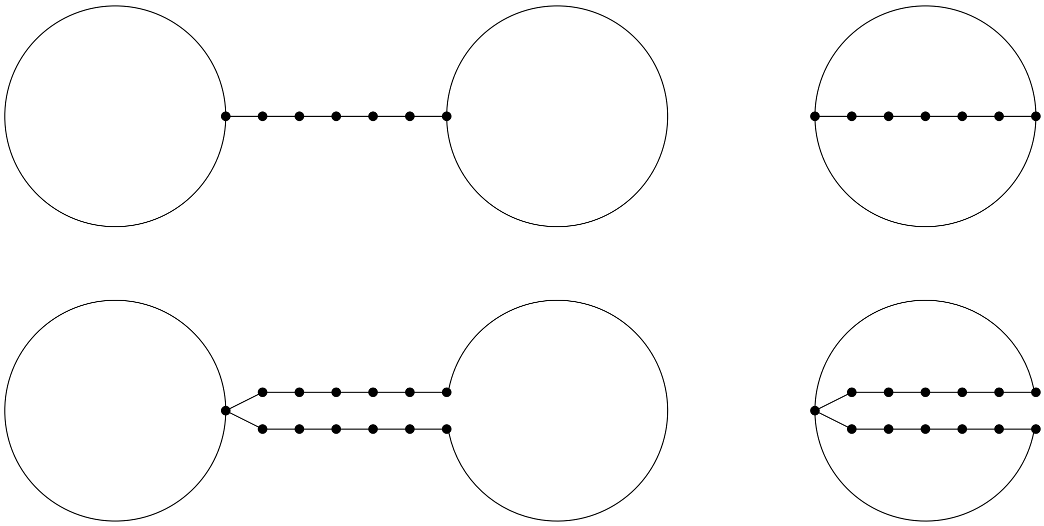


Figure 5.10: Transformations of both kinds of dumbbells with $l \neq 0$.

In the next section we show by massive computer search that a counterexample to Conway's thrackle conjecture needs to have at least 13 vertices.

5.4 Enumerating Thrackles and Searching for Counterexamples

In this section we first present an algorithm to enumerate all non-isomorphic connected thrackles for small values of n. Afterwards we present an improved version of the algorithm that is no longer used for enumerating, but rather for finding possible counterexamples to the thrackle conjecture. First we define three classes of thrackles that are interesting for our purpose.

Definition 5.9. A *tree thrackle* is a thrackle on $n \geq 2$ vertices with exactly $n-1$ edges consisting of exactly one connected component.

Definition 5.10. A *full thrackle* is a thrackle on $n \geq 3$ vertices with exactly n edges consisting of exactly one connected component.

Definition 5.11. An *overfull thrackle* is a thrackle on $n \geq 3$ vertices with exactly $n+1$ edges consisting of exactly one connected component.

We will now show that the sets of all non-isomorphic tree thrackles T_n, full thrackles F_n, and overfull thrackles O_n on n vertices can be determined from the set T_{n-1} by adding a vertex, an edge or two edges to all elements of T_{n-1}. Then we give an algorithm that takes the elements of T_{n-1} as input and produces the sets T_n, F_n and O_n.

Note that assuming the truth of Conway's thrackle conjecture, we expect O_n to be the empty set for all $n \geq 3$.

Theorem 5.13. *Let T_n, F_n, and O_n be the sets of all non-isomorphic tree thrackles, full thrackles and overfull thrackles on n vertices, respectively. There is an algorithm to determine T_n, F_n, and O_n given the set T_{n-1} as input.*

We will prove Theorem 5.13 with the help of Lemmas 5.14 and 5.15 and by developing the algorithm in Section 5.4.2.

Lemma 5.14. *Every element of T_n can be obtained from at least one element of T_{n-1} by adding a missing edge in such a way that it obeys the properties of thrackle drawings.*

Proof. Suppose that G is a thrackle with $n \geq 3$ vertices and $n-1$ edges. The sum of all degrees in G is $2n-2$ since every edges contributes 2 to this sum. Therefore, the average degree in G is $(2n-2)/n < 2$, which implies that there is at least one vertex $v \in G$ with $d(v) = 1$. If we remove v together with the edge incident to it, we get G' a connected thrackle with $n-1$ vertices and $n-2$ edges. Hence, G' is a member of the set T_{n-1}. Adding the vertex v and the corresponding edge back to G' gives the original tree thrackle G. □

Clearly, a tree thrackle might have more than one vertex of degree 1. Thus it might be (and in general will be) possible to obtain a unique element of T_n from several elements of T_{n-1} by adding a vertex together with an edge. For the algorithm this means that we require some kind of fingerprint to distinguish between non-isomorphic thrackles. We will address this issue later in this section when we go into more detail of the actual algorithm.

Further, we need to show that F_n can be obtained from T_n and O_n from F_n by adding edges.

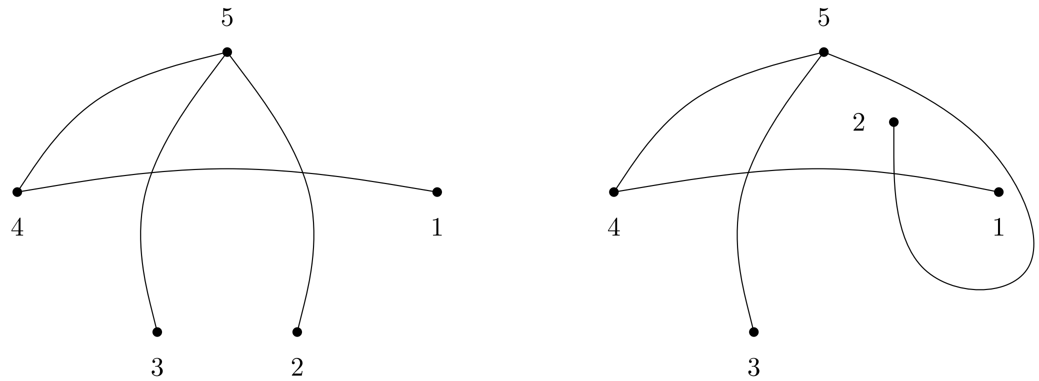
Lemma 5.15. *Every element of F_n (O_n) can be obtained from at least one element of T_n (F_n) by adding a missing edge in such a way that it obeys the properties of thrackle drawings.*

Proof. Suppose that G is a full thrackle on $n \geq 3$ vertices. Clearly, G contains a cycle and removing an arbitrary edge on that cycles leaves the resulting thrackle G' connected. Also the removal of an edge cannot violate the restrictions of thrackle drawings and we get an element of T_n. This implies that G can be obtained from at least one element of T_n by adding a missing edge that shares exactly one point with any other edge. The argumentation for elements of O_n is exactly equivalent. □

5.4.1 Fingerprint for Connected Thrackles

As it was mentioned before, it is possible to encounter one and the same thrackle by extending different thrackles by a vertex and an edge or an edge only. We therefore need a criterion (fingerprint) to distinguish thrackle drawings from one another.

Let us recall the fingerprint used for non-isomorphic drawings of the complete graph (Section 4.1.1). Together with the rotation system, we only needed to determine the order of the crossings along each edge. This is sufficient because the rotation of the crossing is already determined by this order and we therefore capture all information required in order to distinguish non-isomorphic drawings. However, this property is no longer valid for good drawings in general, particularly not for thrackle drawings. See Figure 5.11 for two thrackle embeddings of the same graph. The rotations and crossings along the edges are the same, but the edges highlighted intersect from opposite sides. We therefore get a different rotation for the said crossing and it is easy to verify that, indeed, the drawings are different with respect to isomorphism, since the cell structure of the drawings obviously differs. The three degree one vertices, for instance, are incident to the same cell in one drawing and lie in different cells in the other.



(a) $111234551523425_0 35_0 614_0 614_0 66$

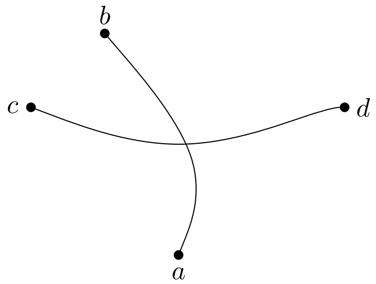
(b) $111234551523425_1 35_0 614_1 614_0 66$

Figure 5.11: The two kinds of rotations at a crossing between ab and cd.

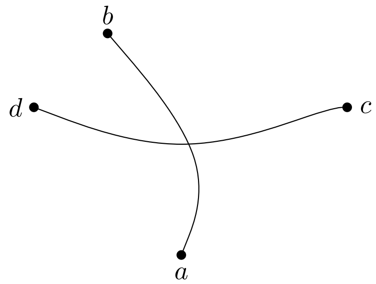
Because of the above, it is necessary to consider the whole extended rotation system of the graph along with the order of the crossings. Anyway, there are not much modifications to the fingerprint used for $D(K_n)$ required. For a crossing we do not only list the indices of the edges that cross, but an additional binary value encoding the rotation of the crossing. Consider a crossing between the edges ab and cd with $a < b$, $c < d$ and $a < c$. When starting at the vertex

a with the lowest index of all vertices involved in the crossing, we can only have one of these two clockwise rotations at the crossing. Either we have (a, c, b, d) or (a, d, b, c). We say that the crossing rotation in these cases is 0 or 1, respectively.

Additionally, the degrees of the vertices are no longer fixed. For this reason we use the degree sequence of the graph as the first part of the fingerprint. This has the further advantage that we can exclude isomorphism for graphs with different degree sequences by only looking at these first n entries of the fingerprint. It is also necessary to have this information in order to assign the rotations to the correct vertices and, in the following, determine the edges that are present in the drawing. Another difference to complete graph drawings is that the information about what edges cross is no longer contained in the rotation system. We therefore list for every edge that is in the graph the two indices of the crossing edges together with the binary value indicating the kind of rotation we have at the crossing. To be able to relate the entries to the correct edges, we delimit them with the value $n + 1$ in the fingerprint. Note that edges without crossings are also considered in the fingerprint. For an example see the fingerprints in the captions of Figures 5.11(a) and 5.11(b). The crossing rotations are written as subscripts of the corresponding indices of the edge. It can be seen that the fingerprints only differ in two of these entries, meaning the drawings differ in one crossing rotation.



(a) 0 crossing.



(b) 1 crossing.

Figure 5.12: The two kinds of rotations at a crossing between ab and cd.

5.4.2 Enumerating Connected Thrackles

The algorithm we use is quite similar to the one used to enumerate good drawings of the complete graph K_n. We again use the half-edge data structure to represent the thrackle drawings and in this case we don't even need to introduce the dummy segments at the vertices, since we do not need to enforce a certain rotation system for the drawings. The two main operations required are the extension of an existing thrackle by a new vertex together with an edge that connects it, and the addition of an edge between two non-adjacent vertices of the thrackle.

Adding a vertex: For extending a certain thrackle with n vertices and m edges by a new vertex together with an edge, we need to place the new vertex in every cell. Then we have to choose every vertex of the thrackle as target of the new edge. This determines the edges that need to be intersected. Starting in a certain cell we proceed to recursively intersect required segments, just as we did in the algorithm presented in Chapter 4. Whenever we have intersected all edges needed to fulfil the conditions of thrackle drawings, we check whether we are currently in a cell incident to the target vertex. If this is the case, a valid thrackle drawing with $n+1$ vertices and $m+1$ edges was found.

Adding an edge: The operation required to add an edge to a thrackle with n vertices and m edges is basically the same as the operation described above. Instead of placing a new vertex in every cell, we start with a segment emanating from every vertex of the thrackle in every possible order in the rotation of the vertex. Clearly, only non-adjacent vertices are chosen as targets and the procedure continues exactly as before. Whenever a valid thrackle is found, it has n vertices and $m+1$ edges.

The diagram in Figure 5.13 shows how the sets to be generated are related to each other. The arrows between the classes mean that any element of the class that the arrow points to can be generated from an element of the class where the arrow starts at by either the operation of adding a vertex together with an edge *+ vertex*, or by adding an edge between non-adjacent vertices *+ edge*. Given the complete set T_n, we apply the first operation on every element of T_n to obtain the complete set T_{n+1}. The set F_n is generated by applying the second operation on every element of T_n as well. O_n is then obtained by again applying the second operation on all elements of F_n. If ever we encounter that the set O_n is non-empty, we have found a coutnerexample to the thrackle conjecture.

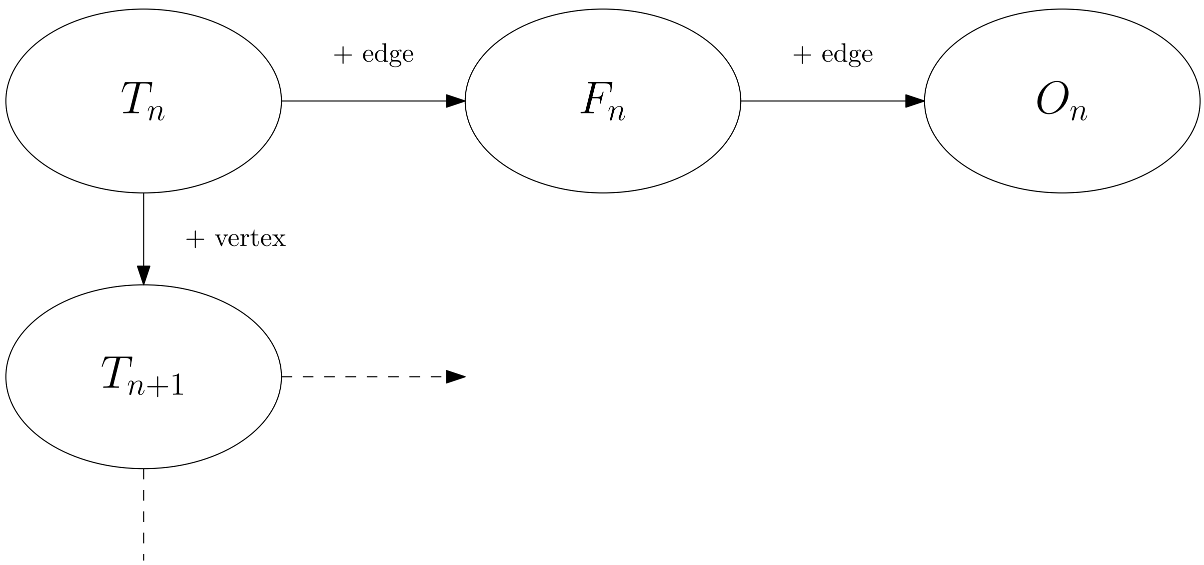


Figure 5.13: Relation between the sets T_n, F_n and O_n.

Note that whenever a new element is encountered, we need to generate its fingerprint according to Section 5.4.1 and compare it to all elements that are already stored in the database.

So everything required to enumerate non-isomorphic thrackles is the set T_2, namely all non-isomorphic tree thrackles on two vertices. This set only contains one element, which is two vertices connected by a single edge. The results obtained by running the algorithm up to $n = 9$

are listed in Table 5.1.

Although running time becomes an increasing problem for this approach as n gets larger, the actual problem is to maintain a database of all non-isomorphic tree and full thrackles. As these numbers rise drastically and searching in the database is necessary whenever a new element is encountered, the algorithm presented above could not be used to obtain results for $n > 9$. For the purpose of finding counterexamples, however, we can make use of several observations which will be presented in the following section.

n	$\|T_n\|$	$\|F_n\|$	$\|O_n\|$
2	1	-	-
3	1	1	0
4	2	1	0
5	5	6	0
6	41	48	0
7	698	994	0
8	22230	38472	0
9	1166917	2580004	0

Table 5.1: Results of enumeration of connected thrackles for $n \leq 9$.

5.4.3 Searching for Counterexamples

As already mentioned above the number of tree and full thrackles rises dramatically already for relatively small n; however, for the purpose of finding counterexamples, it is not necessary to enumerate all non-isomorphic thrackles. Due to the results obtained in Section 5.3, we know that it suffices to generate and extend only path thrackles.

By Lemma 5.12 every minimal counterexample to the thrackle conjecture can be obtained from at least one path thrackle by adding two edges. So instead of generating the sets T_n of tree thrackles on n vertices, it suffices to only maintain the sets P_n, all path thrackles of size n. This enormously reduces computation time for serveral reasons.

First of all $|P_n|$ can be expected to be significantly smaller than $|T_n|$, since $P_n \subseteq T_n$. Furthermore, calculating the fingerprint for every element once it is generated requires permuting through all possible labellings. While in some cases the number of labellings can be reduced because only vertices with equal degree can be mapped onto each other, in the case of cycle thrackles, where all vertices have degree two, we cannot avoid determining the fingerprint for all $n!$ labellings to find the lexicographically minimal one.

In the case of path thrackles, the calculation of the fingerprint can be drastically simplified. The degree sequence of any path thrackle is $(1, 2, 2, ..., 2, 1)$. So there are only two possible labellings that can be applied. One labelling for either of the degree one vertices as a start of the path. Note also that for every path thrackle on n vertices, not only the degree sequence, but also the rotations of the vertices are exactly the same. We can therefore omit these parts and only use the crossings orders together with the crossing rotations as the fingerprint.

To find possible counter examples, we simply extend each element of P_n by two edges in all meaningful ways. For instance we do not need to add two edges to the path in such a way that a vertex with degree one remains, since we know that such a graph cannot be a minimal counterexample.

By this observation we could build up a database of all path thrackles P_n on n vertices for $2 \leq n \leq 12$. The values for $|P_n|$ can be found in Table 5.2.

n	$\|P_n\|$	$\|O_n\|$
2	1	-
3	1	0
4	1	0
5	2	0
6	12	0
7	121	0
8	2399	0
9	73092	0
10	3502013	0
11	*258438398*	*0*
12	*31176142191*	*0*

Table 5.2: Results of enumeration of connected path thrackles for $n \leq 12$.

Still the cardinality of P_n grows dramatically. While computation time is not the biggest problem, storage requirements are the main factor of limitation. We could, however, use another observation that is presented in the following section to reduced space complexity of the algorithm to practically constant. This allowed us to obtain results for $n = 11$ and $n = 12$ (entries written in italic font in Table 5.2) giving us the knowledge that a counterexample to the thrackle conjecture, if existing, must have at least 13 vertices.

Observation 5.16. *Let G be a thrackleable graph on $n < 13$ vertices. Then G has at most as many edges as vertices.*

5.4.4 A Recursive Variant for Generating Path Thrackles

The main problem in the algorithm from Section 5.4.3 is not that all path thrackles need to be stored in a database on the hard disk, although this would be a big problem for $n = 12$ because storing all fingerprints of the 31176142191 path thrackles in a database would require approximately five terabytes of space. The real problem arises because we want to store only unique fingerprints, which means that for every new element, we need to check whether it is already contained in the database or not. Considering this huge amount of data, this is a virtually impossible task today.

Fortunately, for finding counterexamples to the thrackle conjecture, it is not necessary to keep all non-isomorphic path thrackles stored. Instead we run the algorithm recursively and give a condition when to further extend a path thrackle by a vertex such that we guarantee to not extend one and the same path thrackles multiple times.

We run the algorithm on all elements (i.e., path thrackles) with $n-1$ vertices in the database. Whenever we find a new path thrackle with n vertices, we will decide by the condition presented in the following if we count the newly found thrackle and extend it further in a recursive manner.

Consider the illustration below (Figure 5.14). When we found a path thrackle with n vertices $\mathcal{P}_n$, we obtained it by extending a path thrackle $\mathcal{P}_{n-1}$ with one vertex and edge less. Now we only accept $\mathcal{P}_n$ if we could not have reached it by extending a different thrackle with a lexicographically smaller fingerprint than $\mathcal{P}_{n-1}$. So we calculate the two fingerprints $\mathcal{P}'_{n-1}$ and $\mathcal{P}''_{n-1}$, the fingerprints of the thrackles obtained by omitting the first edge and vertex and last edge and vertex of $\mathcal{P}_n$, respectively. If now one of these is lexicographically smaller than $\mathcal{P}_{n-1}$, we can also reach $\mathcal{P}_n$ from this thrackle and therefore reject the newly found drawing.

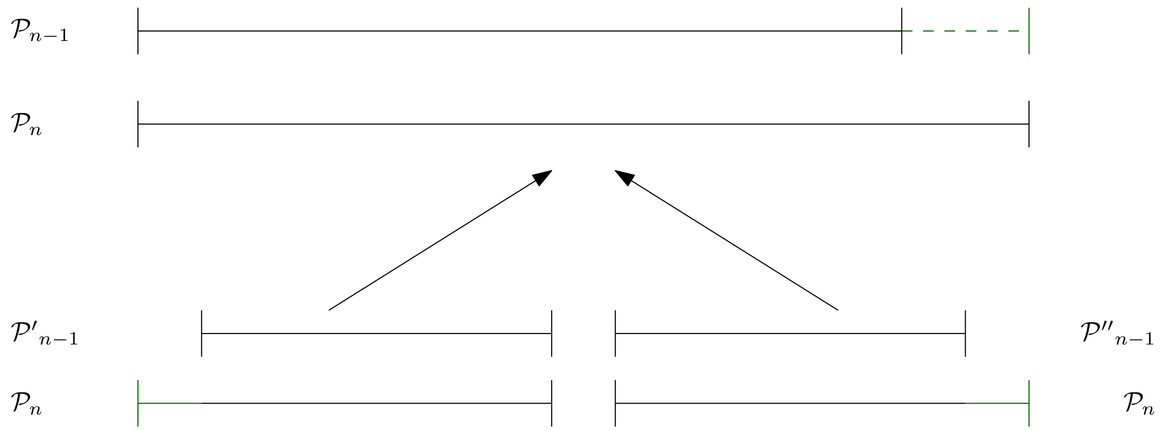


Figure 5.14: Fingerprints of path thrackles and how they are extended.

On the other hand, it is possible that a thrackle has a *symmetric fingerprint*. Meaning, if we omit the first edge and vertex, we get the same fingerprint as when omitting the last edge and vertex. In such a case, we can no longer determine when to accept or reject the thrackle. Therefore we need to store thrackles if they are not rejected and accept it only if it was not stored before. The effect on the storage requirements is that during the algorithm we only have to keep all n path thrackles in memory that can be obtained by extending a single $n-1$ path thrackle. We can then avoid further extending the same thrackles multiple times and memory is freed when the $n-1$ path thrackle was completely processed.

In case a new thrackle is accepted by the algorithm, we proceed as in the previous algorithm by adding two additional edges and test if a counterexample to the thrackle conjecture can be constructed. We could use this reduction of memory usage of the program to enumerate all path thrackles up to $n = 12$ vertices, hence verifying that there is no counterexample with less than $n = 13$ vertices. See the last two entries in Table 5.2 for the cardinalities of P_n.

Further computations were not carried out because, set up as four processes in parallel, it took a bit more than two weeks to produce the results for $n = 12$. Although the actual number of path thrackles for $n = 13$ can only be roughly estimated, one can intuitively tell from Table 5.2 that verifying that there is no counterexample with $n = 13$ vertices is a task that is not doable in reasonable time; at least not without the use of substantially more hardware resources.

5.5 Additional Remarks

Despite having all our efforts to come up with a counterexample to the thrackle conjecture fail, we have recapitulated many observations concerning the structure of a possible thrackle with more edges than vertices. In particular, it was shown that proving or disproving the conjecture is equivalent to proving or disproving that two even cycles that share precisely one vertex can be drawn as a thrackle. Further, our computations suggest that there is no counterexample with fewer than $n = 13$ vertices.

Interestingly, there already exists an abstract overfull thrackle for $n = 7$. By this we mean that a rotation system of the complete graph exists where all 4-tuples of vertices are realizable as either of the two non-isomorphic good drawings of K_4. In this case we can determine from the rotation system whether any two edges cross. The rotation system listed below is such an example and the edges 13, 15, 17, 24, 26, 34, 37, and 56 either share a vertex or cross, hence form a thrackle with 7 vertices and 8 edges. The entire rotation system, however, is not realizable as a good drawing of the complete graph and neither is the subgraph formed by the eight edges mentioned.

It is also interesting to see that the structure of the abstract thrackle as despicted in Figure 5.15 is also of the in principle possible kind. Meaning, two cycles that share an edge and not both having an odd size.

$\mathcal{R}(D(K_7))$:
1: 2 3 4 5 6 7
2: 1 3 4 5 6 7
3: 1 2 4 6 5 7
4: 1 5 7 2 6 3
5: 1 4 2 6 3 7
6: 1 7 2 4 3 5
7: 1 4 6 2 3 5

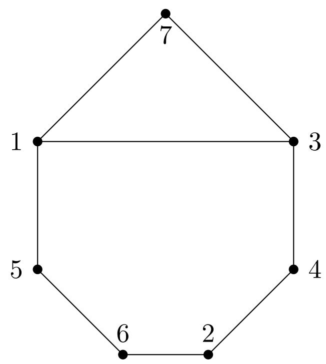


Figure 5.15: Abstract rotation system of K_7 (left) admitting thrackle with $n+1$ edges (right).

6
Related Topics

The following part summarizes results of problems related to good drawings of the complete graph. As mentioned in the introductory chapter, most of the following content was elaborated during a workshop in fall 2013.

6.1 Crossing Maximal Drawings

As already mentioned in Chapter 3, it is easy to see that the maximum number of crossings in any good drawing is $\binom{n}{4}$. This is also the case with geometric drawings, but in this case there is only one configuration that can attain this bound, namely sets with all points in convex position.

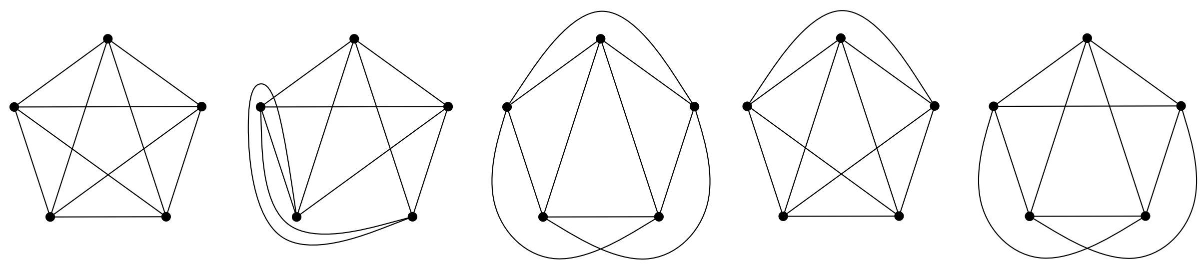


Figure 6.1: The five non-isomorphic drawings (rotation systems) of K_5.

For good drawings, however, this is different. There exist at least two different rotation systems (and thus good drawings) for fixed $n \geq 5$ with the maximum number of crossings. See Figure 6.1 (left) for the two maximizing good drawings for $n = 5$. Observe that one of these drawings (leftmost) is equivalent to the convex case for geometric drawings, whereas the other drawing (second) cannot exist as geometric drawing.

The number of different good drawings for the complete graph of n points with $\binom{n}{4}$ crossings is at least the number of different rotation systems with this property, that is, the number $T_w^{max}(n)$ of weak isomorphism classes of simple drawings of K_n with the maximum number of crossings [44]. Harborth and Mengersen [36] proved a lower bound of $e^{\Omega(\sqrt{n})}$ for $T_w^{max}(n)$. Kynčl [44] improves this estimate to $T_w^{max}(n) \geq 2^{n-5}\frac{(n-3)!}{n} \geq 2^{n(\log(n)-O(1))}$. In [43] Kynčl presents an observation that might help to improve the upper bound of $T_w^{max}(n)$ to $2^{O(n^2)}$. At the moment the upper bound is given by the upper bound for $T_w(K_n)$ (the number of weak isomorphism classes of good drawings that realize K_n) which is $T_w(K_n) \leq 2^{n^2\alpha(n)^{O(1)}}$ [43] (the constant in $O(1)$ is huge, roughly 4^{30^4}).

Table 6.1 presents the number of realizable rotation systems with $\binom{n}{4}$ crossings for $4 \leq n \leq 10$.

n	crossing-maximal realizable rotation systems
4	1
5	2
6	10
7	115
8	2657
9	82957
10	3226173

Table 6.1: The number of crossing-maximal realizable rotation systems for $4 \leq n \leq 10$.

From an exhaustive search for $4 \leq n \leq 8$, we know that all realizable rotation systems for these numbers of points induce a plane Hamiltonian cycle. Hence, also all crossing maximizing rotation systems for that many points induce a plane Hamiltonian cycle. In accordance to the question about whether a plane Hamiltonian cycle is contained in every good drawing, we ask the supposedly simpler question: does every good drawing with $\binom{n}{4}$ crossings induce a plane Hamiltonian cycle?

6.2 Plane Cycles in Drawings of the Complete Graph

The number of Hamilton cycles in the complete graph is very well defined and we give a quick proof of the following statement.

Proposition 6.1. *The complete graph K_n on n vertices contains $(n-1)!/2$ distinct Hamilton cycles.*

Proof. Since we have a complete graph, every edge is present and thus any ordering of the vertices forms a Hamilton cycle if connected in that order and closed by connecting the last vertex to the first one. There are $n!$ different orderings of n vertices, but since it is not important at which of the n vertices the cycle starts, any Hamilton cycle is counted exactly n times. Additionally, reversing the ordering gives one and the same cycle and therefore we get exactly $(n-1)!/2$ distinct Hamilton cycles in any complete graph K_n. □

For drawings of complete graphs, however, a slightly different question might be considered. Namely, the question whether there exists a Hamilton cycle that is formed by arcs that are pairwise non-intersecting, a *plane Hamilton cycle*. In the case of geometric drawings this question is easily settled.

Theorem 6.2. *Every geometric drawing of the complete graph K_n on n vertices contains at least one plane Hamilton cycle.*

Proof. Pick an arbitrary vertex v_1 and let $v_2, ..., v_n$ be the remaining vertices ordered radially around v_1. For $i = 2$ to $n-1$ consider the edges (v_i, v_{i+1}). Together with the edges (v_1, v_2) and (v_n, v_1) we have a Hamilton cycle which is plane due to the chosen ordering of the vertices and the fact that all edges are straight-line segments. □

While showing existence for geometric drawings is straight forward, the more difficult problem of determining the maximum number of crossing-free Hamilton cycles any drawing of K_n admits was first studied by Newborn and Moser [49]. For $n \leq 6$ they could show the exact values for $\Phi(n)$ and $\bar{\Phi}(n)$, the maximum number of crossing-free Hamilton cycles in any good drawing and geometric drawing of K_n, respectively. For up to $n = 9$ lower bounds were given which were subsequently extended up to $n = 13$ by Hayward [37]. His results for $\bar{\Phi}(n)$ were originally tight for $n \leq 8$, but Aichholzer and Krasser [4] could achieve correct tight bounds for $n = 9$ and $n = 10$. The following table summarizes these lower bounds.

n	3	4	5	6	7	8	9	10	11	12	13
$\bar{\Phi}(n)$	1	3	8	29	92	339	1282	4994	18383	75231	306466
$\Phi(n)$	1	3	8	29	96	399	1461	6354	24687	110162	446798

Table 6.2: Lower bounds on $\Phi(n)$ and $\bar{\Phi}(n)$ from [37] and [4]

.

In their just mentioned work, Newborn and Moser also give lower and upper bounds for geometric drawings of K_n.

$$(3/20) \cdot 10^{\lfloor n/3 \rfloor} \leq \bar{\Phi}(n) \leq 2 \cdot 6^{n-2}(\lfloor n/2 \rfloor)! \tag{6.1}$$

The lower bound in Equation 6.1 was improved by Hayward who gave a substantially better asymptotic result.

$$c \cdot 3.2684^n \leq \bar{\Phi}(n), \text{for some } c \in \mathbb{R}^+ \tag{6.2}$$

Clearly, the lower bound for $\bar{\Phi}(n)$ also holds for $\Phi(n)$. For the upper bound Hayward conjectures it to be $c \cdot 4.5^n$ for some $c \in \mathbb{R}^+$ in both cases.

As far as the above publications on the maximum number of plane Hamilton cycles are concerned, none of them mentions whether there are good drawings without any crossing-free Hamilton cycle. However, it appears to be incredibly hard to prove the existence of a single crossing-free Hamilton cycle in any $D(K_n)$. For $3 \leq n \leq 5$ one can easily see that all non-isomorphic good drawings contain a plane Hamilton cycle. Up to $n = 8$ the same could be confirmed with the use of the database created with the algorithm in Section 4.1. We therefore conjecture the following:

Conjecture 6.3. *Every good drawing of the complete graph K_n on n vertices contains a plane Hamilton cycle.*

The aforementioned computations were carried out using rotation systems, since they uniquely determine which pairs of edges cross. Interestingly, among non-realizable rotations systems there were examples that would in principle not admit any crossing-free Hamilton cycle. This implies that Conjecture 6.3 can neither be verified nor disproved without taking realizability of rotation systems into account. Furthermore, compare the existence of an abstract rotation system without crossing free Hamilton cycle to the existence of an abstract thrackle with more edges than vertices in Chapter 5.

It is interesting to note that, to the best of our knowledge, there are no publications dealing with the above conjecture. Only one very interesting doctoral thesis by Nabil Rafla [56] from 1988 could be found. Although, confusingly using the term isomorphic drawings for what we and (as he also mentions himself) others defined as weakly isomorphic drawings, he gave an algorithm to enumerate all weakly isomorphic good drawings of the complete graph on $n \leq 7$ vertices. His approach is very different from ours and relies on the conjectured existence of at least one crossing-free Hamilton cycle in any $D(K_n)$. He obtained the same results as we did, which is no surprise, since we checked Conjecture 6.3 for up to eight vertices. Just to be sure, note that our approach is completely independent of whether the conjecture is true, or not.

Interestingly, the above mentioned thesis was supervised by M. Newborn (exactly the one cited before). With obviously having dealt with crossing-free Hamilton cycles in $D(K_n)$ quite a lot, he seemed to have had no objections against the crucial assumption made in the development of the algorithm. Furthermore, according to the acknowledgements section in the said thesis, it appears that Richard Guy and Paul Erdős were also aware of the contents. Both of whom were prominently involved in topics concerning drawings of complete graphs.

One further question that can be asked in this context is whether any crossing maximal drawing admits exactly one plane Hamilton cycle. These crossing maximal sets are often referred to as “convex sets” as in the geometric setting only point sets in convex position have exactly $\binom{n}{4}$ crossings. Also for straight-line drawings each such set has exactly one crossing-free Hamilton cycle. Figure 6.2 shows, surprisingly, that this is not the case for crossing maximal good drawings of K_n.

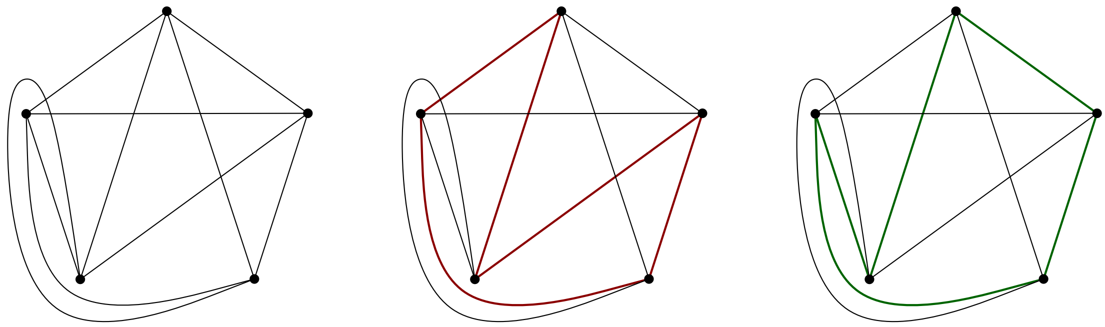


Figure 6.2: Crossing maximal good drawing of K_5 with two distinct plane Hamilton cycles.

6.2.1 Diagonals in Plane Cycles

Despite many efforts, no substantial progress could be made towards resolving Conjecture 6.3; however, a minor related result concerning diagonals in plane cycles could be established.

Definition 6.1. Let C be a plane cycle in a good drawing, and let v and w be two non-adjacent vertices of C. If the interior of the edge vw does not intersect C, then vw is called a *diagonal* of C.

We will make use of a result of Fulek and Ruiz-Vargas [26].

Definition 6.2 (Fulek, Ruiz-Vargas)**.** A *face* of a plane graph is a connected component of the complement of the graph. A vertex is *incident* to a face F if it is contained in the closure of F, but not in F. An edge e is *contained* in F if the interior of e is a subset of F.

Proposition 6.4 (Fulek, Ruiz-Vargas [26], Corollary 2.3)**.** *Let G be a simple topological graph and H be a connected plane subgraph of G with at least two vertices. Let v be a vertex of G that is not in H, and F be the face of H that contains v. Assume that for every vertex w incident to F, we have $vw \in E(G)$. Then there exist two edges in G from v to F that are contained in F.*

The main idea in the proof of Proposition 6.4 is to first show the result for the case where H is a cycle. Then, if H is not a cycle, the approach is to draw a cycle H' sufficiently close to H that separates v from H. For $|C| = 4$, it is easy to see that each vertex is incident to exactly one diagonal. We can use Proposition 6.4 to obtain the following result.

Theorem 6.5. *Let $C = \langle v_1, \dots, v_n \rangle, n \geq 5$, be a plane cycle in a good drawing of the complete graph K_n. Suppose v_1 is not incident to a diagonal of C. Then v_2 and v_3, as well as v_{n-1} and v_n are each incident to a diagonal of C. Moreover the diagonals incident to v_2 and v_3 do not cross and the same holds for the diagonals incident to v_{n-1} and v_n.*

Proof. See Figure 6.3. Consider the path $P = \langle v_3, \dots, v_n \rangle$. Due to Proposition 6.4, there have to be at least two edges from v_1 to P not crossing P. One is $v_1 v_n$. Let the other one be $v_1 v_k$, $n > k \geq 3$. Since v_1 is not incident to a diagonal of C, $v_1 v_k$ has to cross an edge of C, which can only be $v_2 v_3$; it is the only edge neither in P nor incident to v_1. Further, it follows that $k > 3$. We obtain a new cycle $\tilde{C} = \langle v_k, \dots, v_n, v_1 \rangle$, separating v_2 from the sub-path $\langle v_3, \dots, v_{k-1} \rangle$.

Again, we use Proposition 6.4 with v_2 and $\tilde{C}$. There is at least one edge in addition to v_1v_2 from v_2 to $\tilde{C}$ not crossing $\tilde{C}$, which is a diagonal of C incident to v_2. Analogously, we connect v_3 to the plane graph $\tilde{C} \cup \langle v_4, \dots, v_{k-1}\rangle$ to obtain a diagonal from v_3. Observe that the two diagonals that we found do not cross. We can apply the same argument by re-indexing C in the other direction from v_1 for v_{n-1} and v_n. $\square$

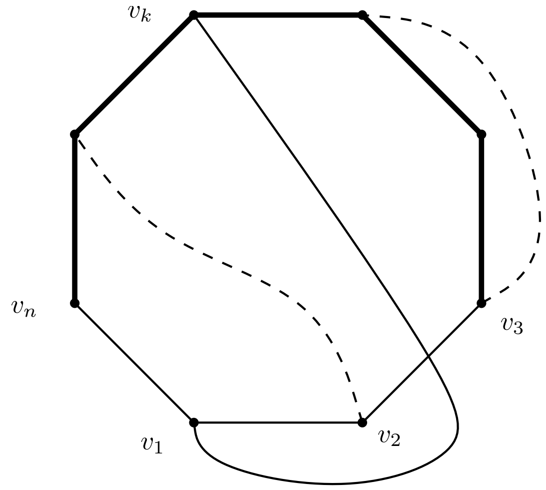


Figure 6.3: A cycle that does not contain a diagonal incident to v_1. The path P is shown bold. Recall that the choice of the unbounded face is arbitrary.

As $\geq \frac{2}{3}$ of the vertices are incident to a diagonal, and a diagonal counts for two vertices, we get the following corollary.

Corollary 6.6. *Every plane cycle of size n in a good drawing of the complete graph contains at least $\lceil n/3 \rceil$ diagonals.*

Figure 6.4 shows that this is tight for 5 vertices.

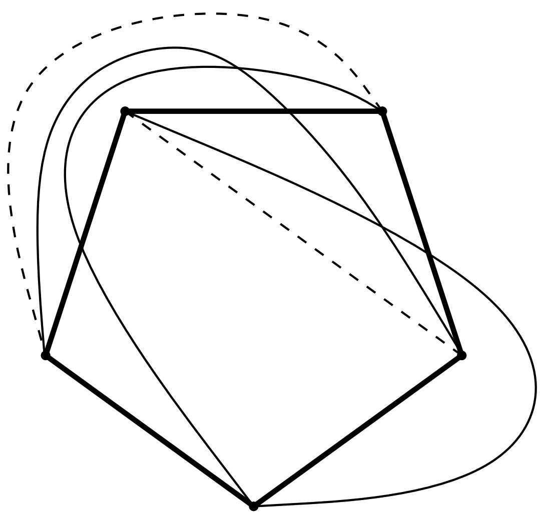


Figure 6.4: Good drawing of K_5 where the bold cycle has $\lceil n/3 \rceil = 2$ diagonals (dashed).

6.3 Plane Matchings in Drawings of the Complete Graph

A *plane matching* in a drawing of a graph is a set of pairwise non-adjacent edges which are drawn in such a way that they do not intersect each other in any point. This section summarizes results concerning the number of such *disjoint edges* one can always find in any good drawing of K_n.

A first result was published by Pach, Solymosi, and Tóth [50] and stated that the number of disjoint edges in any $D(K_n)$ is $\Omega(\log(n)^{1/6})$. Pach and Tóth [53] improved this bound to $\Omega(\log(n)/\log(\log(n)))$ and subsequently posed the problem whether it is true that there exists a constant $c > 0$ such that every good drawing of the complete graph with n vertices has at least n^c disjoint edges (see Problem 4 in Chapter 9 of [13]). The question was first answered in the affirmative by Suk [64] in 2012. Later Fulek and Ruiz-Vargas [26] gave a simpler proof of the same result that the number of disjoint edges in any $D(K_n)$ is $\Omega(n^{1/3})$. We sketch their proof in the following.

To start, consider a star G around an arbitrary vertex u. For every vertex v adjacent to u, we can remove the edge uv from this star and by Proposition 6.4, there exist two non-crossing edges connecting u with the remaining of the star. Thus, at least one edge should be different than uv. By adding this new edge to G, we obtain a new graph G' where v has degree ≥ 2. Let $G = G'$ and repeat the process for every vertex incident to u.

The resulting graph G^* is a plane good drawing where every vertex other than u has degree at least 2.

Case 1. If the maximum degree of a vertex in G^* other than u is at most $n^{2/3}$, then remove u from G^*. From the resulting graph, we can construct a matching by choosing an arbitrary edge and removing it along with all vertices adjacent to its endpoints. Since we remove at most $O(n^{2/3})$ vertices, we can repeat this process at least $\Omega(n^{1/3})$ times before running out of edges, i.e., we can construct a disjoint plane matching of size $\Omega(n^{1/3})$.

Case 2. If there is a vertex v of maximum degree $k = \Omega(n^{2/3})$ other than u, then we can look at the subgraph spanned by u, v, and all k neighbours of v. Let $V' = \{u_0, u_2, \ldots, u_{k-1}\}$ be the common neighbours of u and v sorted according to the rotation order around v. For $1 \leq i < j \leq k$, notice that the edge $u_i u_j$ can cross at most once the path $uu_h v$ for any $1 \leq h \leq k$; see Figure 6.5. Let $E = \{u_i u_j : 0 \leq i < j \leq k-1\}$ and notice that for every edge $u_i u_j \in E$, this edge crosses either the path $uu_0 v$ or the path $uu_{k/2} v$. Assume without loss of generality that at least half of the edges in E intersect the path $uu_0 v$ and let $E' \subset E$ be such a set of edges.

Let H be the subgraph with vertex set V' and edge set E'. We can show that H can be redrawn as a quasi x-monotone graph; see Figure 6.5. We will use the following lemma proved by Ruiz-Vargas and Fulek.

Lemma 6.7. *A simple quasi x-monotone topological graph on n vertices without k pairwise disjoint edges has at most $O(k^2 n)$ edges.*

Since H has at least $k = \Omega(n^{2/3})$ vertices and $\Omega(k^2) = \Omega(n^{4/3})$ edges, by Lemma 6.7 H contains at least $\Omega\left(\sqrt{(n^{\frac{4}{3}-\frac{2}{3}})}\right) = \Omega(n^{1/3})$ pairwise disjoint edges.

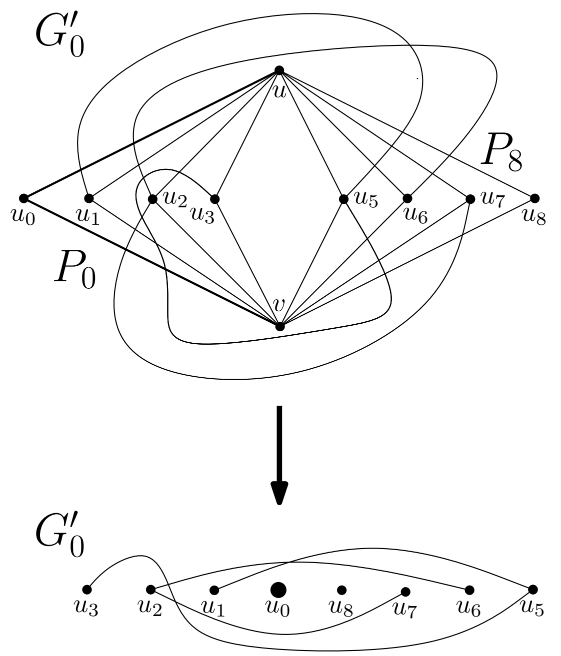


Figure 6.5: Redrawing as a quasi x-monotone graph (Figure 9 in [26])

6.3.1 Bichromatic Matchings

If the points of P are evenly coloured red and blue, then we can ask for the existence of plane bichromatic matchings. Using the computer we obtain the following result.

Lemma 6.8. *Every complete good drawing with either 4, 6, or 8 points and any evenly red and blue colouring contains a planar bichromatic perfect matching as a subgraph.*

6.4 Plane Double Stars

Another plane structure that can be shown to always exists in good drawings of K_n is a double star. Even more interestingly, we will see in the following that we can find such a plane double star with arbitrary edge degrees summing up to n for the two central vertices.

Let $G = (V, E)$ be a good drawing of a complete graph. Let p and q be to vertices of V. We define a relation between any two vertices in $V \setminus \{p, q\}$ in the following way. For any $u, v \in V \setminus \{p, q\}$, we say that $u \prec v$ if and only if the edge pv crosses the edge qu. See Figure 6.6(a).

Observation 6.9. *If $u \prec v$, then $v \not\prec u$.*

Proof. If $u \prec v$, then the edges pv and qu cross; however, the quadruple u, v, p, q can generate at most one crossing in the complete graph that they induce. Therefore, the edges pu and qv cannot cross and hence, $v \not\prec u$. $\square$

Consider the directed graph D having vertex set $V \setminus \{p, q\}$, where there is an arc from u towards v if and only if $u \prec v$.

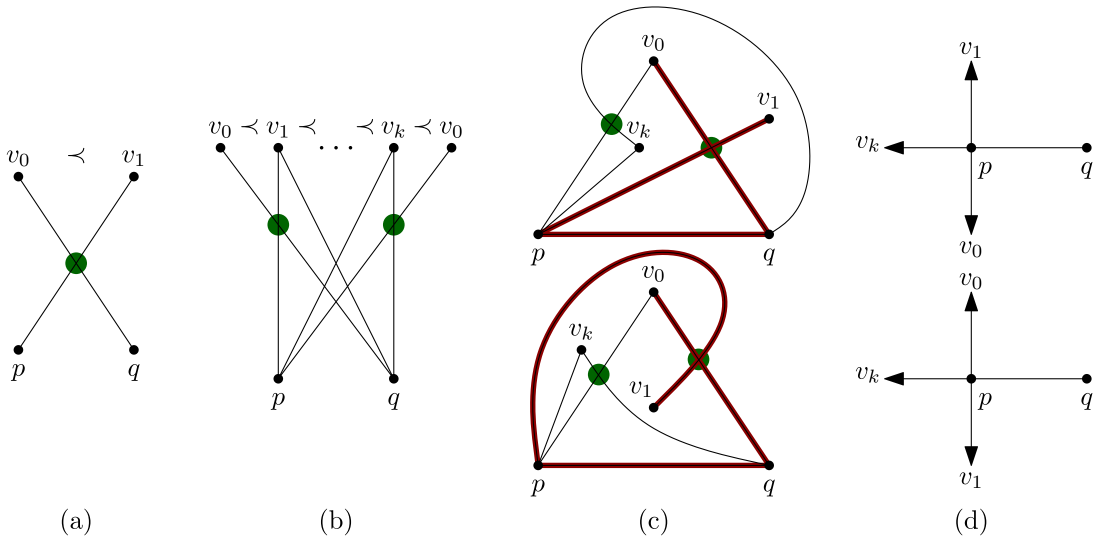


Figure 6.6: There is no cycle in the partial order of edge crossings.

Theorem 6.10. *The graph D is a Directed Acyclic Graph (DAG).*

Proof. Assume for a contradiction that there is a cycle in D. Let $v_0, v_1, \ldots, v_k \in V \setminus \{p, q\}$ be the smallest cycle in D such that $v_0 \prec v_1 \prec \ldots \prec v_k \prec v_0$. See Figure 6.6(b) for a sketch.

Because $v_k \prec v_0$, we know that $v_0 \not\prec v_k$ by Observation 6.9. That is, the edges qv_0 and pv_k cannot cross. Furthermore, the edge pv_k crosses neither the edge pv_0 nor the edge pq by the properties of a good drawing. Therefore, the edge pv_k cannot cross the boundary of the triangle $T = \triangle(p, q, v_0)$.

For each $0 < i < k$, we claim that qv_k do not cross the edge pv_i. Otherwise, if qv_k and pv_i cross, then $v_k \prec v_i$. That is, the sequence $v_i \prec v_{i+1} \prec \ldots \prec v_k \prec v_i$ forms a smaller cycle in D, which contradicts the assumption that we consider the smallest cycle.

Because $v_0 \prec v_1$, we know that the edges qv_0 and pv_1 cross. Therefore, we have two cases for the quadruple p, q, v_0, v_1 depicted in Figure 6.6(c).

Case 1. In this case, v_1 lies outside of T. Recall that qv_k and pv_0 cross. Therefore, (a part of) the edge qv_k is at some point in the interior of T. However, as qv_k is not allowed to cross pv_1, the edge qv_k cannot start in the interior of T, as the boundary of the cell formed by (parts of) the edges pq, qv_0, and pv_1 cannot be crossed; see Figure 6.6(c) top. Therefore, the edge qv_k must start in the outside of T and enter T by crossing pv_0. Since qv_k cannot cross pv_1, we conclude that v_k must lie in the cell that is inside T and bounded by (parts of) pv_0, qv_0, and pv_1. That is, the rotation (radial order) of p is given by (q, v_1, v_k, v_0).

Case 2. In this case, v_1 lies inside of T. The edge pv_1 must enter T by crossing the edge qv_0; see Figure 6.6(c) bottom. In this case, the edge pv_0 is completely contained in the cell bounded by pv_1, pq, and qv_0. The edge qv_k cannot cross neither pv_1, qv_0, nor pq. In order for qv_k to cross pv_0, qv_k must start in the interior of T and hence, it is completely bounded by (parts of) the edges pv_1, pq, and qv_0. That is, v_k must lie in the cell bounded by (parts of) pv_0, pv_1, and qv_0. Therefore, the rotation of p is given by (q, v_0, v_k, v_1).

Regardless of the case, in the rotation of p, the vertices q and v_k cannot be adjacent. See Figure 6.6(d). However, the labeling of the vertices in the cycle was arbitrary, i.e., we can relabel the vertices of the cycle by shifting the indices any fixed amount. In this way, any vertex in the cycle can play the role of v_k and we know that it cannot be adjacent to q in the rotation of p; however, if we consider the subgraph of G spanned by $\{p, q, v_0, v_1, \ldots, v_k\}$, some vertex has to be adjacent to q in the rotation of p which is a contradiction. Consequently, D has no cycle. □

As a consequence of Theorem 6.10, we obtain the following result.

Lemma 6.11. *Let p and q be two arbitrary vertices of the vertex set. Let $\delta(p) \geq 1$ and $\delta(q) \geq 1$ be their edge degrees. For each choice of $\delta(p)$ and $\delta(q)$, with $\delta(p) + \delta(q) = n$, there exists a plane double star with p and q as the two centres, having the predefined edge degrees.*

Proof. Consider the partial order $v_1 \prec v_2 \prec \ldots \prec v_{n-2}$ of the vertices of $V \setminus \{p, q\}$ with respect to p, q. For the double star induced by G, use the edges of G that connect the vertices of $\{v_1, \ldots, v_{\delta(p)-1}\}$ to p, the vertices of $\{v_{\delta(p)}, \ldots, v_{n-2}\}$ to q, and p to q. The edges of the stars around p and q, respectively, do not intersect because they are induced by a good drawing. If an edge pv_i would intersect an edge qv_j then, because of the partial order with respect to p, q, $v_j \prec v_i$. But we chose the edges pv_i and qv_j such that $v_i \prec v_j$ for all $1 \leq i \leq \delta(p) - 1$ and $\delta(p) \leq j \leq n - 2$. Hence, the constructed double star is plane. □

6.5 Strictly Weight Decreasing Untangling

To obtain a plane sub graph – like a plane spanning cycle, a plane spanning tree, or a plane perfect matching – of a given drawing of a complete graph, often the following standard approach is used. Start with a (not necessarily plane) graph from the same graph class and locally "untangle" its crossings. For this untangling we replace two crossing edges by two incident edges which don't cross each other. More precisely, assume that for the four vertices $v_1, \ldots, v_4$ the edges v_1v_2 and v_3v_4 properly intersect and are part of the current graph. Then we replace them by one of the non-intersecting pair v_1v_3 and v_2v_4 or v_1v_4 and v_2v_3.

It is easy to see that for the above mentioned graph classes, this can be done such that the resulting graph still belongs to the same class. In this way we "locally" remove one crossing. If we can repeat this process until no crossings remain, we obtain a plane graph of this class; however, in this process new crossings might be introduced. Moreover, the process might even cycle, that is, removed crossings re-appear later again. In order to guarantee that the process terminates after a finite number of steps, it is usually shown that each untangling reduces some well defined weight of the graph. In the geometric setting, this is, for example, the sum of the length of all edges of the graph. Basic geometry shows that the sum of the length of v_1v_2 and v_3v_4 is strictly larger than that of v_1v_3 and v_2v_4 or v_1v_4 and v_2v_3.

Instead of considering the length of the edges, we can consider a weighted graph G, such that each edge v_iv_j obtains a non-negative weight $w(v_iv_j)$. The total sum of the weights of all edges of G is the weight $w(G)$. Using the same notation as above, assume that for any crossing of the graph the following two relations hold:

$$\begin{aligned}
w(v_1v_2) + w(v_3v_4) &> w(v_1v_3) + w(v_2v_4) \\
w(v_1v_2) + w(v_3v_4) &> w(v_1v_4) + w(v_2v_3)
\end{aligned}$$

Then $w(G)$ is reduced in each untangling step, which implies that the above described approach terminates for G in a finite number of steps.

Can we find such weights for any given good drawing of the complete graph? In the following we answer this question in the negative.

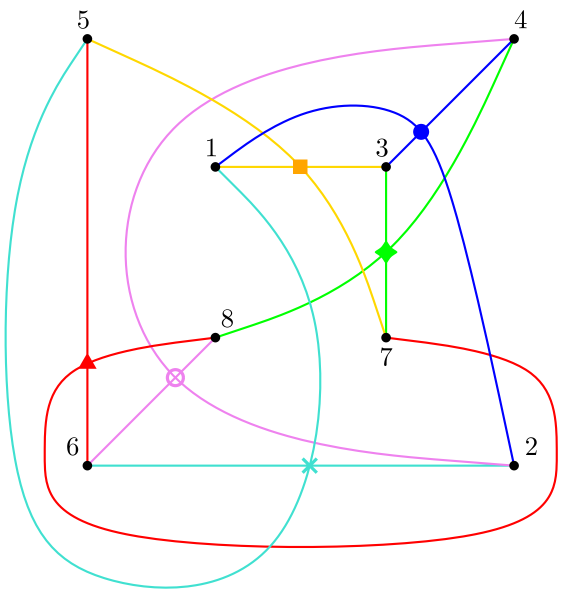


Figure 6.7: A good drawing of a cube-graph where for each face one pair of edges crosses.

Consider the good drawing of Figure 6.7. We will use six of its crossings to obtain a system of six inequalities for the weights of its edges. To this end consider the cube-like structure that this graph has, depicted in Figure 6.8.

This structure leads to the following system:

$$\begin{aligned}
w(v_1v_2) + w(v_3v_4) &> w(v_1v_3) + w(v_2v_4) \\
w(v_1v_3) + w(v_5v_7) &> w(v_1v_5) + w(v_3v_7) \\
w(v_2v_4) + w(v_6v_8) &> w(v_2v_6) + w(v_4v_8) \\
w(v_1v_5) + w(v_2v_6) &> w(v_1v_2) + w(v_5v_6) \\
w(v_3v_7) + w(v_4v_8) &> w(v_3v_4) + w(v_7v_8) \\
w(v_5v_6) + w(v_7v_8) &> w(v_5v_7) + w(v_6v_8)
\end{aligned}$$

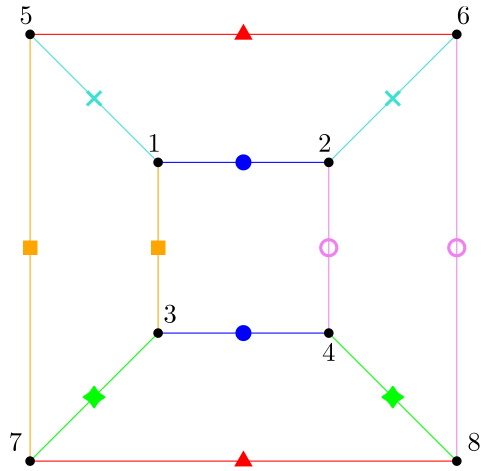


Figure 6.8: Cube structure of the drawing in Figure 6.7, without crossings.

As each involved edge shows up twice, once on each side of the inequalities, summing over all six inequalities results in $0 > 0$, a contradiction. It remains to show that the good drawing of Figure 6.7 can, in fact, be completed to a good drawing of the complete graph. Figure 6.9 shows that this can be done and we thus obtain the following statement.

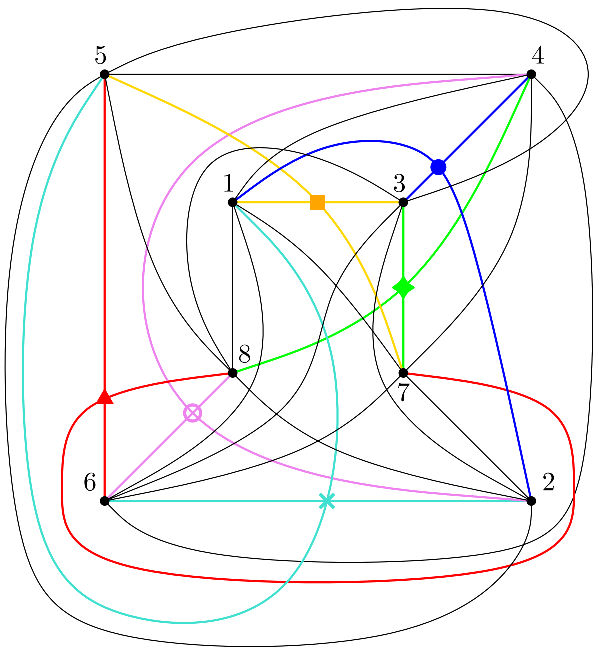


Figure 6.9: The good drawing of Figure 6.7 can be completed to a good drawing of the complete graph on 8 vertices.

Lemma 6.12. *There exist good drawings of the complete graph on $n \geq 8$ vertices which do not allow a weight assignment to the edges of this graph such that each untangling is strictly weight reducing.*

In Figure 6.10 a different completion to a drawing of a K_8 is depicted.

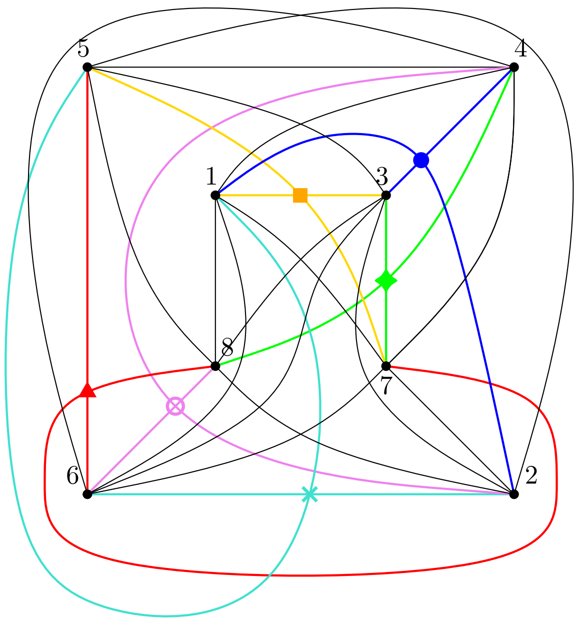


Figure 6.10: The good drawing of Figure 6.7 can be completed to a good drawing of the complete graph on 8 vertices; second version, different rotation system.

We finally checked all 102 realizable rotation systems for $n = 6$ points. For 101 of them we obtained valid integer weights (range $1 \ldots 10$), but for the remaining set this is not possible. From its 15 crossings we derive 30 inequalities and among them are the following six:

$$
\begin{aligned}
w(v_1v_3) + w(v_2v_5) &> w(v_1v_5) + w(v_2v_3)\\
w(v_1v_4) + w(v_2v_6) &> w(v_1v_6) + w(v_2v_4)\\
w(v_1v_5) + w(v_4v_6) &> w(v_1v_4) + w(v_5v_6)\\
w(v_1v_6) + w(v_3v_4) &> w(v_1v_3) + w(v_4v_6)\\
w(v_2v_3) + w(v_5v_6) &> w(v_2v_6) + w(v_3v_5)\\
w(v_2v_4) + w(v_3v_5) &> w(v_2v_5) + w(v_3v_4)
\end{aligned}
$$

As before, summing them up gives the contradiction $0 > 0$. The structure of this system, however, is now the cross polytope, i.e., the dual of the cube. See Figure 6.11 for the related drawings. As we know (and checked), all realizable rotation systems for $n \leq 5$ do not provide such examples.

A question that arises from the previous observations is the following: under which conditions do good drawings of the complete graph allow a weight assignment such that each untangling is weight reducing?

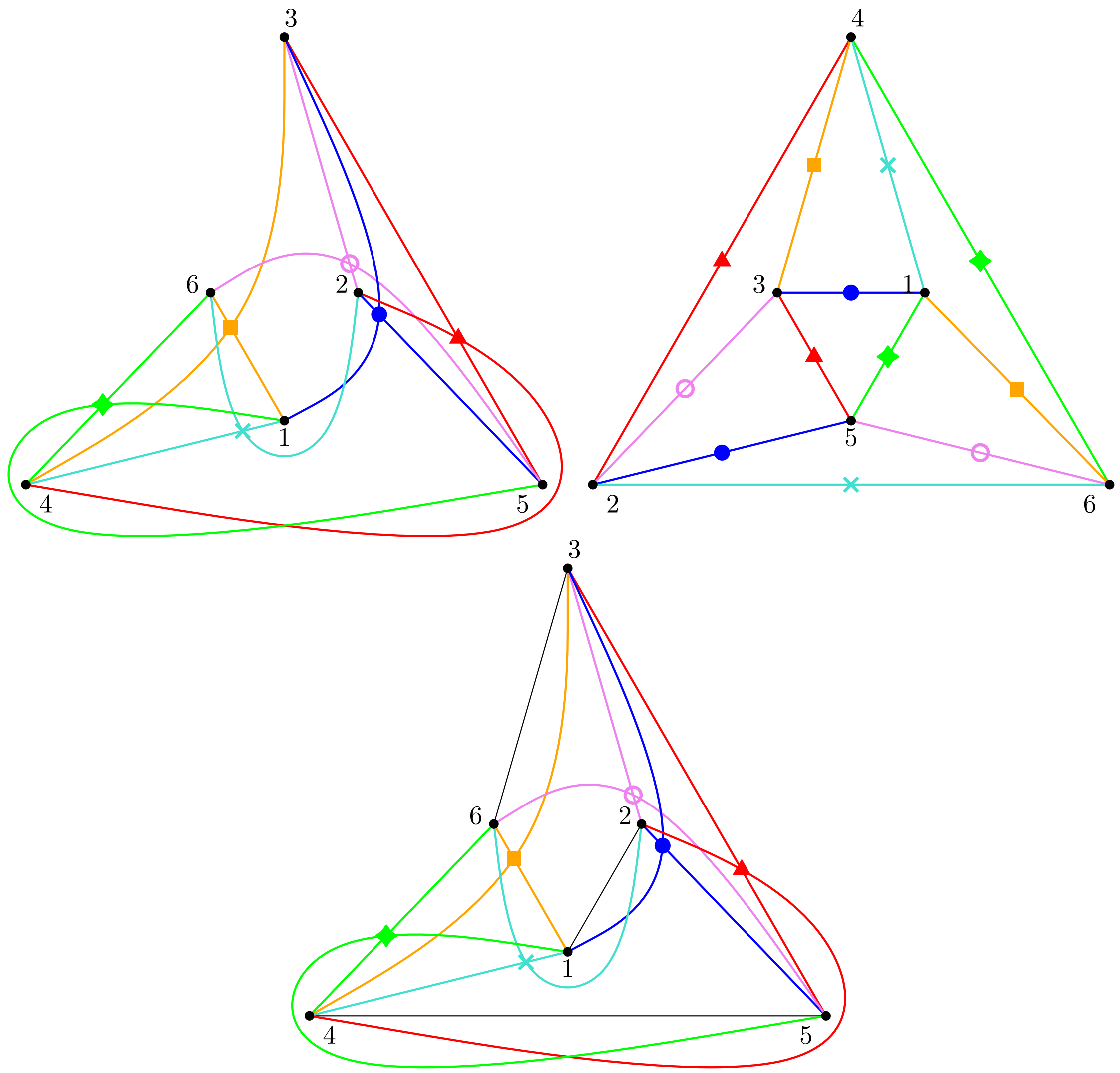


Figure 6.11: The good drawing of the 6-point example, the octaeder structure, and a good drawing of the resulting complete graph.

6.6 Unavoidable Edges

As already mentioned in Chapter 1, another interesting problem is to consider *uncrossed* or *unavoidable* edges in drawings of K_n, i.e., edges that are not crossed by any other edge of the drawing. We mentioned that for $n \geq 8$ there always exists a $D(K_n)$ without uncrossed edges (see Table 1.1). We present in the following the construction of Harborth and Mengersen [35] that proves the said property:

For now suppose that n is even. The vertices are placed as a convex polygon ordered clockwise and all but the following edges are drawn as diagonals: $(i, i+3)$ for $i = 1, ..., n$, $(2i, 2i+2)$ and $(2i-1, 2i+3)$ for $i = 1, ..., \frac{n}{2}$. See Figure 6.12 (leftmost) for the construction so far and notice that all diagonal edges are crossed at least once.

Now, the $(i, i+3)$ edges are drawn either starting inside the polygon if i is odd, or starting outside if i is even. Then each such edge intersects $(i+1, i+2)$ and directly connects to the corresponding endpoint. The current drawing can be seen in Figure 6.12 (centre left).

The remaining edges are simply connected through the exterior of the polygon giving the final drawing without uncrossed edges as in Figure 6.12 (centre right).

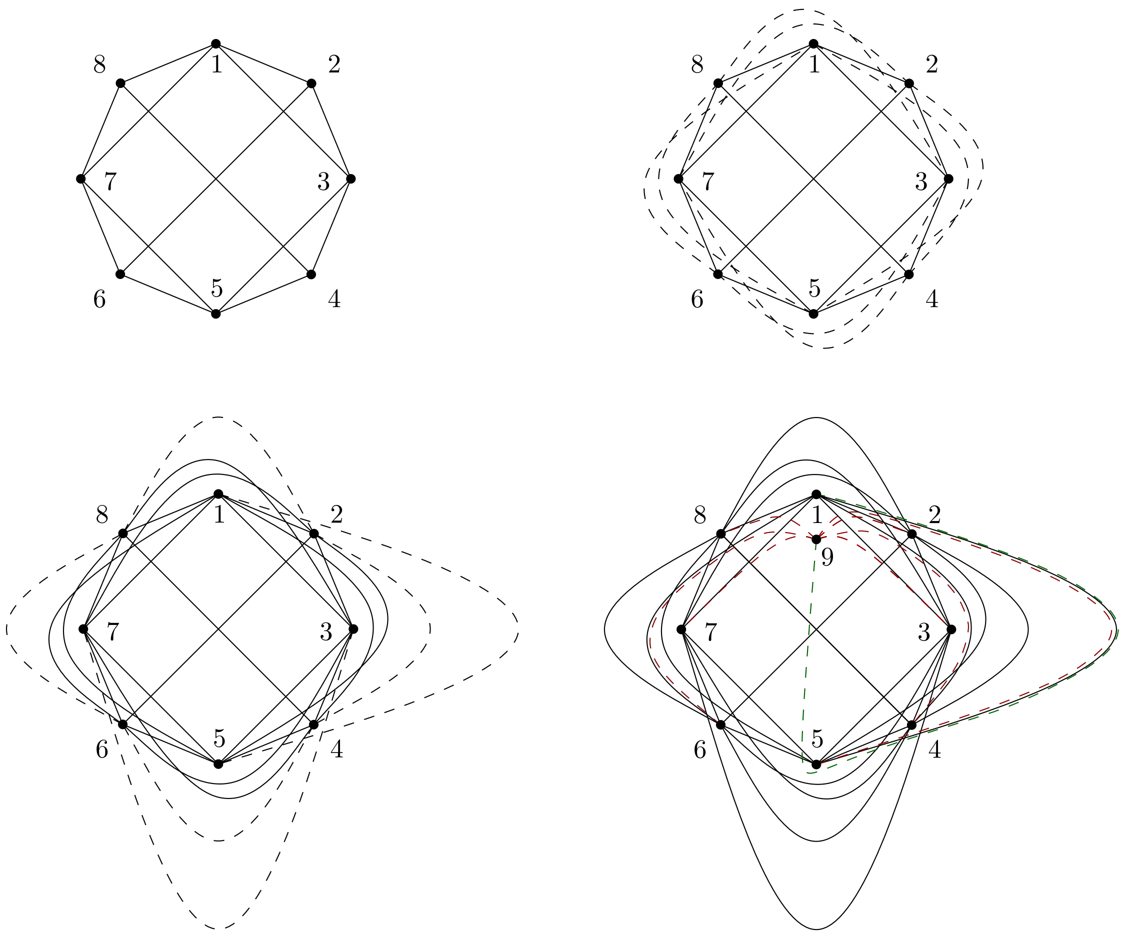


Figure 6.12: Construction of $D(K_8)$ and $D(K_9)$ without uncrossed edges.

It remains to extend the above construction to the case where n is odd. We therefore place the n^{th} vertex close to vertex 1 and draw all edges (n, i) for $i = 2, ..., n-1$ in such a way, that it goes directly very close to the corresponding edge $(1, i)$ and then follows it to the endpoint i. This will certainly induce crossings on all these edges and the only one left is $(1, n)$. See the dashed edges drawn in red in the bottom right picture of Figure 6.12. The last edge can always be drawn such that it follows $(1, 5)$ on the side that was not yet used (green dashed edge). All of these have no mutual crossings and we therefore get a drawing on an odd number of vertices with at least one crossing per edge.

Another result that Harborth and Mengersen attribute to Ringel [58] is that the maximum number of unavoidable edges in any drawing for $n \geq 4$ vertices is $H(n) = 2n - 2$. Moreover, in [13] they ask for bounds on the least integer $l(n)$ such that any good drawing on n vertices contains an edge with fewer than $l(n)$ crossings. Using a rather sophisticated construction, Kynčl and Valtr [44] showed a lower bound of $\Omega(n^{3/2}) \leq l(n)$. Also an upper bound of $l(n) \leq O(n^2/\log(n)^{1/4})$ is given.

Pach and Tóth [52] conjectured that for any complete topological graph (not necessary simple) on $n \geq 5$ vertices, there is some $\delta > 0$ such that there are at least n^δ pairwise crossing edges. Fox and Pach could eventually confirm this conjecture and establish the following theorem.

Theorem 6.13 (Fox, Pach [22]). *For every $\epsilon > 0$ and every integer $t > 0$, there exists $\delta > 0$ and a positive integer n_0 with the following property. If G is a topological graph with $n \geq n_0$ vertices and at least $n^{1+\epsilon}$ edges such that no pair of them intersect in more than t points, then G has n^δ pairwise crossing edges.*

For good drawings of the complete graph we have $t = 1$ and $\epsilon > 0$ and thus the statement holds.

6.7 Empty Triangles

For geometric drawings the definition of a triangle is rather intuitive. In the case of good drawings, we should, however, define a triangle and more specifically the notion of emptiness with care.

Definition 6.3. In a good drawing $D(G)$ of a graph G we denote the induced drawing of any clique of order 3 as a *triangle* Δ.

Clearly, in a good drawing of K_n the induced subgraph of any three vertices constitutes a triangle. Furthermore, such a triangle forms a closed Jordan curve and thus partitions the plane (or sphere) into two connected components. When drawn in the plane, one of the components is bounded and the other is unbounded.

Definition 6.4. A triangle Δ in a good drawing $D(G)$ is said to be empty, if one of the components formed by Δ does not contain any other point of $D(G)$.

We consider the problem of determining $t(n)$, the minimum number of empty triangles in any $D(K_n)$. While in the case of geometric drawings lower and upper bounds are of quadratic order (see [2] and [9] for lower and upper bounds, respectively), Harboth [34] could provide examples that the minimum number of empty triangles in any $D(K_n)$ is at most $2n - 4$. He could show that this is tight for $3 \leq n \leq 6$ and raised the question whether $t(n) = 2n - 4$ in general. While he was only able to show that $t(n) = 2$ for $n \geq 3$, the best bound known to date is due to Aichholzer et al. [3].

Theorem 6.14 (Aichholzer et al., [3]). *For $n \geq 4$, the number of empty triangles in any good drawing $D(K_n)$ of the complete graph K_n with n vertices is at least n.*

Aside from their proof of Theorem 6.14, they mention an interesting relation to rotation systems of complete graph drawings.

Lemma 6.15 (Aichholzer et al., [3])**.** *Let v_1, v_2, v_3 be three vertices of a good drawing $D(K_n)$ of the complete graph K_n on n vertices that form the triangle Δ. Whether Δ is empty is determined by the rotation system $\mathcal{R}(D(K_n))$.*

The above allows to determine the minimum number of empty triangles in any good drawing of the complete graph for $4 \leq n \leq 8$ by testing all different rotation systems for that property. This was in fact done by Aichholzer et al. [3] and their results confirmed the conjecture of Harboth that any $D(K_n)$ contains at least $2n - 4$ empty triangles.

Observation 6.16 (Aichholzer et al., [3])**.** *For $3 \leq n \leq 8$, the number of empty triangles in a good drawing of K_n is at least $2n - 4$.*

This supports that Harboth's question can be answered in the affirmative.

Conjecture 6.17. *Any good drawing $D(K_n)$ of the complete graph K_n contains at least $2n - 4$ empty triangles.*

7 Summary and Open Problems

We have recalled the result of Kynčl [44] that weak isomorphism of good drawings of the complete graph is determined by the rotation system. We made use of his result to implement an algorithm and build up a database of all realizable rotation systems and non-isomorphic drawings of $D(K_n)$ for $3 \leq n \leq 8$. This database can subsequently be used for testing all good drawings of K_n with up to eight vertices for certain interesting properties. It allows to make observations and thus gain more insight into the actual problem of interest, which could eventually lead to a fully analytical proof.

A critical part of this algorithm is evidently verifying realizability. Although Kynčl proved that this problem can be solved in polynomial time [43], we only implemented a non-polynomial time backtracking algorithm for this purpose.

As observations suggest [1], this part could, however, be drastically simplified. While there are non-realizable rotation systems where the rotation system of every K_4 subdrawing is realizable, no such examples could be found that contain only realizable 5-tuples. If one could prove that realizability of all K_5 subdrawings is a necessary and in particular sufficient condition for realizability of the entire $D(K_n)$, this part of the algorithm would reduce to simply testing all $\Theta(n^5)$ 5-tuples of vertices for realizability. Apart from also being more efficient than Kynčl's approach, it is evidently also easier to implement.

As the drawings in Section 4.1.5 suggest, the simple method of embedding the planarized graph obtained from a certain $D(K_n)$ with straigh-line edges does not yield satisfactory results. Aside from aesthetics the visualizations hardly give any better insight into the structure of the drawings. More sophisticated methods that allow for bent edges and take for instance the angular resolution of the drawing into account could be applied. It is, however, questionable that

[1] Personal communication: Oswin Aichholzer, Thomas Hackl, Alexander Pilz, Birgit Vogtenhuber

such algorithms could produce appealing visualizations for complete graphs with many vertices. It is probably unavoidable to extract certain structural properties giving hints for an adequate placement of the vertices and shape of the arcs connecting them.

After summarizing what is known to date about the thrackle conjecture, a database with unique thrackle drawings was created in a similar manner. For $2 \leq n \leq 9$ we could store all connected thrackles, i.e., all thrackle with one edge less than vertices and all thrackles with the same number of edges and vertices. In the attempt of finding a minimal counterexample, we pointed out that it suffices to only consider all thrackle drawings whose underlying graph is a path on n vertices. With further observations we modified the algorithm to construct a database of all path thrackles for $2 \leq n \leq 12$. By failing in disproving Conway's conjecture, we could, however, give evidence that a minimal counterexample must have at least 13 vertices. The best known previous result was due to Curt McMullen. He too established by massive computer search that a counterexample must have at least 11 or 12 vertices. This was only mentioned by John Conway himself in a response to a question in a mathematics forum and he was not certain about the exact number [1].

In Table 5.2 it can be seen that even the number of non-isomorphic path thrackles grows astonishingly fast. An exhaustive enumeration and subsequent test for the existence of counterexamples is no longer feasible for $n \geq 13$. In order to further follow this approach, one would need to gain more insight in the structural properties of possible counterexamples. Maybe the number of path thrackles that actually need to be constructed can be drastically reduced, allowing for covering all the candidates that could be extended to a counterexample for larger n. So far, however, no promising approaches in that direction are in sight.

[1] http://mathforum.org/kb/message.jspa?messageID=1376434

List of Figures

List of Tables

Bibliography

[1] B. Ábrego, O. Aichholzer, S. Fernández-Merchant, P. Ramos, and G. Salazar. Shellable drawings and the cylindrical crossing number of K_n. *ArXiV*, http://arxiv.org/abs/1309.3665:1–9, 2013.

[2] O. Aichholzer, R. Fabila-Monroy, T. Hackl, C. Huemer, A. Pilz, and B. Vogtenhuber. Lower bounds for the number of small convex k-holes. In *Proc. 24^{th} Annual Canadian Conference on Computational Geometry CCCG 2012*, pages 247–252, Charlottetown, PEI, Canada, 2012.

[3] O. Aichholzer, T. Hackl, A. Pilz, P. A. Ramos, V. Sacristan, and B. Vogtenhuber. Empty triangles in good drawings of the complete graph. *ArXiV*, http://arxiv.org/abs/1306.5081:1–9, 2013.

[4] O. Aichholzer and H. Krasser. The point set order type data base: A collection of applications and results. In *CCCG*, pages 17–20, 2001.

[5] K. Appel and W. Haken. *Every Planar Map is Four Colorable.* American Mathematical Society, 1 edition, 1986.

[6] D. Archdeacon and R. B. Richter. On the parity of crossing numbers. *Journal of Graph Theory*, 12(3):307–310, 1988.

[7] C. Auer, F. J. Brandenburg, A. Gleißner, and J. Reislhuber. On 1-planar graphs with rotation systems. Technical report, Tech. Rep. MIP-1207, Fakultät für Informatik und Mathematik, Universität Passau, 2012.

[8] M. Balko, R. Fulek, and J. Kynčl. Crossing numbers and combinatorial characterization of monotone drawings of K_n. *arXiv preprint arXiv:1312.3679*, pages 1–43, 2013.

[9] I. Bárány and P. Valtr. Planar point sets with a small number of empty convex polygons. *Studia Scientiarum Mathematicarum Hungarica*, 41(2):243–269, 2004.

[10] H. Baron. Lösung der Aufgabe 167. *Jahresbericht Deutsch. Math.-Verein*, 45:112, 1935.

[11] P. Bertolazzi, G. Battista, G. Liotta, and C. Mannino. Upward drawings of triconnected digraphs. *Algorithmica*, 12(6):476–497, 1994.

[12] J. Bondy and U. Murty. *Graph Theory*, volume 244 of *Graduate Texts in Mathematics*. Springer, 2008.

[13] P. Brass, W. O. J. Moser, and J. Pach. *Research problems in discrete geometry*. Springer, 2005.

[14] G. Cairns and Y. Nikolayevsky. Bounds for generalized thrackles. *Discrete & Computational Geometry*, 23(2):191–206, 2000.

[15] G. Cairns and Y. Nikolayevsky. Outerplanar thrackles. *Graphs and Combinatorics*, 28(1):85–96, 2012.

[16] G. Di Battista, P. Eades, R. Tamassia, and I. G. Tollis. *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice Hall, Upper Saddle River, NJ, USA, 1st edition, 1998.

[17] A. Donafee and C. Maple. Planarity testing for graphs represented by a rotation scheme. In *Proceedings of the Seventh International Conference on Information Visualization*, pages 491–496, 2003.

[18] J. Edmonds. A combinatorial representation of polyhedral surfaces. *Notices of the American Mathematical Society*, 7:646, 1960.

[19] P. Erdős. On sets of distances of n points. *American Mathematical Monthly*, 53(5):248–250, May 1946.

[20] P. Erdős and R. K. Guy. Crossing number problems. *The American Mathematical Monthly*, 80(1):52–58, 1973.

[21] W. Fenchel. Lösung der Aufgabe 167. *Jahresbericht Deutsch. Math.-Verein*, 45:34–35, 1935.

[22] J. Fox and J. Pach. Coloring K_k-free intersection graphs of geometric objects in the plane. In *Proceedings of the twenty-fourth annual symposium on Computational geometry*, pages 346–354. ACM, 2008.

[23] P. Franklin. A six color problem. *J. Math. Phys.*, 13:363–379, 1934.

[24] R. Fulek and J. Pach. A computational approach to Conway's thrackle conjecture. *Computational Geometry*, 44(6):345–355, 2011.

[25] R. Fulek, M. J. Pelsmajer, M. Schaefer, and D. Štefankovič. Adjacent crossings do matter. *Journal of Graph Algorithms and Applications*, 16(3):759–782, 2012.

[26] R. Fulek and A. J. Ruiz-Vargas. Topological graphs: empty triangles and disjoint matchings. In *Proc. 29th ACM Symposium on Computational Geometry (SoCG'13)*, pages 259–265, Rio de Janeiro, Brazil, 2013.

[27] M. Garey and D. Johnson. Crossing number is NP-complete. *SIAM Journal on Algebraic Discrete Methods*, 4(3):312–316, 1983.

[28] A. Garg and R. Tamassia. On the computational complexity of upward and rectilinear planarity testing. In R. Tamassia and I. Tollis, editors, *Graph Drawing*, volume 894 of *Lecture Notes in Computer Science*, pages 286–297. Springer Berlin Heidelberg, 1995.

[29] E. Gioan. Complete graph drawings up to triangle mutations. In D. Kratsch, editor, *Graph-Theoretic Concepts in Computer Science*, volume 3787 of *Lecture Notes in Computer Science*, pages 139–150. Springer Berlin Heidelberg, 2005.

[30] H.-D. O. Gronau and H. Harborth. Numbers of nonisomorphic drawings for small graphs. *Congressus Numerantium*, 71:105–114, 1990.

[31] J. L. Gross and T. W. Tucker. *Topological Graph Theory*. Wiley-Interscience, New York, NY, USA, 1987.

[32] R. K. Guy. *The decline and fall of Zarankiewicz's theorem*. University of Calgary, Department of Mathematics, 1968.

[33] F. Harary and A. Hill. On the number of crossings in a complete graph. *Proceedings of the Edinburgh Mathematical Society (Series 2)*, 13(04):333–338, 1963.

[34] H. Harborth. Empty triangles in drawings of the complete graph. *Discrete Mathematics*, 191:109 – 111, 1998.

[35] H. Harborth and I. Mengersen. Edges without crossings in drawings of complete graphs. *Journal of Combinatorial Theory, Series B*, 17(3):299 – 311, 1974.

[36] H. Harborth and I. Mengersen. Drawings of the complete graph with maximum number of crossings. In *Proc. of the 23rd Southeastern International Conference on Combinatorics, Graph Theory, and Computing (Boca Raton, FL, 1992), Congressus Numerantium 88*, pages 225–228, Boca Raton, FL, 1992.

[37] R. B. Hayward. A lower bound for the optimal crossing-free hamiltonian cycle problem. *Discrete & Computational Geometry*, 2(1):327–343, 1987.

[38] L. Heffter. Über das Problem der Nachbargebiete. *Mathematische Annalen*, 38(4):477–508, 1891.

[39] Ø. Hjelle and M. Dæhlen. *Triangulations and Applications*. Springer, 2006.

[40] H. Hopf and E. Pannwitz. Aufgabe Nr. 167. *Jahresbericht Deutsch. Math.-Verein*, 43:114, 1934.

[41] D. J. Kleitman. The crossing number of $K_{5,n}$. *Journal of Combinatorial Theory*, 9(4):315 – 323, 1970.

[42] V. P. Korzhik and B. Mohar. Minimal obstructions for 1-immersions and hardness of 1-planarity testing. In I. Tollis and M. Patrignani, editors, *Graph Drawing*, volume 5417 of *Lecture Notes in Computer Science*, pages 302–312. Springer Berlin Heidelberg, 2009.

[43] J. Kynčl. Improved enumeration of simple topological graphs. *Discrete & Computational Geometry*, 50(3):727–770, 2013.

[44] J. Kynčl. Enumeration of simple complete topological graphs. *Eur. J. Comb.*, 30(7):1676–1685, Oct. 2009.

[45] J. Kynčl. Simple realizability of complete abstract topological graphs in P. *Discrete & Computational Geometry*, 45(3):383–399, 2011.

[46] V. Leck. The numbers of nonisomorphic drawings of the connected graphs with 6 vertices, 1998.

[47] L. Lovász, J. Pach, and M. Szegedy. On Conway's thrackle conjecture. *Discrete & Computational Geometry*, 18(4):369–376, 1997.

[48] B. Mohar and C. Thomassen. *Graphs on Surfaces.* Johns Hopkins series in the mathematical sciences. Johns Hopkins University Press, 2001.

[49] M. Newborn and W. Moser. Optimal crossing-free hamiltonian circuit drawings of K_n. *Journal of Combinatorial Theory, Series B*, 29(1):13 – 26, 1980.

[50] J. Pach, J. Solymosi, and G. Tóth. Unavoidable configurations in complete topological graphs. *Discrete & Computational Geometry*, 30(2):311–320, 2003.

[51] J. Pach and E. Sterling. Conway's conjecture for monotone thrackles. *The American Mathematical Monthly*, 118(6):544–548, 2011.

[52] J. Pach and G. Tóth. Unavoidable configurations in complete topological graphs. In *Graph Drawing*, pages 328–337. Springer, 2001.

[53] J. Pach and G. Tóth. Disjoint edges in topological graphs. In *Combinatorial Geometry and Graph Theory*, pages 133–140. Springer, 2005.

[54] J. Pach and G. Tóth. How many ways can one draw a graph? In G. Liotta, editor, *Graph Drawing*, volume 2912 of *Lecture Notes in Computer Science*, pages 47–58. Springer Berlin Heidelberg, 2004.

[55] M. J. Pelsmajer, M. Schaefer, and D. Štefankovič. Crossing number of graphs with rotation systems. In S.-H. Hong, T. Nishizeki, and W. Quan, editors, *Graph Drawing*, volume 4875 of *Lecture Notes in Computer Science*, pages 3–12. Springer Berlin Heidelberg, 2008.

[56] N. H. Rafla. *The Good Drawings D_n of the Complete Graph K_n.* PhD thesis, McGill University, Montreal, Quebec, March 1988.

[57] G. Ringel. Bestimmung der Maximalzahl der Nachbargebiete auf nichtorientierbaren Flächen. *Mathematische Annalen*, 127(1):181–214, 1954.

[58] G. Ringel. Extremal problems in the theory of graphs. In *Theory of Graphs and its Applications, Proc. Symposium Smolenice*, pages 85–90, 1963.

[59] G. Ringel. Das Geschlecht des vollständigen paaren Graphen. In *Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg*, volume 28, pages 139–150. Springer, 1965.

[60] G. Ringel. The combinatorial map color theorem. *Journal of Graph Theory*, 1(2):141–155, 1977.

[61] G. Ringel and J. Youngs. Solution of the Heawood map-coloring problem. *Proceedings of the National Academy of Sciences of the United States of America*, 60(2):438, 1968.

[62] N. Robertson, D. Sanders, P. Seymour, and R. Thomas. The four-colour theorem. *Journal of Combinatorial Theory, Series B*, 70(1):2–44, 1997.

[63] S. Stahl. Generalized embedding schemes. *Journal of Graph Theory*, 2(1):41–52, 1978.

[64] A. Suk. Disjoint edges in complete topological graphs. In *Proceedings of the 2012 symposuim on Computational Geometry*, pages 383–386. ACM, 2012.

[65] J. Sutherland. Lösung der Aufgabe 167. *Jahresbericht Deutsch. Math.-Verein*, 45:33–34, 1935.

[66] C. Thomassen. Embeddings and minors. In R. L. Graham, M. Grötschel, and L. Lovász, editors, *Handbook of combinatorics (vol. 1)*, chapter Embeddings and minors, pages 301–349. Elsevier, Cambridge, MA, USA, 1995.

[67] P. Turán. A note of welcome. *Journal of Graph Theory*, 1(1):7–9, 1977.

[68] B. Vogtenhuber. On Plane Straight-Line Graphs. Master's thesis, IST-TU Graz, Austria, January 2007. Supervisor: Oswin Aichholzer.

[69] K. Weiler. Edge-based data structures for solid modeling in curved-surface environments. *Computer Graphics and Applications, IEEE*, 5(1):21–40, Jan 1985.

[70] H. Whitney. Congruent graphs and the connectivity of graphs. In J. Eells and D. Toledo, editors, *Hassler Whitney Collected Papers*, Contemporary Mathematicians, pages 61–79. Birkhäuser Boston, 1992.

[71] D. R. Woodall. Thrackles and deadlock. *Combinatorial Mathematics and Its Applications*, 348:335–348, 1971.

[72] D. R. Woodall. Cyclic-order graphs and Zarankiewicz's crossing-number conjecture. *Journal of Graph Theory*, 17(6):657–671, 1993.

[73] C. Zarankiewicz. On a problem of P. Turán concerning graphs. *Fundamenta Mathematicae*, 41(1):137–145, 1955.